UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

METODOLOGÍA PARA PROYECTOS DE SOFTWARE ORIENTADO A OBJETOS BASADA EN CMM Y EL PROCESO UNIFICADO DE DESARROLLO DE SOFTWARE

T E S I S

QUE PARA OBTENER EL TÍTULO DE

INGENIERO EN COMPUTACIÓN

P R E S E N T A :

GUZMÁN CASTRO LUIS ALEJANDRO

DIRECTOR: ING. JUAN JOSÉ CARREÓN GRANADOS

CIUDAD UNIVERSITARIA **MÉXICO, D. F., 2003**

Dedicada a María Castro Hernández

Agradezco a María Castro, mi madre
por ayudarme a ser lo que soy el día de hoy
y darme las herramientas para vivir.

Agradezco a Maribel Miranda, mi esposa,
por los días que le robé para finalizar este trabajo.
Te amo Maribel.

Agradezco a mi hermano Carlos
porque sin saberlo
me motivó a continuar estudiando.

Agradezco a toda mi familia y amigos
porque de ellos deriva lo que soy hoy.

Agradezco al Instituto de Ingeniería de la UNAM
por el apoyo académico a lo largo de mi carrera,
en especial a Angélica Lozano y Marco Ambríz.

Te agradezco a ti,
que lees estas líneas,
pues ya eres parte de este esfuerzo

Gracias.

ÍNDICE

INDICE:

1

Introducción

A manera de introducción, este capítulo expone los objetivos del trabajo realizado, así como las consideraciones que se tuvieron para el desarrollo del mismo, finalmente explica su organización.

1.1. Prefacio

Cada día es mas evidente la influencia que tiene el software dentro de los procesos de cualquier organización, sin embargo, no es culpa ni merito solamente de los programadores que el software de poca calidad sea algo cotidiano para todos. No es necesario estar en el centro de una organización de desarrollo de software, ni ser un programador experto, para sentir los efectos de este problema; basta con recibir un estado de cuenta bancaria erróneo, esperar horas para la reservación de vuelos, o incluso encontrar un estado indefinido en el software de control de nuestro autoestéreo para intuir que algo no anda bien en la creación del software.

La exclusión de procesos bien definidos para la creación de software es el acontecimiento lógico principal de nuestro tiempo, refiriendo éste como lógico debido a una mala administración de proyectos que orillan a la creación “al vapor” de software en ocasiones crítico, obligando así a los programadores a evitar procedimientos que a primera apariencia resultaran innecesarios. A pesar de esto es inútil buscar argumentos de buena voluntad, el hecho es que aun no se ha dejado del todo atrás la era artesanal del software, y a pesar de definir metodologías que buscan superar esto, ellas rara vez combinan aspectos técnicos y administrativos de forma tal que sean adoptadas como hábitos de desarrollo.

Es claro también, que la creación de una metodología de desarrollo de software que resuelva estos problemas es una faena difícil, la cual y a pesar de todos nosotros (los afectados por el software de baja calidad) llevará un periodo muy largo de tiempo, y eso solo para su definición, pues se debe tomar en cuanta que su implantación dentro de una organización puede llevar otro gran periodo de tiempo.

Así, surge la inquietud de iniciar la definición de una metodología que auxilie en el desarrollo de software de calidad, la cual sea más palpable en el qué hacer y cómo hacerlo, aplicando en esto las recomendaciones de los marcos de referencia para procesos de calidad de software.

Considerando lo anterior, la metodología que se propone esta basada en estándares internacionales de calidad para la creación de software (CMMI, Capability Maturity Model Integrated del Software Engineering Institute de Carnegie Mellon University) y estándares técnicos para la creación de software orientado a objetos (el Proceso Unificado de Desarrollo de Software y el Lenguaje de Unificado de Modelado). De esta manera se cubre la parte técnica que no detalla CMMI y se complementa la parte administrativa del Proceso Unificado de Desarrollo de Software.

1.2. Consideraciones

En la creación de esta metodología se consideró lo siguiente:

- Utilizar UML como lenguaje de modelado por ser el lenguaje de modelado mas ampliamente aceptado en la actualidad. Su sencillez y orígenes dentro de la informática lo hicieron el lenguaje de modelado elegido para este trabajo.

- Basarse en el Proceso Unificado de Desarrollo de Software fue producto de la compatibilidad con respecto a UML, pero sobre todo de la unificación de las antiguas metodologías principales de desarrollo de software orientado a objetos. También fue

considerado por no enfocarse solo a las habilidades de los participantes de un proyecto, sino a los procesos.

- Utilizar CMMI como marco de referencia. Esto es debido a la experiencia profesional que se tiene con este modelo y su compatibilidad con respecto a las normas ISO. Se eligió su modelo continuo por ser más parecido con las versiones anteriores, de las cuales ya se contaba con cierto conocimiento, además se limito solo al campo de la ingeniería de software, excluyendo los modelos de ingeniería de sistemas, productos integrados y desarrollo de procesos, y de proveedores de servicios.

1.3. Objetivos

El objetivo principal original de este trabajo versa así:

"Crear una metodología que sirva como guía a todos los involucrados en el desarrollo de proyectos de software orientado a objetos para la identificación de sus actividades y responsabilidades, para realizarlas de manera ordenada, controlada, registrada, validada y estandarizada, fomentando así el trabajo en equipo."

Además de este objetivo se consideraron los siguientes objetivos secundarios:

- Proporcionar una guía para el inicio, control, seguimiento y cierre de los proyectos, basada en una serie de actividades propuestas que garanticen el mejor camino para desarrollar un Producto de Software.

- Por medio de la metodología propuesta, asegurar un control de la Configuración de Software del proyecto y de sus Líneas Base para mantener una documentación consistente.

- Utilizar el Lenguaje Unificado de Modelado para modelar, documentar e informar el desarrollo del producto de software dentro de la metodología.

- Cumplir con índices de calidad estandarizados. Esta metodología debe cumplir con el estándar CMMI SW Nivel 2.

- Ampliar los conocimientos adquiridos sobre ingeniería de software y administración de proyectos

- Abrir nuevos horizontes para áreas de desarrollo e investigación profesional.

1.4. Organización

El presente trabajo consta de 9 capítulos y un anexo. Los capítulos están divididos en dos partes. La primer parte se refiere al marco teórico de la propuesta y se compone de los capítulos 2,3 y 4; mientras que la segunda parte describe la propuesta en sí y consta de los capítulos 5, 6, 7 y 8.

Este capítulo (Capítulo 1) contiene la introducción al trabajo, los objetivos y la organización del mismo.

El Capítulo 2 tiene como tema central el Lenguaje Unificado de Modelado (UML) explicando sus orígenes y describiendo sus elementos, las relaciones entre estos elementos, los diagramas en los que se combinan las relaciones y elementos, las reglas que rigen a UML y los mecanismos comunes que sigue.

El Capítulo 3 trata sobre el Proceso Unificado de Desarrollo de Software describiendo la historia de éste, sus bases teóricas, sus flujos de trabajo y sus fases.

El Capítulo 4 describe al Modelo de Madurez de Capacidad (Capability Maturity Model - CMM) en sus orígenes y específicamente en su última versión: el Modelo de Madurez de Capacidad Integrado (Capability Maturity Model Integration – CMMI). Describe los diferentes modelos y representaciones de CMMI, centrándose principalmente en los modelos por etapas y continuo para ingeniería de software, describiendo cada uno de sus componentes.

El Capítulo 5 es el primero que trata sobre la propuesta en sí, específicamente respecto a los roles que se utilizaron en la metodología. Inicia haciendo una comparación de los roles mencionados en el Proceso Unificado y en CMMI, para finalmente crear un conjunto de roles único que describe.

El Capítulo 6 trata sobre aspectos relacionados con los artefactos utilizados en la metodología propuesta, listándolos, organizándolos y describiendo su contenido. Para organizar los artefactos, propone una estructura que debe ser implementada en el repositorio de la configuración de software de la organización.

El Capítulo 7 se enfoca a las actividades propuestas, las cuales organiza y clasifica, para posteriormente listarlas y describirlas, indicando los artefactos de entrada y salida, así como los responsables de su ejecución.

El Capítulo 8 describe como se combinan roles, artefactos y actividades dentro de un flujo de trabajo que finalmente se plasma como un diagrama de actividades UML. Describe las diferentes partes del flujo de trabajo y lista una serie de prerrequisitos para su ejecución.

El Capítulo 9 contiene las conclusiones y recomendaciones sobre el trabajo realizado

Finalmente el Anexo A contiene una validación de la metodología propuesta con respecto a las metas y actividades marcadas por CMMI para su segundo nivel de madurez en ingeniería de software.

Primera Parte

BASES TEÓRICAS

2

El Lenguaje Unificado de Modelado

EL presente capítulo describe la base técnica teórica para la construcción de la metodología de desarrollo de software que se propone. Ésta base teórica es la parte que involucra de forma directa al personal técnico en un proyecto de software, es decir, a analistas, diseñadores, programadores, ingenieros de prueba, etc. El Lenguaje Unificado de Modelado fue creado por las figuras más importantes en el área de software orientado a objetos: Ivar Jacobson, Grady Booch y James Rumbaugh, y hoy ya constituye un estándar en el modelado de dicho software.

2.1. Introducción

El Lenguaje Unificado de Modelado (Unified Modeling Language - UML) es, como su nombre lo dice, un lenguaje y nada más que eso. Debe quedar claro que no es un lenguaje de programación (incluso es independiente del lenguaje de programación que se vaya a utilizar), ni pretende guiar el proceso de desarrollo de software, ni marcar prácticas claves para el aseguramiento de calidad de los productos software.

UML es solo una herramienta, un lenguaje, que nos es útil durante el desarrollo de productos de software, sea cual fuera el proceso o metodología que utilicemos para guiar este desarrollo (aunque se acopla muchísimo mas a metodologías iterativas y basadas en objetos como el Proceso Unificado de Desarrollo de Software). Pero, ¿para qué es útil la herramienta UML? Es útil para lo qué marca su nombre, para "modelar", modelar el producto software que se está desarrollando con el fin de comprenderlo mejor[1], permitiendo hacerlo desde diferentes perspectivas de una manera consistente y que permita que este modelado sea transmitido y comprendido por todos los involucrados en el proyecto de desarrollo de software, desde clientes y usuarios finales, hasta programadores y documentadores. Los modelos creados con UML, como en el caso de todos lo modelos, son simplificaciones de la realidad, simplificaciones realizadas a partir de diversos elementos: gráficos, sintácticos, estructurales, y mas. Como se expone en [Booch et al., 1999], UML es un lenguaje para visualizar, especificar, construir y documentar un sistema de software completo o partes de él.

UML es ya un estándar para el modelado de sistemas de software, es un lenguaje que unifica la manera de expresar los modelos dentro de un proyecto de software y en general en la industria del software y que también pretende unificar la experiencia pasada sobre técnicas de modelado. UML no tiene dueño y esta basado en el común acuerdo de gran parte de la comunidad informática, [Rumbaugh et al., 2000]

UML fue creado a la par del Proceso Unificado de Desarrollo de Software por Grady Booch, James Rumbaugh e Ivar Jacobson en Rational Corporation desde 1994. UML cuenta con aportaciones de diversos lenguajes de modelado para sistemas de software, como el lenguaje de modelado del Método Booch, el lenguaje de modelado de OMT, el lenguaje de modelado de Objetory y el Lenguaje de Especificación y Descripción de la CCITT(Specification and Description Language - DSL), entre otros.

Como se menciona en líneas arriba, UML es la herramienta que sirve para plasmar los modelos generados durante el Proceso de Desarrollo de Software, y es por tanto, la parte más básica a considerar en este trabajo. Posteriormente se describe el proceso de desarrollo de software que se consideró (Proceso Unificado de Desarrollo de Software) y las prácticas claves para el desarrollo de software de calidad (CMMI).

[1] El objetivo de UML es aun mucho mas ambicioso que solo limitarse al desarrollo de software, pretende ser un lenguaje de modelado de propósito general para sistemas discretos

2.2. Modelo Conceptual de UML

UML esta formado de Bloques de Construcción, Reglas y Mecanismos Comunes.

2.2.1. Bloques de Construcción de UML

Los Bloques de Construcción de UML se dividen en tres grandes tipos: Elementos, Relaciones y Diagramas. Los Elementos son las abstracciones básicas en un modelo, las Relaciones ligan los elementos entre sí y los Diagramas agrupan colecciones interesantes de elementos [Booch et al., 1999].

La organización de los Bloques de Construcción de UML se muestra en la Figura 2-1 [Jacobson et al., 2000]:

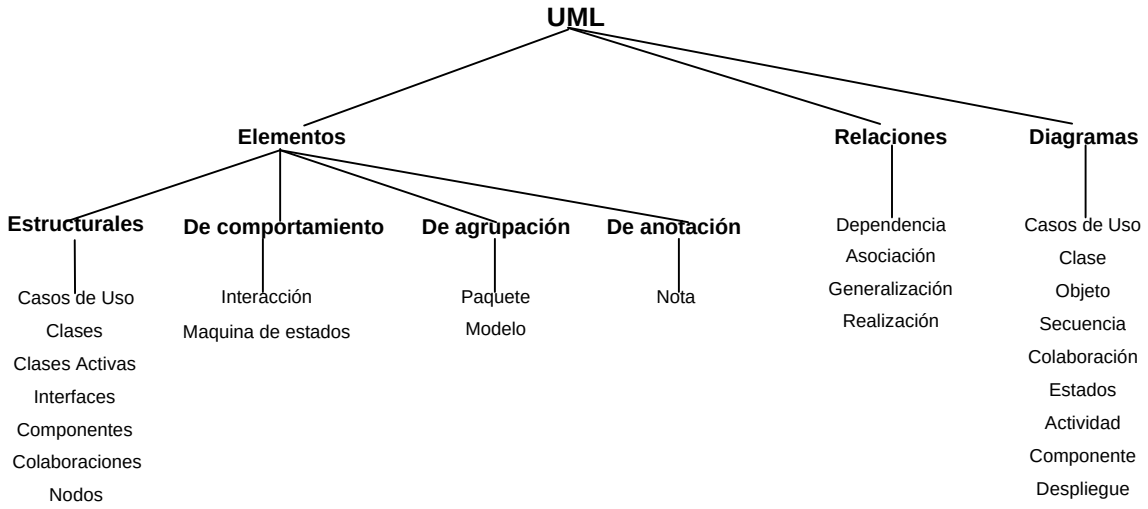


Figura 2-1: Bloques de construcción de UML

- **Elementos en UML:** Existen cuatro tipos de elementos de UML, como se menciona en [Booch et al., 1999]:

    - **Elementos Estructurales**

      En su mayoría son las partes estáticas de un modelo y representan cosas que son conceptuales o materiales. En total hay siete tipos de Elementos Estructurales:

      Casos de Uso
      Son descripciones de un conjunto de secuencias de acciones que un sistema ejecuta y que produce un resultado observable de interés para un actor[1] particular. Se utilizan para estructurar los aspectos de comportamiento de un modelo. Un Caso de Uso es realizado por una Colaboración. Gráficamente se representan como elipses de borde continuo, incluyendo su nombre en el interior.

[1] Un Actor representa un conjunto coherente de roles que los usuarios de los Casos de Uso juegan al interactuar con éstos. Normalmente un Actor representa un rol que es jugado por una persona, un dispositivo hardware o incluso otro sistema al interactuar con nuestro sistema. Aunque se utilizan Actores los modelos, éstos no forman parte del sistema, son externos a él. Gráficamente un actor se representa como un monigote al que se le indica el rol que juega. [Booch et al., 1999]

Figura 2-2: Representación gráfica de un Caso de Uso

Clases
Son descripciones de conjuntos de objetos que comparten los mismos atributos, operaciones, relaciones y semántica. Una Clase implementa una o más Interfaces. Una Clase se representa gráficamente como un rectángulo en el que normalmente se incluye su nombre, atributos y operaciones.

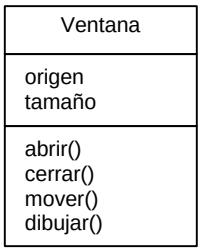


Figura 2-3: Representación gráfica de una Clase

Clases Activas
Son Clases cuyos objetos tienen uno o más procesos o hilos de ejecución y que por lo tanto, pueden dar origen a actividades de control. Una Clase Activa es igual que una Clase, excepto que sus objetos representan elementos cuyo comportamiento es concurrente con otros elementos. Gráficamente se representa como una Clase, pero con la línea de contorno más gruesa, incluyendo normalmente su nombre, atributos y operaciones.

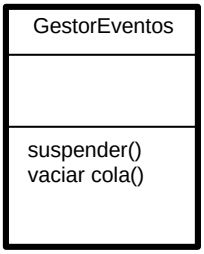


Figura 2-4: Representación gráfica de una Clase Activa

Interfaces
Son colecciones de operaciones que especifican un servicio de una clase o componente, por lo tanto, una Interfaz describe el comportamiento visible externamente de ese elemento. Una Interfaz puede representar el comportamiento completo de una clase o componente, o solo una parte de ese comportamiento. Una Interfaz define un conjunto de especificaciones de operaciones (las declaraciones de las operaciones), pero nunca un conjunto de las implementaciones de esas operaciones. Gráficamente una Interfaz se puede representar de dos formas:

Forma abreviada: Se representa con un circulo con su nombre, rara vez se encuentra aislada, casi siempre se encuentra conectada por medio de una línea a la Clase o Componente que la realiza. Es la representación gráfica mas utilizada.

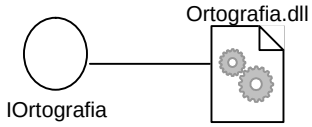


Figura 2-5: Representación gráfica de una Interfaz (forma abreviada)

Forma Canónica: Se utilizan estereotipos de clases (explicados mas adelante) y una línea discontinua dirigida con una flecha vacía (llamada relación de Realización, explicada mas adelante).

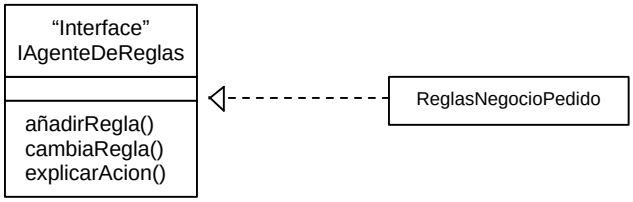


Figura 2-6: Representación gráfica de una Interfaz (forma canónica)

Componentes

Son similares a las clases en cuanto que también describen un conjunto de objetos que comparten los mismos atributos, operaciones, relaciones y semántica, pero los componentes son ya una parte física y reemplazable de un sistema que conforma un conjunto de interfaces y proporciona la implementación de dicho conjunto. En un sistema, se podrán encontrar diferentes tipos de Componentes de despliegue, tales como Componentes COM+ o JavaBeans, así como componentes que sean artefactos[1] del mismo proceso de desarrollo, tales como archivos de código fuente. Un Componente representa típicamente el empaquetamiento físico de diferentes elementos lógicos, como Clases, Interfaces y Colaboraciones. Gráficamente un Componente se representa como un rectángulo con pestañas, incluyendo normalmente solo su nombre.

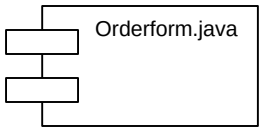


Figura 2-7: Representación gráfica de un Componente

También existen otras cinco formas de representar componentes, por medio de cinco estereotipos de componentes (tratados en la sección de Mecanismos Comunes de UML). Estas cinco representaciones se utilizan solo para representar tipos específicos de componentes:

[1] Pieza de información tangible que (1) es creada, modificada y usada por los trabajadores al realizar actividades; (2) representa un área de responsabilidad, y (3) es candidata a ser tenida en cuenta por el control de la configuración. Un artefacto puede ser un modelo, un elemento de un modelo, o un documento. [Jacobson et al., 2000]. Pieza de información utilizada o producida por un proceso de desarrollo de software, como un documento externo o el producto de un trabajo. Un artefacto puede ser un modelo, una descripción o un software. [Rumbaugh et al., 2000]

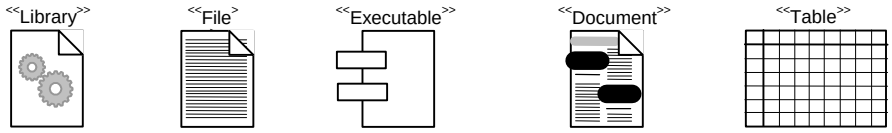


Figura 2-8: Representaciones estereotipadas de Componentes

Colaboraciones
Una Colaboración define una interacción entre otros elementos y/o entre otras colaboraciones que proporcionan un comportamiento cooperativo mayor que la suma de los comportamientos de sus elementos, por lo tanto las Colaboraciones muestran aspectos estructurales y de comportamiento. Una clase dada (o cualquier otro elemento estructural) puede participar en más de una colaboración. Las Colaboraciones representan la implementación de patrones que conforman un sistema. Gráficamente una Colaboración se representa como una elipse de contorno discontinuo que normalmente solo contiene su nombre.

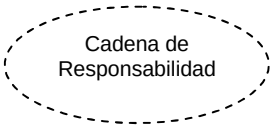


Figura 2-9: Representación gráfica de una Colaboración

Nodos
Un Nodo es un elemento físico que existe en tiempo de ejecución y representa un recurso computacional, que por lo general dispone de algo de memoria y, con frecuencia, capacidad de procesamiento. Un conjunto de componentes puede residir en un solo Nodo y puede también migrar de un Nodo a otro. Gráficamente un componente se representa como un cubo, incluyendo normalmente solo su nombre.

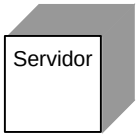


Figura 2-10: Representación gráfica de un Nodo

- **Elementos de Comportamiento**

Los Elementos de Comportamiento son las partes dinámicas de UML. Se puede decir que son los verbos de un modelo y representan comportamiento en el tiempo y el espacio. En total hay dos tipos de Elemento de Comportamiento.

Interacción
Es un comportamiento que comprende un conjunto de mensajes intercambiado entre un conjunto de objetos, dentro de un contexto particular para alcanzar un propósito especifico. El comportamiento de una sociedad de objetos (de una Colaboración) o una operación individual puede especificarse con una Interacción. Una Interacción involucra muchos otros elementos, incluyendo mensajes, secuencias de acción (el comportamiento invocado por un mensaje) y enlaces (conexiones entre objetos). Por lo tanto las Interacciones se utilizan para modelar los aspectos dinámicos de las Colaboraciones. Una Interacción puede modelarse de dos formas: bien destacando la ordenación temporal de los mensajes (representando esto en Diagramas de Secuencia), o bien destacando la secuencia de mensajes en el contexto de una

organización estructural de objetos (representando esto en Diagramas de Colaboración). Pero sea cual fuese la forma de modelar las interacciones, siempre están presentes lo mensajes, los cuales se representan gráficamente como una línea dirigida, incluyendo casi siempre el nombre de su operación.

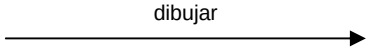


Figura 2-11: Representación gráfica de una Interacción

Maquina de Estados
Es un comportamiento que especifica las secuencias de estados por las que pasa un objeto o una Interacción durante su vida en respuesta a eventos, junto con sus reacciones a estos eventos. Una Maquina de Estados involucra a otros elementos, incluyendo Estados, Transiciones (el flujo de un estado a otro), Eventos (que disparan una Transición) y Actividades (de respuesta a una Transición). Gráficamente un Estado se representa como un rectángulo de esquinas redondeadas, incluyendo normalmente su nombre y sus subestados, si los tiene.

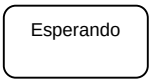


Figura 2-12: Representación gráfica de una Maquina de Estados

- **Elementos de Agrupación**

Son las partes organizativas de los modelos UML. Son las "cajas" en las que se puede descomponer un modelo.

Paquete
Es un mecanismo de propósito general para organizar elementos en grupos. Los elementos estructurales, los elementos de comportamiento e incluso otros elementos de agrupación pueden incluirse en un paquete. Al contrario que los Componentes (que existen en tiempo de ejecución) un Paquete es puramente conceptual (Solo existe en tiempo de desarrollo). Gráficamente un Paquete se visualiza como una carpeta, incluyendo normalmente solo su nombre, y a veces, su contenido.

Figura 2-13: Representación gráfica de un Paquete

Modelo
Un Modelo es un Paquete que abarca una descripción completa de una vista[1] particular de un sistema, incluyendo paquetes. Proporciona una descripción cerrada de un sistema a partir de un punto de vista. No tiene dependencias fuertes con otros paquetes tales como dependencias de implementación o dependencias de herencia. La relación de traza es una forma débil de dependencia entre elementos, en distintos modelos. Generalmente un modelo se estructura en forma de árbol. El paquete raíz

[1] Una vista es la proyección de un modelo que se ve desde una perspectiva o un punto de vista dado, y que omite entidades que no son relevantes desde esa perspectiva. [Booch et al., 1999]

contiene anidados en si mismo paquetes que constituyen el detalle completo del sistema desde un punto de vista dado.

- **Elementos de Anotación.**

  Son las partes explicativas de los modelos UML. Son comentarios que se pueden aplicar para describir, clarificar y hacer observaciones sobre cualquier elemento del modelo.

  Nota
  Una Nota es el elemento principal de anotación y es simplemente un símbolo para mostrar comentarios y restricciones junto a un elemento o una colección de elementos. Típicamente las Notas se utilizan para adornar los diagramas con restricciones o comentarios que se expresan mejor en texto informal o formal. Gráficamente se representa como un rectángulo con una esquina doblada, junto con un comentario textual o gráfico.

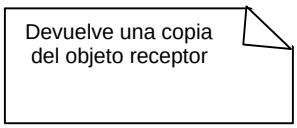


Figura 2-14: Representación gráfica de una Nota

- **Relaciones en UML:** Existen cuatro tipos de relaciones básicas en UML (Dependencias, Asociaciones, Generalizaciones y Realizaciones), además existen variaciones de estos tipos básicos (Refinamiento, Traza, Inclusión y Extensión).

  Las relaciones se utilizan básicamente para mostrar la forma en que se afectan, estructuran, colaboran o componen los diversos elementos del sistema. Al construir redes de relaciones se debe equilibrar las responsabilidades entre los elementos relacionados. Si se abusa demasiado de las relaciones, pueden crearse modelos incomprensibles, pero si se utilizan de manera insuficiente, se puede perder de vista cierto comportamiento del sistema.

  De acuerdo a [Booch et al., 1999] los tipos básicos de relaciones se describen como sigue:

  Dependencias
  Una Dependencia es una relación semántica entre dos elementos, en la cual un cambio a un elemento (el elemento independiente) puede afectar al elemento que lo utiliza (el elemento dependiente), pero no necesariamente a la inversa. La Dependencia se utiliza cuando se quiere representar que un elemento utiliza a otro, y casi siempre dentro del contexto de Clases y Paquetes. Gráficamente una dependencia se representa como una línea discontinua, dirigida del elemento dependiente hacia el elemento independiente, e incluye a veces una etiqueta para indicar el tipo de dependencia y opcionalmente un nombre.

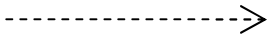


Figura 2-15: Representación gráfica de una Relación de Dependencia

Por ejemplo: en la Figura 2-16 se demuestran dependencias entre clases, donde la Clase A depende de la Clase B y de la Clase C, es decir que los cambios en las Clases B o C afectan el comportamiento de la Clase A, pero no necesariamente a la inversa. Además se indica que a su vez la Clase C depende también de la Clase B. Y se muestra una cuarta dependencia donde una operación de la Clase D depende de la definición de la Clase B

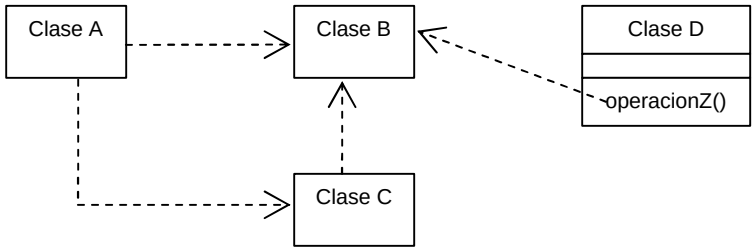


Figura 2-16: Ejemplo de Relaciones de Dependencia entre Clases

Asociación
Es una relación estructural que especifica que los objetos de un elemento están conectados con los objetos de otro. Dada una asociación entre dos Clases, se puede navegar desde un objeto de una Clase hasta un objeto de la otra Clase, y viceversa. Es legal que ambos extremos de la asociación estén conectados a la misma Clase, lo que significa que dado un objeto de la Clase, se puede conectar con otros objetos de la misma Clase. Existe otro tipo de relación llamada Agregación, la cual es un tipo especial de asociación, que representa una relación estructural entre un todo y sus partes. Gráficamente, una Asociación se representa como una línea continua, posiblemente dirigida, que a veces incluye una etiqueta, y a menudo incluye otros adornos, como la multiplicidad y los nombres del rol.

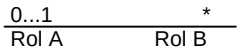


Figura 2-17: Representación gráfica de una Relación de Asociación

Por ejemplo: en la Figura 2-18 se muestra una asociación entre Persona y Empresa, donde Persona juega el rol de Empleado y Empresa juega el rol de Patrón. También se indica que una o más Personas se están relacionadas a cero, una o más Empresas mediante la asociación “TrabajaPara”, con el sentido indicado por la punta de flecha.

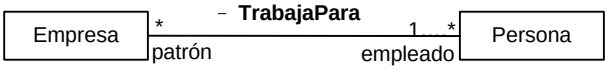


Figura 2-18: Ejemplo de Relación de Asociación entre Clases

Generalización
Es una relación de especialización/generalización en la cual los objetos del elemento especializado (el hijo) pueden sustituir a los objetos del elemento general (el padre). De esta forma el hijo comparte la estructura y el comportamiento del padre. Gráficamente, una relación de Generalización se representa como una línea continua con una punta de flecha vacía apuntando al padre.

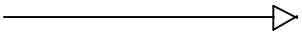


Figura 2-19: Representación gráfica de una Relación de Generalización

Por ejemplo: en la Figura 2-20 muestra una jerarquía de Clases, donde se utiliza la Generalización para representar que la Clase Figura es padre de las Clases Rectángulo,

Circulo y Polígono. A su vez, la Clase Rectángulo es padre de la clase Cuadrado. Así, la Clase Cuadrado, tiene todas las propiedades y comportamiento de la Clase Figura, y también las de la Clase Rectángulo.

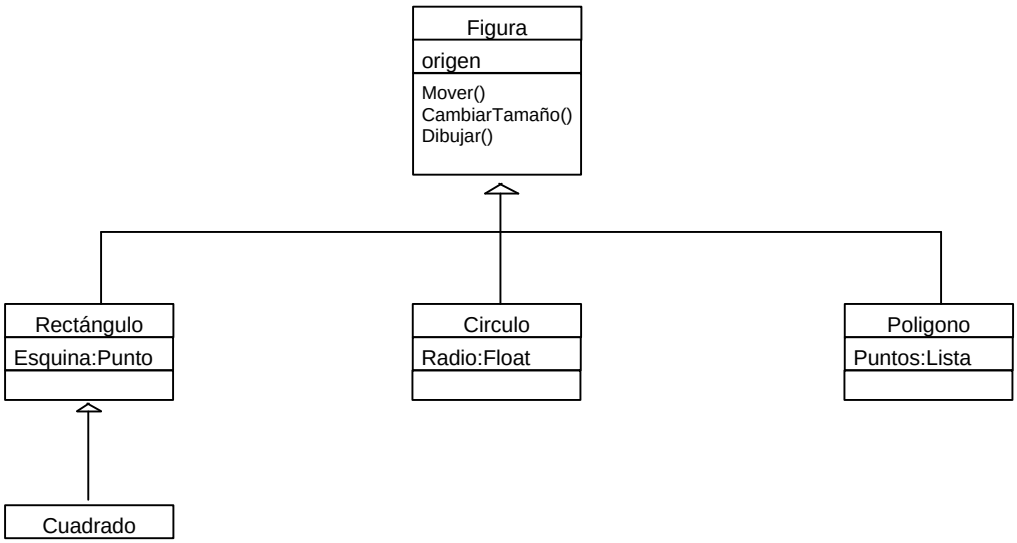


Figura 2-20: Ejemplo de Relaciones de Generalización entre Clases

Realización
Es una relación semántica entre clasificadores, en donde un clasificador especifica un contrato que otro clasificador garantiza que cumplirá, Se pueden encontrar relaciones de Realización en dos sitios: entre Interfaces y las Clases o Componentes que las realizan, y entre los Casos de Uso y las Colaboraciones que los realizan. Gráficamente una relación de Realización se representa como una mezcla entre una Generalización y una relación de Dependencia.

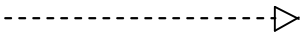


Figura 2-21: Representación gráfica de una Relación de Realización

Por ejemplo: en la Figura 2-22 se muestra una relación de Realización, donde se especifica que la interfaz IAgenteDeReglas es implementada o realizada por la Clase ReglasNegocioPedido.

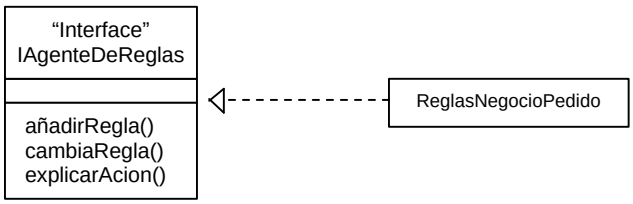


Figura 2-22: Ejemplo de Relación de Realización entre Clase e Interfaz

- **Diagramas en UML:** "Un diagrama es la representación gráfica de un conjunto de elementos, visualizado la mayoría de las veces como un grafo conexo de nodos (elementos) y arcos (relaciones)" [Booch et al., 1999].

  Existen diversos diagramas para sistemas orientados a objetos, cada uno de ellos muestra el sistema en desarrollo o a desarrollar desde una perspectiva o vista en particular, pero en UML se consideran cinco vistas importantes (Vista de Casos de Uso, Vista de Diseño, Vista de Implementación, Vista de Procesos y Vista de Despliegue), para las cuales UML propone nueve diagramas básicos, aunque esto no limita a utilizar solo estos diagramas y se deja libertad de utilizar otros tipos de diagramas.

  Cualquier elemento de un sistema puede aparecer repetido en los diversos diagramas de UML, puede aparecer en un solo diagrama o incluso en ninguno, esto dependerá de la organización de cada sistema. Por lo tanto los nueve diagramas de UML están relacionados entre si, pero deben ser autodefinidos, es decir, se debe comprender un diagrama sin la necesidad de referirse a algún otro diagrama.

  Diagramas de Casos de Uso
  Los Diagramas de Casos de Uso muestran un conjunto de Casos de Uso, Actores (un tipo especial de clases)[1] y sus relaciones. Los Diagramas de Casos de Uso cubren la parte dinámica de la Vista de Casos de Uso de un sistema. Estos diagramas son de suma importancia en el modelado y organización del comportamiento de un sistema. Los Diagramas de Casos de Uso pueden contener Notas y Restricciones (explicadas posteriormente), además de Paquetes, estos últimos se emplean para agrupar elementos del modelo en partes mayores.

  Los diagramas de Casos de Uso se emplean para:

  1. Modelar el contexto de un sistema: implica dibujar una línea alrededor de todo el sistema e identificar qué actores quedan fuera del sistema e interactúan con él. Aquí se emplean los Diagramas de Casos de Uso para especificar los Actores y el significado de sus roles. Por ejemplo, en la Figura 2-23 se muestra el contexto de un sistema de validación de tarjetas de crédito.

[1] Un Actor representa un conjunto coherente de roles que los usuarios de los Casos de Uso juegan al interactuar con éstos. Normalmente un Actor representa un rol que es jugado por una persona, un dispositivo hardware o incluso otro sistema al interactuar con nuestro sistema. Aunque se utilizan Actores los modelos, éstos no forman parte del sistema, son externos a él. Gráficamente un actor se representa como un monigote al que se le indica el rol que juega. [Booch et al., 1999]

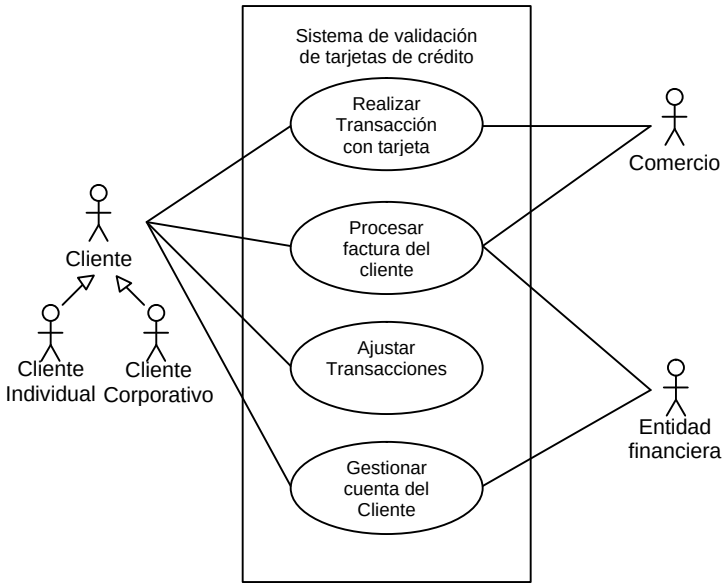


Figura 2-23: Ejemplo de Diagrama de Casos de Uso que muestra el contexto de un sistema

2. Modelar los requisitos del sistema: implica especificar qué debería hacer el sistema (desde un punto de vista externo), independientemente de cómo se haga, es decir se emplean los Diagramas de Casos de Uso para especificar el comportamiento deseado del sistema, mostrando el sistema entero como una caja negra; pudiéndose ver qué hay fuera del sistema y cómo reacciona a los elementos externos, pero no se vera cómo funciona por dentro. Por ejemplo, en la Figura 2-24 se muestran los requisitos para un sistema de validación de tarjetas de crédito.

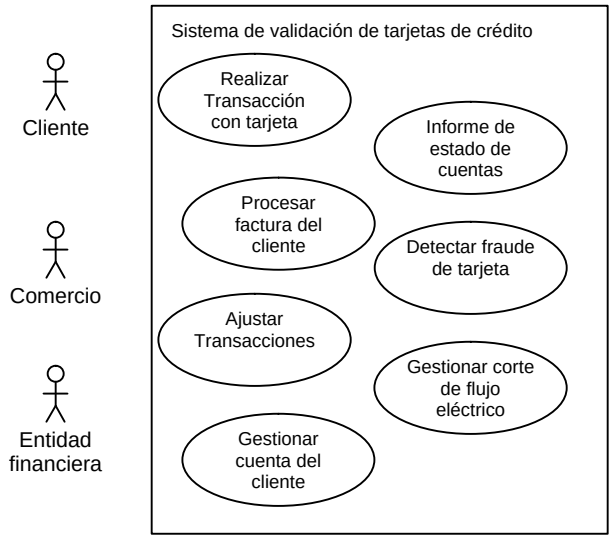


Figura 2-24: Ejemplo de Diagrama de Casos de Uso que modela los requisitos de un sistema

Diagrama de Clases

Muestra un conjunto de Clases, Interfaces y Colaboraciones, así como sus relaciones. Al igual que los demás diagramas, los Diagramas de Clase pueden contener Notas y Restricciones (explicadas posteriormente). Estos diagramas, son los diagramas más comunes en el modelado de sistemas orientados a objetos. Los Diagramas de Clases cubren la parte estática

de la Vista de Diseño de un Sistema. Los Diagramas de Clases sirven de base para otros dos diagramas relacionados: los Diagramas de Componentes y los Diagramas de Despliegue.

Los Diagramas de Clases también pueden contener paquetes o subsistemas, los cuales se usan para agrupar los elementos de un modelo en partes más grandes.

Los Diagramas de Clases se pueden utilizar para:

1. Modelar el vocabulario de un sistema: implica tomar decisiones sobre qué abstracciones son parte del sistema en consideración y cuáles caen fuera de sus límites. Los Diagramas de Clases se utilizan para especificar estas abstracciones y sus responsabilidades.

2. Modelar colaboraciones simples: por ejemplo, cuando se modela la semántica de una transacción en un sistema distribuido, no se puede observar simplemente a una Clase aislada para comprender qué ocurre, en su lugar se tiene un conjunto de Clases que colaboran entre sí. Los Diagramas de Clases se emplean para visualizar y especificar este conjunto de Clases y sus relaciones. Por ejemplo, en la Figura 2-25 se muestra un Diagrama de Clases que representa el modelado de una Colaboración:

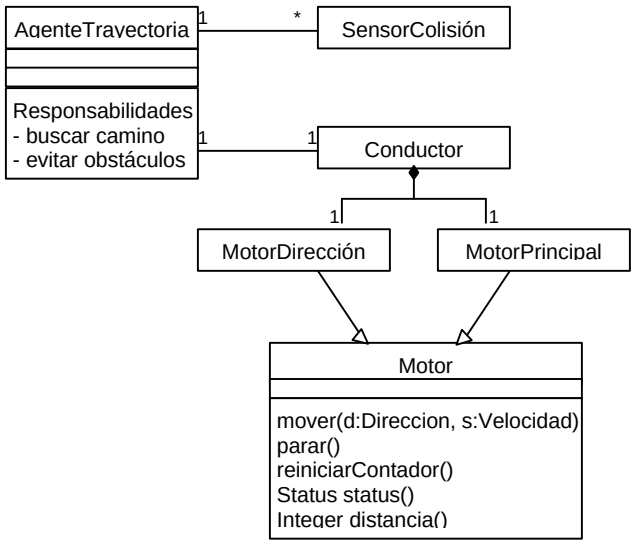


Figura 2-25: Ejemplo de Diagrama de Clases que modela una Colaboración

3. Modelar un esquema lógico de base de datos: En la mayoría de los sistemas de software se deben almacenar datos en forma persistente dentro de una base de datos relacional o en una base de datos orientada a objetos. Se pueden modelar esquemas para estas bases de datos mediante Diagramas de Clases. Por ejemplo, en la Figura 2-26 se muestra el modelado de un esquema lógico de una base de datos de una universidad:

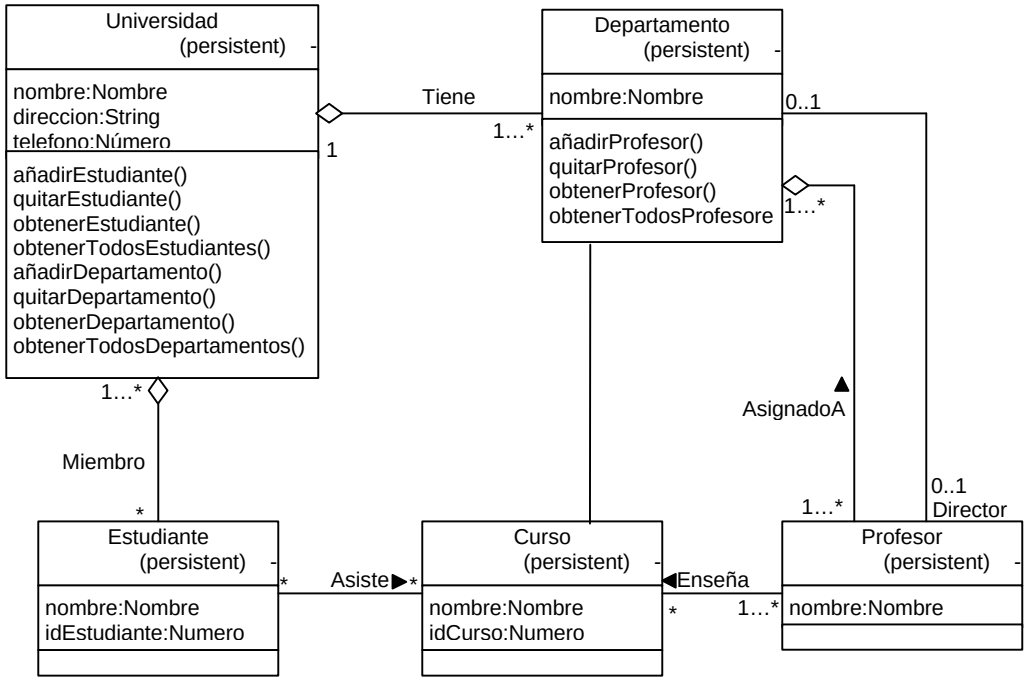


Figura 2-26: Ejemplo de Diagrama de Clases que modela el esquema lógico de una base de datos

En este Diagrama de Clases se esta representando la estructura de una base datos mediante Clases marcadas como persistentes, lo cual indica que las instancias de la Clase se deben almacenar en una base de datos. También se indican los atributos de las Clases de tipos de datos primitivos y sus operaciones de Clase que solo son importantes para mantener la integridad de la base de datos, operaciones que pueden implementarse como procedimientos almacenados.

Diagramas de Objetos

Muestra un conjunto de objetos y sus relaciones. Los Diagramas de Objetos muestran instantáneas de los elementos encontrados en los Diagramas de Clases en un momento concreto. Estos diagramas cubren la parte estática de la Vista de Diseño o la parte estática de la Vista de Procesos de un sistema como lo hacen los Diagramas de Clases, pero desde la vista de casos reales o prototípicos.

Al igual que los demás diagramas, los Diagramas de Objetos pueden contener Notas y Restricciones. Los Diagramas de Objetos también pueden contener Paquetes o Subsistemas, los cuales se usan para agrupar los elementos de un modelo en partes más grandes. A veces se colocaran Clases en los Diagramas de Objetos, especialmente cuando se quieren mostrar las Clase que hay detrás de cada instancia.

Un Diagrama de Objetos es esencialmente una instancia de un Diagrama de Clases o la parte estática de un Diagrama de Interacción. Tanto los Diagramas de Componentes como los Diagramas de Despliegue pueden contener instancias, y si solo contienen instancias también ellos se consideran un tipo especial de Diagramas de Objetos.

Los Diagramas de Objetos se utilizan normalmente para:

1. Modelar estructuras de objetos: implica tomar una instantánea de los objetos de un sistema en un cierto momento. Por ejemplo, en la Figura 2-27 se muestra la estructura de los objetos de una implementación de un robot autónomo.

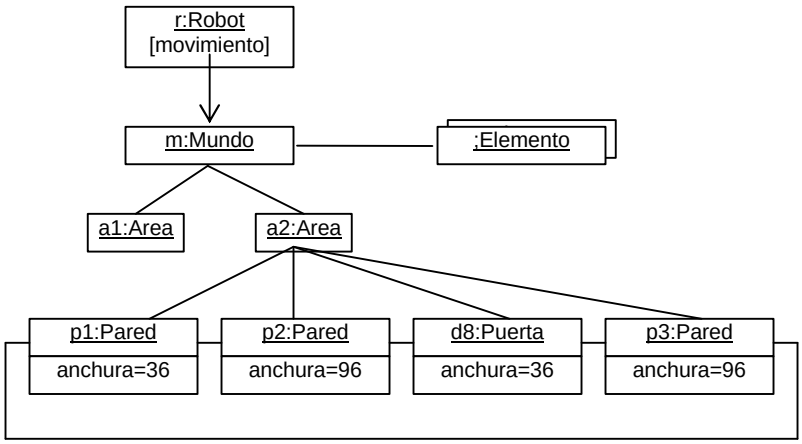


Figura 2-27: Ejemplo de Diagrama de Objetos

Diagramas de Interacción

Un Diagrama de Interacción muestra una Interacción que consta de un conjunto de Objetos y sus relaciones, incluyendo los mensajes que pueden ser enviados entre ellos. Los diagramas de Interacción cubren la parte dinámica de un sistema. Existen dos tipos de Diagramas de Interacción: Los Diagramas de Colaboración y los Diagramas de Secuencia

Diagramas de Colaboración

Un Diagrama de Colaboración es un Diagrama de Interacción que resalta la organización estructural de los objetos que envían y reciben mensajes, y muestra un conjunto de objetos, enlaces entre esos objetos y mensajes enviados y recibidos por esos objetos. Los Diagramas de Colaboración cubren la parte dinámica (o de comportamiento) de un sistema. Un Diagrama de Colaboración de un sistema dado, es semánticamente equivalente a un Diagrama de Secuencia del mismo sistema, es decir un Diagrama de Colaboración se puede convertir en un Diagrama de Secuencia sin perdida de información.

Los Diagramas de Colaboración se utilizan normalmente para:

1. Modelar flujos de control por organización: implica mostrar las relaciones estructurales entre las instancias de la Interacción representada en el diagrama. Por ejemplo, la Figura 2-28 muestra la organización estructural entre objetos de un flujo de control que sirve para matricular un nuevo estudiante en una universidad.

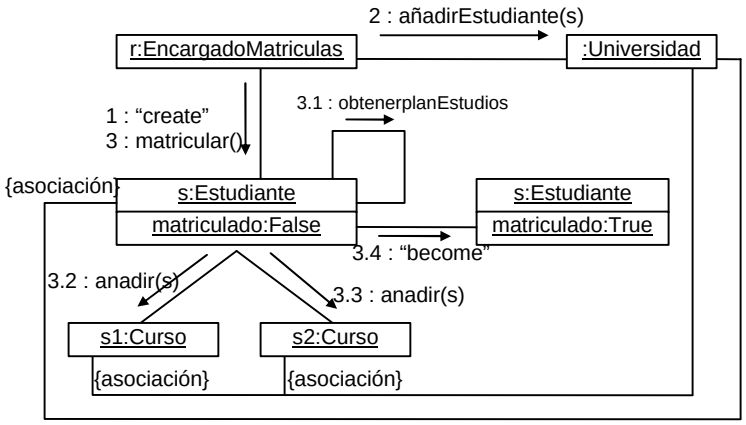


Figura 2-28: Ejemplo de Diagrama de Colaboración

Diagramas de Secuencia

Un Diagrama de Secuencia es un Diagrama de Interacción que resalta la ordenación temporal de los mensajes en un sistema, y muestra un conjunto de objetos y los mensajes enviados y recibidos por ellos. Los objetos suelen ser instancias con nombre o anónimas de Clases, pero también pueden representar instancias de otros elementos, tales como Colaboraciones, Componentes y Nodos. Los Diagramas de Secuencia cubren la parte dinámica (o de comportamiento) de un sistema. Un Diagrama de Secuencia de un sistema dado, es semánticamente equivalente a un Diagrama de Colaboración del mismo sistema, es decir, un Diagrama de Secuencia se puede convertir en un Diagrama de Colaboración si perdida de información.

Un Diagrama de Interacción se forma colocando en primer lugar los objetos que participan en la interacción en la parte superior del diagrama, a lo largo del eje X. Normalmente se coloca a la izquierda el objeto que inicia la interacción, y los objetos subordinados a la derecha. A continuación se colocan los mensajes que estos objetos envían y reciben a lo largo del eje Y, en orden de sucesión en el tiempo, desde arriba hasta abajo. Pueden crearse objetos durante la interacción. Sus líneas de vida comienzan con la recepción del mensaje estereotipado como "create". Los objetos también pueden destruirse durante la interacción. Sus líneas de vida acaban con la recepción del mensaje estereotipado como "destroy" (además también se muestra una señal visual de una gran "X" que marca el final de sus vidas)

Los Diagramas de Secuencia se utilizan normalmente para:

1. Modelar flujos de control por ordenación temporal: implica destacar el paso de mensajes tal y como se desarrolla a lo largo del tiempo. Los Diagramas de Secuencia, a diferencia de los Diagramas de Colaboración, pueden mostrar la dinámica de creación y destrucción de objetos de un sistema a lo largo del tiempo.

   Por ejemplo, la Figura 2-29 presenta un Diagrama de Secuencia que especifica el flujo de control para realizar transacciones a través de un ProxyODBC desde un Cliente.

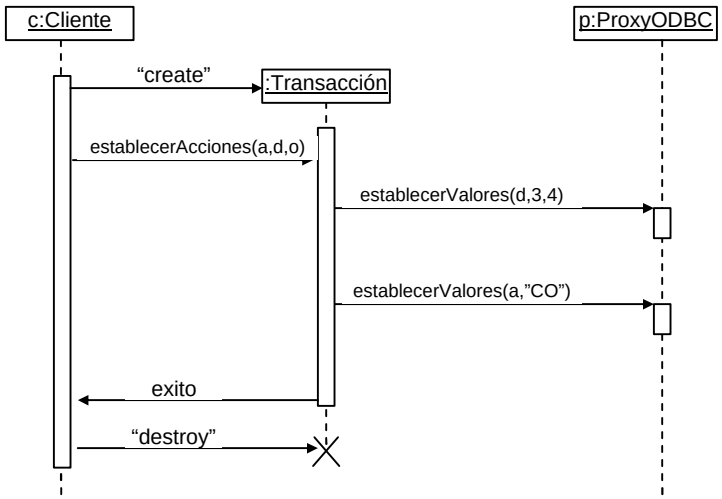


Figura 2-29: Ejemplo de Diagrama de Secuencia

Diagramas de Estados

Un Diagrama de Estados muestra una máquina de estados. Las *máquinas de estados* es un comportamiento que especifica las secuencias de estados por las que pasa una Interfaz, una Clase, un Caso de Uso o una Colaboración a lo largo de su vida en respuesta a eventos, lo cual es especialmente útil en el modelado de sistemas u objetos reactivos. Un *estado* es una situación en la vida de un objeto, durante la cual satisface alguna condición, realiza alguna actividad o espera algún evento. Un *evento* es la especificación de un acontecimiento significativo que ocupa un lugar en el tiempo y en el espacio. En el contexto de las maquinas de estados, un evento es la aparición de un estimulo que puede activar una transición de estado. Una *transición* es una relación entre dos estados que indica que un objeto que esté en el primer estado realizará ciertas acciones y entrará en el segundo estado cuando ocurra un evento especificado y se satisfagan unas condiciones especificadas. Una *actividad* es una ejecución no atómica en curso, dentro de una maquina de estados, Una *acción* es una computación atómica ejecutable que produce un cambio en el estado del modelo o la devolución de un valor

Los Diagramas de Estados cubren la parte dinámica (o de comportamiento) de un sistema y normalmente se utilizan para:

1. Modelar objetos reactivos: un objeto reactivo (o dirigido por eventos) es aquél para el que la mejor forma de caracterizar su comportamiento es señalar cual es su respuesta a los eventos lanzados desde fuera de su contexto. Por ejemplo, en la Figura 2-30 se muestra un Diagrama de Estados que muestra el comportamiento de una maquina de fax.

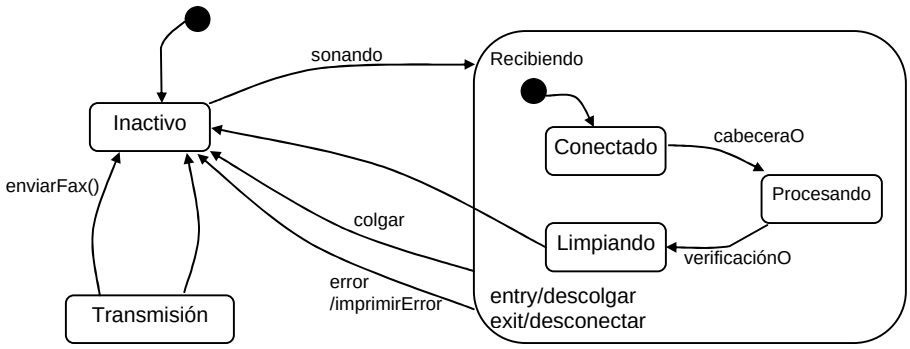


Figura 2-30: Ejemplo de Diagrama de Estados

Diagramas de Actividades

Un Diagrama de Actividades es un tipo especial de Diagrama de Estados que muestra el flujo de actividades (posiblemente concurrentes) dentro de un sistema. Los Diagramas de Estados cubren la parte dinámica (o de comportamiento) de un sistema. Son especialmente importantes al modelar el funcionamiento de un sistema, resaltando el flujo de control entre objetos.

Los Diagramas de Actividades de utilizan normalmente para:

1. Modelar un flujo de trabajo: Para ello se hace hincapié en las actividades, tal y como son vistas por los Actores que colaboran con el sistema. A menudo, en el entorno de sistemas con gran cantidad de software, existen flujos de trabajo y se utilizan para visualizar, especificar, construir y documentar procesos de negocio que implican al sistema que se está desarrollando. En este uso de los Diagramas de Actividades, es particularmente importante el modelado de los flujos de objetos.

   Por ejemplo, la Figura 2-31 muestra un Diagrama de Actividades de un negocio de venta, que especifica el flujo de control que se desarrolla cuando un cliente devuelve un artículo de un pedido por correo. En el diagrama se muestra la interacción de cuatro Actores (Cliente, Televentas, Contabilidad y Almacén), listados en la parte superior de forma horizontal donde a cada uno de ellos se le asigno un carril o calle vertical, donde se encuentran las actividades correspondientes a cada uno de ellos. También se muestra el paso de una instancia (i) del objeto “Articulo”, con su estado correspondiente entre corchetes.

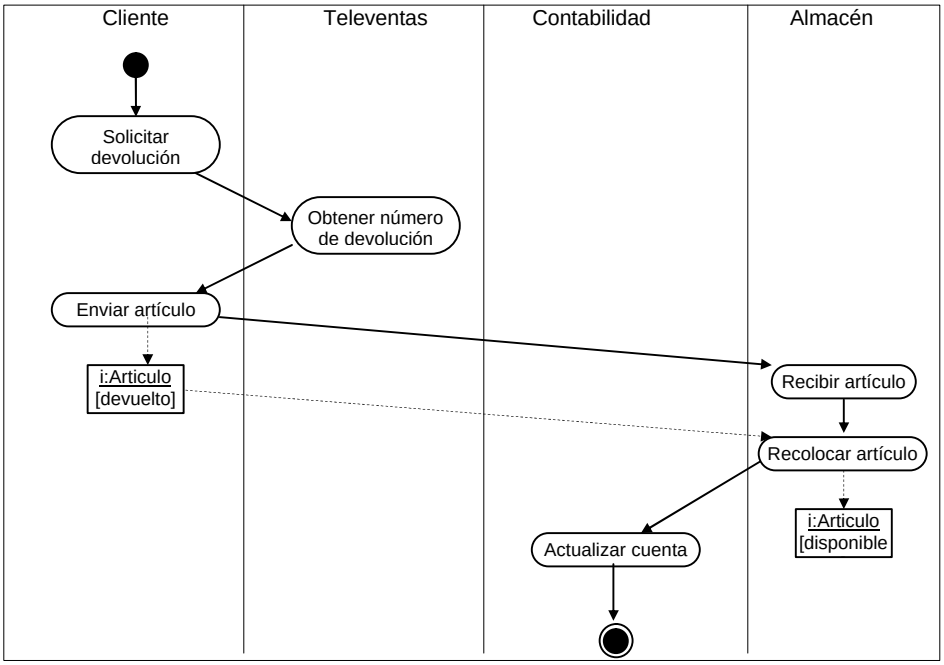


Figura 2-31: Ejemplo de Diagrama de Actividades que modela un flujo de trabajo

2. Modelar una operación: Para ello se utilizan los Diagramas de Actividades como Diagramas de Flujo, para modela los detalles de una computación. En este uso de los Diagramas de Actividades, es particularmente importante el modelado de bifurcaciones, divisiones y uniones del flujo de actividades. El contexto de un diagrama de Actividades utilizado para esta finalidad incluye los parámetros de la operación, así como sus objetos locales. Por ejemplo, en el contexto de la Clase "línea", la Figura 2-32 muestra el Diagrama de Actividades que especifica el algoritmo de la operación "intersección", cuya declaración incluye un parámetro ("l", un parámetro de la clase "línea") y un valor de retorno (de la clase punto).

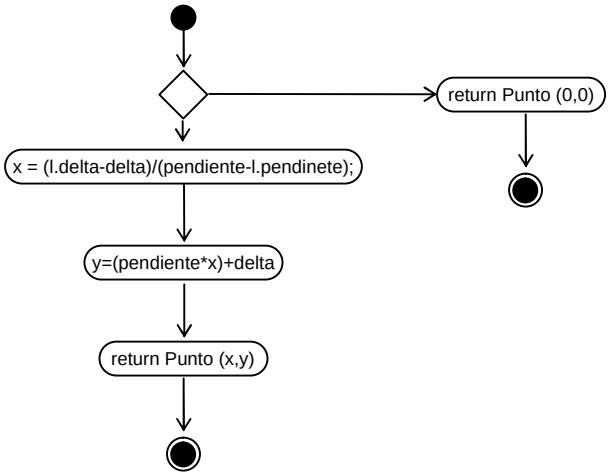


Figura 2-32: Ejemplo de Diagrama de Actividades que modela una operación

Diagrama de Componentes

Un Diagrama de Componentes muestra la organización y la dependencia entre un conjunto de Componentes. Los Diagramas de Componentes muestran la parte estática de la Vista de Implementación de un sistema. Esto implica modelar las cosas físicas que residen en un Nodo, tales como ejecutables, bibliotecas, tablas, archivos y documentos.

Se relacionan con los Diagramas de Clases, en el sentido que un Componente se corresponde por lo común, con una o más Clases, Interfaces o Colaboraciones. Los Diagramas de Componentes también pueden contener Paquetes o Subsistemas, los cuales se utilizan para agrupar elementos del modelo en bloques mayores.

Normalmente se utilizan los Diagramas de Componentes para:

1. Modelar código fuente: Con la mayoría de los lenguajes orientados a objetos actuales el código se produce utilizando entornos integrados de desarrollo, que almacenan el código fuente en archivos. Los Diagramas de Componentes se pueden utilizar para modelar la gestión de configuraciones de estos archivos, los cuales representan los componentes obtenidos como productos de trabajo. Por ejemplo, la Figura 2-33 muestran cinco archivos de código fuente, donde “signal.h” es un archivo de cabecera, se muestran tres de sus versiones, que se remontan desde las nuevas versiones hacia atrás hasta sus antecedentes mas antiguos. Cada variante de este archivo de código fuente se representa con un Valor Etiquetado (detallados mas adelante) que muestra su número de versión. Este archivo de cabecera (signal.h) se utiliza por otros dos archivos (Interp.cpp) y (signal.cpp), los cuales son cuerpos. Uno de esos archivos (Interp.cpp) tiene una dependencia de compilación con otra cabecera (irq.h); a su vez, device.cpp tiene una dependencia de compilación con Interp.cpp. Dado este Diagrama de Componentes, es fácil seguir el rastro del impacto de los cambios.

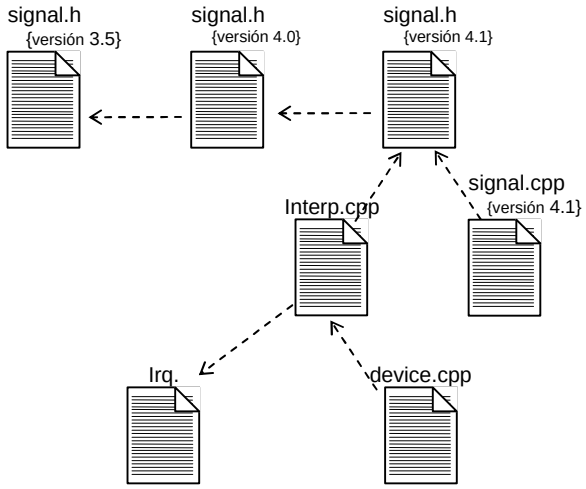


Figura 2-33: Ejemplo de Diagrama de Componentes que modela código fuente

2. Modelar versiones ejecutables: Una versión es un conjunto de artefactos relativamente consistente y completo que se entrega a un usuario externo o interno. En el contexto de los Componentes, una versión se centra en las partes necesarias para entregar un sistema en ejecución. Cuando se modela una versión por medio de Diagramas de Componentes, se está visualizando, especificando y documentando las decisiones acerca de las partes físicas que constituyen el software (es decir sus Componentes de despliegue). Por

ejemplo, en la Figura 2-34 se modela parte de la versión ejecutable de un robot autónomo. El diagrama solo se centra en los componentes de despliegue asociados con las funciones de movimiento y calculo del robot. Se puede ver un componente (motor.dll) que exporta una interfaz (IMotor) que a su vez es importada por otro componente (trayectoria.dll). motor.dll exporta otra interfaz (IAutoTest) que será utilizada probablemente por otros componentes del sistema. También se muestra que trayectoria.dll tiene una dependencia directa con colision.dll.

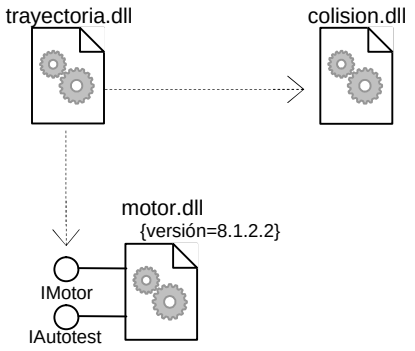


Figura 2-34: Ejemplo de Diagrama de Componentes que modela versiones ejecutables

3. Modelar bases de datos físicas: Una base de datos física puede ser vista como la realización concreta de un esquema, y que pertenece al mundo de los bits. Los Diagramas de Componentes se utilizan para representar las bases de datos físicas. Por ejemplo, la Figura 2-35 muestra un conjunto de tablas de una base de datos, extraídas de un sistema de información de una universidad.

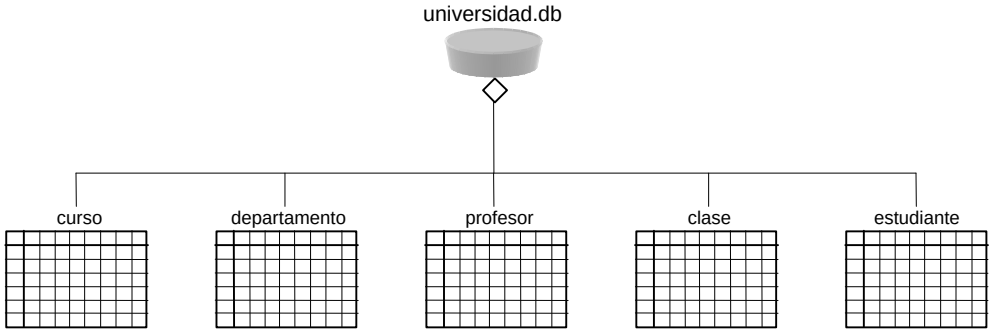


Figura 2-35: Ejemplo de Diagrama de Componentes que modela bases de datos físicas

4. Modelar sistemas adaptables: Algunos sistemas son bastante estáticos, sus componentes entran en escena, participan en la ejecución y desaparecen. Otros sistemas son más dinámicos e implican agentes móviles o componentes que migran con el propósito de equilibrar la carga o la recuperación de fallos. Los Diagramas de Componentes se utilizan, junto a algunos de los diagramas de UML, para modelar el comportamiento de este tipo de sistemas. Por ejemplo, en la Figura 2-36 se modela la replicación de la base de datos mostrada en el ejemplo anterior. Se muestran dos instancias del componente “universidad.db”. Ambas instancias son anónimas, y ambas tienen valores diferentes en su Valor Etiquetado de localización. También hay una nota que especifica explícitamente que instancia es una replica de otra.

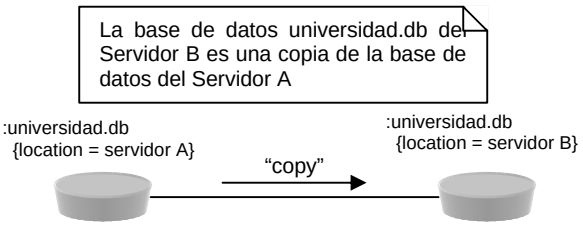


Figura 2-36: Ejemplo de Diagrama de Componentes que modela sistemas adaptables

Diagramas de Despliegue

Un Diagrama de Despliegue muestra la configuración de Nodos de procesamiento en tiempo de ejecución y los componentes que residen en ellos. Los Diagramas de Despliegue cubren la parte estática de la Vista de Despliegue. Se relacionan con los Diagramas de Componentes en que un Nodo incluye, por lo común, uno o más Componentes.

Los Diagramas de Despliegue también pueden contener Componentes, cada uno de los cuales debe residir en algún Nodo. También pueden contener Paquetes o Subsistemas, los cuales se utilizan para agrupar elementos del modelo en bloques más grandes. A veces también se colocaran instancias en los Diagramas de Despliegue, especialmente cuando se quiera visualizar una instancia de una familia de topologías hardware.

Normalmente se utilizan los Diagramas de Despliegue para:

1. Modelar sistemas empotrados: Un sistema empotrado es una colección de hardware con una gran cantidad de software que interactúa con el mundo físico. Los sistemas empotrados involucran software que controla dispositivos como motores, actuadores, pantallas, y que a su vez, están controlados por estímulos externos tales como entradas de sensores, movimientos y cambios de temperatura. Los Diagramas de Componentes se pueden utilizar para modelar los dispositivos y los procesadores que comprenden un sistema empotrado. Por ejemplo, en la Figura 2-37 se muestra el hardware de un simple robot autónomo, se puede ver un nodo (placa base pentium) estereotipado como un procesador. Rodeando a este nodo hay ocho dispositivos, cada uno estereotipado como un dispositivo y representado como un icono que ofrece una señal visual clara de su equivalente en el mundo real.

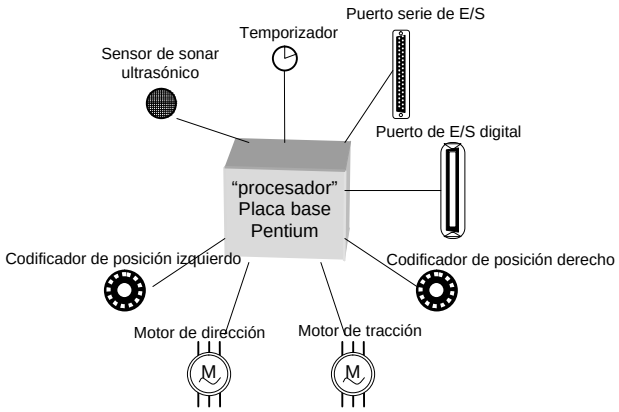


Figura 2-37: Ejemplo de Diagrama de Despliegue que modela un sistema empotrado

2. Modelar sistemas cliente/servidor: un sistema cliente/servidor es una arquitectura muy extendida que se basa en hacer una clara separación de intereses entre la interfaz de usuario del sistema (que reside en el cliente) y los datos persistentes del sistema (que residen en el servidor). Los sistemas cliente/servidor son un extremo del espectro de los sistemas distribuidos y requieren tomar decisiones sobre la conectividad de red de los clientes a los servidores y sobre la distribución física de los componentes software del sistema a través de los nodos. La topología de tales sistemas se pueden modelar mediante Diagramas de Despliegue. Por ejemplo, en la Figura 2-38 se muestra la topología de un sistema de recursos humanos, que sigue una arquitectura cliente/servidor. Este diagrama describe explícitamente la división cliente servidor mediante los paquetes denominados "clientes" y "servidores". El paquete "clientes" contiene dos nodos ("consola" y "terminal"), ambos estereotipados y distinguibles visualmente. El paquete "servidores" contiene dos tipos de nodos ("servidor de caché" y "servidor"), y ambos han sido adornados con algunos de los componentes que residen en ellos. También puede notarse que "servidor de caché" y "servidor" han sido marcados con multiplicidades explicitas, que especifican cuantas instancias de cada uno se esperan en una configuración de despliegue en particular.

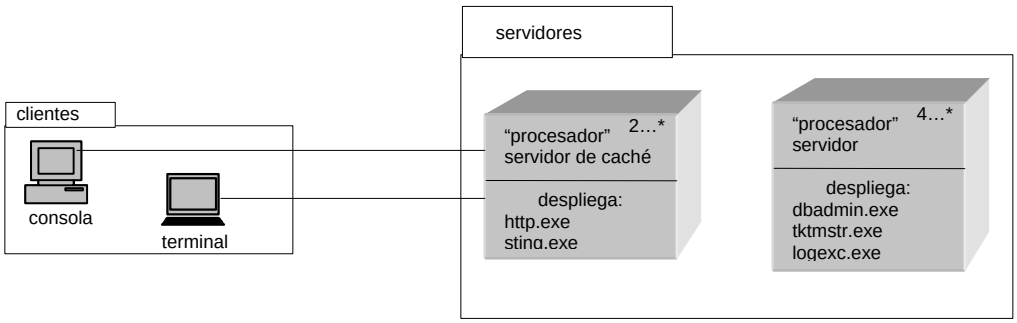


Figura 2-38: Ejemplo de Diagrama de Despliegue que modela un sistema cliente servidor

3. Modelar sistemas completamente distribuidos: En el otro extremo del espectro de los sistemas distribuidos se encuentran aquellos que son ampliamente, si no totalmente, distribuidos y que, normalmente, incluyen varios niveles de servidores. Tales sistemas contienen a menudo varias versiones de los componentes software, algunos de los cuales pueden incluso migrar de nodo en nodo. El diseño de tales sistemas requiere tomar decisiones que permitan un cambio continuo de la topología del sistema. Los Diagramas de Despliegue se pueden utilizar para visualizar la topología actual del sistema y la distribución de Componentes, para razonar sobre el impacto de los cambios de esa topología. Por ejemplo, en la Figura 2-39 se muestra mediante un Diagrama de Despliegue (el cual es también un Diagrama de Objetos, pues solo contiene instancias) la configuración de un sistema completamente distribuido. Se pueden ver tres consolas (instancias anónimas de Nodo estereotipado "consola"), las cuales están conectadas a Internet (el cual es un Nodo único y estereotipado). A su vez, hay tres instancias de "servidores regionales", las cuales sirven como intermediarios para el acceso a los "servidores nacionales", de los cuales solo se muestra uno. Como indica la nota, los "servidores nacionales" están conectados entre sí, pero sus relaciones no se muestran en el diagrama.

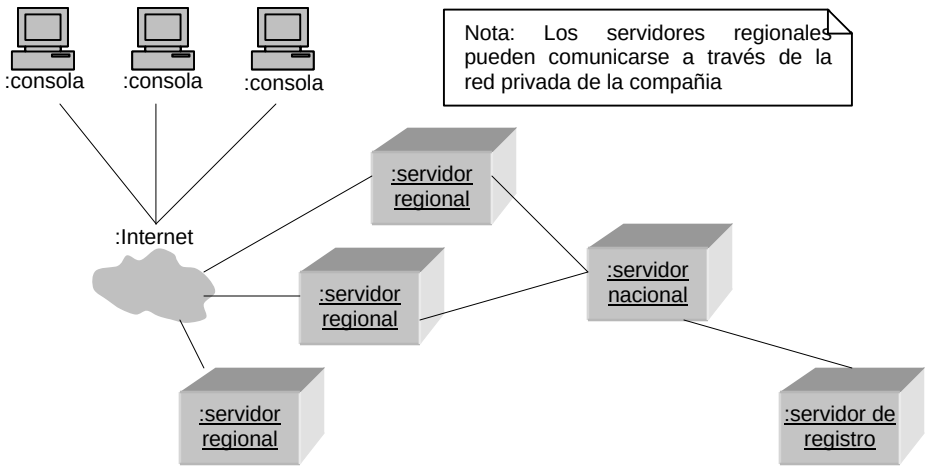


Figura 2-39: Ejemplo de Diagrama de Despliegue que modela un sistema completamente distribuido

Todos los diagramas mostrados líneas arriba se resumen en la Tabla 2-1.

DIAGRAMAS ESTRUCTURALES	
Nombre del Diagrama	Contenido
1. Diagramas de Clases	es. Interfaces y Colaboraciones
2. Diagramas de Objetos	tos o instancias
3. Diagramas de Componentes	ponentes
4. Diagramas de Despliegue	os
DIAGRAMAS DE COMPORTAMIENTO	
Nombre del Diagrama	Propósito o enfoque
1. Diagramas de Casos de Uso	niza el comportamiento del sistema
2. Diagramas de Secuencia	rados en la organización temporal de los mensajes
3. Diagramas de Colaboración	rados en la organización estructural de los objetos que envían y reciben mensajes
4. Diagramas de Estados	rados en el estado cambiante de un sistema dirigido por eventos
5. Diagramas de Actividades	rados en el flujo de control de actividades

Tabla 2-1: Diagramas de UML

La Tabla 2-2 muestra las relaciones Vista-Diagramas para sistemas de diferentes dimensiones.

Vista	aplicaciones monolíticas en un solo Nodo	Para sistemas complejos y distribuidos
Vista de Casos de Uso	• Diagramas de Casos de Uso	• Diagramas de Casos de Uso • Diagramas de Actividades (para modelado del comportamiento)
Vista de Diseño	• Diagramas de Clases (para modelado estructural) • Diagramas de Interacción (para modelado del comportamiento)	• Diagramas de Clases (para modelado estructural) • Diagramas de Interacción (para modelado del comportamiento) • Diagramas de Estados (para modelado del comportamiento)
Vista de Procesos	No se requiere	• Diagramas de Clase (para modelado estructural) • Diagramas de Interacción (para modelado del comportamiento)
Vista de Implementación	No se requiere	• Diagramas de Componentes
Vista de Despliegue	No se requiere	• Diagramas de Despliegue

Tabla 2-2: Diagramas y Listas

2.2.2. Reglas de UML

Al igual que cualquier lenguaje, UML define un conjunto de reglas para que los modelos sean semánticamente autoconsistentes y en armonía con otros modelos relacionados, es decir, para que sean modelos bien formados.

UML tiene cinco tipos de reglas semánticas [Booch et al., 1999]:

Reglas:	Especifican:
Reglas de Nombres	Cómo llamar a los elementos, relaciones y diagramas.
Reglas de Alcance	El contexto que da un significado especifico a un nombre.
Reglas de Visibilidad	Cómo se pueden ver y utilizar esos nombres por otros.
Reglas de Integridad	Cómo se relacionan apropiada y consistentemente unos elementos con otros.
Reglas de Ejecución	Qué significa ejecutar o simular un modelo dinámico.

Tabla 2-3: Reglas de UML

Estas Reglas de UML, especifican y fomentan la construcción de modelos bien formados, pero durante la creación de un sistema, es posible que se lleguen a construir, e incluso manejar y utilizar, modelos que no sean bien formados, pero que aun están en elaboración para llegar a serlo. Por lo tanto, las Reglas de UML no obligan a construir, manejar y utilizar modelos bien formados.

2.2.3. Mecanismos Comunes en UML

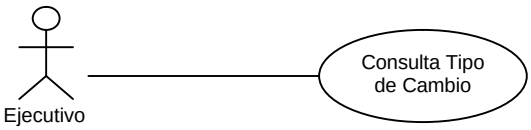
Los Mecanismos Comunes en UML son patrones de características comunes que se aplican de forma consistente a través de todo el lenguaje. Existen cuatro Mecanismos Comunes en UML:

- Especificaciones
- Adornos
- Divisiones Comunes
- Mecanismos de Extensión

- **Especificaciones**

  Este mecanismo se refiere a que detrás de la notación grafica de los elemento de UML existe una especificación textual de la sintaxis y semántica de ese bloque de construcción.

  Por ejemplo, la Figura 2-40 muestra la representación gráfica de un Caso de Uso llamado “Consulta tipo de Cambio”, pero para enunciar los detalles de este Caso de Uso se muestra a continuación su Especificación.



Caso de uso: Consulta de Tipo de Cambio.

Descripción.
Este requerimiento refleja los tipos de cambio de divisas al día. Para este requerimiento, la estructura de las pantallas de salida o respuesta de los diferentes tipos de indicadores arrojan los mismos datos y el indicador "B" (billete) es el más utilizado para la realización de las consultas.

Precondiciones.
- Deben existir los tipos de indicadores para que el ejecutivo pueda realizar la selección.

Flujo de sucesos.
1. El sistema muestra la pantalla de Consulta del tipo de Cambio.
2. El sistema solicita la fecha sobre la que requieren el tipo de cambio, con valor del día por omisión.
3. El ejecutivo tiene la opción de capturar la fecha sobre la que se desea efectuar la consulta.
4. El ejecutivo selecciona el indicador o tipo de moneda, teniendo una "B" como valor por omisión.
5. El ejecutivo captura la entidad.
6. El sistema ejecuta el requerimiento TIPO_DE_CAMBIO cuando el ejecutivo selecciona la opción "Aceptar".
7. El sistema registra en el Diario Electrónico el servicio realizado.
8. El sistema despliega la respuesta donde se muestran los siguientes elementos:
   Un encabezado con los siguientes datos:
   - Fecha Cambio.
   - Zona Cambiaría.
   - Factor Compra.
   - Factor Venta.

   Una tabla que contenga los siguientes campos:
   - Divisa.
   - Cotiza.
   - Comprador.
   - Vendedor.
9. El ejecutivo selecciona la opción "Salir" y el sistema transfiere el control al menú de servicios.

Excepciones.
- 2.a Si la fecha de la consulta es inválida, el sistema utilizará la fecha actual para realizar la operación. Se considera fecha inválida si esta en blanco o si tiene un formato diferente al del sistema o si es posterior a la fecha de la consulta.
- 5.a Si la entidad es inválida, el sistema despliega un mensaje de error y el ejecutivo debe repetir el paso 5.
- 6.a En caso de no llevarse a cabo el requerimiento, el sistema despliega un mensaje de error y comienza de nuevo el caso de uso.

Postcondiciones.
- La operación termina exitosamente ejecutando la Consulta de Tipo de Cambio, o cancelando la consulta.

Figura 2-40: Ejemplo de Especificación para un Caso de Uso

De esta manera es posible construir un modelo de forma incremental, creando primero sus representaciones gráficas y posteriormente agregando semántica por medio de sus Especificaciones.

- **Adornos**

En UML la mayoría de sus elementos tienen una clara y única notación gráfica, que muestra solamente sus aspectos más importantes, pero la especificación de un elemento puede incluir detalles y otros aspectos, muchos de estos detalles se pueden incluir en la notación gráfica como adornos gráficos o textuales. Todos los elementos en la notación UML comienzan con un símbolo básico, al cual puede añadirse una variedad de adornos específicos de ese símbolo.

Por ejemplo, La notación de una clase es simple y revela los aspectos mas importantes de la misma (nombre, atributos y operaciones), pero su especificación puede incluir otros detalles, tales como si es abstracta o la visibilidad de sus atributos y operaciones, en la representación gráfica de la Figura 2-41 se incluyen esos atributos a manera de adornos ("+" para operaciones publicas, "#" para operaciones protegidas y "-" para operaciones privadas).

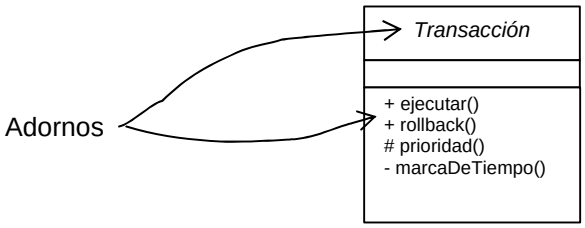


Figura 2-41: Ejemplo de adornos para una clase

- **Divisiones Comunes**

  Son dos las Divisiones Comunes en UML:

  División entre Clase y Objeto: Una Clase es una abstracción del mundo real, un objeto es una manifestación concreta de esa abstracción. Casi todos los bloques de construcción de UML (p.e. Clase, Casos de Uso, Componentes, Nodos, etc.) presentan este tipo de dicotomía Clase/Objeto. Gráficamente, UML distingue un objeto utilizando el mismo símbolo de la Clase y subrayando el nombre del objeto.

  División entre Interfaz e Implementación: Una Interfaz declara un contrato, y una Implementación representa una realización concreta de ese contrato, responsable de hacer efectiva de forma fidedigna la semántica completa de la interfaz, en UML se pueden modelar las interfaces y sus implementaciones. Casi todos los bloques de construcción de UML presentan este tipo de dicotomía Interfaz/Implementación, por ejemplo Casos de Uso/Colaboraciones y operaciones/métodos.

- **Mecanismos de Extensión**

  Los Mecanismos de Extensión de UML permiten extender el lenguaje de forma controlada y se utilizan para adaptar UML a las necesidades específicas de un dominio y una cultura de desarrollo. Existen tres Mecanismos de Extensión:

  Estereotipos
  "Un Estereotipo extiende el vocabulario de UML, permitiendo crear nuevos tipos de bloques de construcción que derivan de los existentes, pero que sean específicos a un problema" [Booch et al., 1999].

  Cuando se aplica un estereotipo sobre un elemento como un Nodo o una Clase se esta extendiendo UML, creando un nuevo bloque de construcción como cualquiera de los existentes, pero con sus propias características (su propio conjunto de valores etiquetados), semántica (sus propias restricciones) y notación (su propio icono) especiales.

  Por ejemplo, para el caso de los Componentes, se mencionó líneas arriba que existen otras representaciones gráficas "estereotipadas" que parten de la representación gráfica básica, mostrado en la Figura 2-42.

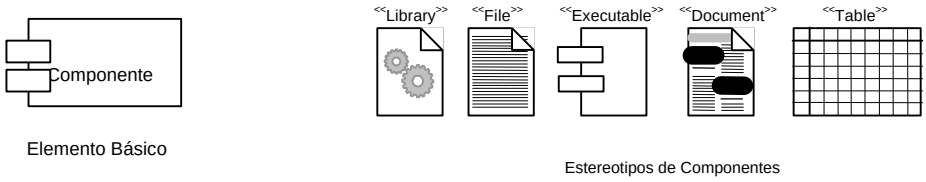


Figura 2-42: Ejemplo de Estereotipos como mecanismos de extensión de Componentes

En su forma mas simple, un estereotipo se representa como un nombre entre comillas tipográficas (por ejemplo <<nombre>>) y se coloca sobre el nombre de otro elemento. Como señal visual se puede definir un icono para el estereotipo y mostrar ese icono a la derecha del nombre (si se utiliza la notación básica para ese elemento) o utilizar ese icono como símbolo básico para el elemento estereotipado (como en el caso del ejemplo anterior).

Por ejemplo, en el entorno de desarrollo de C++ o Java a menudo será necesario modelar las excepciones, que son clases especiales que solo se permite que sean lanzadas o capturadas. Así, si queremos hacer un estereotipo para la excepción Overflow, tendremos lo mostrado en la Figura 2-43.

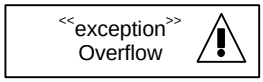


Figura 2-43: Ejemplo de Estereotipo para una excepción

Valores Etiquetados

"Un Valor Etiquetado extiende las propiedades de un bloque de construcción de UML, permitiendo añadir nueva información en la especificación de ese elemento" [Booch et al., 1999].

Mientras que los Estereotipos agregan nuevos elementos a UML, los Valores Etiquetados agregan nuevas propiedades a los elementos de UML existentes (incluso se pueden agregar propiedades a Estereotipos individuales). Se debe aclarar que un Valor Etiquetado no es lo mismo que un atributo de una clase, el Valor Etiquetado es un metadato, pues el valor se aplica al propio elemento no a sus instancias. Por ejemplo, en la Figura 2-44 se utiliza un Valor Etiquetado para especificar el número de procesadores instalado en cada tipo de Nodo en un Diagrama de Despliegue, y para especificar la versión y el autor de una Clase.

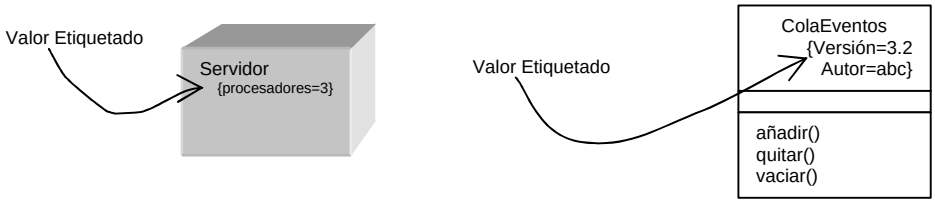


Figura 2-44: Ejemplo de Valores etiquetados

En su forma más simple, un valor etiquetado se ve como una cadena de caracteres entre llaves que se coloca bajo el nombre de otro elemento. Esta cadena incluye un nombre (etiqueta), un separador (el símbolo "=") y un valor (de la etiqueta)

Uno de los usos más comunes de los valores Etiquetados es para especificar propiedades relevantes a la generación de código o gestión de configuraciones. Por ejemplo, se pueden utilizar valores etiquetados para especificar el lenguaje de programación en el cual se debe

implementar una determinada Clase, y como se vio en el ejemplo anterior, se puede utilizar Valores Etiquetados para especificar la versión y el autor de un Componente.

Restricciones

"Una Restricción extiende la semántica de un bloque de construcción de UML, permitiendo añadir nuevas reglas o modificar las existentes" [Booch et al., 1999]. Una restricción específica condiciones que deben cumplirse para que el modelo esté bien formado.

Por ejemplo, en la Figura 2-45 se especifican Restricciones entre Clases, una Restricción que indica que la comunicación entre la asociación de Cartera y CuentaBancaria sea segura, otra Restricción que indica que solo se puede dar una de las dos asociaciones siguientes CuentaBancaria-Empresa o CuentaBancaria-Persona, y por último, otra Restricción que indica que la asociación entre dos Personas debe ser entre Personas de sexo diferente.

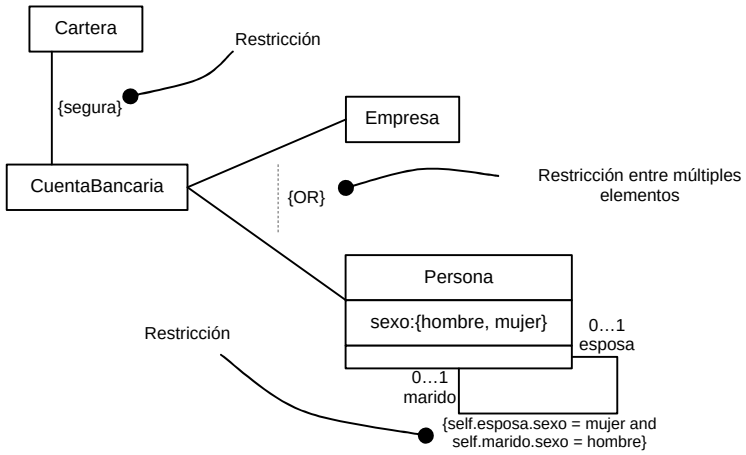


Figura 2-45: Ejemplo de Restricciones entre Clases

A grandes rasgos, todo lo anterior es lo que define a UML, haciéndolo un lenguaje de modelado flexible pero bien definido. En resumen UML es una herramienta de modelado que se forma de Bloques de Construcción, Reglas y Mecanismos Comunes.

3

El Proceso Unificado de Desarrollo de Software

En este capítulo se expone una de las bases teóricas de este trabajo: el Proceso Unificado de Desarrollo de Software. La metodología de desarrollo propuesta por los mismos creadores de UML (Jacobson, Booch y Rumbaugh). Una Vez que UML expone la herramienta de modelado, incluidos todos los artefactos que se deben generar para un proyecto de software orientado a objetos, el Proceso Unificado indica quién hace qué y cuando lo hace, en otras palabras, el Proceso Unificado es la guía para utilizar UML.

3.1. Orígenes

Solo se puede citar algunas de las fuentes que dieron origen al Proceso Unificado (se utiliza "Proceso Unificado" como una abreviación de "Proceso Unificado de Desarrollo de Software") pues son muchas las contribuciones al mismo.

La primer influencia de consideración que tiene el proceso unificado se puede encontrar en el *Método Ericsson* de modelado de sistemas de Ivar Jacobson, definido en 1967 [Rumbaugh et al., 2000]. En el Método Ericsson se introduce un concepto muy importante, el concepto de fragmentación del sistema, esto es, descomponer el sistema en subsistemas interconectados entre ellos, haciendo el sistema más manejable para su análisis. El sistema compuesto por subsistemas y la relación entre ellos se plasmaban en un diagrama donde se documentaban también los mensajes entre subsistemas. También se elaboraban versiones primitivas de diagramas de secuencia, diagramas de colaboración y diagramas de actividades, pero su aportación principal fue el desarrollo basado en componentes.

La CCITT (Consultative Committee for International Telephony and Telegraphy) realizó una gran aportación al definir el *Lenguaje de Especificación y Descripción (Specification and Description Language, SDL)* en 1976 para especificar y describir el comportamiento funcional de los sistemas de telecomunicaciones. El SDL se basaba también en sistemas formados por bloques que se comunicaban entre sí a través de mensajes, además se tenía el concepto de proceso dentro de bloques, procesos que tenían un comportamiento similar a las clases, pues se podían instanciar. En SDL también se manejaban diagramas similares a los diagramas de actividades, diagramas de clases, diagramas de colaboración y diagramas de secuencia de UML. La principal aportación de SDL fue el concepto de procesos que se podían instanciar, dando los primeros pasos hacia un lenguaje que podía modelar lo que hoy conocemos como objetos. Otra aspecto importante de SDL fue el proporcionar un estándar internacional que proponía estandarizar el modelado (fue una versión preliminar a lo que hoy se ofrece en UML).

Con la fundación, en 1987, de Objetory AB por parte de Ivar Jacobson y la definición de su proceso para la creación de sistemas llamado "*Objetory*" se dio un gran paso en el área de metodologías para el desarrollo de sistemas orientados a objetos. Originalmente Objetory AB era una empresa que se dedicaba a telecomunicaciones, pero su proceso para la creación de sistemas logro salir hacia otro tipo de empresas, incluso se difundió en muchos países. Objetory AB tenía gran influencia de Ericsson debido a que Ivar Jacobson trabajó en Ericsson antes de la fundación de Objetory AB.

A partir del Proceso Objetory se comenzó a utilizar el concepto de Casos de Uso, uno de los conceptos fundamentales de UML y del Proceso Unificado. Con la introducción de los Casos de Uso se comienza a dar mayor importancia a la comunicación entre usuarios finales del sistema y los desarrolladores encargados de construirlo. También aportó la creación de una serie de modelos durante el ciclo de vida del proyecto. Cada modelo estaba compuesto de artefactos (diagramas y otros elementos de modelado) relacionados entre sí y creados en un momento dado del ciclo de

vida. Cada modelo que se creaba debería estar basado en el modelo anterior y servir de base para la creación del modelo posterior, es decir, deberían tener *traza*[1] entre ellos. La traza entre modelos implicaba también cierto grado de dependencia, pues los cambios a algún modelo o a artefacto dentro de ellos, deberían reflejarse en artefactos o modelos construidos con base al modificado.

Objetory se desarrollo desde 1988 hasta 1995 en diferentes versiones, versiones que se creaban a partir de la anterior y utilizando el mismo Proceso Objetory para su desarrollo. Esto continuó hasta finales de 1995, año en que Rational Software Corporation adquirió Objetory AB y le dio nuevos enfoques al Proceso Objetory.

La idea principal de Rational Software Corporation al adquirir Objetory AB fue el unificar los procesos para la creación de sistemas, aprovechando la robustez del Proceso Objetory y complementándolo con practicas propias para el desarrollo de sistemas, por ejemplo la abstracción, la ocultación de la información, la reutilización, el prototipado y sobre todo dos conceptos fundamentales: el desarrollo iterativo y el énfasis en la arquitectura. En Rational se veía la arquitectura del sistema como un conjunto de modelos o vistas, lo que permitía ver diversos aspectos del sistemas desde diferentes perspectivas, algunas importantes para usuarios, otras para analistas, otras para diseñadores y otras para programadores, pero que permitían la comunicación entre todos ellos.

Rational complemento el proceso Objetory en las áreas que no estaba bien desarrollado, como la administración de requerimientos, implementación y pruebas, formando así el *Proceso Objetory de Rational 4.1* con fases bien definidas y un desarrollo iterativo controlado.

En Octubre de 1994, antes de la fusión Rational-Objetory AB, Rational Software Corporation incorporo entre su personal a James Rumbaugh, quien venia de Centro de Investigación y Desarrollo de General Electric y era autor de la *Object Modelling Technique* (*OMT*). Otra figura importante que ya colaboraba en Rational, Grady Booch (quién tenia su propio método de modelado de objetos, llamado el *Método Booch*), trabajo con Rumbaugh en la unificación de los métodos de ambos, lo que dio lugar al *Método Unificado 0.8* en octubre de 1995.

Con la unión de Rational y Objetory AB, Booch, Rumbaugh y Jacobson trabajaron juntos para publicar la versión 0.9 del Lenguaje Unificado de Modelado. Para cuando se libero la versión 1.1 de UML en 1997, la colaboración no solo incluía a estos tres personajes, también intervinieron otros metodologistas y empresas de software como IBM, Microsoft y HP, bajo la estandarización del Object Management Group.

El Proceso Objetory de Rational ya utilizaba UML en todos sus modelos, pero con nuevas aportaciones de empresas que adquiría o con las que se fusionaba Rational, el proceso fue madurando cada vez más, y en junio de 1998 se realizó un cambio de nombre para expresar la unificación de diversas técnicas de desarrollo, la unión del trabajo de varios metodologistas y el uso del Lenguaje Unificado de Desarrollo, el nuevo nombre que le dieron fue "*El Proceso Unificado de Rational 5.0*", del cual su versión genérica se conoce como el *Proceso Unificado de Desarrollo de Software*.

[1] Dependencia que indica una relación de proceso o histórica entre dos elementos que representan el mismo concepto, sin reglas para dividir el uno del otro. [Booch et al., 1999]

Como se ha visto, el Proceso Unificado reúne experiencia y conocimientos desde hace más de 30 años, y esto se representa en la Figura 3-1.

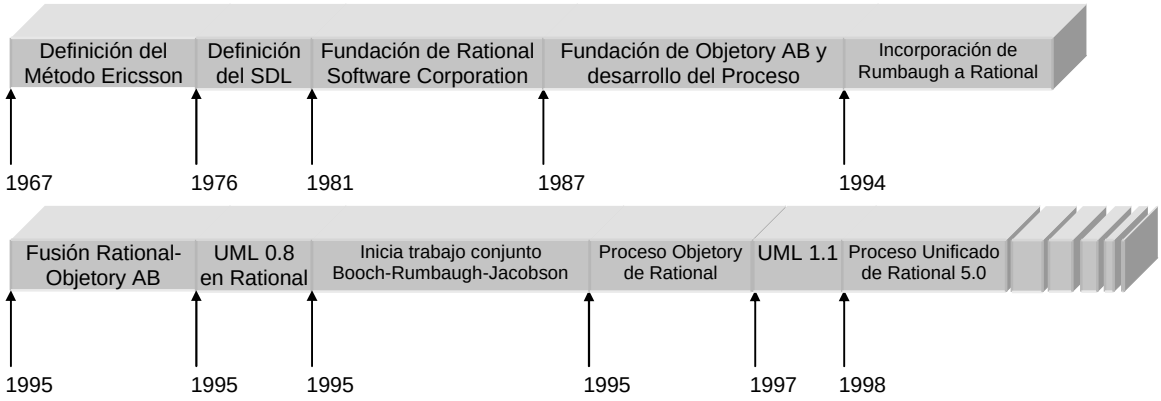


Figura 3-1: Principales desarrollos teóricos y prácticos sobre los que se construyó el Proceso Unificado

3.2. Descripción del Proceso Unificado

Para entender qué es el Proceso Unificado tenemos que definir qué es un proceso. Un proceso es una dirección organizativa para lograr un objetivo, es lo que nos dice *quién* hace *qué*, *cuándo* lo hará y *cómo* lo hará [Jacobson et al., 2000]; el objetivo que se persigue en el área de Ingeniería de Software es construir o mejorar un Producto Software. De esta manera el Proceso Unificado es un proceso de desarrollo de software, y los procesos de desarrollo de software buscan convertir los requerimientos de usuarios en un producto software.

Un proceso debe:

- ✓ Ser actual y ser actualizable
- ✓ Ser comprensible para todos los involucrados en él
- ✓ Estar construido sobre la tecnología actual
- ✓ Contar con herramientas que lo soporten
- ✓ Desarrollarse de acuerdo a las habilidades del personal que participe en él
- ✓ Presentar patrones de organización

Estas consideraciones que deben tomar en cuenta los procesos y las personas que los desarrollan nos permiten obtener beneficios de alto valor como:

- ✓ Reducir los riesgos en los proyectos
- ✓ Aumentar la predicción en los proyectos
- ✓ Fomentar una visión y una cultura comunes para los participantes

El Proceso Unificado en particular, es un proceso de desarrollo de Software que:

- ✓ Esta dirigido principalmente a productos software orientados a objetos
- ✓ Utiliza el Lenguaje de Modelado Unificado como herramienta de modelado
- ✓ Se basa en tres ideas básicas: está dirigido por casos de uso, esta centrado en la arquitectura del sistema y es iterativo e incremental
- ✓ Define el ciclo de vida del proyecto en cuatro fases: Inicio, Elaboración, Construcción y Transición
- ✓ En cada una de estas fases se desarrollan cinco flujos de trabajo: requerimientos, análisis, diseño, implementación y prueba

3.3. Bases del Proceso Unificado

De las características del Proceso Unificado expuestas líneas arriba, la más representativa de él es la base de tres ideas: estar dirigido por Casos de Uso, estar centrado en la Arquitectura y el ser iterativo e incremental. Las tres ideas base se expondrán en esta sección. El hecho de que se exponga esta característica inmediatamente es debido a que se puede considerar el núcleo del Proceso, al no basarse en alguna de estas ideas se altera en gran forma al proceso en sí, y el resultado seria algo que no puede llamarse Proceso Unificado. Expuesta la importancia de ellas, se procederá a explicarlas:

3.3.1. Dirigido por Casos de Uso

Antes de iniciar la explicación, cabe recordar que un Caso de Uso es la especificación del comportamiento de un sistema o parte de él para cubrir una funcionalidad requerida por un usuario. Esta especificación de comportamiento describe un conjunto de secuencias de acciones, incluyendo variantes [Booch et al., 1999]. Su función es la descripción básica, sencilla y primaria de las acciones del sistema (o parte de él) para cubrir necesidades del usuario para que sirva como un medio de comunicación entre el equipo de desarrollo y el o los usuarios. Los Casos de Uso siempre deben elaborarse en conjunto con el usuario(s) y desde la perspectiva del usuario(s).

Sobre éstas representaciones del comportamiento del sistema (para cubrir los requerimientos del usuario) se va a basar todo el desarrollo del producto en el Proceso Unificado, tomando los Casos de Uso como base para crear nuevos artefactos y modelos, y para verificar y probar el producto o partes de él.

Los Casos de Uso se han llegado a hacer populares y se han adoptado de forma casi universal en el desarrollo de sistemas software por dos razones fundamentales [Jacobson et al., 2000]:

- ✓ Son un medio intuitivo y sistemático para capturar requerimientos funcionales desde el punto de vista de lo esperado por el usuario
- ✓ Dirigen el proceso de desarrollo al basar el análisis, el diseño y las pruebas en ellos (los Casos de Uso) y al planificar y coordinar las pruebas y el diseño

El objetivo de los Casos de Usos es identificar las funciones que debe tener el sistema en desarrollo para los usuarios, pero ¿cuáles usuarios? Es muy común que un solo sistema brinde funciones a mas de un tipo de usuario (incluso a usuarios que no son humanos, pudiendo ser otros sistemas de software o hardware), por lo cual, se utiliza el concepto de actor, el actor representa a un tipo de usuarios. Cuando se intenta definir un Caso de Uso, se parte de la pregunta ¿Qué se desea que haga el sistema para cada usuario?

Los Casos de Uso intervienen durante el proceso de creación de un sistema de distintas formas:

- Una vez especificados los Casos de Uso, estos nos proporcionan una forma intuitiva de comunicarnos con los usuarios, de tal suerte que los usuarios no deben aprender una nomenclatura o simbología especial para ayudarnos en la identificación y refinación de sus requerimientos. De está manera también propicia el trabajo en conjunto usuario-cliente-desarrollador.

- Al especificar de manera completa todas las formas de utilizar el sistema por todos los posibles actores, los Casos de Uso limitan al sistema, por lo que se pueden utilizar como base en el contrato con el cliente.

- Después de detectar las necesidades de los usuarios, los Casos de Uso nos ayudan a encontrar las Clases y a definir la interfaz de usuario que tendrá el sistema. También

ayudan a los jefes de proyectos a planificar, asignar y controlar las tareas que los desarrolladores realizan, dividiendo las tareas en base a Casos de Uso.

- Los Casos de Uso son el inicio de la trazabilidad entre modelos y artefactos. Al ser el primer diagrama que se realiza (e incluso el mas importante al plasmar lo que se espera haga el sistema) todos los demás artefactos, diagramas y modelos que se construyen, se hacen tomando como base a los Casos de Uso, de esta manera cualquier cambio o modificación realizada a los artefactos posteriores, se debe reflejar en los Casos de Uso y viceversa.

- Se ha mencionado que el Proceso Unificado es iterativo e incremental (lo cual se explicara mas adelante), sin embargo las iteraciones e incrementos serán controladas y limitadas por los Casos de Uso. Para controlar las iteraciones, se debe seleccionar los Casos de Uso "claves" para el desarrollo, para asignar así jerarquías entre ellos y determinar sobre cuales se iniciaran las primeras iteraciones de desarrollo.

- En la creación del Manual de Usuario, los Casos de Uso definen de qué manera los usuarios van a interactuar con el sistema, proporcionando así las bases para la creación de este documento.

- Durante la implementación del sistema se puede apoyar de los Casos de Uso para detectar los caminos de ejecución de mayor utilización por los usuarios para poner mayor importancia en ellos a fin de mejorar su rendimiento para con los usuarios.

- En las pruebas del sistema los ingenieros de prueba verifican que el sistema implementa en verdad la funcionalidad descrita en los Casos de Uso y que así se satisfagan los requerimientos del sistema. Para lograra esto, se crean Casos de Prueba, los cuales se elaboran a partir de los Casos de Uso.

A continuación se explicara como es que los Casos de Uso intervienen en el desarrollo de los diferentes modelos del sistema.

Durante la captura de requerimientos del sistema se crea el primer modelo, llamado Modelo de Casos de Uso, el cual contiene todas las características mencionadas en líneas anteriores.

En el análisis se crea el Modelo del Análisis, que es una especificación detallada de los requerimientos y una aproximación al Modelo del Diseño. Este modelo describe de manera más precisa los Casos de Uso al elaborarlos[1] en forma de colaboraciones entre Clasificadores Conceptuales[2] y muestra una primera aproximación a la arquitectura final del sistema. El Modelo del Análisis es un modelo conceptual que sirve para la creación del Modelo del Diseño, el cual es un modelo para implementación. De esta manera se tiene una relación directa (dependencia de traza) entre la realización de un Caso de Uso en el Modelo del Análisis y la realización de ese mismo Caso de Uso en el Modelo del Diseño.

El Modelo del Diseño, creado posteriormente y a partir del Modelo del Análisis, es un modelo de implementación, es decir, es un modelo expresado por entero tomando en cuenta la plataforma de desarrollo (lenguaje de desarrollo, tecnología de desarrollo, sistema operativo, etc.) y que contiene elementos que van a ser implementados directamente. En este modelo, cada Caso de Uso es representado a partir de su

[1] se utiliza la palabra "elaborarlos" para indicar que se detallan y analizan los Casos de Uso, en otras ocasiones también se puede referir a esto como "realización de Casos de Uso", frase que implica la representación de un Caso de Uso en alguna etapa o flujo de trabajo mediante los símbolos y diagramas correspondientes.

[2] Un Clasificador es un mecanismo mediante el cual se agrupan características estructurales y de comportamiento. Los clasificadores incluyen interfaces, clases, tipos de datos, componentes y nodos [Jacobson et al., 2000]. Un Clasificador Conceptual, es un clasificador que no va a ser implementado de forma directa (clasificadores del análisis)

realización en el análisis, como colaboraciones entre clasificadores de implementación, por ejemplo clases, subsistemas, interfaces de subsistemas, etc.

Casi a la par de la creación del Modelo del Diseño, se crea el Modelo del Despliegue donde se verifica que los Casos de Uso se puedan implementar como componentes de software que se ejecutan en nodos de procesamiento.

Después de la creación del Modelo del Diseño y del Modelo de Despliegue, los programadores implementan las clases diseñadas en forma de archivos ejecutables (DLL´s, JavaBeans, componentes ActiveX, tablas, etc.) auxiliándose de los Casos de Uso para determinar el orden de implementación e integración de los componentes.

Finalmente durante las pruebas, los ingenieros de pruebas verifican que el sistema implemente de forma correcta la funcionalidad descrita en los Casos de Uso. En las pruebas se crea el Modelo de Pruebas que a su vez contiene Casos de Prueba, estos últimos se obtienen directamente de los Casos de Uso.

Hasta aquí se han explicado la influencia que tienen los Casos de Uso durante todo el desarrollo de los sistemas, desde el inicio y hasta el final, por lo que se confirma como primera característica del Proceso Unificado el hecho de que esté **dirigido por Casos de Uso**.

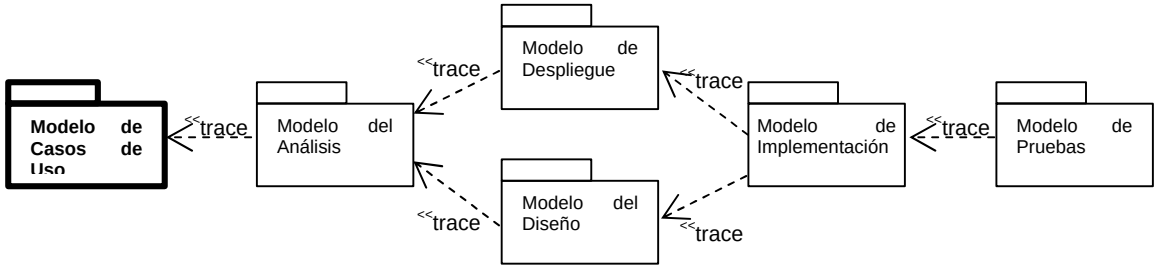


Figura 3-2: Dependencia de traza de todos los modelos con el Modelo de Casos de Uso

3.3.2. Centrado en la Arquitectura

Para poder explicar el hecho de que el Proceso Unificado se centra en la arquitectura, hay que explicar qué es la arquitectura de un sistema.

La arquitectura en la forma mas concreta y de acuerdo a [Jacobson et al., 2000] se define como *el conjunto de decisiones significativas acerca de la organización de un sistema software, la selección de los elementos estructurales a partir de los cuales se compone el sistema, las interfaces entre ellos, junto con su comportamiento, las colaboraciones entre estos elementos para formar elementos progresivamente mayores y el estilo arquitectónico que guía esta organización*.

Analizando la definición anterior, se observa que considera los siguientes puntos:

- ✓ Elementos estructurales que componen al sistema
- ✓ Las interfaces entre los elementos estructurales
- ✓ El comportamiento de estos elementos estructurales
- ✓ Las colaboraciones entre los elementos estructurales
- ✓ La organización de todos los elementos anteriores, y
- ✓ El estilo arquitectónico que guía la organización.

Se describirá cada uno de estos puntos. En primer lugar se habla de "*elementos estructurale*s", estos elementos estructurales (clases, subsistemas, nodos) no son la totalidad de elementos que

forman al sistema, solo son los elementos mas importantes desde un punto de vista de descripción del sistema.

Cada uno de estos elementos no trabaja de forma aislada, sino que se comunican entre ellos. La comunicación entre ellos tiene lugar por medio de *interfaces* de estos elementos. Las interfaces indican qué es lo que los demás elementos "verán" de un elemento dado, por ejemplo, a que información se puede tener acceso y que servicios nos puede brindar, además de definir la manera en que solicitaran datos o servicios de un elemento a otro.

De acuerdo a como se realice el trabajo del cual es responsable un elemento dado dentro del sistema, y a los servicios y datos que le sean solicitados por parte de otro elemento, cada uno se *comportará* de una manera especifica que estará de acuerdo al estimulo externo al que sea sometido o al trabajo que realizará.

En la composición de un sistema se puede hablar de elementos a diversos niveles, por ejemplo elementos pequeños que realicen tareas muy especializadas y especificas, y que por medio de la *colaboración* con otros elementos del mismo tipo formen otro elemento de jerarquía superior que realizará tareas menos especificas y mas diversas (con ayuda de los elementos que lo integra). De esta manera se puede seguir realizando abstracciones de manera que se llegue al sistema en sí que estará formado de elementos de jerarquía muy superior que colaborarán entre ellos para formar al propio sistema.

Así la arquitectura del sistema especifica qué elementos "clave" describen al sistema y sus características (interfaces, comportamiento y colaboraciones) y la forma en que todo ello se *organiza*, de acuerdo a un *estilo arquitectónico* de sistemas de software (o patrón de la arquitectura), que por ejemplo podría ser el patrón Broker, Cliente/Servidor, Three-Tier, Peer-to-Peer o una combinación de ellos. Los estilos arquitectónicos son soluciones estándar a problemas de arquitectura que se presentan comúnmente.

La arquitectura debe considerar partes del sistema que pueden ser reutilizadas de sistemas heredados y adaptar todos sus elementos a esto, además de considerar requerimientos no funcionales que restrinjan al sistema en recursos, políticas, estándares y lo orienten a cumplir determinadas características.

Una vez explicada la arquitectura se puede exponer cómo es que se crea. La arquitectura se crea a partir de la segunda fase que marca el Proceso Unificado, y se forma a partir de vistas de los modelos del sistema. A continuación se explicara esto con más detalle.

El Proceso Unificado marca una serie de modelos que se desarrollan durante las fases de creación del sistema. Un modelo es un conjunto de artefactos que describen cierto aspecto del sistema, así podemos tener los siguientes modelos:

- **Modelo del Negocio**: establece una abstracción de la organización en donde se utilizara el sistema a desarrollar.
- **Modelo del Dominio**: establece el contexto del sistema.
- **Modelo de Casos de Uso**: describe el sistema desde un punto de vista de funcionalidades para el usuario de una forma muy simple.
- **Modelo del Análisis**: donde se refinan y estructuran los requerimientos del sistema.
- **Modelo del Diseño**: donde se describen los casos de uso desde un punto de vista de la solución de los mismos y tomando en cuenta requerimientos no funcionales.
- **Modelo de Procesos**: establece los mecanismos de sincronización y concurrencia del sistema.
- **Modelo de Despliegue**: donde se describe el sistema considerando los nodos que forman la topología de hardware sobre la que se ejecutará el sistema.
- **Modelo de Implementación**: donde se abarcan los componentes usados para el ensamblado y lanzamiento del sistema ejecutable.

- **Modelo de Pruebas**: establece las formas de validar y verificar el sistema.

Todos estos modelos cubren las decisiones importantes sobre el sistema, aunque algunos son opcionales o no siempre se elaboran cuando se crea un sistema, por ejemplo, el Modelo del Negocio puede existir en una organización aun antes de iniciar la construcción de un sistema, o para un sistema monolítico, monousuario y que se ejecute en un solo Nodo, el Modelo de Despliegue y el Modelo de Procesos no tienen mucho sentido.

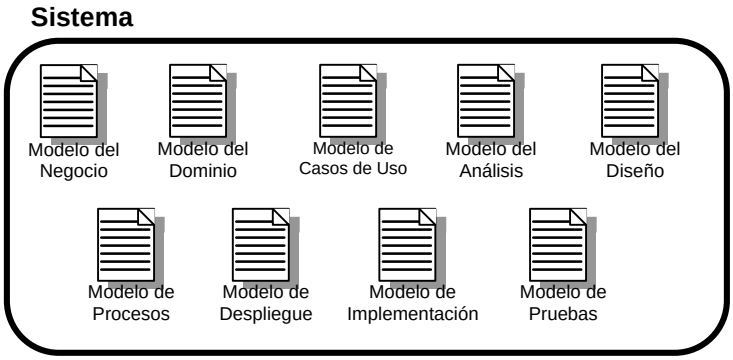


Figura 3-3: Posibles modelos para un sistema

Una vista de un modelo, es una proyección de un modelo, que es visto desde una perspectiva dada o un lugar estratégico y que omite las entidades que no son relevantes para esta perspectiva [Jacobson et al., 2000]. Por lo cual, la arquitectura del sistema estará formada por un conjunto de vistas que muestran el sistema desde diferentes perspectivas. Y precisamente esto último es lo que hace tan importante a la arquitectura de un sistema, el hecho que muestre las partes mas características de un sistema desde diversos puntos de vista, lo que será de gran utilidad para los diversos participantes del proyecto de software, pues cada uno de ellos se enfocara a la perspectiva o vista que le sea de interés para realizar su trabajo.

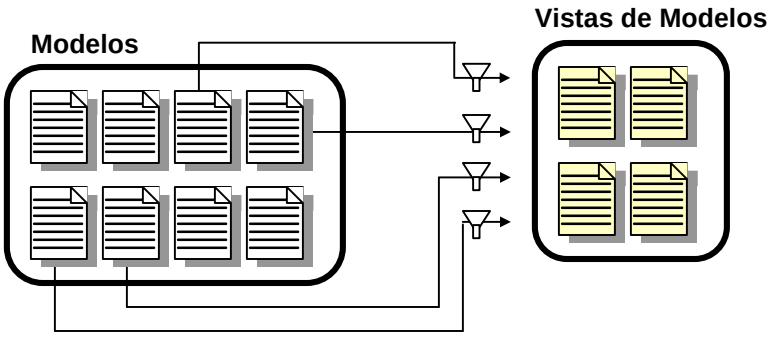


Figura 3-4: Vistas y modelos

Al decir que “el Proceso Unificado esta centrado en la arquitectura” marca la importancia de ésta en el proceso en sí, pues aunque tengamos la dirección de los Casos de Uso para realizar el sistema, lo que nos va a dar suficiente estabilidad y claridad del sistema es la arquitectura. Por ejemplo, los casos de uso nos dicen qué debemos hacer, y la arquitectura nos indica cómo debemos hacerlo, a la vez que nos da una base sólida y temprana para poder construir las funcionalidades que buscan los casos de uso. Se debe pensar a la arquitectura como un descanso de escaleras, que nos va a ayudar a seguir avanzando sobre los demás escalones (los casos de uso).

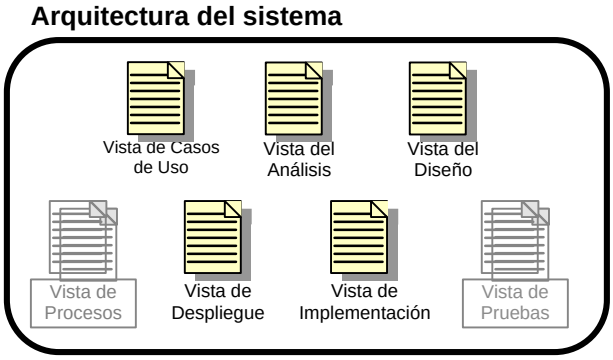


Figura 3-5: Arquitectura y vistas

3.3.3. Iterativo e Incremental

Para la explicación de la naturaleza iterativa e incremental del Proceso Unificado, es necesaria la explicación del ciclo de vida del mismo. Esto se resume en la Figura 3-6:

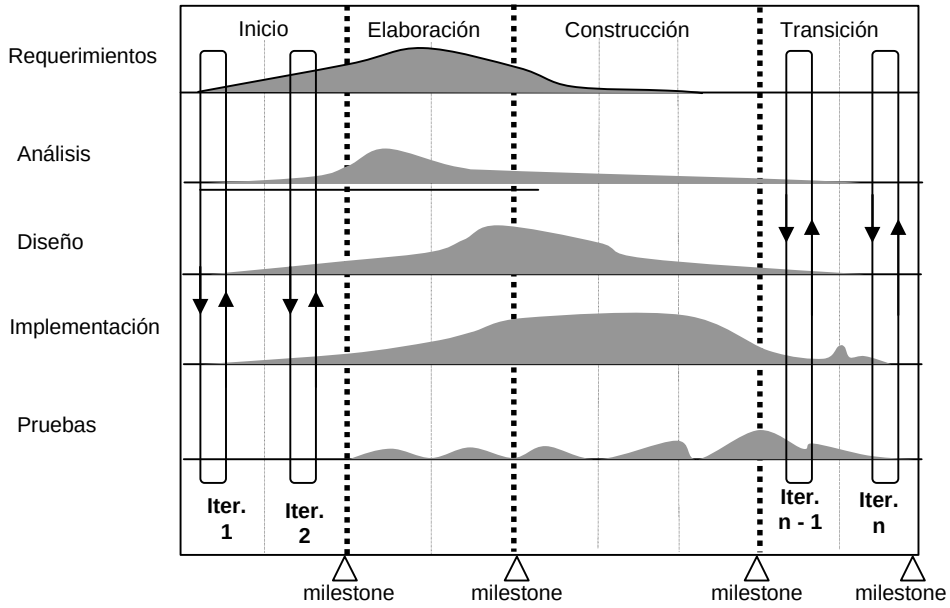


Figura 3-6: Ciclo de vida del Proceso Unificado

En la Figura 3-6 anterior se indican en la parte superior las cuatro fases del proceso (Inicio, Elaboración, Construcción y Transición). Del lado izquierdo se muestran los *flujos de trabajo fundamentales* del Proceso Unificado. La grafica indica que en cada una de las fases se realizaran de manera secuencial los flujos de trabajo, pero con enfoques e importancia diferente, como lo indican las sombras dentro del cuadro. Por ejemplo, en la Fase de Inicio se le da principal importancia al flujo de trabajo de Requerimientos, mientras que al Análisis, Diseño e Implementación se le da una importancia menor, y casi no se toma en cuenta al flujo de trabajo de Pruebas; por otro lado en la Fase de Construcción, la importancia principal la tiene el flujo de trabajo de Implementación y Diseño, mientras que los Requerimientos, el Análisis y Pruebas tienen

una importancia menor. La manera en que se recorrerán los flujos de trabajo en las fases una y otra vez será a través de iteraciones que estarán dirigidas por Casos de Uso.

Las iteraciones buscaran la realización de un Caso de Uso (primero seleccionando para ello los de importancia mayor) a través de su elaboración en los diversos flujos de trabajo, aplicándolos de forma repetitiva hasta convertir el Caso de Uso seleccionado en una parte funcional del sistema.

A través del desarrollo se tendrán una serie de hitos o piedras pilares (milestone), los cuales son puntos en el tiempo (fechas preestablecidas) que proporcionan a todos los involucrados en el proyecto de software criterios para identificar que se ha concluido una fase y se pueda autorizar el paso a la siguiente fase.

Para cada una de las fases mostradas en la grafica anterior se tienen los siguientes criterios [Jacobson et al., 2000]:

Fase de Inicio: el criterio esencial es determinar la viabilidad del proyecto.

- ✓ Identificar y reducir los riesgos críticos para la viabilidad del sistema.
- ✓ Crear una arquitectura candidata a partir del desarrollo de un subconjunto clave de requerimientos.
- ✓ Realizar la estimación inicial de costo, esfuerzo calendario y calidad del producto.
- ✓ Iniciar el análisis del negocio por el cual el proyecto parece rentable.

Fase de Elaboración: el criterio esencial es lograr la capacidad para construir un sistema dentro de un marco de trabajo económico.

- ✓ Identificar y reducir los riesgos que afecten a la construcción del sistema.
- ✓ Especificar la mayoría de los casos de uso.
- ✓ Extender la arquitectura candidata hasta una *Línea Base*.
- ✓ Preparar el Plan de Desarrollo del Proyecto de forma detallada para guiar la construcción.
- ✓ Estimar límites para justificar la inversión.
- ✓ Terminar el análisis del negocio, con lo que se determinará si el proyecto vale la pena.

Fase de Construcción: el criterio esencial es lograr un sistema capaz de una operatividad inicial en el entorno del usuario.

- ✓ Realizar una serie de iteraciones a través de los flujos de trabajo que incrementos y entregas periódicos, con lo cual se puede determinar la vialidad del sistema con la entrega de una serie de ejecutables.

Fase de Transición: el criterio esencial es lograr un sistema que alcance una operatividad final.

- ✓ Modificar el producto, en caso de ser necesario, para solucionar problemas que no se hayan identificado en las fases anteriores.
- ✓ Corregir defectos.

Respecto a las otras dos bases del proceso (dirigido por Casos de Uso y centrado en la arquitectura), una enfocada a la función del sistema y la otra a la forma, se logran mediante las iteraciones en los flujos de trabajo y tomando en cada iteración una funcionalidad requerida por el sistema, es decir, un Caso de Uso, al que se le aplicará con diferente peso los flujos de trabajo de acuerdo a la fase donde se encuentre la iteración.

El resultado de las iteraciones varía de acuerdo a la fase en que nos encontremos. Las primeras iteraciones darán como resultado la determinación del ámbito del trabajo, la eliminación de riesgos críticos y la creación de la línea base de la arquitectura (esto se lograra al elaborar los Casos de

Uso “claves” del sistema). En las siguientes iteraciones se tendrán como resultado incrementos en las funcionalidades del sistema.

Es importante en el proceso iterativo e incremental determinar el orden en que se desarrollaran los Casos de Uso, de manera que se logre desarrollar primero los Casos de Uso que son mas importantes y básicos para todos los demás.

Cuando seguimos el desarrollo iterativo e incremental conseguimos:

- ✓ Atenuar los riesgos. Lo conseguimos al ordenar correctamente los Casos de Uso y elaborar los mas importantes primero casi por completo y no tratar de elaborar todos en conjunto (no llevar todos en su conjunto a través de los flujos de trabajo, sino elaborar uno por uno y en orden correcto). De esta manera se pueden detectar riesgos importantes desde la primeras fases del proyecto y no esperar a que los riesgos aparezcan en todo su conjunto en las ultimas fases de desarrollo, como consecuencia empezaremos a tratar los riesgos desde el principio del proyecto de una manera distribuida en el tiempo.
- ✓ Obtener una arquitectura robusta. Se logra en las primeras fases, a partir de la cual la arquitectura va madurando hasta convertirse en el sistema final. Si en las primeras fases se detectan cambios necesarios en la arquitectura, se puede permitir realizarlos, pues no se ha invertido demasiado.
- ✓ Administrar de requerimientos cambiantes. Comenzar a elaborar las funcionalidades básicas del sistema desde las primeras fases, nos permite mostrar los avances del proyecto al usuario en forma de un sistema con funciones básicas, lo que facilita y agiliza determinar los avances reales y el verificar si se están cumpliendo los requerimientos y determinar nuevos requerimientos que no se habían detectado hasta entonces por parte del cliente. En resumen, se obtiene una retroalimentación temprana y fiel del usuario.
- ✓ Permitir cambios tácticos. Con la detección temprana de riesgos, es posible replanear de forma temprana actividades de administración y del desarrollo técnico, para que estos cambios no afecten al final del desarrollo.
- ✓ Conseguir una integración continua. En cada iteración se elabora un Caso de Uso, el Caso de Uso elaborado, se integra inmediatamente en forma de una funcionalidad, así continuamente y a través de las iteraciones se integran nuevas funcionalidades hasta conseguir el sistema final. Esto es muy diferente al desarrollo tradicional, donde se deja hasta las últimas fases la integración total del sistema, en las cuales se pueden encontrar un gran numero de errores y demorar la entrega del producto final.
- ✓ Conseguir un aprendizaje temprano. El aplicar todos los flujos de trabajo en una pequeña parte del proyecto desde el inicio, permite que se tenga de forma temprana la comprensión de cada uno de ellos por parte de las personas involucradas, mismas que los practicaran mas y consecuentemente comprenderán aun mas en cada iteración sucesiva, por lo cual si se van incorporando nuevo personal en las fases siguientes, las personas que estuvieron involucradas desde el inicio ya tienen conocimiento y experiencia suficiente para guiar a los nuevos.

3.4. Flujos de Trabajo del Proceso Unificado

A continuación se explicarán de forma secuencial los cinco Flujos de Trabajo del Proceso Unificado, se debe aclarar que esta explicación secuencial solo es a manera de exponer las características “genéricas” de los Flujos de Trabajo, pero no quiere decir que se ejecutan exactamente como se marca en esta sección ya que los Flujos de Trabajo se aplican de manera iterativa y adaptada en cada una de las fases del proceso, por lo que en una sola fase de puede aplicar mas de una vez un Flujo de Trabajo dado y en otra fase se puede aplicar ese mismo Flujo de Trabajo distinto numero de veces y adaptado a esta otra fase.

En cada uno de los Flujos de Trabajo se especificarán Actividades, Roles y Artefactos mostrados como un diagrama de actividades estereotipado. Los Flujos de Trabajo se describirán como una secuencia de Actividades ordenadas, las cuales producen salidas que sirven de entrada a la siguiente Actividad. Aunque las Actividades están indicadas de forma secuencial y en orden, no es rígido este orden y secuencia, siempre y cuando se llegue al mismo resultado final equivalente. Tanto las entradas y salidas de las Actividades y el resultado final del Flujo de Trabajo se refieren a Artefactos, y los Artefactos a su vez representan también documentos sobre el sistema que se elaborará[1] (Modelos, Diagramas, especificaciones de Casos de Uso y Actores, etc.). Cada una de las Actividades es realizada por una persona o personas con un perfil específico al cual llamaremos Rol, así un Rol no necesariamente representará a una sola persona.

Los Artefactos y las Actividades se detallan dentro de cada uno de los Flujos de Trabajo que los involucre por primera vez, pero los Roles son una constante en todo el proceso de desarrollo del sistema, y aunque no intervendrán todos de manera simultanea, se considera que son de una importancia muy relevante dentro del Proceso Unificado, por lo cual se describirán a continuación cada uno de los Roles que marca el Proceso Unificado.

3.4.1. Roles del Proceso Unificado

Los Roles designan puestos a los cuales se les puede asignar personas, y los cuales estas personas aceptan. Un Rol es responsable de un conjunto de Actividades y además requiere un conjunto de aptitudes de la o las personas que lo ocuparán.

Dentro de un proyecto un Rol puede ser ocupado por una o mas personas, pero también una persona puede tener más de un Rol asignado

El Proceso Unificado utiliza el siguiente estereotipo para representar un Rol:

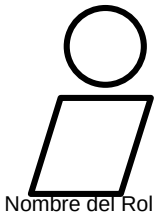


Figura 3-7: Representación gráfica de un rol

[1] El Proceso Unificado se refiere a Artefactos como cualquier tipo de información creada, producida, cambiada o utilizada por los Roles durante el desarrollo del sistema, y aunque hasta este momento solo se han tratado Artefactos que tienen que ver directamente con el sistema (relacionados al producto del proceso de desarrollo), los Artefactos también incluyen información que tienen relación con la calidad y control del proceso de desarrollo del sistema

Los Roles principales que marca el Proceso Unificado son:

1. **Analista de sistemas:** Participa principalmente en el Flujo de Trabajo de Requerimientos y es responsable de limitar el sistemas coordinando la captura de todos los requerimientos funcionales y no funcionales, y encontrando los Actores y Casos de Uso que modelen los requerimientos funcionales de forma completa y consistente, por tanto, es responsable del Modelo de Casos de Uso.

2. **Especificador de casos de uso:** Participa principalmente en el Flujo de Trabajo de Requerimientos y tiene como actividad primordial la especificación detallada de cada uno de los Casos de Uso, y para esto trabaja muy de cerca con los usuarios reales de los Casos de Uso bajo su cargo.

3. **Diseñador de interfaz de usuario:** Participa principalmente en el Flujo de Trabajo de Requerimientos y es su responsabilidad el dar forma visual a las interfaces de usuario (solo la forma, pues la implementación real será realizada dentro de otro flujo de trabajo por otro Rol) mediante prototipos de interfaz de usuario para algunos Casos de Uso y para cada Actor.

4. **Arquitecto:** Participa en casi todos los Flujos de Trabajo (a excepción del Flujo de Trabajo de Pruebas) y en general es responsable de la Arquitectura de todo el sistema y de los modelos de Análisis, Diseño, Desarrollo, Implementación y Despliegue (nótese que no es responsable del Modelo de Casos de Uso y del Modelo de Pruebas, pues estos modelos pasan a ser responsabilidad del Analista de Sistemas y del Diseñador de Pruebas respectivamente), para ello elabora vistas de cada uno de los modelos.

5. **Ingeniero de Casos de Uso:** Participa en el Flujo de Trabajo de Análisis y en el Flujo de Trabajo de Diseño y es responsable del análisis y diseño de las realizaciones de uno o mas Casos de Uso, garantizando su integridad y que cumplan los requerimientos que caen sobre ellos.

6. **Ingeniero de componentes:** Participa en los Flujos de Trabajo de Análisis, Diseño, Implementación y Pruebas. En el Flujo de trabajo de Análisis es responsable de una o más Clases del Análisis y de uno o más Paquetes del Análisis. En el Flujo de Trabajo de Diseño es responsable de una o más Clases del Diseño, de uno o más Subsistemas del Diseño y de las Interfaces correspondientes. En el Flujo de Trabajo de Implementación es responsable de uno o más Componentes, de uno o más Subsistemas de Implementación y de las Interfaces correspondientes. En el Flujo de Trabajo de Pruebas es responsable de algunos de los Componentes de Pruebas que automatizan algunos de los procedimientos de pruebas

7. **Integrador del sistema:** Participa en el Flujo de Trabajo de Implementación y su principal responsabilidad es, como dice su nombre, la integración del sistema, para lo cual planifica la secuencia de Construcciones e integración de Construcciones de cada iteración.

8. **Diseñador de pruebas:** Participa en el Flujo de Trabajo de Pruebas. Es responsable de la integridad del Modelo de Pruebas, de la planeación de las pruebas y la definición de los objetivos de cada prueba. Además selecciona y describe los Casos de Prueba y los procedimientos de prueba que se necesitan, y evalúa las Pruebas de Integración y del Sistema cuando se ejecutan.

9. **Ingeniero de Pruebas de Integración:** Participa en el Flujo de Trabajo de Pruebas. Es responsable de realizar las Pruebas de Integración del Sistema cada vez que se crea una construcción nueva y se agrega al sistema en desarrollo. Las Pruebas de Integración las realizara con base a los Casos de Prueba. También es responsabilidad de documentar los defectos que encuentre durante la realización de las Pruebas de Integración.

10. **Ingeniero de Pruebas del Sistema:** Participa en el Flujo de Trabajo de Pruebas y es responsable de realizar las Pruebas del Sistema necesarias sobre una Construcción que muestre el resultado de una iteración completa y documentar los defectos encontrados. Las Pruebas del Sistema se derivan de los Casos de Prueba. Un Ingeniero de Pruebas del Sistema puede ser cualquier involucrado en el proyecto, aun sin tener conocimiento del funcionamiento interno del sistema, requiriendo solo tener conocimiento del comportamiento externo del sistema.

Además se mencionan brevemente otros Roles, llamémosles secundarios, que aunque no participan directamente en actividades relacionadas con el producto (es decir, con el sistema en desarrollo), si se menciona su participación durante el proceso. Estos Roles secundarios son:

1. **Usuarios:** Este Rol representa personas o sistemas y participa en los Flujos de Trabajo de Requerimientos, Análisis y Pruebas. Tiene por responsabilidad ayudar en la especificación, prueba y refinamiento de requerimientos del sistema.

2. **Ejecutivo de Software:** Es un Rol que se encuentra en la cima de una organización de desarrollo y toma decisiones sobre los caminos que debe tomar su organización en relación al software desarrollado dentro de ella.

3. **Jefe de Proyecto:** Debido a que el ejecutivo de software pertenece a puestos altos dentro de la organización, tendrá poco tiempo para dedicarlo exclusivamente a un solo proyecto. El Jefe de proyecto es el responsable directo del buen termino de un proyecto de software y quien organizará, elegirá y coordinará directamente a todos los Roles

4. **Asesor:** Este Rol trabajara exclusiva y directamente con el Ejecutivo de Software y el Líder de Proyecto para guiar a ambos en la dirección y desarrollo del proyecto utilizando el Proceso Unificado, por lo cual se obvia que es un especialista con experiencia en el Proceso Unificado.

A continuación se muestran los Flujos de Trabajo en forma de un Diagrama de Actividades estereotipado y una breve descripción del flujo.

En el diagrama se tendrán carriles verticales para cada rol que participe en el flujo de trabajo. En cada carril se encuentran las actividades que debe realizar el rol y los artefactos que están bajo la responsabilidad del rol. Algunos de los artefactos que se encuentran en los carriles se muestran en forma menos nítida, lo que representa que este artefacto no es responsabilidad del rol del carril donde se encuentra, pero se coloca en su carril por que es utilizado como entrada en alguna actividad del rol, o bien es elaborado solo parcialmente por él rol propietario del carril, pero es responsabilidad de otro rol el terminar correctamente este artefacto.

El primer carril (de izquierda a derecha) de cada diagrama no pertenecerá a ningún rol y solo se encontraran en él los artefactos que son necesarios para ser utilizados en el flujo actual, pero no se elaboran en él.

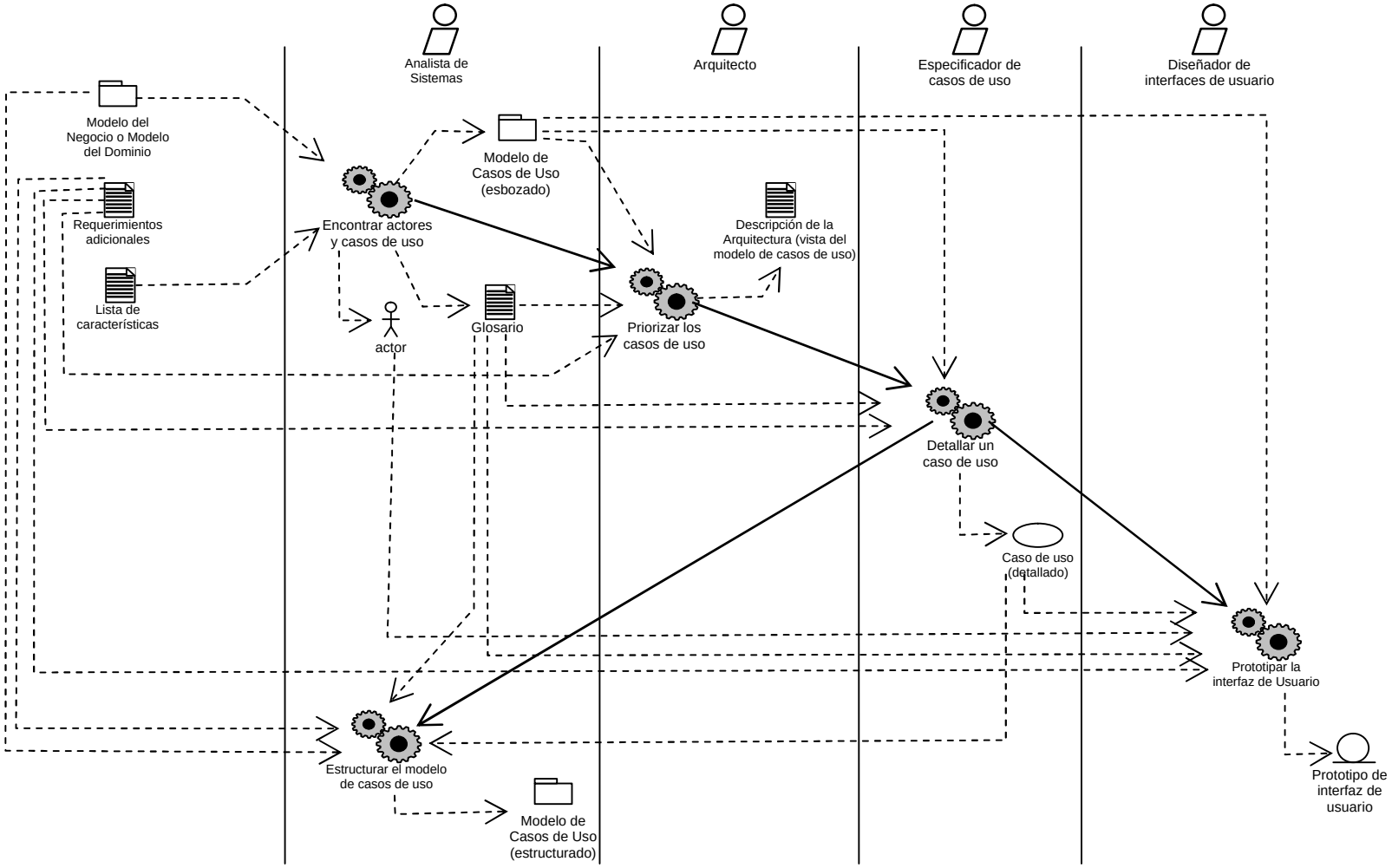


Figura 3-8: Flujo de Trabajo de Requerimientos

3.4.2. Flujo de Trabajo de Requerimientos

En el Flujo de Trabajo de Requerimientos (mostrado en la Figura 3-8) participan los roles Analista de Sistemas, Arquitecto, Especificador de Casos de Uso y Diseñador de Interfaces de Usuario.

Los artefactos que se generan son parte del Modelo de Casos de Uso y son los siguientes: identificación de Actores, descripción de Actores, glosario, identificación de Casos de Uso, descripción de Casos de Uso, descripción de Casos de Uso, prototipo de la interfaz de usuario y descripción de la Arquitectura (vista del modelo de Casos de Uso).

De esta forma el Modelo de Casos de Uso se integra de acuerdo al siguiente diagrama.

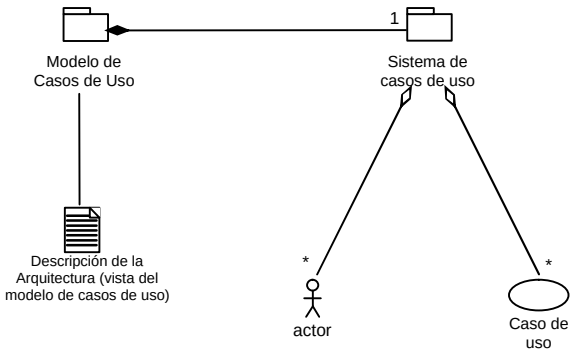


Figura 3-9: Integración del Modelo de Casos de Uso

En el diagrama anterior se muestra cómo esta integrado el Modelo de Casos de Uso. El Modelo de Casos de Uso se relaciona con el Sistema de Casos de Uso (que es un paquete que agrupa a la totalidad de casos de uso que componen el sistema) por medio de una Relación de Composición, la cual es una forma de Relación de Agregación (la Agregación relaciona un “todo” con sus “partes”) con un fuerte énfasis de pertenencia y vidas concurrentes de la parte con el todo, indicando que un Modelo de Casos de Uso se compone de un solo Sistema de Casos de Uso, a su vez, este se relaciona con los Casos de Uso y los Actores por medio de Relaciones de Agregación, que indican que el Sistema de Casos de Uso esta formado por cero o mas Actores y por cero o mas Casos de Uso.

Para explicar el Flujo de Trabajo de Requerimientos se describe cada una de las actividades indicando sus entradas y salidas. También se debe aclarar que en este Flujo de Trabajo en particular, dentro del carril de artefactos requeridos se encuentran artefactos que no provienen de flujos anteriores (Modelo del Negocio o Modelo del Dominio, Requerimientos Adicionales y Lista de Características), y que se generan en el arranque del proyecto por roles que en la mayoría de las ocasiones son ejecutivos de alto nivel o lideres de proyecto.

Otra aspecto que se debe tener siempre presente, es que el Flujo de Trabajo solo representa una secuencia lógica de las actividades utilizando el resultado de la ejecución de una actividad como entrada para ejecutar otra, pero en la práctica se puede trabajar por múltiples vías que producen el resultado fina equivalente, por lo cual una actividad puede ser retomada muchas veces, y cada una de estas puede acarrear la ejecución de una fracción de la actividad.

El rol de Analista de Sistemas inicia con la actividad de Encontrar Actores y Casos de Uso para preparar una primera versión del Modelo de Casos de Uso, con los Actores y Casos de Uso identificados. El Analista de Sistemas debe asegurar que el desarrollo del Modelo de Casos Uso captura todos los requerimientos que son entradas del flujo de trabajo, es decir, la Lista de Características y el Modelo del Dominio o del Negocio. Entonces el Arquitecto identificará los Casos de Uso relevantes arquitectónicamente hablando, para proporcionar entradas a la

priorización de Casos de Uso (y posiblemente otros requerimientos) que van a ser desarrollados en la iteración actual. Hecho esto, el Especificador de Casos de Uso describe todos los Casos de Uso que se han priorizado. Más o menos en paralelo con ellos, el Diseñador de Interfaz de Usuario sugiere las interfaces de usuario adecuadas para cada Actor basándose en los Casos de Uso. Entonces el Analista de Sistemas reestructura el Modelo de Casos de Uso definiendo generalizaciones de funcionalidad entre los Casos de Uso para hacerlo lo mas comprensible posible. [Jacobson et al., 2000]

Los resultados de la primera iteración a través de este flujo de trabajo consisten en una primera versión del Modelo de Casos de Uso, los Casos de Uso y cualquier prototipo de interfaz de usuario asociado. Los resultados de cualquier iteración subsiguiente consistirán entonces en nuevas versiones de estos artefactos. Hay que recordar que todos los artefactos se completan y mejoran incrementalmente a través de las iteraciones. [Jacobson et al., 2000]

Las actividades del Flujo de Trabajo de Requerimientos adoptan formas diferentes en las distintas fases e iteraciones del proyecto, como se muestra en la Figura 3-10:

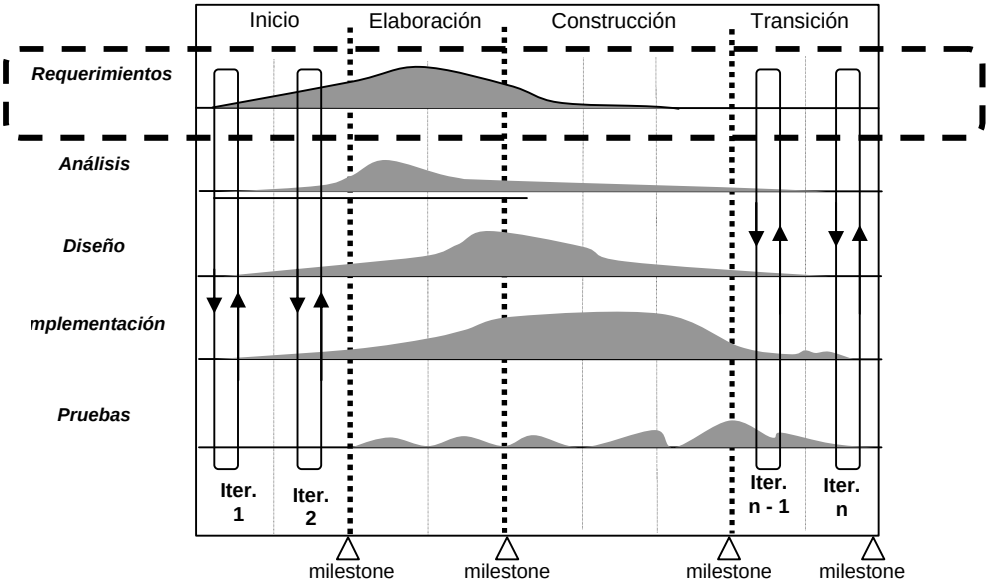


Figura 3-10: Flujo de Trabajo de Requerimientos en el ciclo de vida del Proceso Unificado

Durante la Fase de Inicio, los Analistas identifican la mayoría de los Casos de Uso para delimitar el sistema y el alcance del proyecto y para detallar algunos de ellos (menos del 10 por ciento). Durante la Fase de Elaboración los Analistas capturan la mayoría de los requerimientos restantes para que los desarrolladores puedan estimar el tamaño del esfuerzo de desarrollo que se requerirá. El objetivo es capturar más del 80 por ciento de los requerimientos y haber descrito la mayoría de los Casos de Uso al final de esta fase. Los requerimientos restantes se capturan (e implementan) en la Fase de Construcción. Casi no hay captura de requerimientos en la Fase de Transición, a menos que haya requerimientos que cambien.

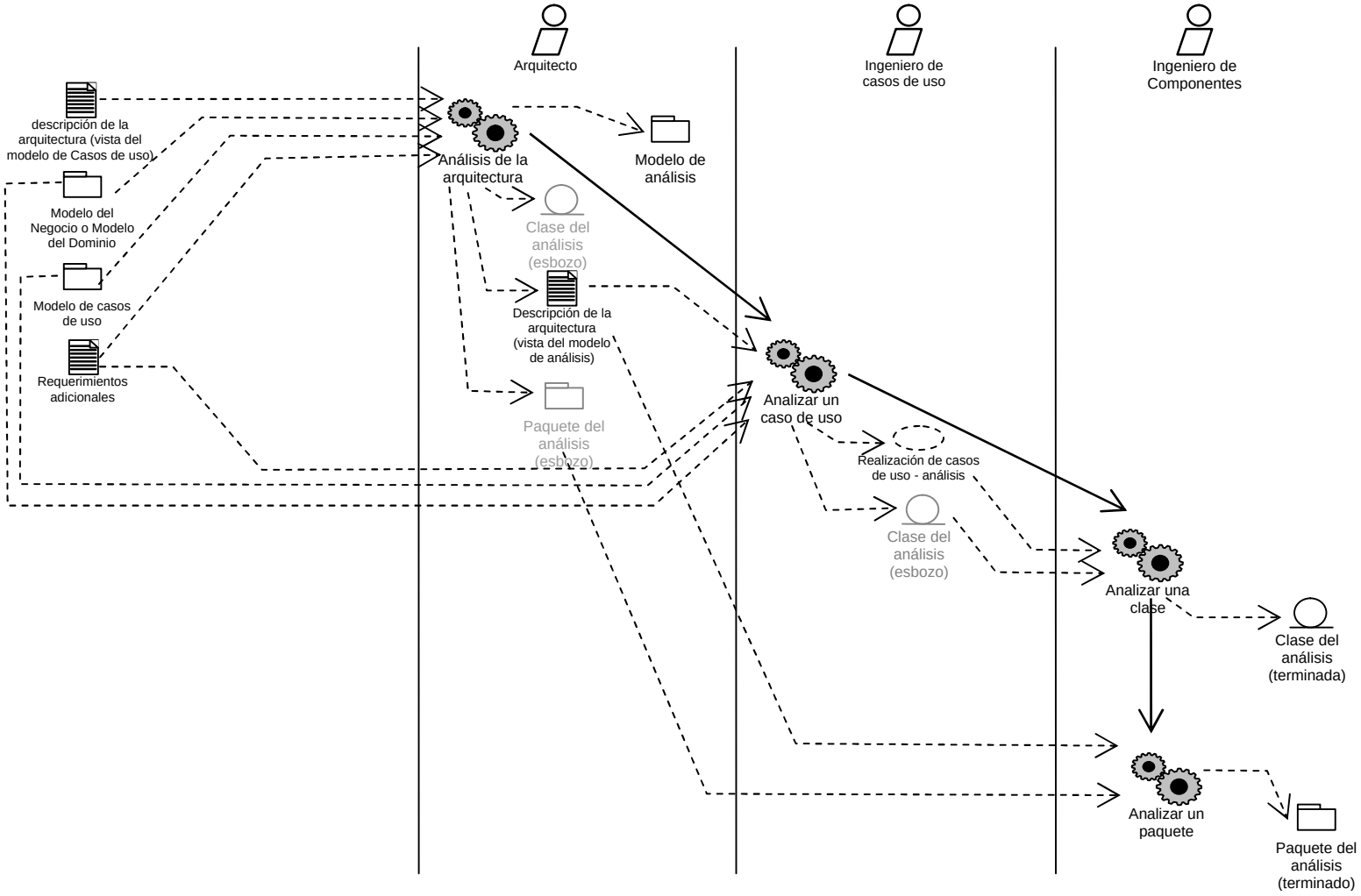


Figura 3-11: Flujo de Trabajo de Análisis

3.4.3. Flujo de Trabajo de Análisis

En el Flujo de Trabajo de Análisis (mostrado en la Figura 3-11) participan los roles Arquitecto, Ingeniero de Casos de Uso e Ingeniero de Componentes.

Los artefactos que se generan forman al Modelo de Análisis y son los siguientes: el Análisis de Casos de Uso, la Realización de Casos de Uso, las Clases del Análisis y Descripción de la arquitectura (vista del Modelo de Análisis).

El Modelo del Análisis se integra de acuerdo al siguiente diagrama:

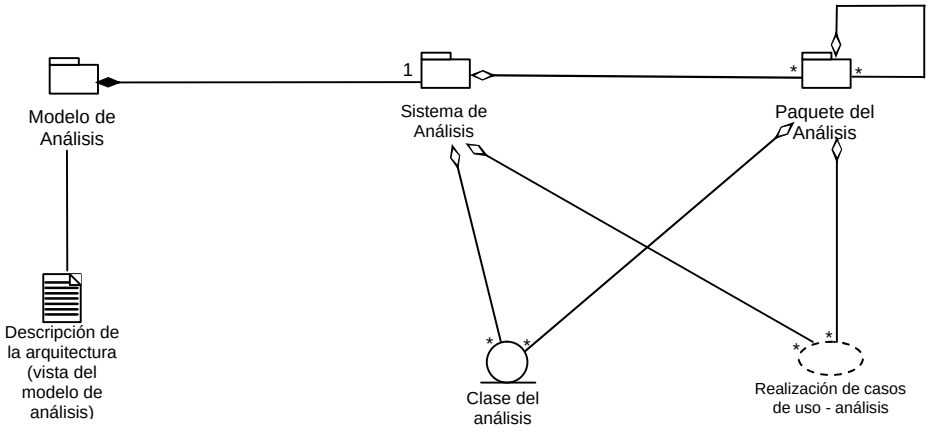


Figura 3-12: Integración del Modelo de Análisis

El Modelo de Análisis se representa mediante un Sistema de Análisis que denota el paquete de más alto nivel del modelo, unidos por medio de una Relación de Composición. La utilización de otros Paquetes de Análisis es por tanto una forma de organizar el Modelo de Análisis en partes mas manejables que representan abstracciones de subsistemas y posiblemente capas completas del diseño del sistema, estos otros posibles Paquetes del Análisis se incluirán dentro del Sistema de Análisis o incluso dentro de otro Paquete de Análisis por medio de una Relación de Agregación. Dentro del Modelo de Análisis, los Casos de Uso se describen mediante Clases de Análisis y sus objetos. Esto se representa mediante colaboraciones dentro del Modelo de Análisis que se llaman Realizaciones de Casos de Uso-Análisis.

El Arquitecto comienza la creación del Modelo de Análisis, identificando los Paquetes de Análisis principales, las Clases de Entidad evidentes y los requerimientos comunes. Después el Ingeniero de Casos de Uso realiza cada Caso de Uso en términos de las Clases de Análisis participante exponiendo los requerimientos de comportamiento de cada clase. El Ingeniero de Componentes especifica posteriormente estos requerimientos y los integra dentro de cada clase creando responsabilidades, atributos y relaciones consistentes para cada clase. Durante el Flujo de Trabajo de Análisis el Arquitecto identificará de manera continua nuevos Paquetes del Análisis, Clases y Requerimientos Comunes a medida que el Modelo de Análisis evoluciona, y el Ingeniero de componente, responsable de los Paquetes de Análisis concretos, continuamente los refina y mantiene. [Jacobson et al., 2000]

Las iteraciones iniciales de la Fase de Elaboración se centran en el Flujo de Trabajo de Análisis, lo cual contribuye a tener una Arquitectura estable y sólida y facilita una comprensión en profundidad de los requerimientos. Mas tarde, al término de la Fase de Elaboración y durante la Fase de Construcción, cuando la arquitectura es estable y se comprenden los requerimientos, el énfasis pasa en cambio al Flujo de Trabajo de Diseño y al Flujo de Trabajo de Implementación. En resumen, el énfasis en el Flujo de Trabajo de Análisis varía de acuerdo a la Figura 3-13.

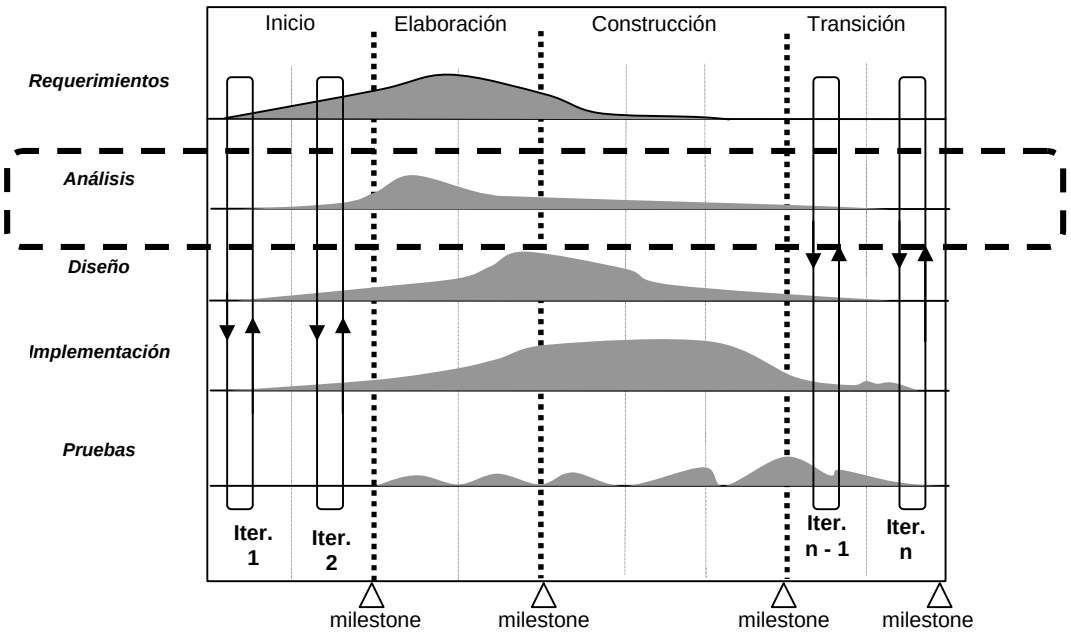


Figura 3-13: Flujo de Trabajo de Análisis en el ciclo de vida del Proceso Unificado

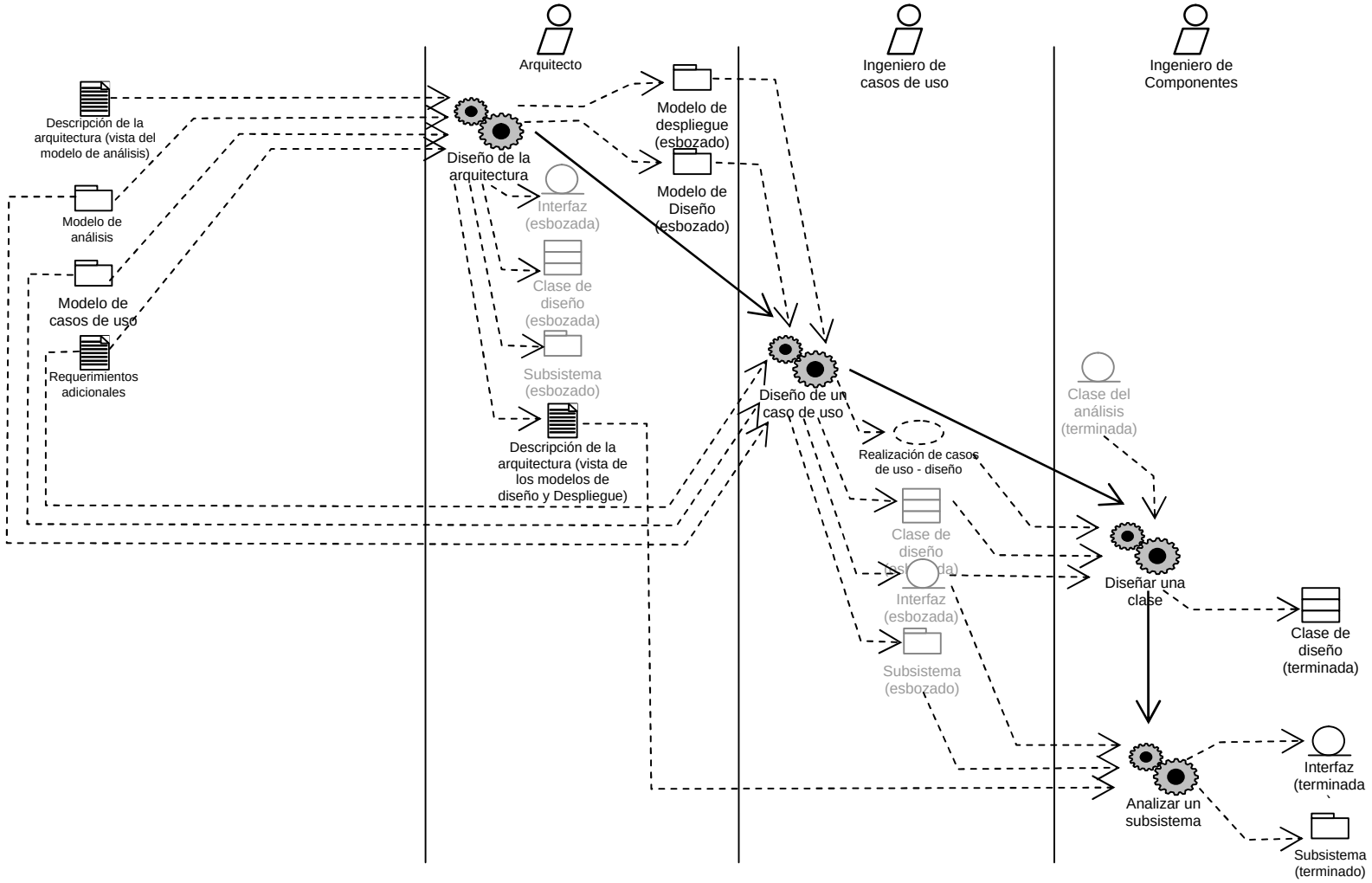


Figura 3-14: Flujo de Trabajo de Diseño

3.4.4. Flujo de Trabajo de Diseño

En el Flujo de Trabajo de Diseño participan los roles Arquitecto, Ingeniero de Casos de Uso e Ingeniero de Componentes.

Los artefactos que se generan son parte del Modelo de Diseño y parte del Modelo de Despliegue, y son los siguientes: realización de Casos de Uso-Diseño, Clases del Diseño, Interfaces, Subsistemas del Diseño y Descripción de la Arquitectura (Vista de los Modelos de Diseño y Despliegue).

De esta forma el Modelo de Diseño se integra de acuerdo a la Figura 3-15.

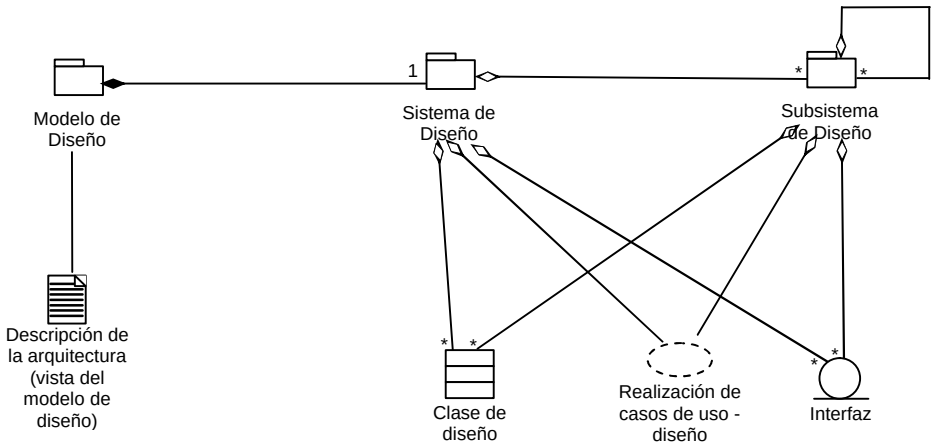


Figura 3-15: Integración del Modelo de Diseño

El Modelo de Diseño se representa por un Sistema de Diseño que denota al subsistema de nivel más alto del diseño. La utilización de otros subsistemas es, entonces, una forma de organización del Modelo de Diseño en porciones más manejables. Los Subsistemas de Diseño y Clases de Diseño representan abstracciones del subsistema y componentes de la implementación del sistema. Estas abstracciones son directas, y representan una sencilla correspondencia entre el diseño y la implementación. En el Modelo de Diseño, los Casos de Uso son realizados por las Clases de Diseño y las Interfases relacionadas, y denotan realizaciones de Casos de Uso-Diseño. [Jacobson et al., 2000]

El Modelo de Despliegue (que también es elaborado en este flujo de trabajo) se integra de acuerdo al siguiente diagrama:

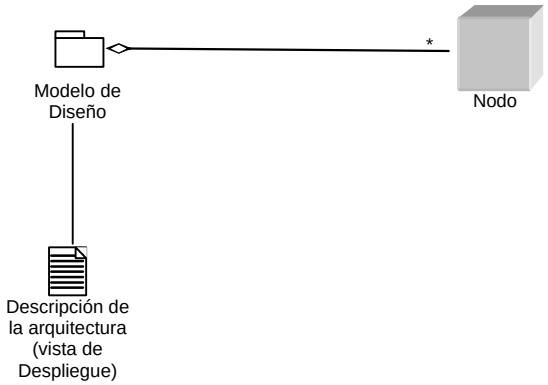


Figura 3-16: Integración del Modelo de Despliegue

El Modelo de Despliegue es un modelo de objetos (y no de clases especiales o abstracciones) que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo. El Modelo de Despliegue se utiliza como entrada fundamental en las actividades de diseño e implementación debido a que la distribución del sistema tiene una influencia principal en su diseño. En el Modelo de Despliegue cada nodo representa un recurso de cómputo, normalmente con procesador o un dispositivo de hardware similar.

El Arquitecto inicia la creación de los Modelos de Diseño y de Despliegue. Ellos esbozan los Nodos del Modelo de Despliegue, los Subsistemas principales y sus interfaces, las Clases de Diseño importantes como las Clases Activas, y los Mecanismos Genéricos de Diseño[1] del Modelo de Diseño. Después el Ingeniero de Casos de Uso realiza cada Caso de Uso en términos de Clases y/o Subsistemas de Diseño participantes y sus Interfaces. Las realizaciones de Casos de Uso resultantes establecen los requisitos de comportamiento para cada clase o subsistema que participe en alguna realización de Caso de Uso. Los Ingenieros de Componentes especifican a continuación los requerimientos, y los integran dentro de cada clase, bien mediante la creación de operaciones, atributos y relaciones consistentes sobre cada clase, o bien mediante la creación de operaciones consistentes en cada interfaz que proporcione el subsistema. A lo largo del Flujo de Trabajo de Diseño, los desarrolladores identificarán, a medida que evolucione el diseño, nuevos candidatos para ser Subsistemas, Interfaces, Clases y Mecanismos Genéricos de Diseño, y los Ingenieros de Componentes responsables de los subsistemas individuales deberán refinarlos y mantenerlos. [Jacobson et al., 2000]

El Diseño es importantísimo desde el final de la Fase de Elaboración y durante las iteraciones de la Fase de Construcción. Esto propicia tener una Arquitectura estable y sólida incluso antes de iniciar la construcción real de la mayor parte del sistema. Pero es importante continuar actualizando y manteniendo el Modelo del Diseño a lo largo de todo el ciclo de vida del software, pero con un grado menor de atención. Esto se ilustra en la Figura 3-17:

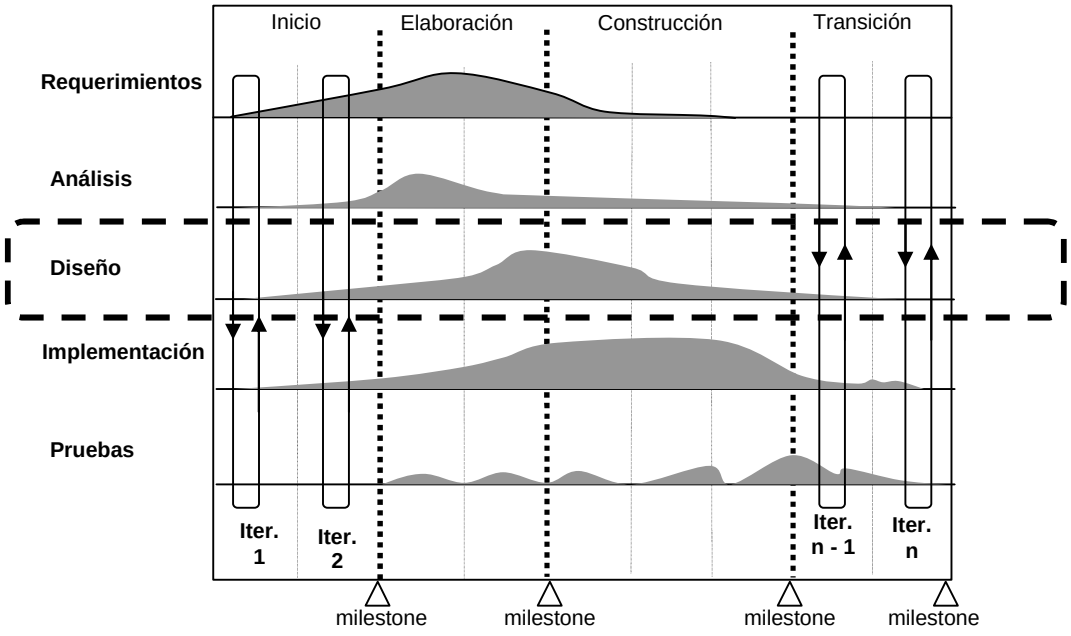


Figura 3-17: Flujo de Trabajo de Diseño en el ciclo de vida del Proceso Unificado

[1] Mecanismos Genéricos de Diseño: Conjunto de Clases del Diseño, Colaboraciones e incluso Subsistemas del Modelo de Diseño que llevan acabo requisitos comunes, como requisitos de persistencia, distribución y funcionamiento, tomando en cuenta las tecnologías de diseño e implementación disponibles. Los Mecanismos Genéricos del Diseño darán solución a los aspectos más difíciles del diseño.

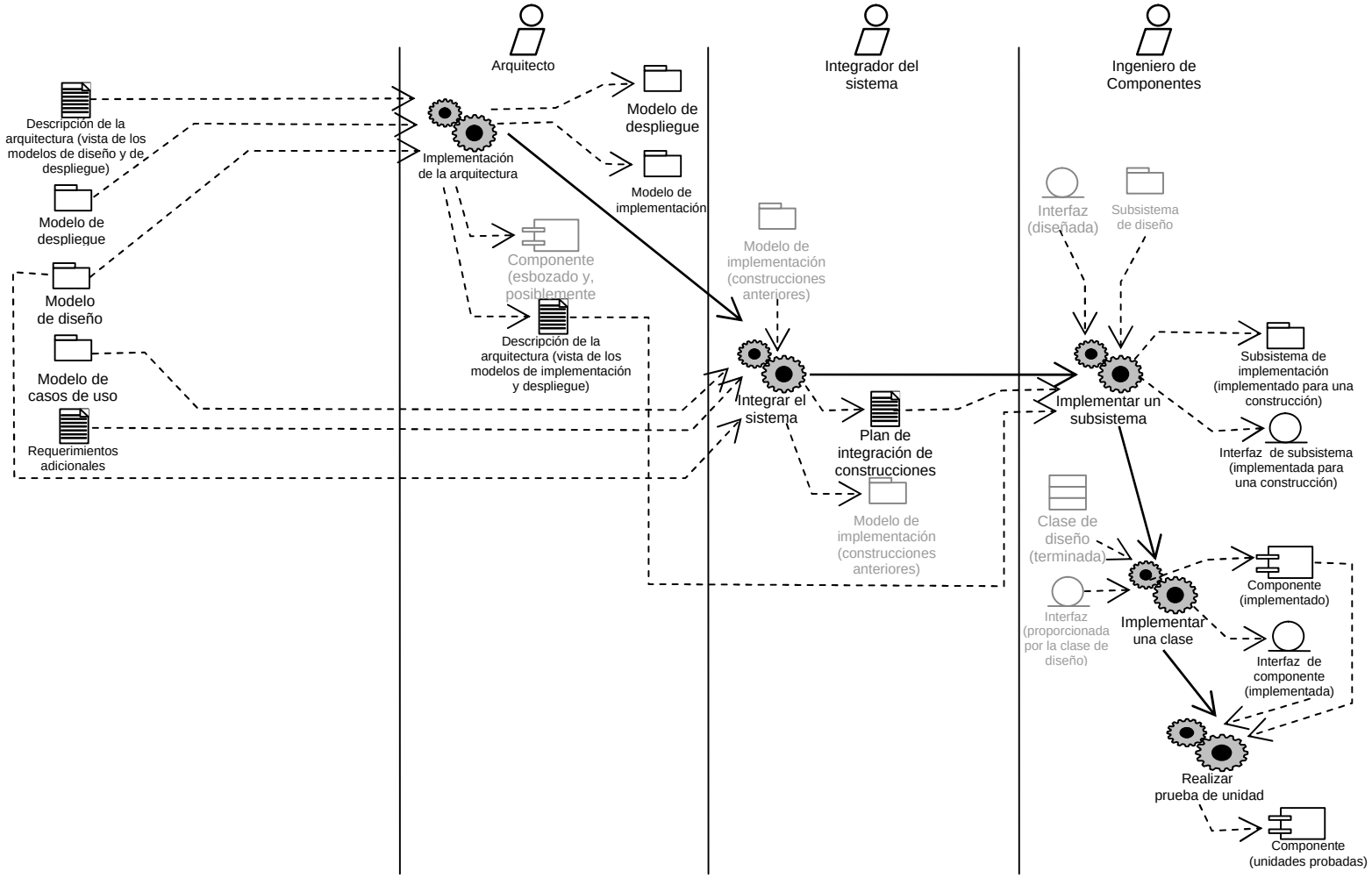


Figura 3-18: Flujo de Trabajo de Implementación

3.4.5. Flujo de Trabajo de Implementación

En el Flujo de Trabajo de Implementación participan los roles Arquitecto, Integrador del Sistema e Ingeniero de Componentes.

Los artefactos que se generan son parte del Modelo de Implementación y parte del Modelo de Despliegue, y son los siguientes: Subsistemas de Implementación, Interfaces, Componentes y Descripción de la Arquitectura (Vista de los Modelos de Implementación y Despliegue).

De esta forma el Modelo de Implementación se integra de acuerdo al siguiente diagrama.

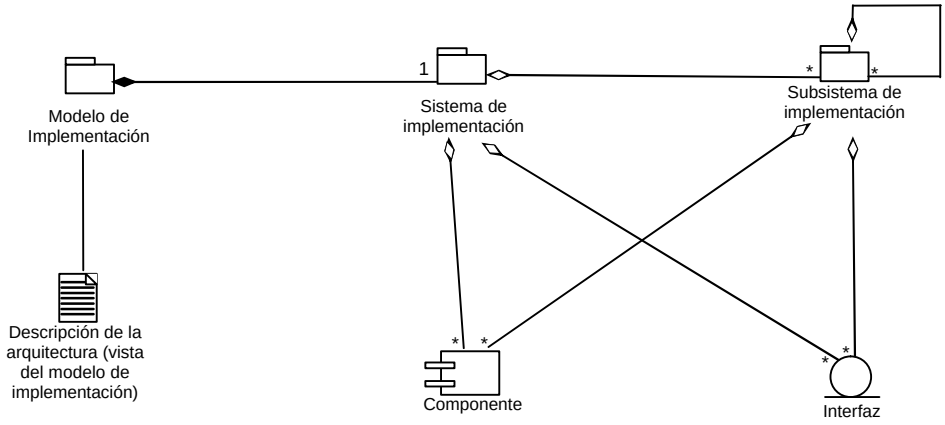


Figura 3-19: Integración del Modelo de Implementación

El modelo de Implementación describe cómo los elementos del Modelo de Diseño, como las Clases, se implementan en términos de Componentes, como ficheros de código fuente, ejecutables, etc. El Modelo de Implementación describe también como se organizan los Componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación y en el lenguaje o lenguajes de programación utilizados, y cómo dependen los componentes unos de otros. [Jacobson et al., 2000]

En el caso del Modelo de Implementación, también se define en forma de jerarquías de subsistemas, donde la jerarquía de más alto nivel corresponde a un Sistema de Implementación que contiene otros subsistemas de jerarquía menor que ayudan a organizar el modelo en partes más manejables.

El Modelo de Despliegue se continúa trabajado en este flujo, por lo cual su estructura es la misma de la sección anterior.

El Arquitecto inicia esbozando los Componentes claves en el Modelo de Implementación. A continuación, el Integrador del Sistema planea las integraciones del sistema necesarias en la presente iteración como una secuencia de Construcciones. Para cada Construcción, el Integrador del Sistema describe la funcionalidad que debería ser implementada y qué partes del Modelo de Implementación (es decir Subsistemas y Componentes) se verán afectadas. Los requerimientos sobre los Subsistemas y Componentes en la Construcción son entonces implementados por Ingenieros de Componentes. Los Componentes resultantes son probados y pasados al Integrador del Sistema para su integración. Entonces, las actividades entran en el Flujo de Trabajo de Pruebas, donde el Integrador del Sistema integra los nuevos Componentes en una Construcción y la pasa a los Ingenieros de Pruebas de Integración para llevar a cabo pruebas de integración. A continuación, lo desarrolladores inician la implementación de la siguiente Construcción, tomando en consideración los defectos de la Construcción anterior. [Jacobson et al., 2000]

Se inicia a trabajar en el Flujo de Trabajo de Implementación desde la Fase de Elaboración, para construir la Línea Base ejecutable de la Arquitectura, pero en las iteraciones de la Fase de Construcción el Flujo de Trabajo de Implementación alcanza los niveles mas importantes de atención para construir el grueso de los requerimientos del sistema, y ya en la Fase de Transición se le da una atención menor y solo la necesaria para atender defectos detectados de forma tardía. Lo anterior se muestra en la Figura 3-20.

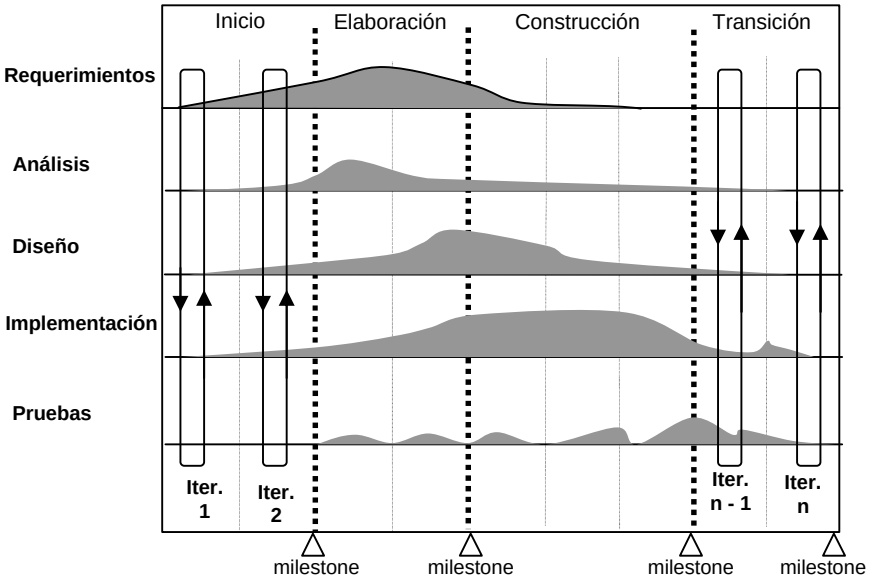


Figura 3-20: Flujo de Trabajo de Implementación en el ciclo de vida del Proceso Unificado

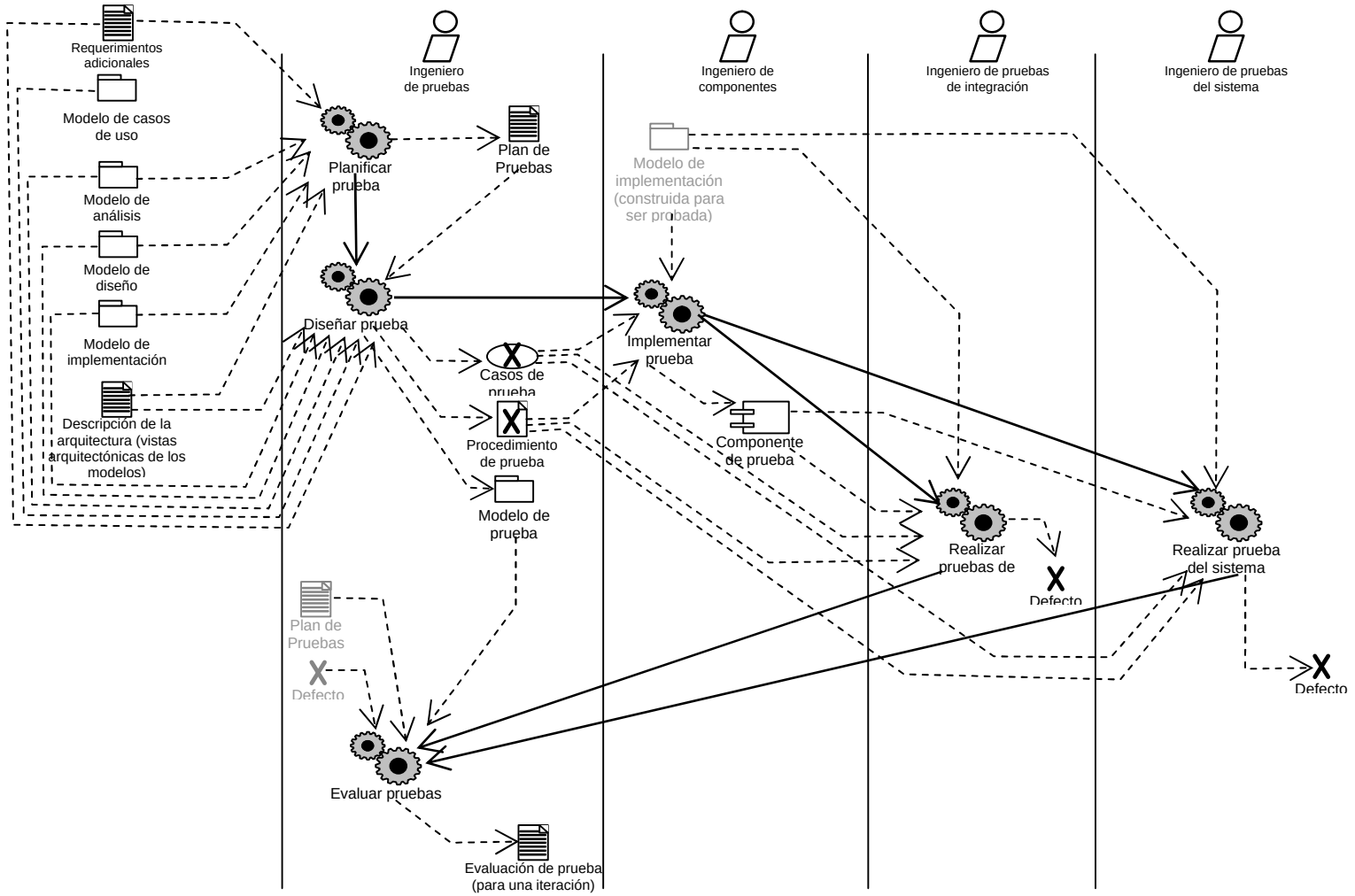


Figura 3-21: Flujo de Trabajo de Pruebas

3.4.6. Flujo de Trabajo de Pruebas

En el Flujo de Trabajo de Pruebas participan los roles Ingeniero de Pruebas, Ingeniero de Pruebas de Integración, Ingeniero de Pruebas del Sistema e Ingeniero de Componentes

Los artefactos que se generan son parte del Modelo de Pruebas, y son los siguientes: Plan de Pruebas, Casos de Pruebas, Procedimientos de Pruebas, Componentes de Pruebas, Defectos, Evaluaciones de Pruebas y Descripción de la Arquitectura (Vista del Modelo de Pruebas).

De esta forma el Modelo de Pruebas se integra de acuerdo al siguiente diagrama.

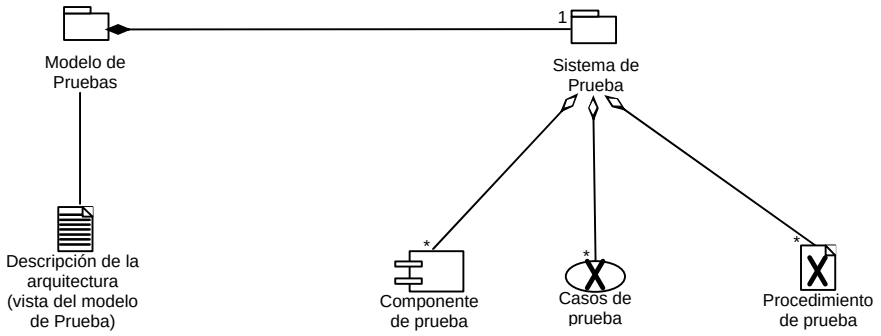


Figura 3-22: Integración del Modelo de Pruebas

Si el Modelo de Pruebas es grande, se pueden utilizar Paquetes en el modelo para manejar su tamaño. El Modelo de Pruebas describe principalmente cómo se prueban los Componentes ejecutables (como las Construcciones) en el Modelo de Implementación con Pruebas de Integración y Pruebas de Sistema. El Modelo de Pruebas puede describir también cómo han de ser probados aspectos específicos del sistema. Así, el Modelo de Pruebas es una colección de Casos de Pruebas, Procedimientos de Prueba y Componentes de Prueba.

El Ingeniero de Prueba inicia planificando el esfuerzo de prueba de cada iteración y describe entonces los Casos de Pruebas necesarios, y los Procedimientos de Pruebas correspondientes para llevar acabo las pruebas. Si es posible, los Ingenieros de Componentes crean a continuación los Componentes de Prueba para automatizar algunos de los Procedimientos de Prueba. Todo esto se hace para cada Construcción entregada como resultado del Flujo de Trabajo de Implementación. [Jacobson et al., 2000]

Durante la Fase de Inicio puede hacerse parte de la planificación inicial de las pruebas cuando se define el ámbito del sistema. Sin embargo, las pruebas se llevan a cabo sobre todo cuando una Construcción (como un resultado de implementación) es sometida a pruebas de integración y de sistema. Esto quiere decir que la realización de pruebas se centra en la Fase de Elaboración, cuando se prueba la Línea Base ejecutable de la Arquitectura, y en la Fase de Construcción, cuando el grueso del sistema esta implementado. En la Fase de Transición el objetivo de las pruebas se desplaza hacia la corrección de defectos durante los primeros usos y a las pruebas de regresión. [Jacobson et al., 2000]. Lo anterior se muestra en la Figura 3-23:

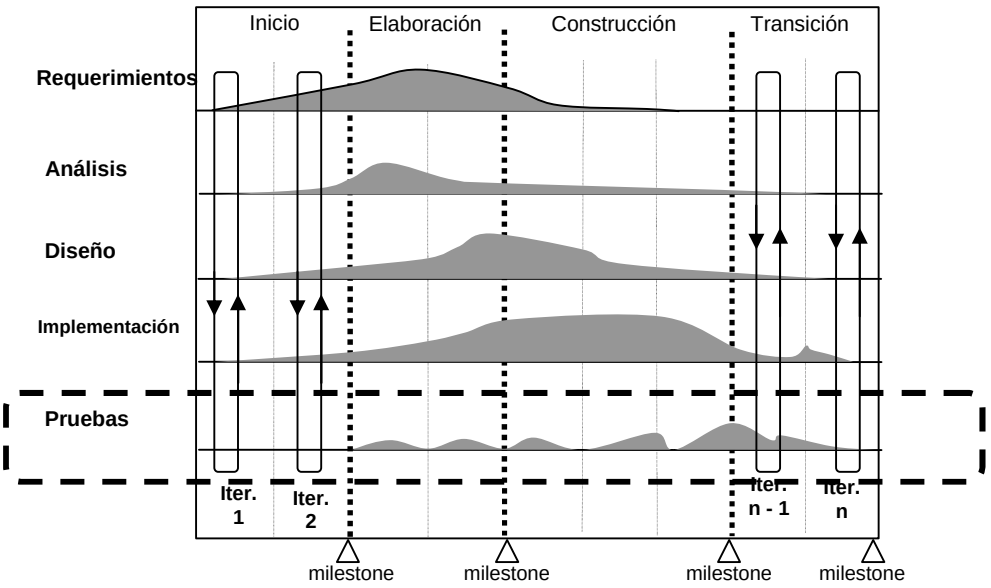


Figura 3-23: Flujo de Trabajo de Pruebas en el ciclo de vida del Proceso Unificado

3.5. Fases del Proceso Unificado

A continuación se expondrá con mas detalle cada una de las cuatro fases del Proceso Unificado, cómo se desarrollan los flujos de trabajo en cada una de ellas, cuales son los requerimientos antes de arrancar las fases, cuales son los productos resultantes de cada fase y otros detalles muy particulares para cada una de ellas.

3.5.1. Fase de Inicio

La Fase de Inicio es la primera de las cuatro Fases del Proceso Unificado, aunque las actividades especificadas en ella parecieran ser las iniciales en el desarrollo del proyecto, existen muchas otras actividades anteriores, que promueven la consolidación del proyecto. Estas actividades anteriores a la Fase de Inicio tienen mucho que ver con el negocio y los procesos del mismo.

La Fase de Inicio también puede tomar formas muy diversas dependiendo de la naturaleza del producto que se desee crear (un sistema grande, un sistema pequeño, un sistema totalmente nuevo, actualizar un sistema existente, etc.), del tipo de organización de desarrollo y de otros muchos aspectos que pueden hacer que la fase sea muy simple o demasiado elaborada.

3.5.1.1. Objetivo de la Fase de Inicio

El objetivo principal de la Fase de Inicio es:

- Desarrollar el análisis del negocio hasta el punto que se permita justificar la puesta en marcha del proyecto.

Además de este objetivo principal, la Fase de Inicio tiene otros objetivos secundarios, algunos de los cuales ayudan a obtener el objetivo principal

- Limitar el alcance del sistema propuesto
- Bosquejar una Arquitectura candidata
- Detectar la mayor cantidad posible de riesgos y así mismo mitigar la mayoría de ellos.

3.5.1.2. Antecedentes de la Fase de Inicio

Como se menciono en líneas arriba, antes de iniciar las actividades de la Fase de Inicio, se llevan a acabo una serie de actividades que dan marcha al proyecto. Estas actividades son:

- Detección de la necesidad de un producto. Esta detección puede tomar formas distintas, por ejemplo, para una empresa de software que desarrolla productos genéricos para venta general, la detección de la necesidad nace de un estudio de mercado completo y de la retroalimentación de sus principales clientes, mientras que para una empresa que se dedique a giros distintos al desarrollo de software, pero que cuente con un departamento de desarrollo de software para uso interno, la detección de la necesidad se realiza por ejecutivos y por lo tanto puede estar demasiado detallada en términos del negocio, pero muy escueta en términos de software; y finalmente para una empresa que se dedica al desarrollo de software a la medida de sus clientes, la detección de la necesidad es en la mayoría de las veces, realizada por los propios clientes y quizás muy poco detalla da en términos del negocio y de software, ante lo cual la Fase de Inicio crece en actividades.

- Desarrollar el Modelo del Dominio[1] y/o el Modelo del Negocio[1]. Dependiendo de la naturaleza del negocio y del producto a elaborar, se debe desarrollar el Modelo del

[1] El Modelo del Dominio captura los tipos más importantes de objetos en el contexto del sistema. Los objetos del dominio representan las “cosas” que existen o los eventos que suceden en el entorno en el que debe trabajar el sistema (totalmente externos al sistema). El Modelo del Domino se describe mediante diagramas de UML, especialmente Diagramas de Clases.

Dominio o el Modelo del Negocio por parte del promotor del proyecto (en algunas ocasiones será el Usuario final o Cliente y en otras será el propio Jefe del Proyecto).

- Planificación de la Fase de Inicio. Antes de comenzar la Fase de Inicio, se debe planificar las actividades que se vayan a elaborar en ella, para esto se debe reunir toda la información recopilada antes de que el proyecto comience, organizar esta información de forma que pueda ser utilizada y definir los objetivos de la Fase de Inicio. Una vez hecho esto, se debe desarrollar un plan provisional para alcanzar estos objetivos y poco a poco ir detallándolo y ajustándolo. También se debe establecer lo más pronto posible el número de iteraciones necesarias para esta fase.

- Ampliación de la Descripción de Requerimientos. Para realizar esto, se debe conformar un equipo de personas que representen los intereses involucrados (Usuarios, Clientes, Jefe del Proyecto, Desarrolladores, etc.) para convertir un simple bosquejo de requerimientos en un a descripción de requerimientos completa y formal, en la cual se muestren peticiones reales, mas que condensadas.

3.5.1.3 Flujos de Trabajo en la Fase de Inicio

A continuación se detalla como se desarrollan las actividades de los cinco flujos de trabajo en las iteraciones de la Fase de Inicio:

- Flujo de Trabajo de Requerimientos

  Este es el Flujo de Trabajo de mayor importancia en la Fase de Inicio, realizándose en él las siguientes actividades (las primeras cuatro actividades solo se realizan en Fase de Inicio).

  Enumerar los requerimientos candidatos: la lista de características (que puede provenir de la experiencia de usuarios o clientes, de un estudio de mercado, de desarrollo de versiones anteriores o de un producto similar) se convierten en requerimientos candidatos.

  Comprender el contexto del sistema: Si se cuenta con un Modelo del Negocio o del Dominio se debe empezar a trabajar en él para comprenderlo, en caso de no existir ninguno de los dos se debe promover su pronta creación.

  Recopilar los requerimientos funcionales: Se debe iniciar la formalización de la lista de requerimientos funcionales.

  Recopilar los requerimientos no funcionales: Además de la lista anterior, se debe trabajar en la lista de requerimientos no funcionales e iniciar la proyección de dichos requerimientos hacia Casos de Uso, al glosario de términos o a requerimientos adicionales en la construcción y distribución del sistema.

  Encontrar Actores y Casos de Uso: a partir de los primeros requerimientos detectados y del Modelo del Negocio (o del Modelo del Dominio) se debe elaborar una relación, lo mas completa posible de todos los casos de uso que conduzcan al Modelo de Casos de Uso completo. Para después, por parte del Jefe de Proyecto, del Analista y el Arquitecto,

En Dominios de Negocios muy pequeños, se puede sustituir el Modelo del Dominio por un simple Glosario de Términos. Se debe tener cuidado de no incluir una clase que sea exclusiva del dominio, en el análisis de clases internas del sistema.

[1] El Modelo del Negocio captura los tipos más importantes de objetos en el contexto del sistema y los procesos internos del negocio (pero externos al sistema), así como los Actores involucrados. Se desarrolla en dos paso: Primero se elabora un Modelo de Casos de Uso del Negocio (que identifica los Actores del negocio y los Casos de Uso del negocio), y después se elabora el Modelo de Objetos del Negocio, compuesto por roles, entidades del negocio y unidades del negocio que juntos realizan los Casos de Uso del Negocio, asociándoles las reglas del negocio y otras reglas impuestas por el negocio. El Modelo del Negocio se distingue del Modelo del Dominio en que el Modelo del Dominio solo se centra en las “cosas” , mientras que el Modelo del Negocio se centra en “cosas” y procesos en que intervienen.

clasificar el subconjunto de Casos de Uso necesarios para llevar a cabo el trabajo de esta fase, con el fin de limitar el trabajo en ella.

Determinar la prioridad de los Casos de Usos: Se crea el plan del proyecto o plan de iteraciones a partir de asignar prioridades a los Casos de Uso que conduzcan a los objetivos de esta fase.

Detallar Casos de Uso: Una vez elaborada la lista de Casos de Uso que componen el Modelo de Casos de Uso, se deberá detallar cerca del 10 por ciento de ellos, es decir los Casos de Uso que permitirán cumplir los objetivos de esta fase.

Construir un prototipo de interfaz de usuario: Durante esta fase, esta actividad no es de interés.

Estructurar el Modelo de Casos de Uso: Durante esta fase esta actividad no es de interés.

- Flujo de Trabajo de Análisis

  El objetivo del Flujo de Trabajo de Análisis es, precisamente, analizar los requerimientos, refinarlos y estructurarlos como un modelo de objetos que sirva como una aproximación al Modelo de Diseño. En esta fase se construirá una parte del Modelo del Análisis (cerca de un 5 por ciento) que ayudará a detallar mejor los Casos de Uso mas importantes y a establecer la Arquitectura candidata (la Línea Base de la Arquitectura se deja hasta la Fase de Elaboración). Para el Flujo de Trabajo de Análisis en la Fase de Inicio se tienen las siguientes actividades.

  Analizar la Arquitectura: A partir del subconjunto de Casos de Uso que nos conducen a los objetivos de esta fase, se crea una versión previa del Modelo de Análisis de forma no necesariamente exhaustiva. Debido a que esta primera versión del Modelo de Análisis puede no utilizarse, no se construya sobre ella el Modelo de Análisis completo en las fases siguientes, y muy probablemente termine por descartarse completamente, solo tiene importancia como guía para la Arquitectura.

  Analizar un Caso de Uso: Para esta actividad se puede analizar y detallar solo cerca del 5 por ciento de los Casos de Uso importantes por medio de los Diagramas de Interacción y de clases pertinentes, que nos permitan detectar recursos que se comparten entre los Casos de Uso (por ejemplo, bases de datos y archivo).

  Analizar una Clase: Durante esta fase, esta actividad no es de interés.

  Analizar un Paquete: Durante esta fase esta actividad no es de interés.

- Flujo de Trabajo de Diseño

  Una vez que se tiene la Arquitectura Candidata, el Flujo de Trabajo de Diseño se encarga de esbozar el Modelo de Diseño de ella e incluirlo en su descripción. En esta fase las actividades para el Flujo de Trabajo de Diseño son:

  *Diseñar la Arquitectura: S*e desarrolla un esbozo inicial del Modelo de Diseño donde se representan los Casos de Uso, que fueron elegidos en el Flujo de Trabajo de Requerimientos, en términos de Colaboraciones entre Subsistemas y Clases, se definen estas colaboraciones y se identifican las Interfaces relacionadas a ellas. También se deben identificar los Mecanismos Genéricos de Diseño[1] a utilizar para cubrir los requerimientos

[1] Mecanismos Genéricos de Diseño: Conjunto de Clases del Diseño, Colaboraciones e incluso Subsistemas del Modelo de Diseño que llevan acabo requisitos comunes, como requisitos de persistencia, distribución y funcionamiento, tomando en

funcionales y no funcionales. Así mismo se crea una versión limitada del Modelo de Despliegue enfocándose en los nodos y conexiones críticos.

Diseñar un caso de uso: En caso de requerirse esta actividad, deberá desarrollarse de forma mínima.

Diseñar una Clase: En caso de requerirse esta actividad, deberá desarrollarse de forma mínima.

Diseñar un Subsistema: En caso de requerirse esta actividad, deberá desarrollarse de forma mínima.

- Flujo de Trabajo de Implementación

  En esta fase se deberá parar la ejecución de los flujos de trabajo tan pronto como se tenga la descripción de una Arquitectura que parezca funcionar, pero en ocasiones para demostrar que puede funcionar se requiere un prototipo ejecutable, y por lo tanto actividades del Flujo de Trabajo de Implementación, esto sucede de acuerdo al criterio del Jefe de Proyecto, y si es así, deberán realizarse muy pocas actividades de este flujo de trabajo.

- Flujo de Trabajo de Pruebas

  A medida que se realizan los trabajos de análisis, diseño e implementación, se pueden vislumbrar cuáles pruebas serán necesarias y así realizar planes provisionales de pruebas. Sin embargo, el Trabajo general de este flujo será mínimo.

3.5.1.4 Productos de la Fase de Inicio

Al terminar la Fase de Inicio se tendrán los siguientes productos de trabajo:

- Lista de Características
- Primer versión del Modelo del Negocio (o del Dominio)
- Esbozos de las primeras versiones de los Modelos de Requerimientos, Análisis y Diseño. Opcionalmente se puede contar con bases para los Modelos de Implementación y/o Pruebas
- Primer esquema de la Descripción de la Arquitectura candidata, incluyendo las vistas de los esbozos de los modelos del punto anterior
- Opcionalmente un prototipo exploratorio que muestre el funcionamiento del sistema
- Lista inicial de riesgos
- Clasificación de Casos de Uso
- Base del Plan General del Proyecto
- Primer borrador de la estimación de beneficios, del reconocimiento del mercado, y las estimaciones del proyecto

cuenta las tecnologías de diseño e implementación disponibles. Los Mecanismos Genéricos del Diseño darán solución a los aspectos más difíciles del diseño.

3.5.2. Fase de Elaboración

Durante la Fase de Elaboración se trabajará teniendo el antecedente de lo realizado en la Fase de Inicio, aunque no se va a reutilizar todo, ni tomar como base para el trabajo que ahora se va a realizar en esta fase, pues parte de lo que se realizó en Fase de Inicio fue solo para demostrar que se podría construir un producto en las fases posteriores, por ejemplo la Arquitectura candidata fue solo eso un "candidato" para lo que se iba a realizar posteriormente.

En la Fase de Elaboración se va a trabajar de manera un tanto general con el sistema en su totalidad, sin profundizar del todo en los aspectos técnicos de partes específicas, sobre todo si estas partes no son críticas en el sistema.

Además en esta fase se describirá detalladamente entre el 40 y 80 por ciento de los Casos de Uso identificados. No es necesario identificar el cien por ciento de los Casos de Uso, ni describir todos los Casos de Uso identificados, solo aquellos que nos lleven a los objetivos de esta fase.

3.5.2.1. Objetivo de la Fase de Elaboración

El objetivo principal de la Fase de Elaboración es:

- Establecer la Línea Base de la Arquitectura

Además de este objetivo principal, la Fase de Elaboración tiene otros objetivos secundarios, algunos de los cuales ayudan a obtener el objetivo principal

- Recopilar la mayor parte de los requerimientos que aun queden pendientes
- Continuar la observación y control de los riesgos críticos que aun queden
- Completar los detalles del Plan General del Proyecto

3.5.2.2. Antecedentes de la Fase de Elaboración

En la Fase de Inicio se ha formulado la Arquitectura candidata, se han identificado los riesgos críticos estudiándolos y minimizándolos, y se ha realizado el análisis del negocio con detalle.

La Fase de Inicio es una especie de convencimiento para con todas las personas involucradas, sobre todo para aquellas que financiarán el proyecto. Por tanto, el tiempo que transcurre entre el fin de la Fase de Inicio y el comienzo de la Fase de Elaboración varia de acuerdo al tiempo que se tarde en que se dé el visto bueno para continuar (si es el caso de que se autorice el proyecto), y es precisamente en este tiempo intermedio, e incluso al final de la Fase de Inicio que se tienen las siguientes actividades:

- Planificar la Fase de Elaboración. Al final de la Fase de Inicio se realiza la planificación de las iteraciones de esta fase, que nos llevaran a alcanzar los objetivos de la misma. Esta planificación puede no ser completa debido a la escasez de conocimientos sobre los recursos que se autoricen, por tanto la planificación es susceptible de detallarse al inicio de la Fase de Elaboración.

- Formación del equipo de trabajo. En compañías con demasiada dinámica entre los miembros de sus equipos de trabajo, es decir que estos equipos intercambien de miembros constantemente, puede ser que parte del equipo que trabajo en Fase de Inicio, o incluso todo, ya este asignado a otros proyectos cuando inicie la Fase de Elaboración, por tanto se debe tratar de mantener la mayor parte del equipo de trabajo original y considerar que se tendrán mas actividades en esta fase, para solicitar nuevo personal que se agregue al proyecto.

3.5.2.3. Flujos de Trabajo en la Fase de Elaboración

A continuación se detalla cómo se desarrollan las actividades de los cinco flujos de trabajo en las iteraciones de la Fase de Elaboración:

- Flujo de Trabajo de Requerimientos

  Este flujo de trabajo aun tiene una gran importancia en la Fase de Elaboración, realizándose en él las siguientes actividades:

  Encontrar Actores y Casos de Uso: El Analista de Sistemas continuará encontrando Actores y Casos de Uso adicionales a los hallados en la fase anterior, pero se recomienda tratar de identificar la totalidad de ellos, con el fin de poder dimensionar el costo del proyecto y/o detectar todos los posibles riesgos de forma temprana.

  Construir un Prototipo de Interfaz de Usuario: En la Fase de Elaboración se realizará esta actividad solo en el caso extraordinario que una Interfaz de Usuario sea importante para la Arquitectura del sistema (por ejemplo, si son importantes para el rendimiento y tiempo de respuesta). Otro motivo para trabajar en esta actividad durante la Fase de Elaboración, es el demostrar el funcionamiento de las interfaces de usuario en caso que no se haya demostrado totalmente la valía del sistema en Fase de Inicio. En general no se realiza esta actividad en Fase de Elaboración.

  Determinar la prioridad de los Casos de Usos: A medida que se siguen encontrando nuevos Casos de Uso y se va estructurando la Línea Base de la Arquitectura, es importante continuar la actividad de priorizar los Casos de Uso para coordinar estas dos actividades.

  Detallar Casos de Uso: En esta fase los esfuerzos se limitarán para realizar descripciones preliminares de Casos de Uso completos y arquitectónicamente importantes. Por lo general, no se detallarán en su totalidad los Casos de Uso seleccionados, solo se limitará a detallar los que se necesitan para esta fase (entre el 40 y 80 por ciento).

  Estructurar el Modelo de Casos de Uso: El Analista de Sistemas, sobre la base de lo que ha hecho, buscará similitudes, simplificaciones y oportunidades para mejorar la estructura del Modelo de Casos de Uso.

- Flujo de Trabajo de Análisis

  El trabajo en este flujo es de suma importancia para la fase, pues como ya se dijo, su objetivo principal es empezar a establecer la Línea Base de la Arquitectura, para lo cual el análisis de los Casos de Uso importantes es primordial, aunque para ello solo se tenga que analizar cerca del 50 por ciento de los Casos de Uso que se han descrito en detalle. Las actividades de este flujo en la Fase de Elaboración son:

  Analizar la Arquitectura: A diferencia de lo que se realizó en la Fase de Inicio (solo se analizó la arquitectura hasta demostrar que era factible desarrollarla), en esta fase se analiza la Arquitectura hasta establecer los inicios de la Línea Base. Para lo anterior, se realiza una partición de alto nivel del sistema, descomponiéndolo en Paquetes (Paquetes Específicos de la aplicación, Paquetes Generales y Paquetes de Servicio) e identificando Clases del Análisis. También se identificarán Colaboraciones genéricas y mecanismos genéricos para todos los Casos de Uso.

  Analizar un Caso de Uso: Solo los Casos de Uso interesantes para la Arquitectura y aquellos de los cuales se debe entender su comportamiento, se van refinar en función de las Clases del Análisis obvias detectadas hasta este momento por el Arquitecto, pero solo

hasta el punto que conduzcan al establecimiento de la Línea Base de la Arquitectura. Considerando que se ha identificado cerca del 80 por ciento de los Casos de Uso y que de estos solo se detallaron entre el 40 y 80 por ciento, el análisis se limitará a un conjunto menor de Casos de Uso. Una vez que los Casos de Uso se han representado por Colaboraciones de Clases del Análisis, los Ingenieros de Casos de Uso, iniciarán la identificación de más Clases del Análisis con base a las que ya han sido detectadas por el Arquitecto, asignándoles responsabilidades y atributos, y mostrando las relaciones entre ellas.

Analizar una Clase: Todo lo elaborado anteriormente se toma en consideración por los Ingenieros de Componentes para refinar las clases ya identificadas, mezclando las responsabilidades que les han sido asignadas a estas clases desde distintos Casos de Uso. También tomarán en consideración los mecanismos genéricos identificados por el Arquitecto para averiguar cómo pueden ser utilizados por cada clase.

Analizar un Paquete: De los Paquetes identificados y esbozados por el Arquitecto, los Ingenieros de Componentes tomaran la responsabilidad de refinarlos y mantenerlos.

- Flujo de Trabajo de Diseño

  En esta fase se diseñará desde un punto de vista de la Arquitectura, trabajando solo en el diseño de Casos de Uso-Diseño, Subsistemas y Clases del Diseño arquitectónicamente importantes (sobre todo en los Subsistemas que vengan de trazas de los Paquetes de Análisis de esta fase). Las actividades de este flujo en la Fase de Elaboración son:

  Diseñar Arquitectura: Para el Diseño de los aspectos arquitectónicamente significativos, el Arquitecto realizará los siguientes sub-actividades:

  - Identificar la Arquitectura en capas: El Arquitecto diseña la Arquitectura en capas considerando las capas de software del sistema y la capa intermedia[1], y selecciona los productos[2] que se van a utilizar finalmente para implementar los mecanismos de diseño.

  - Identificar los Subsistemas y sus Interfaces: Partiendo desde las capas mas altas del sistema (capas de nivel de aplicación) diseñará en la medida de lo posible el Subsistema correspondiente de cada uno de los Paquetes de Análisis, considerando los sistemas heredados y marcos de trabajo que cubran uno o mas Subsistemas de Diseño.

  - Identificar las Clases de Diseño significativas para la Arquitectura: Debe diseñar las Clases de Diseño correspondientes a las Clases de Análisis que sean significativas para la Arquitectura, y muy probablemente identificara y diseñara nuevas Clases de Diseño también relacionadas muy estrechamente con la Arquitectura (posiblemente Clases Activas).

  - Identificar Nodos y configuración de red: solo se realizará esto en el caso que se trabaje con un sistema distribuido, considerando hilos, procesos necesarios, concurrencia y la red física para distribuir la funcionalidad y carga de procesamiento.

  - Actualizar la vista de la Arquitectura de los Modelos de Diseño y Despliegue.

[1] El Software del Sistema y la Capa Intermedia constituyen los cimientos de un sistema, ya que toda la funcionalidad descansa sobre software como sistemas operativos, sistemas de gestión de bases de datos, software de comunicaciones, tecnologías de distribución de objetos, kits de diseño de IGU, y tecnologías de gestión de transacciones. [Jacobson et al., 2000]

[2] Productos software que se compren o se construyan, o que incluso, se hereden.

Diseñar un Caso de Uso: El Ingeniero de Casos de Uso trabajará en las realizaciones de los Casos de Uso-Diseño para cada Caso de Uso significativo para la Arquitectura, esto se realizará en términos de Subsistemas de Diseño, Subsistemas de Servicios y Clases de Diseño; y limitados por los Mecanismos de Diseño. Como consecuencia de esto se podrán identificar y/o detallar las responsabilidades de las Clases de Diseño y Subsistemas de Diseño ya identificados.

Diseñar una Clase: Tomando las responsabilidades identificadas y detalladas en la actividad anterior, los ingenieros de Componentes diseñarán cada una de las Clases, pero aun cuando se termine esta fase, las Clases de Diseño pueden no estar terminadas aun, debido a que no se han diseñado la totalidad de los Casos de Uso donde pudiesen intervenir.

Diseñar un Subsistema: Los Ingenieros de Componentes tomarán los Subsistemas identificados por el Arquitecto, para diseñarlos, retroalimentando al Arquitecto para que posiblemente actualice la Vista de la Arquitectura del Modelo de Diseño.

- Flujo de Trabajo de Implementación

  A partir de la Fase de Elaboración, este flujo de trabajo adquiere importancia, debido a que es en él, y precisamente durante esta fase, donde se implementa la Línea Base de la Arquitectura, a partir de la cual se construirá todo el sistema. Sus actividades son:

  Implementar la Arquitectura: Basándose en la vista de la Arquitectura del Modelo de Diseño y en la Vista de la Arquitectura del Modelo de Despliegue, el Arquitecto identifica los Componentes necesarios para implementar los Subsistemas de Servicio. Los Componentes ejecutables se asignan a los Nodos de red en los que van a ejecutarse. A continuación, el Arquitecto ilustrará esto en la Vista de la Arquitectura del Modelo de Implementación. [Jacobson et al., 2000]

  Implementación de una Clase, implementación de un Componente y realizar Prueba de Unidad: Los Ingenieros de Componentes implementan las Clase de Diseño en términos de Componentes, en donde es muy común implementar mas de una Clase de Diseño en un solo Componente y un Subsistema por medio de mas de un Componente. Conforme se implementan los Componentes, se les realizan Pruebas de Unidad con el fin de detectar errores de su funcionamiento como una unidad.

  Integrar el Sistema: El Integrador de Sistemas crea el plan de integraciones considerando el porcentaje del Sistema que se ha implementado, para después integrarlos como la Línea Base de la Arquitectura.

- Flujo de Trabajo de Prueba

  El objetivo de este flujo es probar la parte del Sistema implementada, ya no solo como unidades separadas, sino que ahora ya se considerará todo el conjunto de Componentes implementados, la interacción de sus partes y su funcionalidad, para lo cual se realizan las siguientes actividades:

  Planificar las Pruebas: Considerando los escenarios de los Casos de Uso y los requerimientos no funcionales, el Ingeniero de Pruebas seleccionará los objetivos que se evaluarán de la implementación de la Línea Base de la Arquitectura.

  Diseñar las Pruebas: El Ingeniero de Pruebas tomará los objetivos de prueba y establecerá los Casos de Prueba y Procedimientos de Prueba para las Integraciones de la implementación de la Línea Base de la Arquitectura.

Implementar pruebas: Si se requieren de procesos automáticos para los Procedimientos de Pruebas o de Componentes auxiliares para ejecutar las Pruebas, es tarea del Ingeniero de Componentes proporcionarlos al Ingeniero de Pruebas.

Realizar Pruebas de Integración: Cada vez que un Ingeniero de Componentes termine un Componente, este será integrado por el Integrador de Sistemas, para que continúe el Ingeniero de Pruebas aplicando Pruebas a esta Integración.

Realizar Pruebas del Sistema: Una vez que el sistema quede integrado por los Casos de Uso Arquitectónicamente significativos, se obtiene la Línea Base de la Arquitectura, lista para ser probada en las funcionalidades de los Casos de Uso que la integran.

Evaluar Pruebas: El resultado de todas las Pruebas realizadas en este flujo de trabajo es interpretado y evaluado por el Ingeniero de Pruebas, notificando los posibles errores y defectos al Arquitecto y a los Ingenieros de Componentes para su pronta corrección.

3.5.2.4. Productos de la Fase de Elaboración

Al terminar de la Fase de Elaboración se tendrán los siguientes productos de trabajo:

- Modelo del Negocio (o del Domino) completo
- Nueva versión de todos los modelos completos hasta cerca del 10 por ciento (a excepción de los Modelos de Casos de Uso y de Análisis que tendrán mas avance de elaboración)
- La Línea Base de la Arquitectura
- La Descripción de la Arquitectura incluyendo las Vistas de los Modelos de Casos de Uso, de Análisis, de Diseño, de Implementación y de Despliegue
- Lista de riesgos actualizada
- Opcionalmente un preliminar del Manual de Usuario
- Las versiones completas de la estimación de beneficios, del reconocimiento del mercado, y las estimaciones del proyecto

3.5.3. Fase de Construcción

En la Fase de Elaboración se creó la Línea Base de La Arquitectura, misma que será utilizada para trabajar sobre ella en la Fase de Construcción, para así obtener el producto software en su versión operativa inicial en el entorno de Usuario (la versión beta del producto software). Así mismo, en esta fase, se cerrarán los Modelos de Análisis, Diseño e Implementación, y se integrará y probará la totalidad del Sistema, reduciendo todos los riesgos (excepto los que se relacionan a la operación del producto, que se tratarán en la siguiente fase).

En la Fase de Inicio y en la Fase de Elaboración se trabajo sobre un enfoque de investigar todo lo relacionado al producto que se deseaba, ahora en la Fase de Construcción se cambiará de enfoque para trabajar precisamente en la construcción propiamente dicha de ese producto.

3.5.3.1. Objetivos de la Fase de Construcción

El objetivo principal de la Fase de Construcción es:

- Desarrollar el Producto Software en su versión operativa inicial

Además de este objetivo principal, la Fase de Construcción tiene otros objetivos secundarios, algunos de los cuales ayudan a obtener el objetivo principal:

- Detallar la totalidad de los Casos de Uso y sus escenarios
- Terminar los Modelos de Análisis, Diseño e Implementación.

- Mitigar todos los riesgos, a excepción de los riesgos de operación.
- Obtener las versiones preliminares del material de usuario y del material de cursos

3.5.3.2. Antecedentes de la Fase de Construcción

Como se ha mencionado anteriormente, en la Fase de Elaboración se ha construido la Línea Base de la Arquitectura ejecutable. Las fases anteriores han reducido los riesgos críticos y significativos hasta un punto que se consiga eliminarlos en la Fase de Construcción. También se han detallado los Casos de Uso más importantes arquitectónicamente.

Asimismo, al final de la Fase de Elaboración, el Jefe de Proyecto planificó la Fase de Construcción, pero una vez que la Fase de Construcción inicie (para lo cual puede darse el caso que pasen meses para que se autorice a continuar), el Jefe de Proyecto deberá ajustar este Plan de la Fase de Construcción a las circunstancias de ese momento.

La carga de trabajo aumenta considerablemente en esta fase (abarcando cerca del 50 por ciento del trabajo total), debido a que se abarca cerca del 10 por ciento del volumen de trabajo del Flujo de Trabajo de Requerimientos, 60 por ciento del Flujo de Trabajo de Análisis y cerca del 90 por ciento de los Flujos de Trabajo de Diseño, Implementación y Prueba. Considerando lo anterior, el Jefe de Proyecto deberá integrar su equipo de trabajo, aumentando recursos hasta casi el doble de la Fase de Elaboración, con el fin de alcanzar los objetivos propuestos para la construcción.

3.5.3.3. Flujos de Trabajo en la Fase de Construcción

A continuación se detalla cómo se desarrollan las actividades de los cinco flujos de trabajo en las iteraciones de la Fase de Construcción:

- Flujo de Trabajo de Requerimientos

  Para esta fase se debe identificar y detallar el cien por ciento de los Casos de Uso, para ello se realizarán las siguientes actividades:

  Encontrar Actores y Casos de Uso: Con respecto a esta actividad, solo resta identificar una mínima cantidad de Casos de Uso y Actores, debido a lo cual se puede requerir actualizar el Modelo de Casos de Uso.

  Construir un Prototipo de Interfaz de Usuario: Por lo general esta actividad no se realiza en las dos primeras fase, pero en la Fase de Construcción se debe construir el Prototipo de Interfaz de Usuario para que sea mostrado a los Usuarios y así ajustarlo a ellos, es decir, en este flujo de trabajo y en esta fase realizaremos el Diseño de la Interfaz de Usuario, y aunque el nombre sugiere lo contrario, esta actividad se realiza dentro del Flujo de Trabajo de Requerimientos y no en el de diseño, y solo hasta que se realice se podrá continuar con los flujos de trabajo siguientes. En ocasiones, en sistemas sencillos y de bajo impacto y distribución, no será necesario hacer un Prototipo de la Interfaz de Usuario, en su lugar se trabajará en conjunto con los Usuarios para el diseño de su interfaz.

  Determinar la prioridad de los casos de usos: Debido a que en esta fase se seguirán identificando una pequeña cantidad de Casos de Uso, la actividad de dar prioridad a estos nuevos Casos de Uso continuará.

  Detallar casos de uso: Durante la construcción se detallarán completamente todos los Casos de Uso y escenarios de Casos de Uso que resten de acuerdo a su prioridad asignada.

  Estructurar el Modelo de Casos de Uso: Si durante esta fase, el Analista de Sistemas desea mejorar el Modelo de Casos de Uso, deberá evaluar el impacto de cualquier cambio,

pues ya se cuenta con una Arquitectura estable y cualquier cambio en este modelo deberá verse reflejado en los Modelos de Análisis y Diseño. Se recomienda que los únicos cambios en esta fase sean los relacionados a Casos de Uso que no hayan sido detectados.

- Flujo de Trabajo de Análisis

  En algunos proyectos, se puede tomar la libertad de no seguir dando mantenimiento al Modelo de Análisis después de la Fase de Elaboración, debido a que se podría considerar un paso intermedio para comenzar a estructurar el Modelo de Diseño, para tales situaciones, en la Fase de Construcción y en la Fase de Transición no se tendrán actividades del Flujo de Trabajo de Análisis. En otros proyectos que construyan productos software demasiado complejos, será recomendable continuar manteniendo el Modelo de Análisis durante todo el ciclo de vida del proyecto, para estos casos se completará el Modelo de Análisis en la Fase de Construcción, terminando de construir casi 60 por ciento que nos faltaba al final de la fase anterior, y para ello se realizarán las siguientes actividades:

  Analizar la Arquitectura: En esta fase el arquitecto solo se limitará a dar mantenimiento (debido a los cambios que surjan a durante esta fase) a la Vista de la Arquitectura del Modelo de Análisis que se realizó en la fase anterior

  Analizar un Caso de Uso: Se continuará analizando los Casos de Uso que resten en esta fase para agregarlos así al Modelo de Análisis.

  Analizar una Clase: En esta fase los Ingenieros de Componentes continuarán el trabajo que iniciaron en la Fase de Elaboración.

  Analizar un paquete: Durante la Fase de Construcción el Arquitecto posiblemente modifique algún Paquete, por lo cual el Ingeniero de Componentes tendrá que mantener los Paquetes afectados.

- Flujo de Trabajo de Diseño

  En Fase de Construcción se continuará completando el 90 por ciento restante del Modelo de Diseño a través de:

  Diseñar Arquitectura: Durante esta fase el Arquitecto puede agregar Subsistemas si son similares o alternativos a los que ya existen, pero es poco probable y riesgoso agregar nuevos Subsistemas de Servicio debido a que estos afectarían directamente a la estructura de la Arquitectura que se elaboró en la fase anterior. También actualizará e introducirá mejoras en la Vista de la Arquitectura de los Modelos de Diseño y Despliegue.

  Diseñar un Caso de Uso: En esta fase el Ingeniero de Casos de Uso continuará trabajando en las realizaciones de los Casos de Uso-Diseño de los Casos de Uso que no se trataron en las fases anteriores, elaborando esto en términos de Clases del Diseño, Subsistemas de Diseño y Subsistemas de Servicio.

  Diseñar una Clase: Los Ingenieros de Componentes terminan de detallar las Clases de Diseño con la información que se va obteniendo a medida que terminan de diseñar todos los Casos de Uso en los que intervienen.

  Diseñar un Subsistema: Cada vez que el Arquitecto identifique nuevos Subsistemas, los Ingenieros de Componentes los diseñaran a detalle.

- Flujo de Trabajo de implementación

  Este es el flujo de trabajo de mayor importancia durante la Fase de Construcción, y es en donde se irán completando en cada iteración los Componentes.

  Implementar la Arquitectura: La Implementación de la Arquitectura (que fue elaborada en la fase anterior) se continuará actualizando y vigilando por parte del Arquitecto.

  Implementación de una Clase, implementación de un Subsistema y realizar Prueba de Unidad: los Ingenieros de Componentes implementarán y probarán la totalidad de Subsistemas y Clases de acuerdo al orden que se les sea establecido.

  Integrar el Sistema: El Integrador de Sistemas creará un Plan de Construcciones que guíe el orden de las implementaciones y Construcciones que se realicen en las iteraciones y en general en la fase.

- Flujo de Trabajo de Prueba

  Este es el segundo flujo de trabajo importante en esta fase, pues por cada incremento que se genere en una iteración se tendrán que hacer pruebas, hasta que al final se pruebe el Sistema completo.

  Planificar las Pruebas: Los Ingenieros de Pruebas seleccionarán los objetivos que se comprobarán en cada incremento, y finalmente en el Sistema completo.

  Diseñar las Pruebas: Una vez que determinen los objetivos a probar, los Ingenieros de Pruebas determinarán cómo probarlos, preparando Casos y Procedimientos de Prueba, de los cuales incluso pueden retomar alguno que ya sea utilizado anteriormente.

  Implementar Pruebas: Una vez que se estableció el qué probar y el cómo probarlo, puede ser que se requiera un Componente para automatizar las pruebas o para sustituir parte del Sistema que no haya sido implementada aun, si es el caso, es responsabilidad del Ingeniero de Componentes el implementarlo.

  Realizar Pruebas de Integración: Por cada Construcción que se genere y se integre a las Construcciones anteriores, se realizarán Pruebas de Integración comprobando los Casos de Prueba y siguiendo Procedimientos de Pruebas previamente establecidos, y de acuerdo a los resultados y evaluación de ellos por parte del Ingeniero de Pruebas y del Integrador de Sistemas se tomará una decisión.

  Realizar Pruebas del Sistema: Al final de cada iteración, la Construcción resultante (que será una versión parcial del Sistema), estará bajo la responsabilidad del encargado de Pruebas del Sistema, que comprobará los casos de prueba al seguir los Casos de Prueba. Esto continuará hasta alcanzar la última iteración, de la cual la Construcción resultante será la versión beta del Sistema completo.

  Evaluar Pruebas: Al final de las Pruebas de Integración y del Sistema, el Ingeniero de Pruebas junto con otros roles competentes, evaluarán los resultados de las Pruebas con el objetivo de tomar las decisiones que permitan alcanzar los objetivos previamente establecidos para las Construcciones parciales y el Sistema en general.

3.5.3.4. Productos de la Fase de Construcción

Al terminar de la Fase de Construcción se tendrán los siguientes productos de trabajo:

- La versión operativa inicial completa del Sistema
- Todos los artefactos, incluyendo los modelos del Sistema
- La descripción de la Arquitectura actualizada
- Versión previa del Manual de Usuario, listo para guiar en la versión beta del Sistema

3.5.4. Fase de Transición

Durante la Fase de Transición el equipo de desarrollo buscará implantar el Sistema en el entorno de operación del Usuario, y al mismo tiempo descubrir, de forma retardada, características que no se hayan detectado y por tanto implementado durante las fases anteriores, riesgos inesperados, problemas no resueltos, fallos no detectados, ambigüedades, lagunas en la documentación y en general buscará la satisfacción total del Usuario. Una vez encontrado algo de lo anterior, no se buscará reformular el producto; se evaluará si es de gran importancia y si se adapta bien al producto ya construido para incluirlo mediante iteraciones adicionales, o bien, se agregará a una lista de características para la siguiente versión del producto. Una vez que se haya resulto lo anterior, el equipo preparará la producción del producto, el empaquetado, la distribución y el lanzamiento a los Usuarios en general.

3.5.4.1. Objetivos de la Fase de Transición

El objetivo principal de la Fase de Transición es:

- Implantar el producto en su entorno de operación, con la satisfacción de todos los Usuarios

Además de este objetivo principal, la Fase de Transición tiene otros objetivos secundarios, algunos de los cuales ayudan a obtener el objetivo principal:

- Gestionar todos los aspectos relativos a la operación en el entorno del Usuario.

3.5.4.2. Antecedentes de la Fase de Transición

Al final de la Fase de Construcción, se construyó el software en su versión operativa inicial. Ahora en la Fase de Transición se debe considerar cómo se conducirá este producto hasta el entorno de operación del Usuario. Se debe evaluar si es necesario utilizar un conjunto de Usuarios beta que nos ayuden a encontrar posibles fallos, o si se va a distribuir el producto directamente con los Usuarios finales por un periodo de evaluación (aplicable para software con un grupo limitado de Usuarios).

Considerando lo anterior y teniendo presente que aun durante esta fase muy probablemente se van a producir cambios de bajo impacto en el producto, se hará una aproximación a la planeación de la fase, pues no se puede saber exactamente cual será la cantidad de trabajo en esta fase, cantidad que dependerá de los defectos encontrados por los Usuarios y por la naturaleza de estos (si producen cambios de bajo impacto, se trabajará en ellos en esta fase, si producen cambios de alto impacto, se deberá elegir si se trabaja en ellos en esta fase o se pospondrán hasta la siguiente versión del producto).

También se debe integrar el equipo de trabajo para esta fase, considerando recursos para realizar pruebas o funcionen como Usuarios beta (estos recursos no necesariamente deben tener conocimiento técnico del producto, basta con que cuenten con conocimiento de la naturaleza de la aplicación) y recursos para realizar posibles cambios al producto (recursos que si requieren conocimiento técnico de la aplicación, y preferentemente del mismo equipo de la fase anterior),

además de los recursos que trabajen en artefactos adicionales para la transición, como programas de instalación y programas de conversión o de migración de datos.

3.5.4.3. Flujos de Trabajo en la Fase de Transición

En esta fase no se describen a detalle las actividades realizadas en cada uno de los flujos de trabajo, debido a que el trabajo es casi nulo en los Flujos de Trabajo de Requerimientos, Análisis y Diseño, limitándose solamente a actualizaciones; y en los Flujos de Trabajo de Implementación y Prueba se genera un ciclo de revisión y corrección entre ellos, esta ultima desencadenando la actualizaciones en los demás flujos de trabajo.

La mayor parte del trabajo en esta fase se enfoca en actividades que no son propiamente de desarrollo (como es el caso de las actividades de los flujos de trabajo) y que se desarrollan en paralelo a las de los flujos de trabajo básicos. El Proceso Unificado menciona básicamente las siguientes actividades durante la Fase Transición:

- Preparar la versión beta a partir de la versión operativa inicial elaborada en la fase anterior, lo que implica elegir a los usuarios beta y el periodo de evaluación de la versión beta. También se debe especificar las instrucciones precisas para la instalación, para la operación, para la solución de posibles problemas, para el reporte de fallos y problemas encontrados, y para la posible migración de datos existentes; finalmente se entregará un paquete con toda esta documentación a nuestros usuarios beta.

- Instalar la versión beta en los lugares elegidos, junto con las actividades de soporte relacionadas en dichos lugares, por ejemplo, la migración de datos.

- Actuar de acuerdo a la información recogida en las instalaciones de prueba. Se debe valorar el tipo de fallo o problema encontrado para saber si desencadena cambios sencillos o cambios muy extensos, y de acuerdo a esto elegir si se corregirá con modificaciones sencillas, si se requerirá de iteraciones adicionales o si se dejará para corregirse en futuras versiones. Lo anterior desencadena actividades de los Flujos de Trabajo de Análisis, Diseño y Construcción.

- Adaptar el producto corregido a las circunstancias de los Usuarios. Una vez que se hayan concluido las pruebas beta, se debe considerar ahora todo el conjunto de posibles Usuarios y su entorno de operación, por lo que posiblemente se requieran actividades de soporte, como migración de bases de datos, además de configuraciones particulares del software desarrollado.

- Finalización de los artefactos. Se deben cerrar y actualizar todos los artefactos de todos los modelos, incluyendo la Arquitectura, y comprobando que todo tenga consistencia, esto involucrará a la mayoría de los flujos de trabajo.

- Formalización del fin del proyecto. Esto se obtendrá hasta que el cliente este satisfecho con el producto y los artefactos elaborados.

3.5.4.4. Productos de la Fase de Transición

Al terminar de la Fase de Transición se tendrán los siguientes productos de trabajo:

- El software ejecutable y el software de instalación
- Documentos legales, como contratos, licencias, renuncias de derechos y garantías
- Versión completa y corregida de todos los artefactos, incluyendo modelos
- Descripción completa y actualizada de la arquitectura
- Manuales y material de formación del Usuario final, Operador y Administrador del Sistema

El Proceso Unificado marca al final de todo proyecto la evaluación de las fases del mismo y de todo el proyecto en conjunto, registrando cualquier experiencia insatisfactoria y también los éxitos, tanto relacionados al producto y los relacionados al proceso de desarrollo. Toda esta información, aunque no se marca una formalización como tal, si se indica que se utilizará para mejorar la organización de desarrollo, a sus miembros y al proceso de desarrollo.

En este capítulo se expone un breve resumen del Proceso Unificado, basado en [Jacobson et al., 2000], que intenta mostrar los aspectos característicos de éste y que se utilizaron como base para la metodología propuesta.

4

El Modelo de Madurez de Capacidad

Este capítulo describe el Modelo de Madurez de Capacidad (Capability Maturity Model - CMM), creado por el Software Engineering Institute (SEI) de la Universidad de Carnegie Mellon, el cual es un modelo de referencia para evaluar y mejorar la capacidad de los procesos de administración y desarrollo de proyectos de software.

4.1. Introducción

En los últimos años las sociedades modernas han experimentado una dependencia creciente hacia los productos de software, desde individuos hasta organizaciones de negocios completas requieren cada vez más de productos de software para realizar sus actividades diarias. A pesar de lo anterior el problema de la "crisis del software" (iniciado a finales de los 60´s) no ha disminuido como se esperaba. Para darle solución se ha trabajado en tres aspectos complementarios: herramientas, metodologías y modelos de referencia. Estos esfuerzos buscan incrementar la eficiencia, el control y la calidad en el desarrollo de software.

En aspectos de herramientas, se ha buscado dar solución por medio de diversas propuestas, una de ellas es el Lenguaje Unificado de Modelado; en cuanto a metodologías, el esfuerzo mas reciente es el Proceso Unificado de Desarrollo de Software; mientras que en el aspecto de los modelos de referencia de procesos, se han desarrollado diversos modelos para evaluar y mejorar la capacidad de los procesos de desarrollo de software.

Los modelos de referencia de procesos que se han desarrollado en la última década difieren de los modelos desarrollados anteriormente en que el desarrollo de software ya no solo es visto como un conjunto netamente de tareas de ingeniería, sino que ahora el desarrollo de software se ve como un proceso integrado que puede ser controlado, medido y mejorado [Krishnan et al., 1999]. Uno de estos modelos de referencia es el CMM.

4.2. Orígenes

En noviembre de 1986, el Software Engineering Institute (SEI), con la asistencia de Mitre Corporation, inició el desarrollo de un marco de trabajo, que llamó Marco de Trabajo para Madurez de Procesos (Framework Process Maturity), con el objetivo de ayudar a las organizaciones a mejorar sus procesos de software. Originalmente el proceso se inició en respuesta a una solicitud para proveer al gobierno federal de los Estados Unidos con un método para evaluar la capacidad de sus contratistas de software. En septiembre de 1987 el SEI libera una descripción breve del marco de trabajo, junto con un cuestionario para evaluar la madurez de los procesos. Desgraciadamente el cuestionario fue tomado como el modelo en sí, en lugar de ser tomado como una herramienta para evaluar la madurez de los procesos [Paulk et al. 1993].

Con la experiencia de cuatro años con el marco de trabajo y el cuestionario, el SEI generó el Capability Maturity Model for Software (SW-CMM) utilizando el conocimiento adquirido de la evaluación de procesos de software y la retroalimentación de industrias y gobierno. La primera versión del SW-CMM, versión 1.0, fue revisada por la comunidad de software durante 1991 y 1992, a la cual se le realizaron algunas modificaciones que generaron en 1993 la versión 1.1 de SW-CMM.

Para 1997 el SEI ya había desarrollado diversos modelos de referencia para diversas disciplinas, por ejemplo, para Ingeniería de Software había desarrollado el SW-CMM, para Ingeniería de Sistemas había desarrollado el SE-CMM, para el desarrollo de productos integrado había desarrollado el IPD-CMM, para mejorar el desempeño del personal desarrolló el People-CMM,

para la adquisición de software el SA-CMM. Al mismo tiempo, fuera del SEI se habían elaborado nuevos modelos relacionados, por ejemplo EIA/IS 731, ISO/IEC 15504 e ISO9000:2000.

En 1997 el SEI contaba ya con el Draft C de SW-CMM v2.0, pero en ese mismo año decide tomar este documento como una de las bases para generar la primer versión del nuevo CMMI (Capability Maturity Model Integration). El motivo para esta decisión fue integrar en un solo marco de trabajo todos los modelos de referencia para diversas disciplinas, y así solucionar el cada vez mas creciente problema de confusión y distanciamiento entre modelos; al mismo tiempo se buscó que los productos de trabajo de este marco de trabajo fueran compatibles con el estándar ISO/IEC 15504.

Después de pasar por la versión 0.1 y 0.2, en agosto del 2000 se libera la suite CMMI v1.0. Finalmente en 2002 se libera la mas reciente versión de la suite CMMI, la versión 1.1, que incluye modelos de referencia para Ingeniería de Software, para Ingeniería de Sistemas, para productos integrados y desarrollo de procesos, y para proveedores de servicios. En el futuro se planea incluir mas disciplinas dentro del marco de trabajo.

4.3. Definición de CMM

CMM no contiene procesos o descripciones de procesos. El propósito de CMM es proveer guías para mejorar los procesos de una organización y proveerle de la habilidad de manejar el desarrollo, adquisición y mantenimiento de productos o servicios de calidad. CMM ayuda a las organizaciones a establecer sus objetivos y prioridades de mejora de procesos, puede mejorar procesos y proveer guías para asegurar procesos estables, capaces y maduros [“Capability Maturity Model Integration (CMMI), Version 1.1”, 2002].

CMMI es la integración de los modelos CMM. Difiere de CMM, en que tiene dos representaciones diferentes: escalonada y continua (definidas por similitud con modelos anteriores y por compatibilidad con otros modelos, respectivamente). El objetivo de CMMI fue resolver los problemas que involucraba el manejar mas de un modelo de procesos (uno para cada disciplina y no completamente compatibles), problemas tales como, actividades duplicadas en los modelos e inconsistencias debido a la falta de estandarización entre modelos, al mismo tiempo que buscaba ser compatible con otros modelos internacionales.

4.4. Modelos de CMMI

La suite de productos CMMI actualmente se ha enfocado a cuatro áreas de conocimiento o disciplinas, por lo cual el SEI ha desarrollado cuatro modelos, de acuerdo a la tabla siguiente:

Nombre del Modelo	Disciplinas incluidas
CMMI-SW	Ingeniería de Software
CMMI-SE/SW	Ingeniería de Sistemas e Ingeniería de Software
CMMI-SE/SW/IPPD	Ingeniería de Sistemas, Ingeniería de Software y Productos Integrados y Desarrollo de Procesos
CMMI-SE/SW/IPPD/SS	Ingeniería de Sistemas, Ingeniería de Software, Productos Integrados y Desarrollo de Procesos y Proveedores de Servicios

Tabla 4-1: Disciplinas y modelos de CMMI

Para cada uno de estos modelos se han desarrollado dos representaciones distintas para proveer flexibilidad de aplicación y dar continuidad a la visión de los modelos que sirvieron de base para el desarrollo de CMMI. Ambas representaciones son equivalentes, y cada una de ellas tiene características muy particulares que las hacen elegibles de acuerdo a la naturaleza de la organización donde se desee emplear o de acuerdo a la estrategia que se planee emplear para la mejora de procesos.

La Representación Continua de los modelos tiene las siguientes características:

- Permite seleccionar el orden de mejora de acuerdo al que más se apegue a los objetivos de negocio de la organización y a mitigar los riesgos de las áreas de la organización.
- Provee una migración fácil del EIA/IS 731[1] hacia CMMI.
- Permite comparación entre y a través de organizaciones basada en Áreas de Procesos
- Ofrece una fácil comparación de la mejora de procesos con ISO/IEC 15504[2], debido a que la organización de las Áreas de Procesos es similar.
- Provee flexibilidad para enfocarse en Áreas de Procesos especificas de acuerdo a los objetivos y metas del negocio.

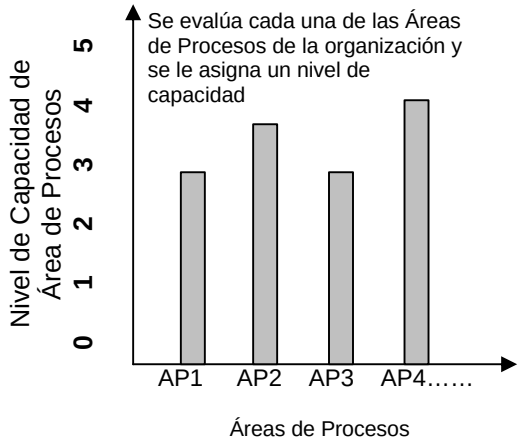


Figura 4-1: Representación Continua de CMMI

[1] Electronic Industries Alliance Interim Standard (EIA/IS) 731
[2] International Organization for Standardization and International Electrotechnical Commission (ISO/IEC) 15504

La Representación Escalonada de los modelos tiene las siguientes características:

- Provee una secuencia probada de mejora a partir de las prácticas básicas de gestión, pasando después a través de rutas probadas y predefinidas de niveles sucesivos, cada uno de los cuales sirviendo de base para el siguiente.
- Permite comparación entre y a través de organizaciones inutilizando niveles de madurez.
- Proporciona una migración fácil desde SW-CMM hacia CMMI.
- Provee una única forma de valuar, que resume los resultados de evaluaciones, permitiendo una fácil comparación entre organizaciones.
- Provee una guía para implementar grupos de Áreas de Procesos de forma secuencial.

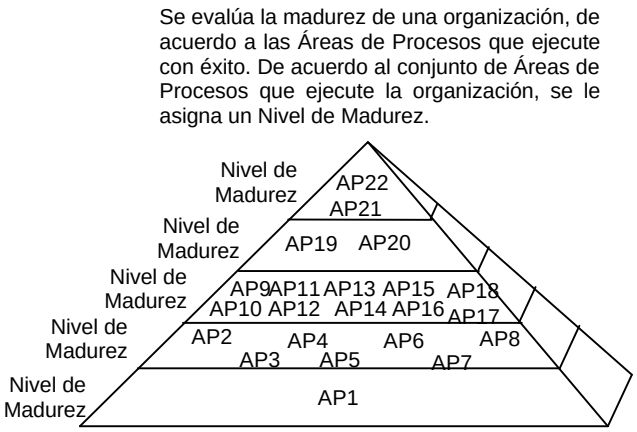


Figura 4-2: Representación Escalonada de CMMI

Una relación completa de los modelos CMMI y sus representaciones es la mostrada en la Tabla 4-2:

Disciplina	Modelos
Ingeniería de Sistemas Ingeniería de Software Productos Integrados y Desarrollo de Procesos Proveedores de Servicios	• **CMMI-SE/SW/IPPD/SS staged representation** • **CMMI-SE/SW/IPPD/SS continuous representation**
Ingeniería de Sistemas Ingeniería de Software Productos Integrados y Desarrollo de Procesos	• **CMMI-SE/SW/IPPD staged representation** • **CMMI-SE/SW/IPPD continuous representation**
Ingeniería de Sistemas Ingeniería de Software	• **CMMI-SE/SW staged representation** • **CMMI-SE/SW continuous representation**
Ingeniería de Software	• **CMMI-SW staged representation** • **CMMI-SW continuous representation**

Tabla 4-2: Disciplinas, modelos y representaciones de CMMI

La elección de uno de estos modelos se determina de acuerdo a la disciplina en la que se quiera aplicar y a la experiencia anterior que se tenga con otros modelos de referencia o a la estrategia de mejora de procesos que se planee.

4.5. Componentes de los Modelos de CMMI

A pesar que los modelos tienen dos representaciones, ambas comparten el mismo conjunto de componentes:

- Áreas de Procesos
  - METAS ESPECÍFICAS
  - PRÁCTICAS ESPECÍFICAS
  - METAS GENÉRICAS
  - PRÁCTICAS GENÉRICAS
  - Productos típicos de trabajo
  - Sub-prácticas
  - Notas
  - Ampliaciones de disciplinas
  - Elaboraciones de prácticas genéricas
  - Referencias

4.5.1. Áreas de Procesos

Un Área de Procesos es un grupo de prácticas relacionadas en un área que cuando se realizan de manera conjunta satisfacen un conjunto de metas que se consideran importantes para mejorar significativamente esa área. Todas las Áreas de Procesos de CMMI son comunes para la Representación Escalonada y para la Representación Continua. En la Representación Escalonada, las Áreas de Procesos son organizadas en Niveles de Madurez. En la Representación Continua, las Áreas de Procesos son organizadas por Categorías de Áreas de Procesos. [Paulk et al. 1993], [“Capability Maturity Model Integration (CMMI), Version 1.1”, 2002]

4.5.2. Metas Específicas

Las Metas Especificas se aplican a un Área de Procesos y se ocupan de las características únicas que describen cuales deben ser implementadas para satisfacer el Área de Procesos. Las Metas Especificas son componentes requeridos del modelo y se utilizan en las evaluaciones para ayudar a determinar si un Área de Procesos se satisface [Paulk et al. 1993], [“Capability Maturity Model Integration (CMMI), Version 1.1”, 2002].

4.5.3. Prácticas Específicas

Una Practica Especifica es una actividad que es considerada importante para alcanzar la Meta Especifica asociada a un Área de Procesos. Las Prácticas Específicas describen las actividades esperadas para resultar en el logro de las Metas Específicas de un Área de Procesos. Las Prácticas Especificas son componentes esperados del modelo. [Paulk et al. 1993], [“Capability Maturity Model Integration (CMMI), Version 1.1”, 2002].

4.5.4. Metas Genéricas

Alcanzar una Meta Genérica en un Área de Procesos significa mejorar el control en planear e implementar los procesos asociados esa Área de Procesos, indicando si estos procesos son efectivos, repetibles y duraderos. Las Metas Genéricas son componentes requeridos del modelo, y son utilizadas en evaluaciones para determinar si un Área de Procesos se satisface. Las Metas Genéricas son llamadas “Genéricas” por que la misma declaración de la Meta aparece para múltiples Áreas de Procesos. [Paulk et al. 1993], [“Capability Maturity Model Integration (CMMI), Version 1.1”, 2002].

En la Representación Escalonada cada Área de Procesos tiene solo una Meta Genérica. La Meta Genérica que un Área de Proceso contiene depende del Nivel de Madurez al que pertenece el

Área de Proceso. En esta representación, se tienen solo dos Metas Genéricas, por lo cual, una Meta Genérica puede ser la misma para distintos Niveles de Madurez. [Paulk et al. 1993]

En la Representación Continua cada Nivel de Capacidad (del 1 al 5) tiene solo una Meta Genérica que describe la institucionalización que la organización debe lograr en ese Nivel de Capacidad; por lo tanto se tienen cinco Metas Genéricas, las cuales aparecen en cada Área de Procesos. ["Capability Maturity Model Integration (CMMI), Version 1.1", 2002].

4.5.5. Prácticas Genéricas

Las Prácticas Genéricas proporcionan la institucionalización para asegurar que los procesos asociados al Área de Procesos serán efectivos, repetibles y duraderos. En la Representación Escalonada, las Prácticas Genéricas son categorizadas por las Metas Genéricas y las Características Comunes (de las cuales se describirán mas adelante con la Representación Escalonada), y son componentes esperados en los modelos CMMI. En la Representación Continua, las Prácticas Genéricas son categorizadas por el Nivel de Capacidad, y son componentes esperados en los modelos CMMI. También en la Representación Continua, cada Practica Genérica se relaciona a una Meta Genérica. [Paulk et al. 1993], ["Capability Maturity Model Integration (CMMI), Version 1.1", 2002].

4.5.6. Productos Típicos de Trabajo

Los Productos Típicos de Trabajo son componentes informativos del modelo que proveen ejemplos de salidas de una Práctica Específica o de una Práctica Genérica. Estos ejemplos son llamados "Productos Típicos de Trabajo" debido a que existen otros productos de trabajo que no se mencionan. [Paulk et al. 1993], ["Capability Maturity Model Integration (CMMI), Version 1.1", 2002].

4.5.7. Sub-prácticas

Las Sub-Prácticas son descripciones detalladas que proveen una guía para interpretar Prácticas Específicas o Genéricas. Las Sub-prácticas se pueden redactar de forma imperativa, pero realmente son componentes informativos en los modelos CMMI, lo que significa que solo proveen ideas que pueden ser útiles para la mejora de procesos. [Paulk et al. 1993], ["Capability Maturity Model Integration (CMMI), Version 1.1", 2002].

4.5.8. Notas

Las notas son otros de los componentes informativos del modelo que ayudan a entender los componentes del modelo y previendo detalles sobre los mismos.

4.5.9. Ampliaciones de Disciplinas

Las amplificaciones de Disciplinas son componentes informativos del modelo que contienen información relevante a una disciplina en particular y pueden ser añadidas a otros componentes del modelo cuando sea necesario. [Paulk et al. 1993], ["Capability Maturity Model Integration (CMMI), Version 1.1", 2002].

4.5.10. Elaboraciones de Prácticas Genéricas

Las Elaboraciones de Prácticas Genéricas son componentes informativos del modelo que aparecen en cada Área de Procesos para proveer una guía de cómo deben ser aplicadas de manera única las Prácticas Genéricas al Área de Procesos [Paulk et al. 1993].

4.5.11. Referencias

Las Referencias son componentes informativos del modelo que dirige hacia información mas detallada relacionada al Área de Procesos [Paulk et al. 1993].

4.6. Representación Escalonada de CMMI

El Modelo Escalonado de CMMI se describe en un documento que contiene seis capítulos que dan la visión general del mismo:

1. Introducción
2. Estructura del Modelo
3. Terminología del Modelo
4. Niveles de Madurez, Características Comunes y Prácticas Genéricas
5. Interpretación del Modelo
6. Utilización del Modelo

Además de otros dos capítulos con explicación más detallada:

7. Áreas de Procesos

    - Nivel de Madurez 2: Administrado
    - Nivel de Madurez 3: Definido
    - Nivel de Madurez 4: Administrado cuantitativamente
    - Nivel de Madurez 5: Optimizado

Apéndices

- Referencias
- Acrónimos
- Glosario
- Elementos requeridos y esperados del Modelo
- Participantes del Proyecto CMMI

La Representación Escalonada esta basada en la madurez de una organización, la cual es la combinación de capacidades de un conjunto de procesos relacionados. La Representación Escalonada esta compuesta de Niveles de Madurez[1] que incluyen las principales Áreas de Procesos que se deben satisfacer para alcanzar un nivel. Para alcanzar Niveles de Madurez superiores, es necesario alcanzar los Niveles de Madurez inferiores, debido a que son las bases, y por lo tanto no se puede saltar. Esta representación tiene un orden definido para dirigir la mejora de procesos. Los nombres de los Niveles de Madurez han sido adaptados a la terminología de ISO/IEC 15504, a diferencia de los modelos anteriores de CMM, pero su descripción permanece igual. Las Áreas de Procesos se han redefinido para alinearse mejor con los procesos de ISO/IEC 15504. En la Representación Escalonada, cada una de las veintidós Áreas de Procesos es asignada a alguno de los Niveles de Madurez del 2 al 5 [Bowen et al., 2002], como se muestra en la Tabla 4-3:

[1] Estos Niveles de Madurez representan la madurez de una organización.

Nivel	Enfoque	Áreas de Procesos
1. Inicial	Procesos no predecibles, poco controlados y reactivo.	
2. Administrado	Administración de Procesos Básica a nivel de proyectos.	1. Administración de Requerimientos 2. Planeación del Proyecto 3. Monitoreo y Control del Proyecto 4. Administrador de acuerdos con proveedores 5. Medición y Análisis 6. Aseguramiento de la calidad de procesos y productos 7. Administración de la Configuración.
3. Definido	Estandarización de Procesos a nivel de organización	8. Desarrollo de Requerimientos 9. Solución Técnica 10. Integración del Producto 11. Verificación 12. Validación 13. Enfoque de Procesos Organizacionales 14. Definición de Procesos Organizacionales 15. Entrenamiento Organizacional 16. Administración Integrada de Proyecto 17. Administración de Riesgos 18. Análisis de Decisión y Resolución
4. Administrado Cuantitativamente	Administración Cuantitativa	19. Desempeño de Procesos de la Organización 20. Administración Cuantitativa de Proyectos
5. Optimizado	Mejora Continua de Procesos	21. Innovación Organizacional y Despliegue 22. Análisis Causal y Resolución

Tabla 4-3: Áreas de Procesos y Niveles de Madurez en la representación escalonada de CMMI

Una característica importante de los modelos CMMI, es la distinción de Metas Especificas y Metas Genéricas que simulan las dos dimensiones del modelo ISO/IEC 15504 [Bowen et al., 2002].

En la Representación Escalonada, para cada Área de Procesos hay una o mas Metas Específicas describiendo qué actividades deben ser implementadas para satisfacer esa Área de Procesos, esas actividades son las Prácticas Especificas [Bowen et al., 2002].

Las Prácticas Especificas se relacionan con una Meta Específica, y son actividades que se consideran importantes para alcanzar la meta.

Además de las Metas Específicas en un Área de Procesos, existen también una o más de las dos Metas Genéricas siguientes:

- Institucionalizar un proceso administrado, y
- Institucionalizar un proceso definido.

La primera meta aplica a todas las Áreas de Procesos en el Nivel de Madurez 2, y la segunda meta aplica a todas las Áreas de Procesos incluidas en los Niveles de Madurez 3, 4 y 5.

También en esta representación, existen diez Prácticas Genéricas asociadas a la primer Meta Genérica “Institucionalizar un proceso administrado”. Además existen dos Prácticas Genéricas mas asociadas la segunda Meta Genérica “Institucionalizar un proceso definido”.

El total de las doce Prácticas Genéricas se organizan en cuatro Características Comunes, de esta manera las Características Comunes categorizar cada una de las Prácticas Genéricas, y lo hacen de una forma muy similar a la categorización de las Áreas de Procesos, asegurando que un Área de Procesos dada sea efectiva, repetible y duradera, además de que la proveen de la infraestructura de soporte necesaria. Las cuatro Características Comunes de la Representación Escalonada son:

1. **Acuerdos a realizar**
   Se refiere al establecimiento de políticas administradas y aseguramiento del apoyo para lograr la mejora de procesos.

2. **Habilidad a satisfacer**
   Asegura que el proyecto y/o la organización tiene los recursos que se requieren para el propósito de mejora de procesos; administrando y manteniendo los planes, recursos, asignación de responsabilidades y de autoridad, y capacitación.

3. **Directiva de Implementación**
   Se refiere a la medida, control y realización de los procesos

4. **Verificación de Implementación**
   Asegura que las actividades del proyecto y/o de la organización se realizan conforme a requerimientos, procesos y procedimientos.

En la Figura 4-3 se muestra de forma esquemática la representación escalonada:

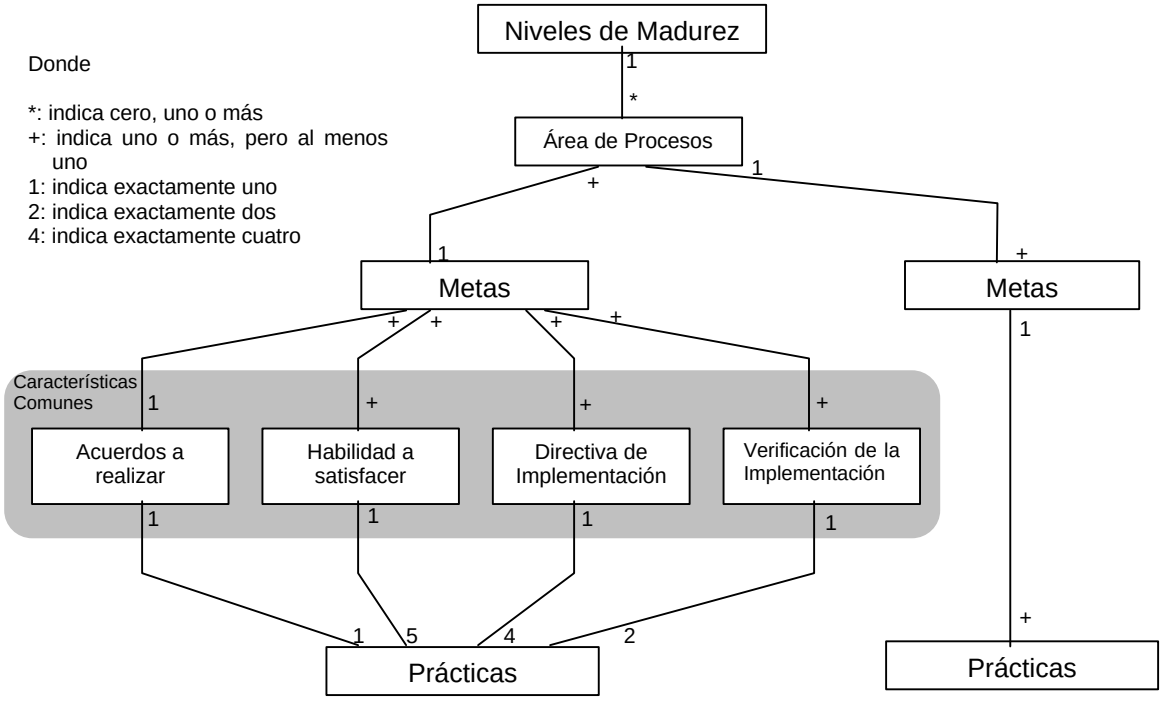


Figura 4-3: Esquema de la representación escalonada de CMMI

4.7. Esquema de Representación Escalonada del Modelo CMMI-SW

A continuación se muestra un esquema de la Representación Escalonada del Modelo CMMI-SW, en el cual se listan solo los componentes requeridos y esperados del modelo, y se utilizan las siguientes abreviaturas:

Áreas de Procesos: AP
- Meta Genérica: MG
- Práctica Genérica: PG
- Meta Específica: ME
- Práctica Específica: PE
- Características Comunes: CC
  - Acuerdos a realizar: AR
  - Habilidad a satisfacer: HS
  - Directiva de Implementación: DI
  - Verificación de Implementación: VI

Así mismo, por sencillez en el esquema se coloca a las Metas Genéricas y Prácticas Genéricas fuera del Área de Procesos a la que pertenecen y se coloca bajo el Nivel de Madurez correspondiente, indicando que todas las Áreas de Procesos de ese nivel tienen esas Metas Genéricas y Prácticas Genéricas. Aclarado este punto, el esquema es el siguiente:

NIVEL DE MADUREZ 1

NIVEL DE MADUREZ 2

MG 2: Institucionalizar un proceso administrado
- PG 2.1 Establecer una política organizacional (AR1)
- PG 2.2 Planear el proceso (HS1)
- PG 2.3 Proveer recursos (HS2)
- PG 2.4 Asignar Responsabilidades (HS3)
- PG 2.5 Capacitar Personal (HS4)
- PG 2.6 Administrar Configuraciones (DI1)
- PG 2.7 Identificar e implicar a todos los interesados (DI2)
- PG 2.8 Monitorear y controlar el proceso (DI3)
- PG 2.9 Evaluar objetivamente el apego al proceso (VI1)
- PG 2.10 Revisar el status con administradores de alto nivel (VI2)

AP: Administración de Requerimientos

ME1: Administrar requerimientos
- PE 1.1 Obtener una comprensión de requerimientos
- PE 1.2 Obtener un acuerdo sobre los requerimientos
- PE 1.3 Administrar cambios a los requerimientos
- PE 1.4 Mantener una trazabilidad bidireccional de los requerimientos
- PE 1.5 Identificar inconsistencias entre el trabajo del proyecto y los requerimientos

AP: Planeación del Proyecto

ME1: Establecer estimaciones
- PE 1.1 Estimar el alcance del proyecto
- PE 1.2 Establecer estimaciones del producto de trabajo y de las características de las tareas
- PE 1.3 Definir el Ciclo de Vida del Proyecto
- PE 1.4 Determinar estimaciones de esfuerzo y costo

ME2: Desarrollar un Plan del Proyecto
- PE 2.1 Establecer el presupuesto y el calendario
- PE 2.2 Identificar los riesgos del proyecto
- PE 2.3 Plan para la Administración de Datos
- PE 2.4 Plan para los Recursos del Proyecto
- PE 2.5 Plan para las Habilidades y Conocimientos Necesarios
- PE 2.6 Plan para Implicar a los interesados
- PE 2.7 Establecer un Plan del Proyecto

ME3: Obtener acuerdos para el Plan
- PE 3.1: Revisar planes que afecten al proyecto
- PE 3.2: Reconciliar los niveles de recursos y de trabajo
- PE 3.3: Obtener compromisos con el plan

AP: Monitoreo y Control del Proyecto

ME1: Monitorear el proyecto contra el plan
- PE 1.1: Monitorear los parámetros planeados del proyecto
- PE 1.2: Monitorear acuerdos
- PE 1.3: Monitorear riesgos del proyecto
- PE 1.4: Monitorear administración de datos
- PE 1.5: Monitorear la implicación de los interesados
- PE 1.6: Conducir revisiones al proyecto
- PE 1.7: Conducir revisiones en milestones[1]

ME2: Administrar las acciones correctivas
- PE 2.1: Analizar el caso
- PE 2.2: Tomar una acción correctiva
- PE 2.3: Administrara la acción correctiva

AP: Administrador de acuerdos con proveedores

ME1: Establecer acuerdos con proveedores
- PE 1.1: Determinar el tipo de adquisición
- PE 1.2: Seleccionar proveedor
- PE 1.3: Establecer acuerdos con el proveedor

ME2: Satisfacer acuerdos con proveedores
- PE 2.1: Revisar ofertas de productos
- PE 2.2: Ejecutar acuerdos con proveedores
- PE 2.3: Aceptar el producto adquirido
- PE 2.4: Transición del producto

AP: Medición y Análisis

ME1: Alinear medidas y actividades de análisis
- PE 1.1: Establecer objetivos de medidas
- PE 1.2: Especificar medidas
- PE 1.3: Especificar colección de datos y procedimientos de almacenamiento
- PE 1.4: Especificar procedimientos de análisis

ME2: Proveer resultados de medidas
- PE 2.1: Colectar datos de medidas
- PE 2.2: Analizar datos de medidas
- PE 2.3: Almacenar datos y resultados
- PE 2.4: Comunicar datos

AP: Aseguramiento de la calidad de procesos y productos

ME1: Evaluar objetivamente procesos y productos de trabajo
- PE 1.1: Evaluar objetivamente procesos
- PE 1.2: Evaluar objetivamente productos de trabajo y servicios

ME2: Proveer una visión objetiva
- PE 2.1: Comunicar y asegurara la resolución de casos que no cumplieron
- PE 2.2: Establecer registros

AP: Administración de la Configuración.

ME1: Establecer Líneas Base
- PE 1.1: Identificar objetos de la configuración
- PE 1.2: Establecer un sistema de administración de configuración
- PE 1.3: Crear y liberar Líneas Base

ME 2: Seguimiento y control de cambios
- PE 2.1: Seguimiento de solicitudes de cambio
- PE 2.2: Control de objetos de configuración

ME3: Establecer integridad
- PE 3.1: Establecer registros de administración de configuración
- PE 3.2: Realizar auditorias a la configuración

NIVEL DE MADUREZ 3

MG3: Institucionalizar un proceso definido
- PG 2.1 Establecer una política organizacional (AR1)
- PG 3.1 Establecer un proceso definido (HS5)
- PG 2.2 Planear el proceso (HS1)
- PG 2.3 Proveer recursos (HS2)

[1] Los milestones se refieren a revisiones preestablecidas de acuerdo a periodos de tiempo o al logro de determinados objetivos parciales durante el desarrollo del proyecto.

- PG 2.4 Asignar Responsabilidades (HS3)
- PG 2.5 Capacitar Personal (HS4)
- PG 2.6 Administrar Configuraciones (DI1)
- PG 2.7 Identificar e implicar a todos los interesados (DI2)
- PG 2.8 Monitorear y controlar el proceso (DI3)
- PG 3.2 Colectar información de mejora (DI4)
- PG 2.9 Evaluar objetivamente el apego al proceso (VI1)
- PG 2.10 Revisar el status con administradores de alto nivel (VI2)

AP: Desarrollo de Requerimientos

ME1: Desarrollar requerimientos del cliente
- PE 1.1: Obtener necesidades
- PE 1.2: Desarrollar requerimientos del cliente

ME2: Desarrollar requerimientos del producto
- PE 2.1: Establecer requerimientos de producto y de sus componentes
- PE 2.2: Asigne los requerimientos de los componentes del producto
- PE 2.3: Identifique requerimientos de interfaz

ME3: Analizar y validar los requerimientos
- PE 3.1: Establecer los conceptos y escenarios operacionales
- PE 3.2: Establecer una descripción de la funcionalidad requerida
- PE 3.3: Analizar requerimientos
- PE 3.4: Analizar Requerimientos y alcanzar el balance
- PE 3.5: Validar los requerimientos con métodos comprensivos

AP: Solución Técnica

ME1: Seleccionar soluciones de los componentes del producto
- PE 1.1: Desarrollar detalladamente las soluciones alternativas y los criterios de selección
- PE 1.2: Desarrollar los conceptos y escenarios operacionales
- PE 1.3: Seleccionar la solución de los componentes del producto

ME2 Desarrollar el diseño
- PE 2.1: Diseñar el producto o los componentes del producto
- PE 2.2: Establecer un paquete técnico de datos
- PE 2.3: Diseñar la s interfaces utilizando criterios
- PE 2.4: Realizar un análisis de elaboración, compra o reutilización de componentes del producto

ME3: Implementar el diseño del producto
- PE 3.1: Implementar el diseño
- PE 3.2: Desarrollar la documentación de soporte del producto

AP: Integración del Producto

ME1: Preparar la integración del producto
- PE 1.1: Determinar la secuencia de integración
- PE 1.2: Establecer el ambiente de integración del producto
- PE 1.3: Establecer los procedimientos y criterios de integración del producto

ME2: Asegurar la compatibilidad de interfaces
- PE 2.1: Revisar que las descripciones de interfaces estén completas
- PE 2.2: Administrar interfaces

ME3: Ensamblar los componentes del producto y liberar el producto
- PE 3.1: Confirmar que los componentes del producto están preparados para la integración
- PE 3.2: Ensamblar los componentes del producto
- PE 3.3: Evaluar componentes del producto ensamblados
- PE 3.4: Empaquetado y liberación del producto o de los componentes del producto

AP: Verificación

ME1: Preparar la verificación
- PE 1.1: Seleccionar productos de trabajo a verificar
- PE 1.2: Establecer el entorno de verificación
- PE 1.3: Establecer procedimientos y criterios de verificación

ME2: Realizar revisiones entre pares
- PE 2.1: Preparar revisiones entre pares
- PE 2.2: Dirigir la revisiones entre pares
- PE 2.3: Analizar datos de revisiones entre pares

ME3: Verificar productos de trabajo seleccionados
- PE 3.1: Realizar la verificación
- PE 3.2: Analizar los resultados de verificación e identificar acciones correctivas

AP: Validación

ME1: Preparar la validación
- PE 1.1: Seleccionar los productos a validar
- PE 1.2: Establecer el entorno de validación
- PE 1.3: Establecer procedimientos y criterios de validación

ME2: Validar el producto o los componentes del producto
- PE 2.1: Realizar la validación
- PE 2.2: Analizar los resultados de la validación

AP: Enfoque de Procesos Organizacionales

ME1: Determinar oportunidades de mejora de procesos
- PE 1.1: Establecer necesidades de procesos organizacionales
- PE 1.2: Valorar los procesos de la organización
- PE 1.3: Identificar mejoras en los procesos de la organización

ME2: Planear e implementar actividades de mejora de procesos
- PE 2.1: Establecer planes de acción de procesos
- PE 2.2: Implementar planes de acción de procesos
- PE 2.3: Desplegar los activos de procesos organizacionales
- PE 2.4: Incorporar experiencias relacionadas a los procesos en el activo de procesos organizacionales

AP: Definición de Procesos Organizacionales

ME1: Establecer activos de procesos organizacionales
- PE 1.1: Establecer procesos estándar
- PE 1.2: Establecer descripciones del ciclo de vida del modelo
- PE 1.3: Establecer criterios y guías a la medida

PE 1.4: Establecer repositorio de medidas de la organización
PE 1.5: Establecer librería de activos de los procesos de la organización

AP: Entrenamiento Organizacional

ME1: Establecer una capacidad organizacional de entrenamiento
PE 1.1: Establecer las necesidades estratégicas de entrenamiento
PE 1.2: Determinar cuales necesidades de entrenamiento son responsabilidad de la organización
PE 1.3: Establecer una plan organizacional táctico de entrenamiento.
PE 1.4: Establecer la capacidad de entrenamiento

ME2: Proveer el entrenamiento necesario
PE 2.1: Proporcionar el entrenamiento
PE 2.2: Establecer registros de entrenamiento
PE 2.3: Determinar la efectividad del entrenamiento

AP: Administración Integrada de Proyecto

ME1: Utilizar los procesos definidos del proyecto
PE 1.1: Establecer los procesos definidos del proyecto
PE 1.2: Utilizar activos de procesos de la organización para planear actividades del proyecto
PE 1.3: Integrar planes
PE 1.4: Administrar planes utilizando los planes integrados
PE 1.5: Contribuir al activo de procesos de la organización

ME2: Coordinar y colaborar con los interesados relevantes
PE 2.1: Administrar la implicación de los interesados
PE 2.2: Administrar dependencias
PE 2.3: Resolver casos de coordinación

AP: Administración de Riesgos

ME1: Preparar la administración de riesgos
PE 1.1: Determinar las fuentes y las categorías de los riesgos
PE 1.2: Definir parámetros de riesgos
PE 1.3: Establecer una estrategia de administración de riesgos

ME2: Identificar y analizar riesgos
PE 2.1: Identificar riesgos
PE 2.2: Evaluar, priorizar y categorizar riesgos

ME3: Mitigar riesgos
PE 3.1: Desarrollar planes de mitigación de riesgos
PE 3.2: Implementar planes de mitigación de riesgos

AP: Análisis de Decisión y Resolución

ME1: Evaluar alternativas
PE 1.1: Establecer guías para análisis de decisión
PE 1.2: Establecer criterio de evaluación
PE 1.3: Identificar alternativas de solución
PE 1.4: Seleccionar métodos de evaluación
PE 1.5: Evaluar alternativas
PE 1.6: seleccionar soluciones

NIVEL DE MADUREZ 4

MG3: Institucionalizar un proceso definido
PG 2.1 Establecer una política organizacional (AR1)
PG 3.1 Establecer un proceso definido (HS5)
PG 2.2 Planear el proceso (HS1)
PG 2.3 Proveer recursos (HS2)
PG 2.4 Asignar Responsabilidades (HS3)
PG 2.5 Capacitar Personal (HS4)
PG 2.6 Administrar Configuraciones (DI1)
PG 2.7 Identificar e implicar a todos los interesados (DI2)
PG 2.8 Monitorear y controlar el proceso (DI3)
PG 3.2 Colectar información de mejora (DI4)
PG 2.9 Evaluar objetivamente el apego al proceso (VI1)
PG 2.10 Revisar el status con administradores de alto nivel (VI2)

AP: Desempeño de Procesos de la Organización

ME1: Establecer líneas base y modelos de desempeño
PE 1.1: Seleccionar procesos
PE 1.2: Establecer medidas del desempeño de procesos
PE 1.3: Establecer objetivos de calidad y de desempeño del proyecto
PE 1.4: Establecer líneas base de desempeño del proceso
PE 1.5: Establecer modelos de desempeño del proceso

AP: Administración Cuantitativa de Proyectos

ME1: Administrar cuantitativamente el proyecto
PE 1.1: Establecer objetivos del proyecto
PE 1.2: Componer el proceso definido
PE 1.3: Seleccionar los subprocesos que serán administrados estadísticamente
PE 1.4: Administrar el desempeño del proceso

ME2: Administrar estadísticamente el desempeño de subprocesos
PE 2.1: Seleccionar mediciones y técnicas analíticas
PE 2.2: Aplicar métodos estadísticos para comprender variaciones
PE 2.3: Monitorear el desempeño de los subprocesos seleccionados
PE 2.4: Registrar datos de administración estadística

NIVEL DE MADUREZ 5

MG3: Institucionalizar un proceso definido
PG 2.1 Establecer una política organizacional (AR1)
PG 3.1 Establecer un proceso definido (HS5)
PG 2.2 Planear el proceso (HS1)
PG 2.3 Proveer recursos (HS2)
PG 2.4 Asignar Responsabilidades (HS3)
PG 2.5 Capacitar Personal (HS4)
PG 2.6 Administrar Configuraciones (DI1)
PG 2.7 Identificar e implicar a todos los interesados (DI2)
PG 2.8 Monitorear y controlar el proceso (DI3)
PG 3.2 Colectar información de mejora (DI4)
PG 2.9 Evaluar objetivamente el apego al proceso (VI1)
PG 2.10 Revisar el status con administradores de alto nivel (VI2)

AP: Innovación y Despliegue Organizacional
ME1: Seleccionar mejoras
- PE 1.1: Recolectar y analizar las mejoras propuestas
- PE 1.2: Identificar y analizar innovaciones
- PE 1.3: Mejoras experimentales
- PE 1.4: Seleccionar las mejoras para desplegar

ME2: Desplegar mejoras
- PE 2.1: Planear el despliegue
- PE 2.2: Administrar el despliegue
- PE 2.3: Medir efecto de la mejora

AP: Análisis Causal y Resolución
ME1: Determinar las causas de defectos
- PE 1.1: Seleccionar defectos a analizar
- PE 1.2: Analizar causas

ME2: Tratar las causas de los defectos
- PE 2.1: Implementar acciones propuestas
- PE 2.2: Evaluar los efectos de los cambios
- PE 2.3: Registrar datos

4.8. Representación Continua de CMMI

El Modelo Continuo de CMMI se describe en un documento que contiene seis capítulos que dan una visión general del mismo:

1. Introducción
2. Estructura del Modelo
3. Terminología del Modelo
4. Niveles de Capacidad y Componentes Genéricos del modelo
5. Interpretación del Modelo
6. Utilización del Modelo

Además de otros dos capítulos con explicación más detallada:

7. Áreas de Procesos

    - Administración de Procesos
    - Administración de Proyectos
    - Ingeniería
    - Soporte

    Apéndices

    - Referencias
    - Acrónimos
    - Glosario
    - Elementos requeridos y esperados del Modelo
    - Participantes del Proyecto CMMI
    - Equivalencias con la Representación Escalonada

La Representación Continua esta basada en la Capacidad de Procesos de cada Área de Procesos, esto se puede ver como una evaluación de dos dimensiones:

Figura 4-4: Evaluación bidimensional de la representación continua de CMMI

En la dimensión de procesos se indica "qué" se hace, especificado en las veintidós Áreas de Procesos. Mientras que en la dimensión de la capacidad se indica "qué tan bien" se realizan los procesos de un Área de Procesos, valorado esto en seis Niveles de Capacidad, que indica un estado evolutivo bien definido que describe la capacidad de los procesos y cuyos nombres y número son muy similares a los niveles de capacidad de ISO/IEC 15504. Cada nivel es la base

para alcanzar los niveles superiores, de esta forma no se pueden saltar niveles en la evolución de la mejora de procesos. Los Niveles de Capacidad son los siguientes:

0. Incompleto
1. Realizado
2. Administrado
3. Definido
4. Administrado cuantitativamente
5. Optimizado

Considerando esto, el perfil de procesos de una empresa puede ser representado de la siguiente manera:

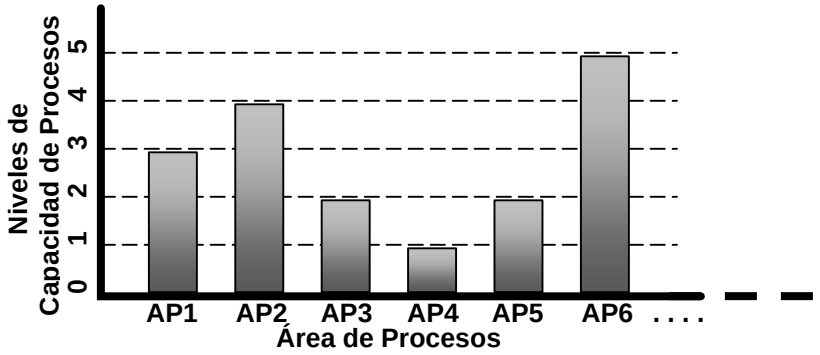


Figura 4-5: Ejemplo del perfil de procesos de una organización en la Representación Continua de CMMI

Esta representación también incluye Metas Genéricas, Prácticas Genéricas, Metas Específicas y Prácticas Específicas, y estas se organizan de la siguiente manera:

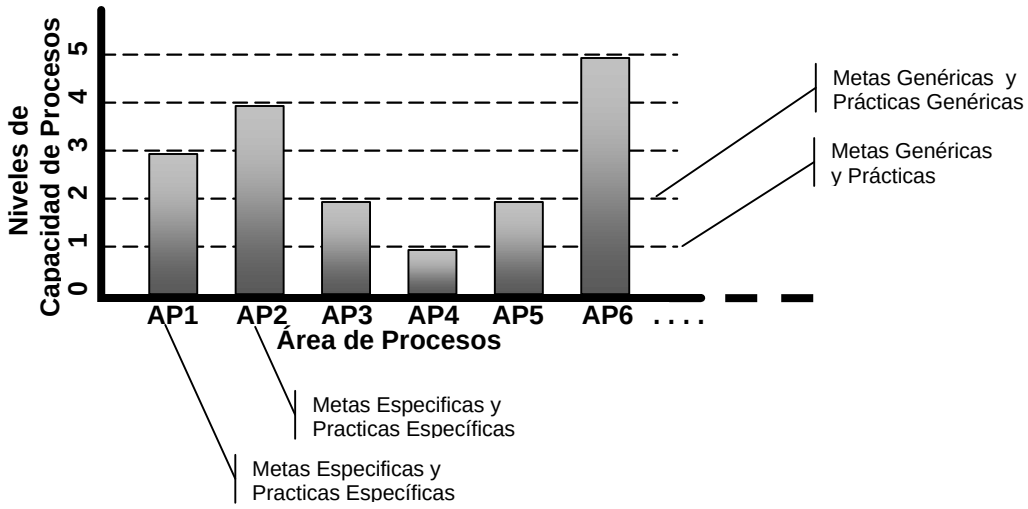


Figura 4-6: Metas y Prácticas en un perfil continuo de CMMI

De forma similar a la Representación Escalonada, en esta representación las Metas Específicas y las Prácticas Específicas se asocian a las Áreas de Procesos, sin embargo difiere de esta en que cada Practica Especifica esta conectada acierto Nivel de Capacidad, y solamente para este nivel y los superiores, esta practica es aplicada. Por otro lado, las Metas Generales y Prácticas Generales se relacionan a los Niveles de Capacidad.

La Representación Continua ofrece la flexibilidad a las organizaciones de elegir que Áreas de procesos mejorar, así como el grado al cual desea mejorar cada una de ellas. Esto posibilita elegir el orden de mejora que mas se adapte a los objetivos de negocio de la organización. Esta representación consiste también de las veintidos Áreas de Procesos de la Representación Escalonada, pero ya no se agrupan en Niveles de Madurez, sino en cuatro Categorías de Áreas de Procesos:

Categoría	Área de Procesos
Administración de Procesos	1. Enfoque de Procesos Organizacionales 2. Definición de Procesos Organizacionales 3. Entrenamiento Organizacional 4. Desempeño de Procesos de la Organización 5. Innovación Organizacional y Despliegue
Administración de Proyectos	6. Planeación del Proyecto 7. Monitoreo y Control del Proyecto 8. Administrador de acuerdos con proveedores 9. Administración Integrada de Proyecto 10. Administración de Riesgos 11. Administración Cuantitativa de Proyectos
Ingeniería	12. Administración de Requerimientos 13. Desarrollo de Requerimientos 14. Solución Técnica 15. Integración del Producto 16. Verificación 17. Validación
Soporte	18. Administración de la Configuración. 19. Aseguramiento de la calidad de procesos y productos 20. Medición y Análisis 21. Análisis de Decisión y Resolución 22. Análisis Causal y Resolución

Tabla 4-4: Categorías de Áreas de Proceso en la representación escalonada de CMMI

En la Figura 4-7 se muestra la estructura de los elementos en la Representación Continua:

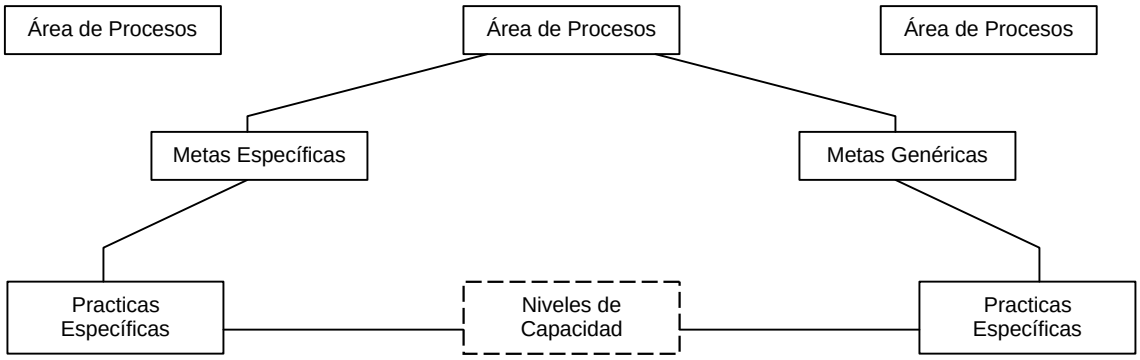


Figura 4-7: Organización de elementos en la Representación Continua de CMMI

En la Figura 4-7 se muestra que las Metas Especificas organizan Prácticas Especificas, y que las Metas Genéricas organizan Prácticas Genéricas. Como ya se mencionó, cada una de las Metas Genéricas y Prácticas Genéricas, corresponde a un Nivel de Capacidad, mientras que las Metas Específicas y las Prácticas Específicas aplican a un Área de Procesos individual. Las Metas Genéricas y las Prácticas Genéricas aplican a múltiples Áreas de Procesos. Las Metas Genéricas y las Prácticas Genéricas definen una secuencia de Niveles de Capacidad que representan mejoras en la implementación y efectividad de todos los procesos que se eligen para ser mejorados. En esta representación, a todas las Prácticas Especificas en el Nivel de Capacidad 1, se les llama Prácticas Base; mientras que a todas las Prácticas Especificas en el Nivel de Capacidad 2 o superior, se les llama Practicas Avanzadas [Bowen et al., 2002].

4.9. Esquema de Representación Continua del Modelo CMMI-SW

A continuación se muestran los componentes genéricos del modelo para software en su Representación Continua por Niveles de Capacidad, utilizando las abreviaciones definidas para la Representación Escalonada:

Nivel de Capacidad	Meta Genérica	Práctica Genérica
Nivel de Capacidad 0 Incompleto		
Nivel de Capacidad 1 Realizado	MG1: Satisfacer las Metas Genéricas	PG1: Realizar las Practicas Base
Nivel de Capacidad 2 Administrado	MG 2: Institucionalizar un proceso administrado	PG 2.1 Establecer una política organizacional (AR1) PG 2.2 Planear el proceso (HS1) PG 2.3 Proveer recursos (HS2) PG 2.4 Asignar Responsabilidades (HS3) PG 2.5 Capacitar Personal (HS4) PG 2.6 Administrar Configuraciones (DI1) PG 2.7 Identificar e implicar a todos los interesados (DI2)

Nivel de Capacidad	Meta Genérica	Práctica Genérica
		PG 2.8 Monitorear y controlar el proceso (DI3) PG 2.9 Evaluar objetivamente el apego al proceso (VI1) PG 2.10 Revisar el status con administradores de alto nivel (VI2)
Nivel de Capacidad 3 Definido	MG3: Institucionalizar un proceso definido	PG 3.1 Establecer un proceso definido (HS5) PG 3.2 Colectar información de mejora (DI4)
Nivel de Capacidad 4 Administrado cuantitativamente	MG4: Institucionalizar un proceso administrado cuantitativamente	PG 4.1 Establecer objetivos cuantitativos para los procesos PG 4.2 Estabilizar el desarrollo de los subprocesos
Nivel de Capacidad 5 Optimizado	MG5: Institucionalizar un proceso optimizado	PG 5.1 Asegurar la mejora continua de procesos PG 5.2 Corregir las causas de problemas

Figura 4-8: Componentes genéricos por Nivel de Capacidad en la Representacion Contnua de CMMI

Por otro lado, y también utilizando las abreviaturas de la sección anterior[1], se muestra a continuación las Metas y Prácticas Especificas de cada Área de Procesos organizadas por categoría.

Administración de Procesos		
Área de Procesos	**Meta Especifica**	**Práctica Especifica**
Enfoque de Procesos Organizacionales	ME1: Determinar oportunidades de mejora de procesos	PE 1.1-1: Establecer necesidades de procesos organizacionales PE 1.2-1: Valorar los procesos de la organización PE 1.3-1: Identificar mejoras en los procesos de la organización
	ME2: Planear e implementar actividades de mejora de procesos	PE 2.1-1: Establecer planes de acción de procesos PE 2.2-1: Implementar planes de acción de procesos

[1] También se respeta la misma numeración de la Representación Escalonada, peor se agrega un guión y un numero para indicar el Nivel de Capacidad de la Practica Especifica

		PE 2.3-1: Desplegar los activos de procesos organizacionales PE 2.4-1: Incorporar experiencias relacionadas a los procesos en el activo de procesos organizacionales
Definición de Procesos Organizacionales	ME1: Establecer activos de procesos organizacionales	PE 1.1-1: Establecer procesos estándar PE 1.2-1: Establecer descripciones del ciclo de vida del modelo PE 1.3-1: Establecer criterios y guías a la medida PE 1.4-1: Establecer repositorio de medidas de la organización PE 1.5-1: Establecer librería de activos de los procesos de la organización
Entrenamiento Organizacional	ME1: Establecer una capacidad organizacional de entrenamiento	PE 1.1-1: Establecer las necesidades estratégicas de entrenamiento PE 1.2-1: Determinar cuales necesidades de entrenamiento son responsabilidad de la organización PE 1.3-1: Establecer un plan organizacional táctico de entrenamiento. PE 1.4-1: Establecer la capacidad de entrenamiento
	ME2: Proveer el entrenamiento necesario	PE 2.1-1: Proporcionar el entrenamiento PE 2.2-1: Establecer registros de entrenamiento PE 2.3-1: Determinar la efectividad del entrenamiento
Desempeño de Procesos de la Organización	ME1: Establecer líneas base y modelos de desempeño	PE 1.1-1: Seleccionar procesos PE 1.2-1: Establecer medidas del desempeño

		de procesos PE 1.3-1: Establecer objetivos de calidad y de desempeño del proyecto PE 1.4-1: Establecer líneas base de desempeño del proceso PE 1.5-1: Establecer modelos de desempeño del proceso
Innovación y Despliegue Organizacional	ME1: Seleccionar mejoras	PE 1.1-1: Recolectar y analizar las mejoras propuestas PE 1.2-1: Identificar y analizar innovaciones PE 1.3-1: Mejoras experimentales PE 1.4-1: Seleccionar las mejoras para desplegar
	ME2: Desplegar mejoras	PE 2.1-1: Planear el despliegue PE 2.2-1: Administrar el despliegue PE 2.3-1: Medir efecto de la mejora
Administración de Proyectos		
Área de Procesos	**Meta Especifica**	**Práctica Especifica**
Planeación del Proyecto	ME1: Establecer estimaciones	PE 1.1-1 Estimar el alcance del proyecto PE 1.2-1 Establecer estimaciones del producto de trabajo y de las características de las tareas PE 1.3-1 Definir el Ciclo de Vida del Proyecto PE 1.4-1 Determinar estimaciones de esfuerzo y costo
	ME2: Desarrollar un Plan del Proyecto	PE 2.1-1 Establecer el presupuesto y el calendario PE 2.2-1 Identificar los riesgos del proyecto PE 2.3-1 Plan para la Administración de Datos PE 2.4-1 Plan para los Recursos del Proyecto PE 2.5-1 Plan para las Habilidades y Conocimientos

		Necesarios PE 2.6-1 Plan para Implicar a los Interesados PE 2.7-1 Establecer un Plan del Proyecto
	ME3: Obtener acuerdos para el Plan	PE 3.1-1 Revisar planes que afecten al proyecto PE 3.2-1 Reconciliar los niveles de recursos y de trabajo PE 3.3-1 Obtener compromisos son el plan
Monitoreo y Control del Proyecto	ME1: Monitorear el proyecto contra el plan	PE 1.1-1 Monitorear los parámetros planeados del proyecto PE 1.2-1 Monitorear acuerdos PE 1.3-1 Monitorear riesgos del proyecto PE 1.4-1 Monitorear administración de datos PE 1.5-1 Monitorear la implicación de los interesados PE 1.6-1 Conducir revisiones al proyecto PE 1.7-1 Conducir revisiones en milestones[1]
	ME2: Administrar las acciones correctivas	PE 2.1-1 Analizar el caso PE 2.2-1 Tomar una acción correctiva PE 2.3-1 Administrara la acción correctiva
Administrador de acuerdos con proveedores	ME1: Establecer acuerdos con proveedores	PE 1.1-1 Determinar el tipo de adquisición PE 1.2-1 Seleccionar proveedor PE 1.3-1: Establecer acuerdos con el proveedor
	ME2: Satisfacer acuerdos con proveedores	PE 2.1-1 Revisar ofertas de productos PE 2.2-1 Ejecutar acuerdos con proveedores PE 2.3-1 Aceptar el producto adquirido PE 2.4-1 Transición del

[1] Los milestones se refieren a revisiones preestablecidas de acuerdo a periodos de tiempo o al logro de determinados objetivos parciales durante el desarrollo del proyecto.

		producto
Administración Integrada de Proyecto	ME1: Utilizar los procesos definidos del proyecto	PE 1.1-1 Establecer los procesos definidos del proyecto PE 1.2-1 Utilizar activos de procesos de la organización para planear actividades del proyecto PE 1.3-1 Integrar planes PE 1.4-1 Administrar planes utilizando los planes integrados PE 1.5-1 Contribuir al activo de procesos de la organización
	ME2: Coordinar y colaborar con los interesados relevantes	PE 2.1-1 Administrar la implicación de los interesados PE 2.2-1 Administrar dependencias PE 2.3-1 Resolver casos de coordinación
Administración de Riesgos	ME1: Preparar la administración de riesgos	PE 1.1-1 Determinar las fuentes y las categorías de los riesgos PE 1.2-1 Definir parámetros de riesgos PE 1.3-1 Establecer una estrategia de administración de riesgos
	ME2: Identificar y analizar riesgos	PE 2.1-1 Identificar riesgos PE 2.2-1 Evaluar, priorizar y categorizar riesgos
	ME3: Mitigar riesgos	PE 3.1-1 Desarrollar planes de mitigación de riesgos PE 3.2-1 Implementar planes de mitigación de riesgos
Administración Cuantitativa de Proyectos	ME1: Administrar cuantitativamente el proyecto	PE 1.1-1 Establecer objetivos del proyecto PE 1.2-1 Componer el proceso definido PE 1.3-1 Seleccionar los subprocesos que serán administrados estadísticamente PE 1.4-1 Administrar el

		desempeño del proceso
	ME2: Administrar estadísticamente el desempeño de subprocesos	PE 2.1-1 Seleccionar mediciones y técnicas analíticas PE 2.2-1 Aplicar métodos estadísticos para comprender variaciones PE 2.3-1 Monitorear el desempeño de los subprocesos seleccionados PE 2.4-1 Registrar datos de administración estadística
Ingeniería		
Área de Procesos	**Meta Especifica**	**Práctica Especifica**
Administración de Requerimientos	ME1: Administrar requerimientos	PE 1.1-1 Obtener una comprensión de requerimientos PE 1.2-2 Obtener un acuerdo sobre los requerimientos PE 1.3-1 Administrar cambios a los requerimientos PE 1.4-2 Mantener una trazabilidad bidireccional de los requerimientos PE 1.5-1 Identificar inconsistencias entre el trabajo del proyecto y los requerimientos
Desarrollo de Requerimientos	ME1: Desarrollar requerimientos del cliente	PE 1.1-1 Colectar necesidades de los interesados PE 1.1-2 Obtener necesidades PE 1.2-1 Desarrollar requerimientos del cliente
	ME2: Desarrollar requerimientos del producto	PE 2.1-1 Establecer requerimientos de producto y de sus componentes PE 2.2-1 Asigne los requerimientos de los componentes del producto PE 2.3-1 Identifique requerimientos de interfaz

	ME3: Analizar y validar los requerimientos	PE 3.1-1 Establecer los conceptos y escenarios operacionales PE 3.2-1: Establecer una descripción de la funcionalidad requerida PE 3.3-1: Analizar requerimientos PE 3.4-3 Analizar Requerimientos y alcanzar el balance PE 3.5-1 Validar requerimientos PE 3.5-2 Validar requerimientos con métodos comprensivos
Solución Técnica	ME1: Seleccionar soluciones de los componentes del producto	PE 1.1-1 Desarrollar soluciones alternativas y criterios de selección PE 1.1-2 Desarrollar detalladamente las soluciones alternativas y los criterios de selección PE 1.2-2 Desarrollar los conceptos y escenarios operacionales PE 1.3-1 Seleccionar la solución de los componentes del producto
	ME2 Desarrollar el diseño	PE 2.1-1 Diseñar el producto o los componentes del producto PE 2.2-3 Establecer un paquete técnico de datos PE 2.3-1 Establecer descripciones de interfaces PE 2.3-3 Diseñar las interfaces utilizando criterios PE 2.4-3 Realizar un análisis de elaboración, compra o reutilización de componentes del producto
	ME3: Implementar el diseño del producto	PE 3.1-1 Implementar el diseño PE 3.2-1 Desarrollar la documentación de soporte del producto

Integración del Producto	ME1: Preparar la integración del producto	PE 1.1-1 Determinar la secuencia de integración PE 1.2-2 Establecer el ambiente de integración del producto PE 1.3-3 Establecer los procedimientos y criterios de integración del producto
	ME2: Asegurar la compatibilidad de interfaces	PE 2.1-1 Revisar que las descripciones de interfaces estén completas PE 2.2-1 Administrar interfaces
	ME3: Ensamblar los componentes del producto y liberar el producto	PE 3.1-1 Confirmar que los componentes del producto están preparados para la integración PE 3.2-1 Ensamblar los componentes del producto PE 3.3-1 Evaluar componentes del producto ensamblados PE 3.4-1 Empaquetado y liberación del producto o de los componentes del producto
Verificación	ME1: Preparar la verificación	PE 1.1-1 Seleccionar productos de trabajo a verificar PE 1.2-2 Establecer el entorno de verificación PE 1.3-3 Establecer procedimientos y criterios de verificación
	ME2: Realizar revisiones entre pares	PE 2.1-1 Preparar revisiones entre pares PE 2.2-1 Dirigir la revisiones entre pares PE 2.3-2 Analizar datos de revisiones entre pares
	ME3: Verificar productos de trabajo seleccionados	PE 3.1-1 Realizar la verificación PE 3.2- 2Analizar los resultados de verificación e identificar

		acciones correctivas
Validación	E1: Preparar la validación	PE 1.1-1 Seleccionar los productos a validar PE 1.2-2 Establecer el entorno de validación PE 1.3-3 Establecer procedimientos y criterios de validación
	E2: Validar el producto o los componentes del producto	PE 2.1-1 Realizar la validación PE 2.2-1 Analizar los resultados de la validación
Soporte		
Área de Procesos	**Área de Procesos**	**Área de Procesos**
Administración de la Configuración	ME1: Establecer Líneas Base	PE 1.1-1 Identificar objetos de la configuración PE 1.2-1 Establecer un sistema de administración de configuración PE 1.3-1 Crear y liberar Líneas Base
	ME 2: Seguimiento y control de cambios	PE 2.1-1 Seguimiento de solicitudes de cambio PE 2.2-1 Control de objetos de configuración
	ME3: Establecer integridad	PE 3.1-1 Establecer registros de administración de configuración PE 3.2-1 Realizar auditorias a la configuración
Aseguramiento de la calidad de procesos y productos	ME1: Evaluar objetivamente procesos y productos de trabajo	PE 1.1-1 Evaluar objetivamente procesos PE 1.2-1 Evaluar objetivamente productos de trabajo y servicios
	ME2: Proveer una visión objetiva	PE 2.1-1 Comunicar y asegurara la resolución de casos que no cumplieron PE 2.2-1 Establecer registros
Medición y Análisis	ME1: Alinear medidas y actividades de análisis	PE 1.1-1 Establecer objetivos de medidas PE 1.2-1 Especificar

		medidas PE 1.3-1 Especificar colección de datos y procedimientos de almacenamiento PE 1.4-1 Especificar procedimientos de análisis
	ME2: Proveer resultados de medidas	PE 2.1-1 Colectar datos de medidas PE 2.2-1 Analizar datos de medidas PE 2.3-1 Almacenar datos y resultados PE 2.4-1 Comunicar datos
Análisis de Decisión y Resolución	ME1: Evaluar alternativas	PE 1.1-1 Establecer guías para análisis de decisión PE 1.2-1 Establecer criterio de evaluación PE 1.3-1 Identificar alternativas de solución PE 1.4-1 Seleccionar métodos de evaluación PE 1.5-1 Evaluar alternativas PE 1.6-1 seleccionar soluciones
Análisis Causal y Resolución	ME1: Determinar las causas de defectos	PE 1.1-1 Seleccionar defectos a analizar PE 1.2-1 Analizar causas
	ME2: Tratar las causas de los defectos	PE 2.1-1 Implementar acciones propuestas PE 2.2-1 Evaluar los efectos de los cambios PE 2.3-1 Registrar datos

Figura 4-9: Metas y Prácticas especificas por categorías de Áreas de Procesos

4.10. Comentarios finales sobre CMMI

CMMI, con sus dos representaciones, ofrece dos alternativas para interpretarlo e implementarlo dentro de una organización, la elección de cuál de las dos representaciones utilizar depende mucho del status de la organización y de los responsables de implementar CMMI. Esto es, para organizaciones que ya se encuentran en el Nivel de Madurez 2 de CMM para Software, le será mas fácil adoptar la Representación Escalonada debido a que le será mas familiar; por otro lado si la organización ya utiliza el modelo ISO 15504, le será mas familiar la Representación Continua, así mismo para organizaciones nuevas, es recomendable utilizar la Representación Continua de CMMI, que permite elegir cuáles Áreas de Procesos mejorar.

Sobre las Áreas de Procesos, se puede comentar que es muy difícil que una organización encuentre una relación directa de sus procesos con las Áreas de Procesos de CMMI, por lo cual debe iniciar en hacer un mapeo de sus procesos a las Áreas de Procesos que mas se parezcan en CMMI.

Segunda Parte

PROPUESTA

5

Propuestas sobre Roles

Este capítulo describen los roles que participan dentro de la metodología propuesta, tomando en cuenta las bases teóricas del Proceso Unificado y de CMMI.

5.1. Introducción

Desarrollar un sistema de software de calidad es un proceso complejo que requiere de un grupo de personas, pues se necesitan diversos conocimientos y habilidades que no se encuentran todas en una sola persona. Así, cada uno de los integrantes del grupo aportará parte del total de las habilidades y conocimientos necesarios para el desarrollo del sistema de acuerdo a su experiencia y capacidades personales. La aportación de capacidades y habilidades se define formalmente en un "Rol".

Con los roles que se proponen se busca guiar a los involucrados en un proyecto de desarrollo de software en su participación. Los roles están basados en los mencionados en Proceso Unificado y en CMMI, pero se busca hacer un conjunto mas reducido de roles, que mantenga claridad de organización y consistencia en la definición de sus actividades.

5.2. Comparación de roles del Proceso Unificado y roles de CMMI

A continuación se hará una comparación de los roles propuestos por el Proceso Unificado y los roles propuesto por CMMI, con el propósito de encontrar un conjunto mínimo de roles que cubra las necesidades de ambos.

Como se mencionó en el Capítulo 3, los roles principales que marca el Proceso Unificado son:

1. Analista de Sistemas
2. Especificador de Casos de Uso
3. Diseñador de Interfaz de Usuario
4. Arquitecto
5. Ingeniero de Casos de Uso
6. Ingeniero de Componentes
7. Integrador del Sistema
8. Diseñador de Pruebas
9. Ingeniero de Pruebas de Integración
10. Ingeniero de Pruebas del Sistema

Además, los roles secundarios del Proceso Unificado son:

1. Usuarios
2. Ejecutivo de Software
3. Jefe de Proyecto
4. Asesor

Por su parte CMMI, a pesar de que en su objetivo no se contempla el listado explicito de roles específicos, en su modelo escalonado de ingeniería de software se nombran los siguientes roles:

1. Proveedor o Contratista (Contractor o supplier)
2. Cliente (Customer):
3. Administrador (Manager)
4. Administrador del Proyecto (Project Manager)
5. Administrador Principal (Senior Manager)

6. Interesados (Stakeholder)
7. Interesados relevantes (Relevant Stakeholder):

En donde Administrador (Manager) es un rol genérico para designar a una persona que provee control y dirección técnica y administrativa dentro de un área de responsabilidad. Por otro lado, Interesados (Stakeholder) e Interesados relevantes (Relevant Stakeholder) también son roles genéricos que se refieren a personas que tienen interés de algún tipo en el proyecto y a personas que tienen interés y además actividades de algún tipo en el proyecto, respectivamente. Estos roles genéricos no se tomarán en cuenta.

CMMI se basa en Áreas de Procesos que, como ya se mencionó en el Capítulo 4, son grupos de prácticas relacionadas dentro de un área, por lo cual, aunque no definen roles específicos, si se pueden tomar como entidades que agrupan actividades de roles específicos. A su vez, las Áreas de Proceso también se agrupan en 5 categorías, la siguiente tabla muestra las categorías y Áreas de Procesos para el nivel de madurez 2 de Ingeniería de Software

Categoría	Área de Procesos de Nivel 2
Administración de Procesos	
Administración de Proyectos	1. Planeación del Proyecto 2. Monitoreo y Control del Proyecto 3. Administrador de Acuerdos con Proveedores
Ingeniería	4. Administración de Requerimientos
Soporte	5. Administración de la Configuración. 6. Aseguramiento de la Calidad de Procesos y Productos 7. Medición y Análisis

Tabla 5-1: Categorías de Áreas de Proceso de CMMI

Tomando en consideración lo anterior, los roles y áreas mencionados en el Proceso Unificado y en CMMI pueden ser comparados de la siguiente manera

Categoría CMMI	Área de Procesos CMMI Nivel 2	Roles mencionados en CMMI	Roles mencionados en Proceso Unificado
Administración de Procesos			
Administración de Proyectos	Planeación del Proyecto Monitoreo y Control del Proyecto Administración de Acuerdos con Proveedores	Administrador del Proyecto (Project Manager)	Jefe de Proyecto
Ingeniería	Administración de Requerimientos		Especificador de casos de uso Analista de sistemas Arquitecto Ingeniero de casos de uso Diseñador de interfaz de usuario Ingeniero de componentes Integrador del sistema Diseñador de pruebas Ingeniero de pruebas de integración
Soporte	Administración de la Configuración. Aseguramiento de la Calidad de Procesos y Productos Medición y Análisis		Ingeniero de pruebas del sistema
Otros que no caen dentro de las categorías de CMMI		Cliente Administrador Principal Proveedor o Contratista	Usuarios Ejecutivo de Software Asesor

Tabla 5-2: Comparación entre roles de CMMI y del Proceso Unificado, organizados por categorías y Áreas de Proceso de CMMI

5.3. Roles adicionales

A pesar de que la comparación anterior incluye a los roles del Proceso Unificado y a los roles mencionados en CMMI, existen algunas Áreas de Procesos que no son cubiertas por rol alguno (ni del Proceso, ni mencionado en CMMI), por lo anterior se propone la Tabla 5-3 con roles que no se listan en el Proceso y en CMMI, pero que se encuentran en una propuesta de Scout E. Donaldson y Stanley G. Siegel [Donalson et al., 2001]. Esta segunda tabla de roles también contiene una columna donde se colocan los roles finales que se utilizan en la metodología propuesta.

Categoría CMMI	Área de Procesos CMMI Nivel 2	Roles mencionados en CMMI	Roles del Proceso Unificado	Roles de Scott E. Donaldson	Roles finales utilizados en la metodología propuesta
Administración de Procesos					
Administración de Proyectos	Planeación del Proyecto Monitoreo y Control del Proyecto Administración de acuerdos con proveedores	Administrador del Proyecto (Project Manager)	Jefe de Proyecto	Administrador del Proyecto (Project Manager) y /o Administrador del Programa (Program Manager)	ADMINISTRADOR DEL PROYECTO
Ingeniería	Administración de Requerimientos			Administrador de Desarrollo (Development Manager)	ADMINISTRADOR DE DESARROLLO
			Arquitecto		
			Especificador de casos de uso		ESPECIFICADOR DE CASOS DE USO
			Diseñador de interfaz de usuario		
			Analista de sistemas		ANALISTA DE SISTEMAS
			Ingeniero de casos de uso		INGENIERO DE CASOS DE USO
			Ingeniero de componentes		INGENIERO DE COMPONENTES
			Integrador del sistema		INTEGRADOR DEL SISTEMA
			Diseñador de pruebas		DISEÑADOR DE PRUEBAS

Categoría CMMI	Área de Procesos CMMI Nivel 2	Roles mencionados en CMMI	Roles del Proceso Unificado	Roles de Scott E. Donaldson	Roles finales utilizados en la metodología propuesta
			Ingeniero de pruebas de integración		INGENIERO DE PRUEBAS DE INTEGRACIÓN
			Ingeniero de pruebas del sistema		
	Administración de la Configuración.				ADMINISTRADOR DE LA CONFIGURACIÓN
Soporte	Aseguramiento de la calidad de procesos y productos			Administrador del Aseguramiento del Producto (Producto Assurance Manager)	ADMINISTRADOR DE LA CALIDAD
	Medición y Análisis				
Otros que no caen dentro de las categorías de CMMI			Usuarios		USUARIOS
		Cliente		Cliente	CLIENTE
		Administrador Principal	Ejecutivo de Software	Administrador Principal (Señor Manager)	ADMINISTRADOR PRINCIPAL
				Administrador de Negocios (Business Manager)	
		Proveedor o Contratista	Asesor		PROVEEDOR O CONTRATISTA
				Editor técnico (Technical Editor)	EDITOR TÉCNICO

Tabla 5-3: Roles finales para la metodología propuesta

Tomando como base la tabla anterior, se organizaron los roles finales como se muestra en la Figura 5-1.

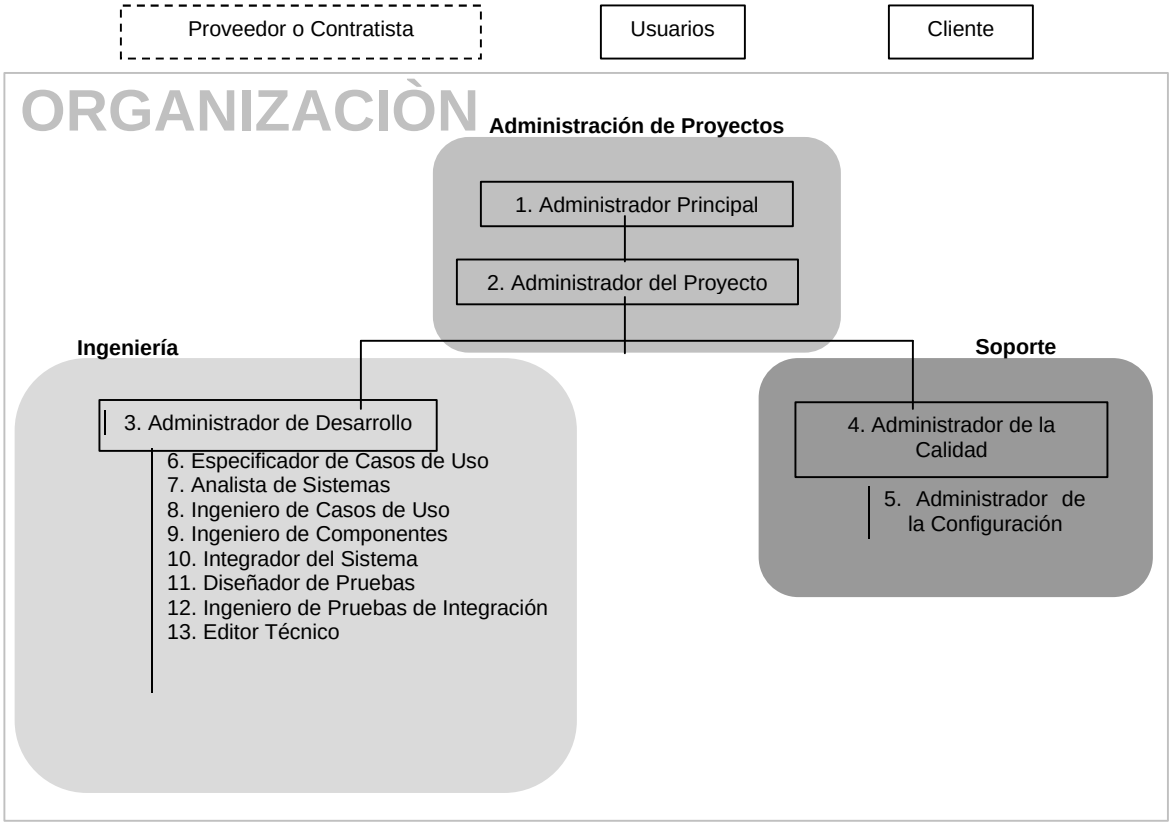


Figura 5-1: Organigrama de roles de la metodología propuesta

En el organigrama se muestran 13 roles internos para una organización de desarrollo de software y 3 roles externos. Los 13 roles internos se agruparon de acuerdo a las categorías de Áreas de Procesos de CMMI.

Se debe aclarar que es posible que no se requieran todos los roles en un proyectos de desarrollo de software, lo que dependerá del tamaño y del tipo de desarrollo. También es posible que una persona tome las actividades y responsabilidades de más de un rol al mismo tiempo (por ejemplo, en proyectos de desarrollo muy pequeños), sin embargo, esto debe ser tratado con cuidado, pues es importante que la persona conozca completamente los objetivos, actividades y responsabilidades de los roles.

Así mismo, es posible tener más de una persona desarrollando las actividades de un mismo rol (por ejemplo, en proyectos de desarrollo grandes), pero en este caso, además de tener presentes los objetivos, actividades y responsabilidades del rol, se debe mantener una comunicación sólida entre las personas que conformen el rol.

5.4. Descripción de roles propuestos

A continuación se describen brevemente cada uno de los roles que finalmente se proponen. Sin embargo esto es solo a manera de descripción breve, pues dentro de la metodología se proponen documentos de descripción de roles que deben detallar el objetivo del rol, sus actividades, el perfil de las personas que lo tomen y sus responsabilidades, entre otros.

Las descripciones se organizan de acuerdo a las categorías de las Áreas de Procesos de CMMI:

5.4.1. Área de Administración del Proyecto

1. Administrador Principal:
   Es el rol que provee la dirección a nivel organizacional y es responsable de promulgar directivas y políticas que regirán a la organización, incluyendo los proyectos dentro de ella. También es quién concreta los proyectos con los clientes y promueve las condiciones necesarias para que concluyan con éxito. Este rol valora la complejidad, costo e importancia de los proyectos dentro de la organización y asigna los recursos necesarios, también asigna al proyecto un Administrador de Proyecto apropiado. A este rol le reporta directamente el Administrador del Proyecto.

2. Administrador del Proyecto:
   El Administrador del Proyecto provee dirección continua al grupo de ingeniería y de soporte durante todo el desarrollo del producto y es responsable del éxito del proyecto. También distribuye los recursos de manera adecuada entre ingeniería y soporte, y promueve una interacción continua y equilibrada entre ellos. A él le reportan directamente el Administrador de Desarrollo y el Administrador de Calidad.

5.4.2. Área de Ingeniería

3. Administrador de Desarrollo:
   Es el responsable de la dirección de aspectos técnicos del proyecto y del proceso de producción. Selecciona el personal que forma parte del Grupo de Ingeniería, les motiva, asigna sus tareas y lleva el seguimiento y supervisión de los aspectos técnicos de sus actividades para que el proyecto se concrete dentro de las expectativas de presupuesto, tiempo y requerimientos. Dentro de los aspectos técnicos, su función principal es mantener consistente la arquitectura del sistema ante los cambios y correcciones que surjan en los diversos modelos y vistas que se elaboran en las primeras iteraciones del proyecto.

6. Especificador de Casos de Uso:
   Este rol realiza sus actividades al ayudar a detectar requerimientos, detallarlos por medio de Casos de Uso, y establecer la definición visual que tendrá finalmente la interfaz de usuario, pero solamente a nivel de diseño gráfico, ya que la implementación de la interfaz de usuario real es elaborada por el Ingeniero de Componentes; para esto trabaja en conjunto con el Analista de Sistemas, con el Cliente y Usuarios finales del sistema. Reporta directamente al Administrador de Desarrollo.

7. Analista de Sistemas:
   El Analista de Sistemas es el responsable de delimitar el sistema, encontrando los Casos de Uso (requerimientos) y Actores, y asegurando que el Modelo de Casos de Uso este completo y consistente. Trabaja en conjunto con el Especificador de Casos de Uso, el Cliente y el Usuario; y le reporta al Administrador de Desarrollo

8. Ingeniero de Casos de Uso:
   Es responsable tanto del análisis como del diseño de los Casos de Uso, al mismo tiempo que ayuda a esbozar las Clases, Interfaces y Subsistemas que participan en los Casos de Uso,

pero en ultima instancia, estos elementos son responsabilidad del Ingeniero de Componentes. Este rol reporta al Administrador de Desarrollo.

9. Ingeniero de Componentes:
   Cada Ingeniero de Componentes es responsable de una o más Clases del Análisis y de uno o más Paquetes del Análisis, así como de una o más Clases del Diseño, de uno o más Subsistemas del Diseño y de las Interfaces correspondientes. También durante la implementación es responsable de uno o más Componentes, de uno o más Subsistemas de Implementación y de las Interfaces correspondientes. Durante las pruebas es responsable de algunos de los Componentes de Pruebas que automatizan los procedimientos de pruebas. Reporta al Administrador de Desarrollo

10. Integrador del Sistema:
    Participa en la implementación y su principal responsabilidad es, como dice su nombre, la integración del sistema, para lo cual planifica la secuencia de construcciones e integración de construcciones a seguir. Reporta al Administrador de Desarrollo

11. Diseñador de Pruebas:
    Es responsable de la planeación de las pruebas y la definición de los objetivos de cada prueba. Además selecciona y describe los Casos de Prueba y los Procedimientos de Prueba que se necesitan, y evalúa las Pruebas de Integración y del Sistema cuando se ejecutan. Reporta al Administrador de Desarrollo

12. Ingeniero de Pruebas de Integración:
    Es responsable de realizar las Pruebas de Integración y del Sistema cada vez que se crea una Construcción nueva y se agrega al sistema en desarrollo. Las Pruebas de Integración y del Sistema se realizan con base a los Casos de Prueba. Para las Pruebas del Sistema específicamente, se requiere tener conocimiento del comportamiento externo del sistema, mientras que para las Pruebas de Integración si se requiere de cierto conocimiento interno del sistema. También es responsabilidad de él documentar los defectos que encuentre durante la realización de las Pruebas de Integración. Reporta directamente al Administrador de Desarrollo.

13. Editor Técnico:
    El Editor Técnico es responsable de que los documentos realizados no sean ambiguos, utilicen el lenguaje apropiado para la audiencia dirigida, sea internamente y externamente consistente, cumpla los estándares establecidos y que no contenga errores gramaticales ni de ortografía. Trabaja en estrecha relación con el Administrador de la Configuración y reporta al Administrador de Desarrollo.

5.4.3. Área de Soporte

4. Administración de la Calidad:
   El objetivo de este rol, es proveer al Administrador del Proyecto y al Administrador Principal con un conjunto de verificaciones y balances respecto a la integridad del producto y el proceso. Desarrolla sus actividades de forma constructiva, no confrontativa y no tendenciosa. Este rol reporta directamente al Administrador del Proyecto y al Administrador Principal.

5. Administrador de la Configuración:
   Dentro del Área de Soporte, una actividad de gran importancia es la Administración de la Configuración del Software, esta actividad es responsabilidad del Administrador de la Configuración y entre sus responsabilidades se contempla llevar un control formal de los cambios al software y mantener consistentemente las líneas bases del proyecto, así como otros productos de trabajo que no son parte de la Línea Base. Reporta directamente al Administrador de Calidad y al Administrador del Proyecto.

5.4.4. Roles Externos a la Organización

14. Cliente:
    Este rol es externo a la organización de desarrollo de software y es quién en la mayoría de las ocasiones inicia el proceso de desarrollo de software por medio de una solicitud formal del proyecto, especificación de requerimientos y acuerdos con la organización de desarrollo. También es quién, junto con el Usuario, en última instancia acepta algunos productos de trabajo y finalmente el sistema de software completo. Este rol interactúa con la organización de desarrollo vía el Administrador Principal y el Administrador del Proyecto.

15. Usuario:
    Este rol, externo a la organización de desarrollo de software, puede ser cubierto por una persona o personas, o incluso por algún otro sistema, pero independientemente de su naturaleza, su función principal y final es la de interactuar con el sistema en desarrollo, además de ayudar a especificar los requerimientos junto con el cliente, verificar el cumplimiento de los requerimientos en el sistema final y en cierto casos reportar errores del producto final.

16. Proveedor o Contratista:
    Este rol externo a la organización, es opcional y dependerá de la naturaleza del proyecto y sobre todo, de los riesgos que implique el desarrollo para considerarlo o no. Este rol se refiere a cualquier entidad que suministre productos o servicios que serán adquiridos por la organización de desarrollo, lo cual será formalizado con el Administrador del Proyecto por medio de un contrato o acuerdo.

Cada uno de estos 16 roles intervienen en alguna de las actividades que se indican en el flujo de trabajo de la metodología propuesta, y forman el primer bloque de construcción de la misma.

6

Propuestas sobre Artefactos

Este capítulo describe los artefactos que se generan y utilizan dentro de la metodología propuesta, tomando en cuenta las bases teóricas de UML, del Proceso Unificado y de CMMI.

6.1. Introducción

En la especificación del Proceso Unificado se nombran, definen, y se establece la secuencia de elaboración de los artefactos que se utilizan, por ejemplo, se menciona al Modelo de Casos de Uso, se define y se establece que debe ser elaborado por el Analista de Sistemas después de identificar Actores y Casos de Uso. Lo anterior auxilia de sobre manera en la forma que se aplica el proceso. Desgraciadamente y debido a su naturaleza, no sucede lo mismo con CMMI, esto es debido a que se define como un marco de referencia para la calidad de los procesos de desarrollo de software, donde no se definen explícitamente los artefactos que ayuden en el incremento de calidad en los procesos. En lugar de una definición de artefactos a utilizar, se listan solo a manera de ejemplo algunos productos de trabajo típicos de cada Área de Proceso. A partir de este listado de productos típicos de trabajo y de las guías y recomendaciones para mejoras de procesos, en este capítulo se propone un conjunto de artefactos que, junto con los artefactos del Proceso Unificado, formarán parte de la metodología propuesta.

6.2. Organización de Artefactos

Para organizar el total de artefactos que se proponen se recurrirá a los Paquetes UML, que son utilizados como elementos de agrupación.

Básicamente se tiene un Repositorio General que sirve como una agrupación principal para todos los artefactos de una organización de desarrollo de software, incluyendo artefactos de todos los proyectos. Para representar este artefacto se recurre a un estereotipo de los Paquetes UML, representando por la Figura 6-1:

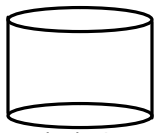


Figura 6-1: Representación gráfica del Repositorio General

Dentro de este repositorio general se tendrán tres paquetes comunes:

- Paquete de Definiciones Generales: donde se contendrá definiciones generales de la metodología, independientes de cualquier proyecto, como por ejemplo, definiciones de políticas, estándares, procedimientos, artefactos, roles y métricas.

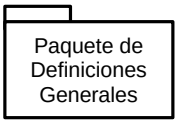


Figura 6-2: Representación gráfica del Paquete de Definiciones Generales

- Paquete de Descripción de Repositorio: que contendrá artefactos que ayuden en la descripción del repositorio (general y de cada proyecto), como por ejemplo, descripción de niveles de promoción de artefactos y la lista de artefactos que se mantendrán bajo la Administración de la Configuración

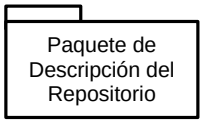


Figura 6-3: Representación gráfica del Paquete de Descripción del Repositorio

- Paquete de Registro Histórico: que contendrá información histórica de acuerdo a la experiencia de proyecto pasados y para apoyo de proyectos futuros. Un ejemplo de lo que puede contener es el registro histórico de las experiencias con distintos proveedores.

Figura 6-4: Representación gráfica del Paquete de Registro Histórico

Además de estos tres paquetes comunes a todos los proyectos, el Repositorio General contendrá repositorios específicos de cada proyecto, que se pueden implementar como particiones del repositorio general, y contendrán información relacionada a cada uno de los proyectos en particular. Para representar un repositorio específico de proyecto genérico se utilizará el siguiente estereotipo de paquete:

Figura 6-5: Representación gráfica del Repositorio del Proyecto

Dentro de cada Repositorio del Proyecto se tendrán dos paquetes:

- Paquete de Artefactos del Producto: que contendrá artefactos relacionados específicamente al producto final que se va a elaborar en el proyecto. Por ejemplo, aquí se contendrán los artefactos del Proceso Unificado que son relativos al producto, como el Modelo de Casos de Uso, el Modelo del Análisis, el Modelo del Diseño, etc. con sus respectivos contenidos.

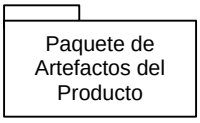


Figura 6-6: Representación gráfica del Paquete de Artefactos del Producto

- Paquete de Artefactos Administrativos: que contendrá artefactos relativos a la administración de los procesos del proyecto, por ejemplo artefactos relacionados a cada una de las Áreas de Procesos de CMMI, como son planes del proyecto, reportes de auditoría de calidad, reportes de revisión al repositorio, informes, minutas, criterios de selección de proveedores, etc.

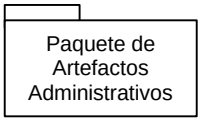


Figura 6-7: Representación gráfica del Paquete de Artefactos Administrativos

Finalmente, dentro del Paquete de Artefactos del Producto se tendrán otros paquetes que organizarán los artefactos ahí contenidos, y que coinciden con los modelos especificados por el Proceso Unificado; mientras que para el Paquete de Artefactos Administrativos se organizarán sus artefactos por medio de paquetes para cada una de las áreas de procesos de CMMI, se tendrá por ejemplo el Paquete de la Administración de Requerimientos, el Paquete de Planeación del Proyecto, etc.

A continuación se proporciona una lista de todos los artefactos propuestos clasificada por los paquetes definidos, tal como se propone que se implementen dentro del Repositorio General de la Configuración de Software:

Repositorio general

Paquete de Definiciones Generales:
Políticas
Estándares
Definición de procedimientos
Definición de roles
Definición de artefactos
Definición de métricas

Paquete de Descripción de Repositorio:
Descripción de niveles de promoción
Lista de productos y elementos de la configuración

Paquete de Registro Histórico:
Registro Histórico de Proveedores

Repositorio de proyecto
Paquete de Artefactos Administrativos
Paquete de la Administración de Requerimientos
Solicitud del proyecto
Inconsistencias con requerimientos
Carta de liberación
Comentarios y observaciones del producto final
Propuesta de respuesta a retroalimentaciones
Paquete de Planeación del Proyecto
Estimaciones del proyecto
Estimado de recursos de cómputo
Organigrama de ingeniería
Cotización del proyecto
Lista de riesgos
Mecanismos para la adquisición de conocimientos y habilidades

- **Plan de desarrollo del proyecto**
- **Plan de aseguramiento de la calidad**
- **Plan de la administración de la configuración**
- **Plan de contención**
- **Plan de contingencia**
- **Plan de desarrollo de proveedor**

Paquete de Monitoreo y Control del Proyecto

- **Contrato del proyecto**
- **Informe de avance técnico**
- **Informe de defectos**

Paquete de la Administración de Proveedores

- **Lista de nuevas herramientas, productos o servicios para adquirir**
- **Requerimientos a externos**
- **Criterios de evaluación de proveedores**
- **Lista de posibles proveedores**
- **Propuestas/ofertas de proveedores**
- **Riesgos y capacidades de proveedor(es) seleccionado(s)**
- **Contrato con proveedor**
- **Lista de artefactos que entregará proveedor**
- **Evaluación de proveedor(es)**

Paquete de Medición y Análisis

- **Métricas de productos**
- **Métricas de recursos**
- **Métricas del proceso**
- **Métricas del proyecto**
- **Interpretación de métricas**
- **Evaluaciones parciales del proyecto**
- **Evaluación general del proyecto**

Paquete de Aseguramiento de la Calidad

- **Reportes de auditoría por actividad**
- **Reportes de auditoría por artefacto**
- **Reportes de auditoría por rol**

Paquete de la Administración de la Configuración

- **Descripción del repositorio**
- **Lista de productos y elementos adicionales de la configuración**
- **Procedimientos especiales de la configuración**
- **Reporte del contenido del repositorio**
- **Informe de la revisión al repositorio**
- **Solicitud de cambio**
- **Análisis del impacto del cambio**

Paquete de Artefactos del Producto

Línea base

- **Modelo del negocio**
- **Modelo del dominio**
  - **Glosario de términos**
- **Modelo de casos de uso**
  - **Especificación de requerimientos**
  - **Casos de uso**
  - **Matriz de traza de requerimientos**
  - **Prototipo de interfaz de usuario**
- **Modelo del análisis**
  - **Realizaciones de Casos de Uso - Análisis**
  - **Clases del análisis**
  - **Paquetes del análisis**
- **Modelo del diseño**

- **Reglas específicas de diseño**
- **Especificación de bibliotecas comunes**
- **Realizaciones de Casos de Uso - Diseño**
- **Clases del diseño**
- **Interfaces diseñadas de clases**
- **Subsistemas de diseño**
- **Interfaces diseñadas de subsistemas**
- **Definición de variables y constantes de configuración de subsistemas**

Modelo de despliegue

- **Especificación de ambientes y herramientas**
- **Proceso de migración**
- **Unidad de despliegue**

Modelo de implementación

- **Componente**
- **Modelo de Datos**
- **Reglas específicas de codificación**
- **Subsistemas de implementación**
- **Interfaces implementadas de componentes**
- **Interfaces implementadas de subsistemas**
- **Plan de la integración de la construcción**
- **Construcción**

Modelo de Pruebas

- **Plan de pruebas**
- **Casos de pruebas**
- **Componente de prueba**
- **Procedimientos de prueba**
- **Evaluación de pruebas (incluida la Evaluación de pruebas de aceptación)**

Material de capacitación

- **Lista de material de soporte y capacitación**
- **Tablas de contenido de manuales, presentaciones y cursos**
- **Manual de usuario**
- **Manual técnico**

En la lista anterior, aparecen en negritas los artefactos utilizados en la metodología, lo que se encuentra escrito sin negritas solo se utiliza para organización de los artefactos. Por ejemplo, el Repositorio General, es el paquete contenedor de todos los artefactos de la empresa, pero no se menciona como artefacto de la metodología propuesta

6.3. Lista y descripción de artefactos

Los artefactos que se proponen en la metodología se listan a continuación con una breve descripción:

1. **Análisis del impacto del cambio**
   Se debe documentar el número de control, referencia a la Solicitud de Cambio o Informe de Defectos que lo originó
   En la sección de Análisis Técnico, se debe describir detalladamente su impacto en la funcionalidad del sistema, en el diseño, en el código, en la documentación técnica y en la definición de las pruebas.
   En la sección del Análisis de Impacto se debe determinar y documentar su costo total (en tiempo, dinero y esfuerzo) y si la propuesta de cambio es acorde a las políticas y estándares del proyecto y del proceso.

Se debe contar con un apartado para plasmar las autorizaciones (en caso de que se autorice) del Administrador del Proyecto, Administrador de la Calidad, Administrador de la configuración y Administrador de Desarrollo.
En caso de no autorizarse, se debe proponer una alternativa de solución o cambio.

2. Carta de liberación

La Carta de Liberación es un documento firmado de conformidad por el Cliente, y elaborado por el Administrador del Proyecto o al Administrador Principal, por medio del cual da por concluido el proyecto en términos legales, y que como mínimo debe contener: Nombre del proyecto, Fecha de inicio del proyecto, Fecha de terminación del proyecto, Período de garantía, Términos de garantía, Lista de productos finales terminados que se entregan, Firma del Administrador Principal, Firma del Administrador del Proyecto, Firma de aceptación del Cliente.

3. Casos de pruebas

Un Caso de Prueba especifica la forma en que se probará el sistema, incluyendo la entrada o resultado con el que se va a probar, las condiciones bajo las que ha de probarse y los resultados esperados. Se pueden definir Casos de Prueba que prueben lo que se realiza en un Caso de Uso específico e implementado en la iteración actual, o se puede probar la interacción entre componentes definidos durante el diseño. Estos artefactos son responsabilidad del Diseñador de Pruebas.

4. Casos de uso

Estos documentos proporcionan una descripción detallada de los Casos de Uso, incluyendo como comienza (precondiciones), interactúa con los diversos Actores y termina (postcondiciones) cada uno de los Casos de Uso, además de describir las vías de comportamiento alterno permitidas y no permitidas para el flujo normal de eventos y los requerimientos especiales. Conjuntamente a la descripción textual de los Casos de Uso, se debe incluir el Diagrama de Casos de Uso (y opcionalmente Diagramas de Estados, Actividades o Interacción para los Casos de Uso). Su creación es responsabilidad del Especificador de Casos de Uso y del Cliente.

5. Clases del análisis

Una Clase del Análisis representa una abstracción de una o varias Clases y/o Subsistemas de Diseño del sistema, se centra en el tratamiento de requisitos funcionales, su comportamiento se define por medio de responsabilidades[1] a nivel alto e informal (no por interfaces), contiene atributos conceptuales de alto nivel (los atributos de las Clases de Diseño son de tipos de algún lenguaje de programación), y siempre encajan dentro de alguno de los tres estereotipos básicos: de interfaz, de control y de entidad. La identificación de las Clases del Análisis es responsabilidad del Ingeniero de Casos de Uso, Ingeniero de Componentes y del Administrador de Desarrollo.

6. Clases del diseño

Una Clase de Diseño es una abstracción de un conjunto de objeto que participan en una funcionalidad o servicio, descrita en el lenguaje de implementación elegido especificando la visibilidad de sus atributos y operaciones, sus métodos, los estados que puede tomar, sus relaciones con otras clases, sus dependencias, sus requerimientos de implementación y sus interfaces. En algunas ocasiones estas Clases del Diseño ya están estereotipadas en los lenguajes de programación. Su definición y mantenimiento es responsabilidad del Ingeniero de Componentes, del Administrador de Desarrollo y del Ingeniero de Casos de Uso.

7. Comentarios y observaciones del producto final

Este producto puede ser generado por el Editor Técnico, por el Administrador de Desarrollo o por el Administrador del Proyecto, los cuales interactúan constantemente con el Cliente y

[1] Una responsabilidad es una descripción textual de un conjunto cohesivo del comportamientos (interfaces) de una clase

pueden colectar los comentarios y observaciones que se generen sobre el producto final después de la entrega formal.

8. **Componente**
   El Componente es un empaquetamiento físico de elementos de un modelo como son las clases en el Modelo del Diseño. Los Componentes tienen los siguientes estereotipos estándar: ejecutable, archivo, biblioteca, tabla, documento. Los Componentes tienen relaciones de traza con elementos del modelo que implementan y pueden implementar varios elementos. Son responsabilidad de los Ingenieros de Componentes.

9. **Componente de prueba**
   El propósito de los Componentes de Prueba es implementar una funcionalidad específica de prueba para facilitar o automatizar las pruebas. Existen dos tipos: stubs (clases o componentes que simulan una funcionalidad aun no implementada del todo en esta iteración) y drivers (para facilitar la automatización de los Casos de Prueba). Son responsabilidad de los Ingenieros de Componentes.

10. **Contrato con proveedor**
    Este documento formaliza los acuerdos entre el o los Proveedores y la Organización de Desarrollo de Software para satisfacer los productos o servicios requeridos como externos, se debe incluir el nombre o razón social del Proveedor y de la Organización de Desarrollo de Software, nombre de sus representantes legales, direcciones de ambos, responsables del proyecto, puntos de contacto, fecha de contrato, costo monetario total del Proyecto, forma de pago, calendario de actividades de revisión y entregas, dependencias criticas entre las partes, garantías, términos de soporte y capacitación posterior al desarrollo, y firmas de aceptación.

11. **Contrato del proyecto**
    Este documento formaliza los acuerdos entre el Cliente y la Organización de Desarrollo de Software, se debe incluir el nombre o razón social del Cliente y de la Organización de Desarrollo de Software, nombre de sus representantes legales, direcciones de ambos, responsables del proyecto, puntos de contacto, fecha de contrato, costo monetario total del Proyecto, forma de pago, calendario de actividades de revisión y entregas, dependencias criticas entre las partes, garantías, términos de soporte y capacitación posterior al desarrollo, y firmas de aceptación.

12. **Cotización del proyecto**
    Este documento, que es responsabilidad del Administrador Principal y del Administrador del Proyecto, se elabora inmediatamente después de la creación de la primer versión mas estable del Plan de Desarrollo del Proyecto, en él se detalla el costo monetario total del proyecto, se muestra la asignación de recursos por áreas (desarrollo, administración y aseguramiento de la calidad) junto con su justificación, y los términos en los que son validos estos costos. Es utilizado como entrada en la actividad de negociación del proyecto para la firma del contrato.

13. **Criterios de evaluación de proveedores**
    En este documento se definen las métricas necesarias para seleccionar un proveedor, así como la prioridad de cada una de ellas. Estas definiciones deben ser compatibles con los requerimientos del proyecto y con los estándares y políticas de la organización de desarrollo de software. Este documento es responsabilidad del Administrador de Desarrollo y del Administrador del Proyecto.

14. **Definición de artefactos**
    Son documentos requeridos antes de conducir un proyecto mediante esta metodología, pues establece las definiciones de los productos de trabajo que servirán de entradas o salidas a las actividades marcadas.

Debe contener identificador, nombre del artefacto, tipo de artefacto (de ingeniería o de gestión), objetivo, rol o roles responsables de su elaboración, actividades relacionadas, descripción detallada, formatos utilizados (si aplica) y dependencias con otros artefactos.
La creación de las Definiciones de Artefactos es un compromiso de toda la organización de desarrollo de software.

15. Definición de métricas

Este documento cuenta con cuatro secciones que definen y describen el conjunto de métricas (primitivas y derivadas) disponibles para medir proyectos, productos, recursos y el propio proceso de desarrollo. La elección de las métricas a utilizar dependerá de la naturaleza del proyecto, cliente y producto a medir. Para cada métrica se debe establecer su objetivo. La definición de métricas es un requerimiento antes de aplicar la metodología y es responsabilidad de la organización de desarrollo.

16. Definición de procedimientos

Son documentos requeridos antes de conducir un proyecto mediante esta metodología, pues establece los procedimientos para completar cada una de las actividades. Para la ejecución de cada una de las actividades, se debe establecer una definición de los procedimientos que especifique el "cómo" debe realizarse una actividad (la actividad misma marca el "qué" debe hacerse) para completar objetivos y productos dentro del proceso de desarrollo.
Debe contener el identificador, objetivos, actividad a implementar, roles involucrados, periodicidad, conocimientos y perfil requerido, productos de entrada, productos de salida, sub-actividades a realizar, y posiblemente métricas a colectar en la actividad (tiempo estimado, recursos a utilizar).
La creación de las Definiciones de Procedimientos es un compromiso de toda la organización de desarrollo de software.

17. Definición de roles

Son documentos requeridos antes de conducir un proyecto mediante esta metodología, pues establece la definición de cada uno de los roles empleados en ella, especificando el tipo de rol, el perfil requerido por el rol, sus actividades, sus responsabilidades y dependencias con otros roles.
La creación de las Definiciones de Roles es un compromiso de toda la organización de desarrollo de software.

18. Definición de variables y constantes de configuración de subsistemas

En este documento que es responsabilidad del ingeniero de Componentes y del Administrador de Desarrollo, se detalla la información que parametriza el sistema, especificando el lugar de almacenamiento de los parámetros, modo de acceso a los parámetros, tipo de información contenida en ellos, valores posibles y la descripción de estos valores.

19. Descripción de niveles de promoción

En este documento se deben definir los estados de promoción de los Elementos, Componentes y Unidades de la Configuración (por ejemplo, borrador, revisado, liberado, etc.) dentro del Repositorio del Proyecto y grados de afectación de un cambio sobre un producto de trabajo (por ejemplo, rehacer, actualizar, o corregir). Así mismo se deben definir los saltos de versión (Por ejemplo, cuándo se realizara un cambio de versión de una unidad, de una décima o de una centésima) de un producto de trabajo. Este documento puede heredar de alguna política o estándar para todo proyecto o puede ser definido especialmente para un solo proyecto y es responsabilidad de la Administración de la Configuración del Software.

20. Descripción del repositorio

Este documento (responsabilidad del Administrador de la Configuración) describe detalladamente el repositorio de la configuración de software para un proyecto determinado. Especifica el nombre del repositorio, ruta de acceso, responsable del repositorio del proyecto,

lista de Elementos, Unidades y Componentes de la Configuración con sus respectivos permisos para cada grupo de trabajo o roles.

21. Especificación de ambientes y herramientas

Este documento, responsabilidad del Administrador de Desarrollo, identifica los diferentes ambientes que se van a necesitar para el funcionamiento del sistema final y sus requerimientos de hardware y software. Para cada requerimiento de software, se especifica la versión que se utilizará, y se justificarán sus características requeridas. Además tendrá una sección de comunicaciones, donde se describirán los requerimientos de hardware, de software y de comunicaciones (protocolos, ancho de banda) de un sistema en caso de que sea distribuido.

22. Especificación de bibliotecas comunes

Este documento es responsabilidad del Administrador de Desarrollo. Se crea a partir de la detección de funcionalidades comunes (en forma de Clases, Paquetes de Servicio o incluso de Subsistemas), para detallar cuáles bibliotecas comunes se tendrán, qué funcionalidad brindarán y cómo van a funcionar estas bibliotecas (Clases e Interfaces contenidas utilizando para ello, narrativas y diagramas). Es importante que este documento se actualizado y promovido con frecuencia entre el grupo de ingeniería.

23. Especificación de requerimientos

Este es un documento dinámico, por lo cual se ira completando poco a poco y posiblemente redefiniendo. Es elaborado principalmente por el Cliente, con ayuda de la organización de desarrollo.

Deberá contener una sección donde describa el negocio u organización del Cliente, donde se incluirá: misión del negocio u organización, políticas sobre sus proveedores, y una valoración actual de su cultura y ambiente de software.

Además En una sección sobre el proyecto, se debe especificar sus puntos de contacto para este proyecto, una lista de las tareas principales que desea que se elabore, sus productos entregables correspondientes a cada tarea y las fechas de entrega de los mismos.

En otra sección sobre el producto, se debe describir detalladamente cada uno de los requerimientos funcionales y no funcionales del sistema a desarrollar y, en caso de ser posible, una valoración del producto dentro de la organización del Cliente.

24. Estándares

Los estándares son documentos requeridos antes de conducir un proyecto mediante esta metodología, pues establece el contexto para otros elementos de la metodología, y se refieren a las especificaciones formales y obligatorias desarrolladas y utilizadas para prescribir consistencia en el desarrollo y la documentación.

Estándar de Documentación: En él se establecen las normas para unificar el lenguaje y formato utilizados en los documentos generados durante el proyecto

Estándar de Programación: En éste estándar se define la unificación en estilo de indentación, nomenclatura de identificadores, comentarios y en general de toda codificación generada en el proyecto.

La creación de los Estándares es un compromiso de toda la organización de desarrollo de software.

25. Modelo de Datos

El Modelo de Datos es parte del Modelo de Implementación, y contiene Diagramas de Clases persistentes, descripción del acceso a datos, definición de características de almacenamiento, estimación de volumen crecimiento de datos, relaciones entre tablas, reglas de integridad, comportamientos de Clases Persistentes (procedimientos almacenados), y scripts para creación de bases de datos, tablas y relaciones. Este artefacto es responsabilidad del Administrador de Desarrollo y del Ingeniero de Componentes.

26. Estimaciones del proyecto

Este documento se creará gradualmente a través de diversas actividades al inicio del proyecto y antes de la firma del contrato. Contendrá las principales tareas funcionales del futuro sistema, las tareas de soporte y de administración necesarias, así como una organización estructural de dichas tareas, en paquetes de trabajo. También se documentará el perfil y número de recursos necesarios para cada paquete de trabajo. De acuerdo a lo anterior también se debe indicar cuales paquetes de trabajo serán subcontratados y cuales podrán ser reutilizados.

En otro apartado se describirá el tamaño (número de funciones, líneas de código, clases necesarias, número de paginas, volumen de datos, etc.) aproximado de cada una de las tareas y de sus productos principales.

Finalmente se especificará el costo (temporal y monetario) aproximado de cada producto y tarea.

Este documento es, como su nombre lo indica, de naturaleza un tanto informal, y no será parte de la Línea Base, debido a que no se realizó un análisis muy detallado y su función termina hasta la elaboración de la Propuesta para el Cliente.

Este documento es responsabilidad del grupo de administradores.

27. Estimado de recursos de cómputo

En cada una de las áreas del proyecto (administración, calidad, configuración, y desarrollo) se hacen estimados del equipo de cómputo a utilizar, en este documento dinámico se reúnen todos los requerimientos de cómputo para el proyecto y es elaborado y mantenido por el Administrador del Proyecto, en consenso y ayuda con los demás administradores. Deberá incluir una lista del equipo de cómputo a utilizar, características, alternativa, justificación, responsable, tiempos de utilización, y recursos humanos que lo utilizarán.

28. Evaluación de pruebas (incluida la Evaluación de pruebas de aceptación)

La Evaluación de Pruebas recolecta, organiza y presenta los resultados de las pruebas, las mediciones de prueba realizadas y las evaluaciones acerca del sistema y sus componentes. También contiene recomendaciones sobre las pruebas siguientes y sobre el producto. Es responsabilidad del Diseñador de Pruebas, aunque para su creación pueden intervenir Ingenieros de Pruebas.

29. Evaluación general del proyecto

Este documento responsabilidad del Administrador del Proyecto contiene indicadores generales que resumen las métricas obtenidas de productos, recursos y el proyecto como tal, con el fin de proporcionar una visión general de ellos y sirva de apoyo para futuras mejoras de la metodología. También contiene propuestas para la mejora del proceso utilizado.

30. Evaluaciones parciales del proyecto

En este documento cada uno de los participantes en el proyecto genera su evaluación del mismo de acuerdo a su punto de vista, listando deficiencias de recursos, procesos, actividades y de administración, así como proponiendo acciones para mejorar cada uno de los defectos encontrados. Se recomienda que esta evaluación sea anónima, aun que en el caso de roles asignados a un solo recurso humano, esto puede no tener sentido.

31. Glosario de términos

Este documento, que se completará incrementalmente a través de las actividades e iteraciones, define términos comunes importantes que se utilizan al describir y construir el sistema buscando un consenso común entre Programadores, Analistas, Diseñadores, Administradores y el Cliente, evitando posibles confusiones. Su construcción inicia a partir del Modelo del Negocio y del Modelo del Dominio. Puede incluir dos secciones, una para el contexto del sistema y otra para el sistema en sí.

La principal responsabilidad de este documento la tiene el Editor Técnico.

32. Inconsistencias con requerimientos

En este documento se debe documentar las inconsistencias de los artefactos y actividades realizados y con los requerimientos, el Plan de Desarrollo y las reglas específicas del proyecto a nivel de revisiones técnicas. Su elaboración es responsabilidad del Administrador de Desarrollo.

33. Informe de avance técnico

Este documento elaborado periódicamente por el Administrador de Desarrollo documenta a nivel técnico el avance que se tiene sobre la elaboración del producto final con respecto al Plan de Desarrollo y los requerimientos, listando métricas recolectadas de productos y recursos, y describiendo brevemente las inconsistencias y errores encontrados, además referenciado a los documentos que describen mejor las métricas y las inconsistencias. También puede contener solicitudes y recomendaciones de cambios.

34. Informe de defectos

Este documento lo puede generar cualquier rol, y debe contener un número de control, nombre de quién lo detectó, Caso de Prueba que lo origino (si aplica), descripción detallada del defecto, versión de la Línea Base en la que se detectó, impacto del defecto, solución sugerida, urgencia de corrección, y nombre de receptor.

35. Informe de la revisión al repositorio

El objetivo de este documento es informar lo encontrado en las revisiones de la configuración, incluida la estructura del repositorio, los artefactos contenidos, la disponibilidad de los artefactos y el estado de los artefactos. Su elaboración es responsabilidad del Administrador de la Configuración.

36. Interfaces diseñadas de clases

Las Interfaces Diseñadas de Clases se utilizan para especificar las operaciones que proporcionan las Clases del Diseño. Las Clases del Diseño que proporcionan una interfaz, deben proporcionar también métodos que realicen las operaciones de la interfaz. Son responsabilidad del Ingeniero de Componentes, del Administrador de Desarrollo y del Ingeniero de Casos de Uso.

37. Interfaces diseñadas de subsistemas

Las Interfaces Diseñadas de Subsistemas se utilizan para especificar las operaciones que proporcionan los subsistemas. Un Subsistema del Diseño que proporcione una interfaz debe contener también las Clases del Diseño u otro Subsistema del Diseño que proporcione la interfaz. La mayoría de las interfaces entre subsistemas se consideran importantes para la Arquitectura debido a que definen las interacciones permitidas entre los Subsistemas, por lo cual, en algunos casos es útil diseñar interfaces que sean estables entre Subsistemas, antes de implementar los propios Subsistemas para que las interfaces sean utilizadas como requisitos y medio de sincronización entre equipos de desarrollo. Son responsabilidad del Ingeniero de Componentes, del Administrador de Desarrollo y del Ingeniero de Casos de Uso.

38. Interfaces implementadas de componentes

Las Interfaces Implementadas de Componentes especifican las operaciones implementadas por Componentes de implementación. Un Componente que implementa una interfaz, debe implementar correctamente todas las operaciones definidas por la interfaz.

39. Interfaces implementadas de subsistemas

Las Interfaces Implementadas de Subsistemas especifican las operaciones implementadas por subsistemas de implementación. Un Subsistema que implementa una interfaz, debe implementar correctamente todas las operaciones definidas por la interfaz.

40. Interpretación de métricas

Este documento de elaboración periódica es responsabilidad del Administrador del Proyecto y contiene una lista de indicadores de progreso calculados a partir de las métricas (primitivas y derivadas) de recursos y productos y de los reportes de auditoría generados por el Administrador de la Calidad. También contiene conclusiones, decisiones y recomendaciones obtenidas a partir del análisis de los indicadores.

41. Línea base

Es un conjunto de artefactos, revisados y aprobados, que constituyen las bases acordadas para el desarrollo posterior del producto, y que solamente pueden ser modificados por medio de un procedimiento formalmente definido.

42. Lista de artefactos que entregará proveedor

En este documento se formalizara la lista de artefactos que entregará el Proveedor a la Organización de Desarrollo de Software, su fecha de entrega, y términos de entrega.

43. Lista de material de soporte y capacitación

Este documento es responsabilidad del Editor Técnico y contendrá una lista del material a elaborar con fines de soporte y capacitación, junto con sus justificaciones, el responsable de su elaboración (a nivel de recurso humano, no a nivel de rol) y el público o audiencia para quien se piensa elaborar.

44. Lista de nuevas herramientas, productos o servicios para adquirir

Documento dinámico que muestra una lista de herramientas, productos y/o servicios para adquirir, conteniendo una descripción detallada de las adquisiciones y fecha limite de adquisición. Su elaboración es responsabilidad del Administrador del Proyecto y del Administrador de Desarrollo

45. Lista de posibles proveedores

Este documento es definido para cada proyecto que requiera la adquisición de un producto o servicio externo. Lista los proveedores que posiblemente satisfagan lo requerido por la organización de desarrollo, y contendrá información de contacto con el proveedor candidato y una valoración de sus capacidades. Es responsabilidad del Administrador de la Configuración y del Administrador de Desarrollo.

46. Lista de productos y elementos adicionales de la configuración

Es similar a la “Lista de productos y elementos de la configuración”. En este documento se identifican los Elementos Componentes y Unidades de la Configuración especiales y específicos a un proyecto dado. Se les debe asignar un identificador único, quienes harán uso de ellos, definir si van a cambiar en el tiempo, identificar las dependencias entre ellos, y si son críticos para el proyecto. Su elaboración es responsabilidad del Administrador de la Configuración.

47. Lista de productos y elementos de la configuración

En este documento (responsabilidad del Administrador de la Configuración) se identifican los Elementos de la Configuración y Componentes o Unidades de la Configuración que los integran, y que serán puestos bajo la Administración de la Configuración; cada uno de ellos se puede corresponder a productos que serán entregados al Cliente, productos de trabajo internos, productos adquiridos, herramientas, u otros elementos utilizados para crear y describir estos y otros productos de trabajo; todos ellos definidos e identificados para todo proyecto.

Los Elementos de la Configuración pueden ser descompuestos en Componentes de la Configuración o en Unidades de la Configuración. Esta organización lógica provee fácil identificación y acceso controlado. Pero es importante aclarar que los términos “Elemento de la Configuración”, “Componente de la Configuración” y “Unidad de la Configuración” son

utilizados solo en el contexto de la Administración de la Configuración, y así es utilizado en la literatura relacionada.
En este documento se debe asignar un identificado único a cada Elemento, Componente y Unidad de la Configuración, y se debe identificar cuáles de ellos serán utilizados por dos o mas grupos de trabajo, cuáles pueden cambiar a través del tiempo (ya sea por su naturaleza o por algún error o cambio de requerimientos), las dependencias entre ellos, y cuáles son críticos para un proyecto

48. Lista de riesgos

En este documento dinámico se documentarán todos los riesgos identificados (puntos críticos, peligros, amenazas, vulnerabilidades, etc.), su análisis (causas, impacto, probabilidades, etc.), su clasificación y sus prioridades entre ellos. La elaboración es realizada por el grupo de administradores, pero la responsabilidad principal de su elaboración y actualización la lleva el Administrador del Proyecto. Una vez establecida la versión inicial del documento (y para cada actualización) es importante que se plasme en una sección específica el acuerdo de los administradores en cuanto a completitud y exactitud del documento.

49. Manual de usuario

Este documento es una guía de la utilización del producto o productos del proyecto por parte de Usuarios de perfil no técnico (o a un nivel poco profundo de conocimiento de sistemas). Se pueden utilizar los Casos de Uso, el Modelo de Casos de Uso, y el Prototipo de Interfaz de Usuario para guiar y estructurar su contenido. Es responsabilidad del Editor Técnico.

50. Manual técnico

El Manual Técnico es una guía para solución de problemas y mejora del producto o productos del proyecto, intentado para usuarios con un perfil técnico, capaces de instalar, solucionar o mejorar el producto. En su contenido se debe incluir las vistas de Casos de Uso, de Análisis, de Diseño, de Implementación, Despliegue y Pruebas de la Arquitectura del producto, junto con todos los elementos que permitan su comprensión. Es responsabilidad del Editor Técnico, quién puede consultar a los demás roles para la elaboración.

51. Material de capacitación

El Material de Capacitación es un conjunto de diversos artefactos utilizados para capacitar a los usuarios sobre el producto a realizar, estos artefactos pueden incluir, presentaciones de diapositivas, apuntes de cursos, ejercicios y prácticas, especificaciones de ambientes de capacitación, etc., lo cual va a depender de la naturaleza del proyecto y del cliente. Este material es responsabilidad del Editor Técnico, quién puede consultar a los demás roles para la elaboración.

52. Matriz de traza de requerimientos

El objetivo de este documento dinámico es identificar mediante qué actividades, Casos de Uso u otros artefactos se satisfacen cada uno de los requerimientos funcionales y no funcionales. El nivel de descomposición de artefactos para hacer la trazabilidad de los requerimientos, se deja abierto a consideración del Administrador de Desarrollo, pero se debe especificar el nivel, siendo el único forzoso el nivel de artefactos de Casos de Uso. Este documento (a nivel de Casos de Uso) es responsabilidad del Analista de Sistemas.

53. Mecanismos para la adquisición de conocimientos y habilidades

Su elaboración es responsabilidad del Administrador de Desarrollo y deberá contener una lista de conocimientos y habilidades técnicas necesarias por parte de los recursos humanos técnicos (miembros del Grupo de Ingeniería) para participar en el proyecto. Para cada uno de estos conocimientos y habilidades se debe definir una estrategia para adquirirlos, por ejemplo integrar a un recurso humano específico para que participe activamente en el proyecto, integrar a un recurso humano específico para capacitación, adquisición de documentación necesaria junto con la calendarización de horas de aprendizaje, adquisición de cursos externos, etc.

54. Métricas de productos

Este documento resume las medidas utilizadas para cuantificar diversos aspectos de los productos, entendiendo como producto cualquier documento, modelo, pieza de software, diagrama, etc. que sea salida de alguna actividad o conjunto de actividades. Para los productos se pueden tener métricas (primitivas o derivadas) que midan progreso, calidad, integridad, trazabilidad, volatilidad, esfuerzo, etc., de cuerdo a la naturaleza del producto. La definición de las métricas primitivas que se deben colectar y el cálculo de las métricas derivadas de productos se establecen de acuerdo a la organización de desarrollo, al cliente, al proyecto y a los productos a ser medidos (documentado en la Definición de Métricas). La responsabilidad de las Métricas de Productos es del Administrador de Desarrollo y del Administrador del Proyecto.

55. Métricas de recursos

Este documento resume el desempeño de los recursos materiales y humanos en las actividades que les corresponde dentro del proyecto en forma de métricas (primitivas y derivadas), con el fin de tomar acciones correctivas o de reconocimiento y estimulo. Las métricas primitivas que se pueden colectar para recursos humanos son experiencia, habilidad, costo, desempeño (consideradas a través del tiempo, y de acuerdo al rol, grado y equipo al que pertenecen); para los recursos materiales las métricas pueden ser calidad, costo, desempeño, recursos consumidos, recursos remanentes, etc. Las métricas derivadas se calcularán partiendo de las métricas primitivas y de acuerdo a lo definido por la organización de desarrollo (documentado en la Definición de Métricas). La responsabilidad de las Métricas de Recursos es del Administrador de Desarrollo y del Administrador del Proyecto

56. Métricas del proceso

Las métricas del proceso son responsabilidad del Administrador del Proyecto y miden diversos aspectos del proceso en base a las métricas de productos, recursos y proyecto, y a las evaluaciones parciales del proyecto generadas por cada uno de los roles. Algunos aspecto del proceso que se pueden medir son: duración del proyecto, numero de involucrados, nivel de los recursos humanos, número de rotaciones de recursos humanos, número de artefactos del proceso, peso (o importancia) asociado a cada artefacto, número de actividades realizadas para la elaboración de cada artefacto, peso asociado a cada actividad, nivel de satisfacción sobre el producto final, número de juntas internas, número de juntas externas, número de auditorías de la Administración de la Calidad, número de revisores técnicas, número de revisiones administrativas, número de desviaciones del proceso, número de cambios aplicados y solicitados por el Cliente, etc. A partir de estas medidas se calculan indicadores que muestren la madurez e integridad del proceso utilizado.

57. Métricas del proyecto

Las métricas del proyecto son responsabilidad del Administrador del Proyecto y del Administrador de Desarrollo y miden diversos aspectos del proyecto en base a métricas derivadas de métricas de productos y recursos, y de los reportes de auditorías del Administrador de la Calidad. Algunos aspecto del proyecto que se pueden medir son: Modularidad (promedio de errores por cambios o implementaciones de modelos), Adaptabilidad (promedio del esfuerzo por cambios o implementaciones de modelos), Madurez (numero y duración de pruebas por cambios correctivos), facilidad de mantenimiento, estabilidad, etc. En este documento se deben incluir acciones o recomendaciones para corregir el desempeño del proyecto.

58. Modelo de casos de uso

Este modelo de UML permite que los desarrolladores de software y el cliente lleguen a un acuerdo sobre los requisitos, y proporciona la entrada fundamental del análisis, del diseño y de las pruebas. Se construye incrementalmente y su elaboración inicia de forma temprana para proporcionar bases para la negociación del Contrato del Proyecto. Se compone de los Casos de Uso, sus prioridades y sus relaciones como un todo que represente la funcionalidad del sistema completo, junto con la especificación de requerimientos, además de contar con una

sección para describir detalladamente a los Actores. Así mismo, es la guía para todo el ciclo de vida del proyecto, de ahí su importancia. Su elaboración, consistencia y mantenimiento es responsabilidad del Administrador de Desarrollo principalmente.

59. Modelo de despliegue

El Modelo de Despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los Nodos de cómputo. El Modelo de Despliegue se utiliza como entrada fundamental en actividades de implementación y en algunas actividades de diseño. En el Modelo de Despliegue cada Nodo representa un recurso de cómputo, los Nodos poseen relaciones que representan medios de comunicación entre ellos, se pueden describir diferentes configuraciones de red, la funcionalidad de un Nodo se define por los componentes que se distribuyen sobre ese Nodo, y el Modelo de Despliegue en si mismo representa una correspondencia entre la Arquitectura software y la Arquitectura hardware. Puede contener diagramas de clases estereotipadas como Nodos. Su creación y su mantenimiento deben ser responsabilidad del Administrador de Desarrollo.

60. Modelo de implementación

El Modelo de Implementación describe cómo los elementos del Modelo del Diseño se implementan en términos de Componentes y Subsistemas de Implementación, describe también cómo se organizan los Componentes de acuerdo con los mecanismos de modularización y estructuración disponibles en el entorno de implementación y en el lenguaje o lenguajes de programación utilizados, y la dependencia entre los Componentes. El Modelo de Implementación contiene Componentes, Interfaces Implementadas de Componentes, Subsistemas de Implementación, e Interfaces Implementadas de Subsistemas en forma de alguno de los estereotipos estándar de Componentes, así como las descripciones textuales de dependencias, ilustradas con diversos diagramas UML. Una parte importante del Modelo de Implementación es el Modelo de Datos, que contiene la descripción en detalle de la implementación de la base de datos a utilizar en el sistema. Este modelo es responsabilidad del Administrador de Desarrollo.

61. Modelo del análisis

Este artefacto, responsabilidad del Administrador de Desarrollo, ayuda a refinar los requerimientos por medio de objetos conceptuales (Paquetes del Análisis y Clases del Análisis), utilizando el lenguaje del desarrollador (no el Lenguaje de Desarrollo a utilizar) para dar una descripción interna del sistema, mediante la realización de los Casos de Uso a través de los objetos abstractos y sus relaciones. Puede contener Diagramas de Realizaciones de Casos de Uso, Diagramas de Clases del Análisis, Diagramas de Estado, Diagramas de Interacción y Diagramas de Actividades, junto con las descripciones de las Clases del Análisis y Paquetes del Análisis que participan en ellos

62. Modelo del diseño

El Modelo del Diseño es un modelo de objetos que describe la realización física de los Casos de Uso centrándose en cómo los requerimientos funcionales y no funcionales tienen impacto en el sistema a desarrollar. Además, el Modelo del Diseño sirve de abstracción de la implementación del sistema, y es de ese modo utilizada como una entrada fundamental en las actividades de implementación y pruebas. El Modelo del Diseño es un artefacto global compuesto que abarca todas las Clases del Diseño, Subsistemas del Diseño, Colaboraciones y relaciones entre todo ello; en otras palabras, es el subsistema de diseño de nivel mas alto. Este artefacto es responsabilidad del Administrador de Desarrollo. Puede contener Diagramas de Realizaciones de Casos de Uso, Diagramas de Clases del Diseño, Diagramas de Estado, Diagramas de Interacción y Diagramas de Actividades, junto con las descripciones de las Clases del Diseño y Subsistemas del Diseño que participan en ellos.

63. Modelo del dominio

El Modelo del Dominio es un caso especial del Modelo del Negocio, cuyo ámbito de descripción se centra mas específicamente con los procesos relacionados con el sistema

software a desarrollar, dentro de un negocio. El Modelo del Dominio es un instrumento para comprender los procesos relacionados al sistema software que se va a desarrollar, a través de modelos de UML, específicamente Casos de Uso del Dominio, Diagramas de Interacción, Diagramas de Actividades y Diagramas de Clases del Dominio. Su creación y mantenimiento es responsabilidad del Administrador de Desarrollo y del Analista de Sistemas.

64. Modelo del negocio

El Modelo del Negocio es un instrumento para comprender los procesos de negocio de la organización en donde se utilizará el sistema software que se va a desarrollar, a través de modelos de UML, específicamente Casos de Uso del Negocio, Diagramas de Interacción, Diagramas de Actividades y Diagramas de Clases del Negocio. Su creación y mantenimiento es responsabilidad del Administrador de Desarrollo y del Analista de Sistemas.

65. Modelo de pruebas

Describe principalmente cómo se prueban los componentes ejecutables en el Modelo de Implementación con Pruebas de Integración y Pruebas de Sistema. El Modelo de Pruebas es una colección de Casos de Pruebas, Procedimientos de Pruebas, Componentes de Prueba y Planes de Prueba. Si el Modelo de Pruebas es grande, puede ser útil introducir paquetes en el modelo para manejar su tamaño. Es responsabilidad del Diseñador de Pruebas.

66. Organigrama de ingeniería

Este puede ser un documento dinámico, donde se describirá la organización, jerarquía (en forma de organigrama), responsabilidades (rol asignado), interacciones, información de localización, fecha de asignación y posible fecha de liberación de cada uno de los miembros del Grupo de Ingeniería del proyecto. El contenido se debe actualizar periódicamente para reflejar los recursos humanos que continúan en el proyecto o que se liberaran. La información contenida servirá para colectar las métricas del proyecto sobre rotación de recursos. Su creación y mantenimiento es responsabilidad del Administrador de Desarrollo.

67. Paquetes del análisis

Los Paquetes del Análisis proporcionan un medio para organizar los artefactos del Modelo del Análisis en piezas manejables. Un Paquete del Análisis puede constar de Clases del Análisis, de Realizaciones de Casos de Uso-Análisis, y de otros Paquetes del Análisis. Los Paquetes del Análisis deben ser cohesivos y débilmente acoplados. Además, los Paquetes del Análisis pueden representar una separación de intereses de análisis, deben basarse en los requerimientos funcionales y en el dominio del problema (por lo tanto deben ser identificables por personas con conocimientos del dominio), no deben ser basados en requerimientos no funcionales. Existe un tipo de Paquetes del Análisis reutilizables llamados Paquetes de Servicios, que representan servicios.

68. Plan de aseguramiento de la calidad

El Plan de Aseguramiento de la Calidad deriva directamente del Plan de Desarrollo de Software y es elaborado y actualizado por el Administrador de la Calidad para dirigir las actividades de la Administración de la Calidad de manera coherente con el plan principal. Debe contener una visión general del proyecto, estrategia a utilizar en la Administración de la Calidad, requerimientos de recursos, cronograma y lista de artefactos a entregar.

69. Plan de contención

En la elaboración de este documento dinámico participa todo el grupo de administradores, pero es responsabilidad del Administrador de Proyecto. Debe incluir las técnicas y métodos para evitar, reducir y controlar la ocurrencia de cada riesgo de alto impacto (los riesgos de bajo impacto pueden ser solamente monitoreados), así como los responsables para dirigir estas técnicas o métodos, y la relación costo/beneficio de contener cada los riesgos de alto impacto

70. Plan de contingencia

Este documento dinámico es un complemento al Plan de Contención, y es elaborado por el grupo de administradores, pero es responsabilidad del Administrador del Proyecto. En él se detalla las actividades y técnicas dirigidas a limitar el efecto de los riesgos en caso de que se presenten, se definen los responsables de limitarlos, y los costos que acarrearía.

71. Plan de desarrollo de proveedor

El Plan de Desarrollo del Proveedor solo será solicitado solo a los proveedores seleccionados y contendrá una descripción del cómo se satisfacerá la adquisición, incluyendo: visión general del producto/servicio a proporcionar, riesgos y plan de mitigación de riesgos, estrategia de administración del proyecto, estrategia o metodología de desarrollo, estrategia para el aseguramiento del producto, requerimientos de recursos, cronograma y lista de artefactos a entregar. Este plan será considerado en la planeación del proyecto de desarrollo y su solicitud será responsabilidad del Administrador de Desarrollo y del Administrador del Proyecto.

72. Plan de desarrollo del proyecto

Este documento dinámico es la base para la administración del proyecto, es elaborado por el grupo de administradores y su actualización es responsabilidad del Administrador del Proyecto. Contendrá: Una visión general del proyecto, la estrategia de administración del proyecto, la estrategia o metodología de desarrollo, la estrategia para el aseguramiento de la calidad del producto, el cronograma de actividades y lista de artefactos a entregar al cliente.

73. Plan de la administración de la configuración

El Plan de la Administración de la Configuración deriva directamente del Plan de Desarrollo de Software y es elaborado y actualizado por el Administrador de la Configuración para dirigir las actividades de la Administración de la Configuración de Software de manera coherente con el plan principal. Debe contener una visión general del proyecto, estrategia a utilizar en la Administración de la Configuración, requerimientos de recursos, cronograma y lista de artefactos a entregar.

74. Plan de la integración de la construcción

Este documento, que es responsabilidad del Integrador del Sistema, contiene un análisis del impacto de integrar los requerimientos de los Subsistemas de Implementación y de los Componentes elaborados a la Construcción actual, de acuerdo a este impacto recomienda acciones a tomar, como por ejemplo, el posponer la integración de ciertos subsistemas componentes, definir un orden de integración de los subsistemas y componentes actuales, creación de stubs, etc.

75. Plan de pruebas

El Plan de Pruebas es responsabilidad del Diseñador de pruebas y describe las estrategias, recursos y planificación de las pruebas a realizar en una iteración dada.

76. Políticas

Las Políticas son documentos requeridos antes de conducir un proyecto mediante esta metodología, pues establece el contexto para los demás elementos de la metodología. Algunas de sus directrices serán heredadas directamente de la organización en general, desde fuera del contexto de la organización de desarrollo de software. Su objetivo es definir las directrices a las cuales esta sujeta la organización de desarrollo de software para incrementar la comunicación entre los participantes de los proyecto y reducir confusiones. Su elaboración es responsabilidad del grupo de administradores de la organización.

Deben contener: Objetivo, Miembros de Comité de Definición, Antecedentes y Contexto, Directrices y firma de aceptación de los administradores.

Políticas de la Administración de requerimientos: Esta política establece las expectativas organizativas para la administración de requerimientos e inconsistencias detectadas entre

requerimientos y los planes y productos de trabajo. [“Capability Maturity Model Integration (CMMI), Version 1.1”, 2002]

Políticas de la Planeación del proyecto: Esta política establece las expectativas organizativas para estimar los parámetros de planeación, para el establecimiento de acuerdos internos y externos, y para el desarrollo del plan para dirigir el proyecto. [“Capability Maturity Model Integration (CMMI), Version 1.1”, 2002]

Políticas del Monitoreo y control del proyecto: Esta política establece las expectativas organizativas para el monitoreo del desempeño contra el Plan de Desarrollo del Proyecto, así como de la administración de acciones correctivas cuando el desempeño o los resultado se desvían significativamente del plan. [“Capability Maturity Model Integration (CMMI), Version 1.1”, 2002]

Políticas de la Administración de proveedores: Esta política establece las expectativas organizativas para establecer, mantener y satisfacer los acuerdos de los proveedores. [“Capability Maturity Model Integration (CMMI), Version 1.1”, 2002]

Políticas de la Medición y el análisis: Esta política establece las expectativas organizativas para empatar los objetivos y las actividades de medición con necesidades de información identificadas y para proporcionar resultados de mediciones. [[“Capability Maturity Model Integration (CMMI), Version 1.1”, 2002]

Política de la Administración de la Calidad: Esta política establece las expectativas organizativas para evaluar objetivamente si los procesos y los productos de trabajo asociados se apegan a las descripciones aplicables de procesos, estándares y procedimientos, y asegurar que el incumplimiento sea manejado. Esta política también establece expectativas organizativas para el aseguramiento de la calidad de procesos y productos en todos los proyectos. El aseguramiento de la calidad de procesos y productos debe tener suficiente independencia de la administración del proyecto para brindar objetividad en la identificación y divulgación de incumplimientos de calidad. [“Capability Maturity Model Integration (CMMI), Version 1.1”, 2002]

Política de la Administración de la Configuración: Esta política establece las expectativas organizativas para el establecimiento y el mantenimiento de las Líneas Base, el seguimiento y control de cambios a los productos de trabajo (bajo la Administración de la Configuración), y el establecimiento y mantenimiento de la integridad de las Líneas Base. [“Capability Maturity Model Integration (CMMI), Version 1.1”, 2002]

77. Procedimientos de prueba

Un Procedimiento de Prueba especifica cómo realizar uno o varios Casos de Prueba o partes de estos, definiendo tiempos, responsabilidades, instrucciones, propósito, métodos de verificación y ajustes en caso de resultados discrepantes originados por factores externos al sistema. Su elaboración es responsabilidad del Diseñador de Pruebas.

78. Procedimientos especiales de la configuración

En este documento se establecen los procedimientos especiales específicos para un proyecto específico que servirán en la Administración de la Configuración. Debe contener el identificador, objetivos, roles involucrados, periodicidad, conocimientos y perfil requeridos, productos de entrada, productos de salida, sub-actividades a realizar, y posiblemente métricas a (tiempo estimado, recursos a utilizar). Su creación es responsabilidad del Administrador de la Configuración.

79. Proceso de migración

Este artefacto es opcional y es responsabilidad del Administrador de Desarrollo. Especifica, si aplica, la forma en que se realizará la conversión del sistema actual al nuevo, definiendo el tipo

de estrategia a utilizar (piloto, paralelo, sustitución completa, etc.), condiciones necesarias para la migración, acciones a tomar en caso de falla de migración.

80. Propuesta de respuesta a retroalimentaciones

En este documento, responsabilidad del Administrador del Proyecto y del Administrador de Desarrollo, se documentan las propuestas de mejoras encontradas para el producto final, ya sea bajo el mismo contrato o por medio de un nuevo contrato.

81. Propuestas/ofertas de proveedores

Aunque este documento es elaborado por el Proveedor y es entregado a la Organización de Desarrollo de Software, esta debe definir el formato e información que deben contener en los Requerimientos a Proveedores (por ejemplo, que contenga su experiencia en aplicaciones similares, capacidades en la administración de proyectos similares, instalaciones y recursos disponibles y un plan para la satisfacción de la adquisición). La elaboración de este documento es responsabilidad del Proveedor.

82. Prototipo de interfaz de usuario

Este artefacto, que es responsabilidad del Especificador de Casos de Uso, ayuda a comprender y especificar las interacciones entre Actores humanos y el sistema. Se pueden incluir junto al prototipo un documento que contenga los esquemas de pantalla y sus relaciones. Debido a que el Prototipo de Interfaz de Usuario muestra la interacción de los actores humanos con el sistema, debe basarse en los Casos de Uso identificando con cuales Casos de Uso tienen relación directa los Actores humanos y de qué manera, por lo que se recomienda un Prototipo de Interfaz de Usuario por cada Caso de Uso.

83. Realizaciones de Casos de Uso – Análisis

Este artefacto responsabilidad del Ingeniero de Casos de Uso y del Administrador de Desarrollo, describe como un Caso de Uso particular es realizado en términos de colaboraciones de Clases del Análisis y sus objetos. Proporciona una traza directa hacia un Caso de Uso concreto del Modelo de Casos de Uso. Deben contener una descripción textual del flujo de eventos, Diagramas de Interacción (tanto Diagramas de Secuencia y Diagramas de Colaboración), Diagramas de Clases, y los requerimientos derivados de todo esto.

84. Realizaciones de Casos de Uso – Diseño

Una Realización de Caso de Uso-Diseño es una colaboración en el Modelo de Diseño que describe cómo se realiza un Caso de Uso específico, y cómo se ejecuta en términos de Clases de Diseño y sus objetos. Una Realización de Caso de Uso-Diseño proporciona una traza directa a una Realización de Caso de Uso-Análisis en el Modelo del Análisis. Contiene una descripción textual del flujo de eventos y de los requisitos de implementación de un Caso de Uso, Diagramas de Clases que muestran sus Clases del Diseño participantes, y Diagramas de Interacción que muestran la realización de un flujo o escenario concreto de un Caso de Uso en términos de interacciones entre objetos del diseño. Es su creación es responsabilidad del Ingeniero de Casos de Uso.

85. Registro Histórico de Proveedores

En el Repositorio General de la Organización de Desarrollo debe existir de manera independiente a los proyecto un Registro Histórico de Proveedores, donde se puede incluir: registro histórico evaluación de proveedores, registro histórico de propuestas de proveedores entregadas a la organización, registro histórico de riesgos y capacidades de los proveedores y registro histórico de contratos con proveedores. Este artefacto es responsabilidad de la Administración de Configuración de Software.

86. Reglas específicas de codificación

Las Reglas Especificas de Codificación estandarizan a nivel de proyecto el aspecto del código a generar, de forma coherente con el Modelo del Dominio y los Estándares de Programación que rigen a la organización de desarrollo. Este documento es responsabilidad del

Administrador de Desarrollo y contiene la forma estándar de nombrar los diferentes elementos dentro del código (variables, constantes, funciones, etc.) y restricciones a los lenguajes de codificación que se utilizarán en cada capa.

87. Reglas específicas de diseño

Este artefacto es creado y actualizado por el Administrador de Desarrollo, y sirve como medio de estandarización a nivel de proyecto sobre aspectos del diseño e implementación, como por ejemplo, abreviaciones y nomenclaturas utilizadas en el contexto de este proyecto, restricciones y limites en el diseño estructural y de comportamiento de la aplicación, enfoque arquitectónico al cual deben ajustarse todo el diseño, nombre de tablas y procedimientos almacenados a utilizar en la base de datos, etc.

88. Reporte del contenido del repositorio

Documenta la estructura y los artefactos contenidos en el Repositorio de la Configuración del Proyecto (incluida la Línea Base y sus componentes) en una fecha determinada, indicando su versión, disponibilidad, estado, responsable, fecha de ultima modificación, fecha de creación, etc. Es responsabilidad del Administrador de la Configuración.

89. Reportes de auditoría por actividad

Este documento es generado a partir de una auditoría por parte del Administrador de Calidad a la realización de una actividad, registrando los pasos que se siguieron en la auditoría, su consistencia con el proceso de desarrollo utilizado, con la Definición del Procedimiento correspondiente, con los costos y tiempos planeados, y se agrega una recomendación sobre la realización de la actividad.

90. Reportes de auditoría por artefacto

Este documento es generado a partir de una auditoría por parte del Administrador de Calidad a algún artefacto o grupo de artefactos. En este reporte se registra de un artefacto los pasos que se siguieron en la auditoría, su apego a estándares, su conformidad con la propia definición del artefacto, su apego al costo y tiempos estimados para su elaboración, su ortografía y su gramática, su completez, y se agrega una recomendación sobre el artefacto o artefactos. Cabe señalar que uno de los Reportes de Auditoría por Artefacto mas importantes es el relacionado al Repositorio de la Configuración.

91. Reportes de auditoría por rol

Este documento es generado a partir de una auditoría por parte del Administrador de Calidad a las actividades y desempeños de un rol, que puede incluir a uno o más recursos humanos. Se registra los recursos humanos que conforman el rol, así como los pasos que se siguieron en la auditoría, el apego de su desenvolvimiento a la definición del rol, a los Estándares y a las Políticas. También debe incluir recomendaciones que tiendan a mejorar el desempeño del rol.

92. Repositorio del proyecto

El repositorio de la configuración del proyecto es donde se almacenaran Elementos, Componentes y Unidades de la Configuración, incluye el medio de almacenamiento, el sistema de configuración de software y control de cambios, las herramientas para administrar al sistema de configuración de software y control de cambios, la base de datos de solicitudes de cambios y la configuración de niveles de control y promoción. Es responsabilidad del Administrador de la Configuración.

93. Requerimientos a externos

Este documento (responsabilidad del Administrador de Desarrollo y del Administrador del Proyecto) define los requerimientos que deben ser satisfechos por los proveedores para cubrir las necesidades de la organización de desarrollo con respecto a sus adquisiciones. También sirve como base de negociación con los proveedores y lista lo qué la organización de desarrollo se compromete a proporcionar al proveedor (instalaciones, documentación, servicios, etc.).

94. Riesgos y capacidades de proveedor(es) seleccionado(s)

Considerando el Registro Histórico de Proveedores, el Plan de Desarrollo del Proveedor y la evaluación y comparación de las propuestas de los proveedores, el Administrador de Desarrollo y el Administrador del Proyecto redactarán los Riesgos y Capacidades de Proveedor(es) Seleccionado(s) tomados desde el punto de vista de la Organización de Desarrollo de Software, pues aunque se incluye una descripción de esto en los documentos proporcionados por el proveedor, esta nueva perspectiva puede agregar nuevos puntos no detectados hasta entonces. Este documento será considerado para la identificación de riesgos del proyecto.

95. Solicitud de cambio

Un cambio no solo sucede a raíz de una modificación en algún requerimiento o por la identificación de un requerimiento nuevo, también puede surgir debido a errores o defectos en los productos de trabajo, en la realización de una actividad o desempeño de algún recurso (humano o material).

Se debe documentar el número de control, cual es el origen del cambio, referencia al informe de defectos relacionado (si aplica), descripción del cambio, justificación del cambio, su nivel de urgencia, su impacto sobre algún requerimiento, producto, actividad, recurso y/o calendarización, los posibles riesgos involucrados, y posibles alternativas.

Para cada artefacto afectado se debe determinar el grado de afectación (si el cambio involucra rehacerlo o solo actualizarlo), quién es su responsable, el costo (monetario y de tiempo) y las dependencias entre los artefactos afectados; de tal forma que también sirva de guía para la Administración de la Configuración al momento de autorizar las modificaciones en el Sistema de Administración de la Configuración. Así mismo se debe especificar si se afecta la Línea Base y si es necesario "congelarla".

Se debe describir (si aplica) la Estrategia de Pruebas que se va a seguir después de aplicar el cambio.

Finalmente se debe especificar el receptor de la solicitud de cambio.

96. Solicitud del proyecto

Es un documento que formalmente hace la solicitud del proyecto al Administrador Principal por parte del Cliente, independientemente de que se llegue a un acuerdo por ambas partes para realizarlo o no (si se llega a firmar un contrato o no). Sirve (junto con la Especificación de Requerimientos) para que la organización que desarrollará el software se prepare para proponer estimados del proyecto y una cotización formal y más real, que conducirá a la posible firma de un contrato.

Deberá contener el Nombre del solicitante, Fecha de solicitud, Descripción general del sistema solicitado, Términos especiales bajo los cuales se realizaría el proyecto, Firma del cliente solicitante y firma de enterado del Administrador Principal.

97. Subsistemas de diseño

Los Subsistemas de Diseño (que son responsabilidad del Administrador de Desarrollo, del Ingeniero de Componentes y del Ingeniero de Casos de Uso) son una forma de organizar los artefactos del Modelo del Diseño en piezas más manejables. Un Subsistema del Diseño puede constar de Clases del Diseño, Realizaciones de Casos de Uso-Diseño, Interfaces y otros Subsistemas de Diseño. Por otro lado, un Subsistema del Diseño puede proporcionar interfaces que representan la funcionalidad que exporta en términos de operaciones. Deben ser cohesivos y débilmente acoplados, representan una separación de aspectos de diseño, pueden representar componentes de "grano grueso" en la implementación del sistema, y pueden representar producto software reutilizado o heredado y encapsulado dentro de ellos. Existe un tipo especial de subsistema que se llaman Subsistemas de Servicio que vienen de una relación de traza con los Paquetes de Servicios del Modelo del Análisis, y existe otro especial para los datos, donde se diseña la base de datos a utilizar.

98. Subsistemas de implementación

Los subsistemas de Implementación proporcionan una forma de organizar los artefactos del Modelo de Implementación en trozos más manejables. Un Subsistema de Implementación puede estar formado por componentes, interfaces y otros subsistemas. Además un Subsistema de Implementación puede implementar y proporcionar las interfaces que representan la funcionalidad que exportan en forma de operaciones. Su creación es responsabilidad del rol Ingeniero de Componentes.

99. Tablas de contenido de manuales, presentaciones y cursos

Las tablas de contenido del material de capacitación y soporte puede ser considerara como un artefacto aparte, o bien como una parte del propio material al que se refiere de acuerdo a la naturaleza del proyecto y del Cliente. Es responsabilidad del Editor Técnico y su objetivo es estructurar el contenido de cada uno de los materiales de soporte y capacitación que se elaborarán.

100. Unidad de despliegue

Una unidad de despliegue consiste de una construcción, documentos y artefactos de instalación. Una unidad de despliegue es asociada comúnmente un nodo en toda la red de computadoras y periféricos. La creación de las unidades de despliegue es responsabilidad del Administrador de la Configuración

101. Construcción

Una Construcción es una versión ejecutable del sistema, de una parte específica del mismo. El desarrollo del sistema transcurre a través de una sucesión de Constricciones cada vez mas completas hasta alcanzar el sistema completo. En Cada una de las iteraciones se producirá una construcción que es igual a la construcción de la iteración anterior más un incremento de funcionalidad, proporcionado por la compilación y enlace de los Componentes y Subsistemas de Implementación de la iteración.

102. Minuta

Debe contener: Objetivo, Asistentes, Moderador de junta, fecha, hora, duración, decisiones y compromisos tomados

103. Evaluación de proveedor(es)

Durante la etapa de la evaluación post-mortem del proyecto, el Administrador d Desarrollo valorará la participación y desempeño de los proveedores que intervinieron durante la elaboración del proyecto, con el objetivo de agregar esta información al Registro Histórico de Proveedores y considerarla en futuras selecciones de proveedor.

Explicando más detalle la relación que guardan cada uno de los planes utilizados, se listan los planes a continuación en orden de jerarquía:

Plan de Desarrollo del Proyecto
- **Plan de Contención**
- **Plan de Contingencia**
- **Plan de Aseguramiento de la Calidad**
- **Plan de la Administración de la Configuración**
  - Plan para la Administración de Datos
- **Plan de desarrollo del proveedor**
- **Plan de la integración de la construcción**
- **Plan de pruebas**
- Plan para la Administración de Requerimientos
- Plan para los Recursos del Proyecto
- Plan para las Habilidades y Conocimientos Necesarios
- Plan para Implicar a los Interesados
- Plan para Mantenimiento del Plan de Desarrollo del Proyecto
- Plan para Monitoreo y control del Proyecto
- Plan para la Administración de Acuerdos con Proveedores
- Plan para la Medición y Análisis

El Plan de Desarrollo incluye o se derivan de él otros planes, los planes mostrados en negritas son planes mencionados explícitamente como artefactos de la metodología propuesta, mientras que los planes que no están en negritas, son planes que debe contener el Plan de Desarrollo del Proyecto, y no se toman como artefactos por aparte.

De forma similar se puede explicar las políticas, a las cuales se menciona de manera general en la lista y descripción de artefactos, pero que a continuación se muestran las políticas específicas que se pueden incluir, nuevamente se colocan en negritas los artefactos mencionados explícitamente en la metodología:

Políticas
- Políticas de la Administración de requerimientos
- Políticas de la Planeación del proyecto
- Políticas del Monitoreo y control del proyecto
- Políticas de la Administración de proveedores
- Políticas de la Medición y el análisis
- Políticas de Aseguramiento de la calidad
- Políticas de la Administración de la configuración de software

---

En este capítulo se describe el segundo bloque de construcción de la metodología propuesta, relativo a los artefactos que se utilizarán en ella.

7

Propuestas sobre Actividades

Este capítulo describe las actividades a realizar dentro de la metodología por los distintos roles para utilizar o generar artefactos, tomando en cuanta las bases teóricas de UML, del Proceso Unificado y de CMMI.

7.1. Introducción

En los dos capítulos anteriores se han descrito dos de los bloques de construcción de la metodología propuesta, estos son los roles y los artefactos. Ahora toca el turno de describir el tercer bloque de construcción que se refiere a las actividades.

Al igual que los artefactos del Proceso Unificado, en él sus actividades se nombran, definen, y se establece su secuencia de ejecución. Para citar un ejemplo, podemos hablar de la actividad de creación del prototipo de la interfaz de usuario, a ésta actividad se le nombra, define, y se establece que debe ser realizada después de la identificación de los Casos de Uso básicos y antes de la creación del Modelo del Análisis.

En CMMI se definen las Prácticas y Subprácticas Específicas y Genéricas para cada una de las Áreas de Procesos de un nivel de madurez. Por medio de estas Prácticas y Subprácticas se definen y describen a detalle las actividades que se consideran importantes y se espera se realicen para alcanzar el objetivo de un Área de Procesos. Las Prácticas a través de las Subprácticas son muy claras, y son excelentes candidatas a considerar directamente como actividades en la metodología propuesta, sin embargo, algunas de ellas son consideraras o incluidas en actividades del Proceso Unificado, y otras son demasiado redundantes o demasiado particulares para ser consideradas como una sola actividad.

En este capítulo se propone el conjunto de actividades para la metodología propuesta, que incluyan a las actividades del Proceso Unificado y las practicas de CMMI para el nivel de madurez 2, buscando evitar redundancias y actividades demasiado particulares, y sobre todo considerando los roles y artefactos propuestos.

7.2. Organización de actividades

Para organizar el conjunto de actividades que se proponen, de nuevo se recurrirá a los paquetes de UML para agruparlas y clasificarlas.

Se puede partir de agrupar las actividades en dos paquetes:

- Paquete de Actividades Administrativas: que contendrá las actividades que están relacionadas a la administración de los procesos del proyecto de desarrollo de software, como por ejemplo, seleccionar al Administrador de Desarrollo, desarrollar el Plan de Desarrollo del Proyecto, revisar artefactos, etc.

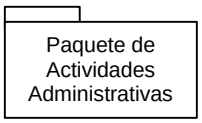


Figura 7-1: Representación gráfica del Paquete de Actividades Administrativas

- Paquete de Actividades Técnicas: que contendrá las actividades que están relacionadas con la creación del producto final del proyecto, como por ejemplo, identificar actores y Casos de Uso, Diseñar los Casos de Uso, integrar el sistema, etc.

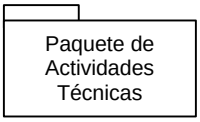


Figura 7-2: Representación gráfica del Paquete de Actividades Técnicas

A su vez el Paquete de Actividades Administrativas contendrá otros paquetes que organizarán las actividades ahí contenidas, y que coinciden con cada una de las áreas de procesos de CMMI nivel 2. Así mismo, dentro del Paquete de Actividades Técnicas se tendrán otros paquetes que organizarán sus actividades de acuerdo a los modelo del Proceso Unificado, mas un paquete adicional referente al material de capacitación.

A continuación se proporciona una lista de todas las actividades propuestas clasificadas por medio de los paquetes definidos:

Paquete de Actividades Administrativas

Paquete de la Administración de Requerimientos

1-Solicitar proyecto y proporcionar requerimientos (cliente)
2-Revisar la solicitud y los requerimientos (Administrador principal)
6-Analizar y ajustar la solicitud y los requerimientos (Cliente, Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y Administrador de desarrollo)
27-Realizar negociaciones y acuerdos (Cliente y Administrador principal)
82-Entrega formal del producto al cliente (Cliente, Administrador Principal, Administrador del Proyecto y Administrador de desarrollo)
85-Preparar propuestas de respuesta a las retroalimentaciones (Administrador del proyecto y Administrador de desarrollo)
87-Presentar respuestas a retroalimentaciones (Administrador del proyecto)

Paquete de Planeación del Proyecto

3-Seleccionar al Administrador del proyecto y al Administrador de la calidad (Administrador principal)
4-Seleccionar al Administrador de desarrollo (Administrador del proyecto)
5-Seleccionar al Administrador de la configuración (Administrador de la calidad)
8-Seleccionar el equipo de ingeniería (Administrador de desarrollo)
7-Identificar conocimientos y habilidades necesarios (Administrador de desarrollo)
22-Elaborar la evaluación de riesgos (Administrador principal, Administrador del proyecto, Administrador de la calidad y Administrador de desarrollo)
23-Elaborar el Plan de desarrollo del proyecto (Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y Administrador de desarrollo)
24-Calcular estimados monetarios (Administrador principal y Administrador del proyecto)
70-Actualizar Plan de trabajo y Organigrama (Administrador del proyecto)

Paquete de Monitoreo y Control del Proyecto

66-Monitorear avance del proyecto y colectar Métricas de productos y recursos (Administrador de desarrollo)

76-Convocar junta interna de revisión e informe general de avance. (Administrador del proyecto)
78-Convocar junta externa de revisión e informe general de avance (Administrador del proyecto)
79-Informar del avance de proyecto al Administrador principal (Administrador del proyecto, Administrador principal)
84-Monitorear retroalimentación de uso operacional (Administrador del proyecto y Administrador de desarrollo)

Paquete de la Administración de Proveedores

17-Identificar herramientas necesarias y productos o servicios externos (Administrador de la configuración y Administrador de desarrollo)
18-Definir criterios para evaluar proveedores potenciales (Administrador del proyecto y Administrador de desarrollo)
19-Identificar proveedores potenciales y distribuirles requerimientos solicitados (Administrador del proyecto y Administrador de desarrollo)
20-Evaluar propuestas, riesgos y capacidades de cada proveedor potencial (Administrador del proyecto y Administrador de desarrollo)
21-Seleccionar proveedor(es) (Administrador del proyecto y Administrador de desarrollo)
28-Establecer acuerdos con proveedor (es) (Administrador del proyecto y administrador de desarrollo)

Paquete de Medición y Análisis

68-Completar métricas de recursos (Administrador del proyecto)
69-Analizar Métricas y administrar excepciones y problemas (Administrador del proyecto)
77-Colectar métricas del proyecto (Administrador del proyecto y Administrador de desarrollo)
88-Promover evaluación post-mortem (Administrador del proyecto)
89-Evaluación Post-mortem del proyecto (Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y administrador de desarrollo)
90-Colectar métricas del producto terminado, generales de recursos, del proceso y del proyecto. (Administrador del proyecto)

Paquete de Aseguramiento de la Calidad

10-Revisar el Modelo del dominio y el Modelo del negocio (Administrador de la calidad)
13-Revisar preliminar del Modelo de casos de uso (Administrador de la calidad)
16-Revisar el establecimiento del Repositorio de la configuración de software (Administrador de la calidad)
25-Revisar documentos para negociación (Administrador de la calidad)
67-Revisar actividades, roles y artefactos (Administrador de la calidad)
86-Revisar propuestas de respuesta a las retroalimentaciones (Administrador de la calidad)

Paquete de la Administración de la Configuración

14-Identificar productos de trabajo, elementos adicionales y procedimientos especiales para la configuración de software (Administrador de la configuración)
15-Establecer el Repositorio de la configuración del proyecto (Administrador de la configuración)
26-Integrar Línea base inicial y otros artefactos iniciales al repositorio (Administrador de la configuración)
62-Integrar Línea base (Administrador de la configuración)
71-Recibir, registrar y convocar la evaluación de Solicitudes de cambio e informes de problemas. (Administrador de la configuración)
72-Definir impacto de cambio o de la solución al problema (Administrador de la configuración, Administrador del proyecto, Administrador de la calidad y Administrador de desarrollo)

73-Controlar y dar seguimiento a los cambios o soluciones autorizadas (Administrador de la configuración)
74-Integrar artefactos del Repositorio (Administrador de la configuración)
75-Revisar la integridad del Repositorio (Administrador de la configuración)
91-Cerrar repositorio del proyecto (Administrador de la configuración)

Paquete de Actividades Técnicas

Paquete del Modelo del Negocio y del Modelo del Dominio

9-Generar el Modelo del dominio y el Modelo del negocio (Cliente, Administrador de desarrollo y Analista del sistema)

Paquete del Modelo de Casos de Uso

11-Identificar actores y casos de uso (Cliente y Analista de sistema)
12-Priorizar los casos de uso (Administrador de desarrollo)
29-Identificar nuevos Actores y Casos de uso (Analista de sistema y Cliente)
30-Priorizar los nuevos Casos de uso (Administrador de desarrollo)
31-Detallar Casos de uso (Especificador de casos de uso)
32-Prototipar la Interfaz de usuario (Especificador de casos de uso)
33-Estructurar el Modelo de casos de uso (Administrador de desarrollo)

Paquete del Modelo del Análisis

34-Analizar la Arquitectura (Administrador de desarrollo)
35-Analizar Casos de uso (Ingeniero de Casos de uso)
36-Analizar Clases (Ingeniero de Componentes)
37-Analizar Paquetes (Ingeniero de Componentes)
38-Estructurar el Modelo del análisis (Administrador de desarrollo)

Paquete del Modelo del Diseño

39-Diseñar la Arquitectura (Administrador de desarrollo)
41-Diseñar Casos de uso (Ingeniero de Casos de uso)
42-Diseñar Clases (Ingeniero de componentes)
43-Diseñar subsistemas (Ingeniero de componentes)
45-Estructurar el Modelo de diseño (Administrador de desarrollo)

Paquete del Modelo de Despliegue

40-Iniciar la estructuración del Modelo de despliegue (Administrador de desarrollo)
52-Terminar la estructuración del Modelo de despliegue (Administrador de desarrollo)
80-Producir Unidades de despliegue (Administrador de la configuración)

Paquete del Modelo de Implementación

46-Implementar la Arquitectura (Administrador de desarrollo)
47-Planear la integración en la iteración (Integrador del sistema)
48-Implementar Clases (Ingeniero de componentes)
50-Implementar e integrar subsistemas (Ingeniero de componentes)
51-Integrar el sistema (Integrador del sistema)
53-Estructurar el modelo de implementación (Administrador de desarrollo)

Paquete del Modelo de Pruebas

44-Diseñar pruebas (Diseñador de pruebas)
49-Realizar Pruebas de unidad (Ingeniero de componentes)
54-Implementar los componentes de pruebas (Ingeniero de componentes)
55-Planificar pruebas (Diseñador de pruebas)
56-Supervisar la generación o el funcionamiento del ambiente de pruebas (Diseñador de pruebas)
57-Realizar pruebas de integración (Ingeniero de pruebas de integración)

58-Realizar pruebas del sistema (Ingeniero de pruebas de integración)
59-Evaluar pruebas y estructurar el Modelo de pruebas (Diseñador de pruebas)
60-Realizar pruebas de aceptación (Cliente)
61-Supervisar pruebas de aceptación (Administrador de desarrollo, Diseñador de pruebas)
81-Coordinar (realizar) prueba final de aceptación en sitio (Usuario, Cliente, Administrador del proyecto y Administrador de desarrollo)

Paquete del Material de Capacitación

63-Iniciar o continuar el análisis del Material de soporte y Material de capacitación necesarios para el usuario final (Editor técnico)
64-Estructurar o reestructurar el Material de soporte y Material de capacitación para el usuario final (Editor técnico)
65-Desarrollar el Material de soporte Material de capacitación para el usuario final (Editor técnico)
83-Proporcionar capacitación a usuarios finales (Editor técnico y Administrador de desarrollo)

7.3. Lista y descripción de actividades

A continuación se listan las actividades propuestas por la metodología, junto con una breve descripción, artefactos de entrada, artefactos de salidas y roles que las realizan:

1-Solicitar proyecto y proporcionar requerimientos (cliente)

El Cliente es quien inicia el ciclo de vida del proyecto, por medio de la entrega a la Organización de Desarrollo, específicamente al Administrador Principal, de la Solicitud formal del Proyecto y de la Especificación de Requerimientos. En esta actividad solo se realiza la solicitud del proyecto, misma que es evaluada por la organización de desarrollo, la cual emitirá posteriormente una propuesta de solución al cliente, para que posiblemente lleguen a acuerdos entre los dos por medio de un contrato.

Entradas:
Salidas: 96.Solicitud del proyecto
23.Especificación de requerimientos

2-Revisar la solicitud y los requerimientos (Administrador principal)

Posterior a la solicitud del proyecto por parte del Cliente, el Administrador Principal valora dicha solicitud y sus requerimientos.

Entradas: 96.Solicitud del proyecto
23.Especificación de requerimientos
Salidas:

3-Seleccionar al Administrador del proyecto y al Administrador de la calidad (Administrador principal)

Una vez que el Administrador Principal ha analizado los requerimientos y la solicitud, y en base a ellos, selecciona a un Administrador del Proyecto y a un Administrador de la Calidad cuyos perfiles y experiencia les permita participar como los roles correspondientes en el proyecto.

Entradas: 96.Solicitud del proyecto
23.Especificación de requerimientos
Salidas:

4-Seleccionar al Administrador de desarrollo (Administrador del proyecto)

La primera actividad del Administrador del Proyecto es analizar la Solicitud del Proyecto y la Especificación de Requerimiento, para que en base a ellas seleccione a un Administrador de Desarrollo acorde al proyecto.

Entradas: 96.Solicitud del proyecto
23.Especificación de requerimientos
Salidas:

5-Seleccionar al Administrador de la configuración (Administrador de la calidad)

La primera actividad del Administrador de la Calidad es analizar la Solicitud del Proyecto y la Especificación de Requerimiento, para que en base a ellas seleccione a un Administrador de la Configuración acorde al proyecto.

Entradas: 96.Solicitud del proyecto
23.Especificación de requerimientos
Salidas:

6-Analizar y ajustar la solicitud y los requerimientos (Cliente, Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y Administrador de desarrollo)

El Administrador Principal y el Administrador del Proyecto convocan una junta (con el cliente, el Administrador de la Calidad, el Administrador de la Configuración y el Administrador de Desarrollo) para analizar y posiblemente ajustar o completar la Especificación de Requerimientos, con el fin de llegar a una compresión mas precisa y común de los

requerimientos. Además inicia la creación de las Estimaciones del Proyecto, por medio de una estructuración simple del sistema en base a sus funciones principales.

Entradas: 96.Solicitud del proyecto
23.Especificación de requerimientos
Salidas: 26.Estimaciones del proyecto
96.Solicitud del proyecto (ajustada)
23.Especificación de requerimientos (ajustada)

7-Identificar conocimientos y habilidades necesarios (Administrador de desarrollo)

Sobre la base de la Solicitud de Requerimientos, la Especificación de Requerimientos y de las Estimaciones del Proyecto se debe determinar las habilidades y conocimientos necesarios de los Recursos Humanos a nivel técnico que participarán en el proyecto; además de definir, en caso de ser necesario, los mecanismos necesarios para adquirir dichos conocimientos y habilidades. En casos excepcionales (como promoción o integración de personal) se requerirá capacitar a los recursos humanos un Área de Procesos en específico

Entradas: 26.Estimaciones del proyecto
96.Solicitud del proyecto (ajustada)
23.Especificación de requerimientos (ajustada)
Salidas: 53.Mecanismos para la adquisición de conocimientos y habilidades

8-Seleccionar el equipo de ingeniería (Administrador de desarrollo)

Considerando la documentación anterior, sobre todo las Estimaciones del Proyecto y los Mecanismos para la Adquisición de Conocimientos y Habilidades, el Administrador de Desarrollo elige los recursos humanos para integrar el Grupo de Ingeniería para desarrollar el sistema a nivel técnico, plasmando la organización, jerarquía, responsabilidades e interacciones de dicho grupo en el documento Organigrama de Ingeniería.

Entradas: 53.Mecanismos para la adquisición de conocimientos y habilidades
26.Estimaciones del proyecto
96.Solicitud del proyecto
23.Especificación de requerimientos (ajustada)
Salidas: 66.Organigrama de ingeniería

9-Generar el Modelo del dominio y el Modelo del negocio (Cliente, Administrador de desarrollo y Analista del sistema)

Con ayuda del Cliente, el Administrador de Desarrollo y el Analista de Sistemas describen y especifican el contexto donde funcionará el sistema software a construir, primero a nivel de negocio u organización con el Modelo del Negocio, y posteriormente a un nivel mas especifico al contexto del sistema software dentro del negocio con el Modelo del Dominio. Durante este proceso se inicia la construcción del Glosario de Términos y se completan aun más las Estimaciones del Proyecto.

Entradas: 96.Solicitud del proyecto (ajustada)
23.Especificación de requerimientos (ajustada)
26.Estimaciones del proyecto
Salidas: 63.Modelo del dominio
64.Modelo del negocio
31.Glosario de términos
26.Estimaciones del proyecto

10-Revisar el Modelo del dominio y el Modelo del negocio (Administrador de la calidad)

Esta es la primera actividad propia del rol del Administrador de la Calidad. En ella se revisa que el modelo del dominio y el Modelo del Negocio generados estén en conformidad con Estándares, Políticas, los procedimientos definidos para su creación (Definición de Procedimiento), los lineamientos de los artefactos que incluyen (Definición de Artefactos) y términos incluidos en el Glosario. Como resultado se obtienen los modelos corregidos y reportes sobre la calidad de los artefactos y el desempeño de las actividades para crearlos (Reportes de Auditoría por Artefacto y Reportes de Auditoría por Actividad, respectivamente).

El objetivo de esta Auditoría temprana de artefactos es que se cuente con los modelos lo suficientemente correctos de cara a una próxima negociación con el Cliente para obtener el contrato.

Entradas: 16.Definición de procedimientos
76.Políticas
24.Estándares
14.Definición de artefactos
63.Modelo del dominio
64.Modelo del negocio
31.Glosario de términos

Salidas: 89.Reportes de auditoría por actividad
90.Reportes de auditoría por artefacto
63.Modelo del dominio
64.Modelo del negocio
31.Glosario de términos

11-Identificar actores y casos de uso (Cliente y Analista de sistema)

Los objetivos de esta actividad son delimitar el sistema en su entorno, esbozar quién y qué actores interactúan con el sistema y que funcionalidad se espera del sistema. Esta actividad es de las más decisivas para obtener adecuadamente los requerimientos del sistema, por lo cual es necesaria la participación del Cliente y la utilización del Modelo del Negocio y del Modelo del Negocio. De manera general la actividad consta de la identificación de Actores, identificación de Casos de Uso, descripción breve de cada Caso de Uso principal y descripción breve del Modelo de Casos de Uso preliminar.

Como resultado de esta actividad, se obtiene una versión preliminar del Modelo de Casos de Uso a manera de esbozo para generar las Estimaciones del Proyecto y generar las cotizaciones de cara a la negociación del Contrato del Proyecto.

Durante el desarrollo de esta actividad, también se puede iniciar la elaboración de la Matriz de Traza de Requerimientos, donde se especifica cual Caso de Uso satisface cierto requerimiento.

Entradas: 63.Modelo del dominio
64.Modelo del negocio
31.Glosario de términos
23.Especificación de requerimientos

Salidas: 52.Matriz de traza de requerimientos
31.Glosario de términos
58.Modelo de casos de uso

12-Priorizar los casos de uso (Administrador de desarrollo)

El propósito de esta actividad es determinar el orden de desarrollo de los Casos de Uso a través de las iteraciones. Esta información se plasma en el Modelo de Casos de Uso y en las Estimaciones del Proyecto, y se utiliza durante la planificación del proyecto, donde se indicará que Casos de Uso se desarrollarán dentro de las iteraciones calendarizadas.

Entradas: 31.Glosario de términos
26.Estimaciones del proyecto
23.Especificación de requerimientos
58.Modelo de casos de uso

Salidas: 26.Estimaciones del proyecto
58.Modelo de casos de uso (esbozo con prioridades de casos de uso)

13-Revisar preliminar del Modelo de casos de uso (Administrador de la calidad)

En esta actividad se revisa que el Modelo de Casos de Uso este en conformidad con Estándares, Políticas, los procedimientos definidos para su creación (Definición de Procedimiento), los lineamientos de los artefactos que incluye (Definición de Artefactos) y términos incluidos en el Glosario. Como resultado se obtiene el modelo corregido y reportes sobre la calidad de los artefactos incluidos y el desempeño de las actividades para crearlo

(Reportes de Auditoría por Artefacto y Reportes de Auditoría por Actividad, respectivamente). El objetivo de esta auditoría temprana del artefacto es que se cuente con el modelo lo suficientemente correcto de cara a la planeación y a una próxima negociación con el Cliente para obtener el contrato.

Entradas: 76.Políticas
24.Estándares
16.Definición de procedimientos
14.Definición de artefactos
58.Modelo de casos de uso (esbozo con prioridades de casos de uso)

Salidas: 89.Reportes de auditoría por actividad
90.Reportes de auditoría por artefacto
58.Modelo de casos de uso (esbozo con prioridades de casos de uso)

14-Identificar productos de trabajo, elementos adicionales y procedimientos especiales para la configuración de software (Administrador de la configuración)

En esta actividad el Administrador de la Configuración identifica elementos específicos del proyecto y adicionales a los que comúnmente administra, esto es debido a que los procedimientos o políticas no pueden contemplar al cien por ciento todos los elementos de configuración que deben ser administrados, pudiéndose tener el caso de ciertos proyectos especiales que requieran elementos especiales y administrables para la configuración de software.

Al mismo tiempo se documenta formalmente todos los productos y elementos (incluidos los especiales) que va a administrar a través del desarrollo del proyecto, los procedimientos especiales específicos del proyecto que se deriven para la Administración de la Configuración, y con esto se completa aun mas las Estimaciones del Proyecto.

Entradas: 58.Modelo de casos de uso (esbozo con prioridades de casos de uso)
23.Especificación de requerimientos
26.Estimaciones del proyecto
31.Glosario de términos

Salidas: 46.Lista de productos y elementos adicionales de la configuración
47.Lista de productos y elementos de la configuración
26.Estimaciones del proyecto
78.Procedimientos especiales de la configuración

15-Establecer el Repositorio de la configuración del proyecto (Administrador de la configuración)

Se debe establecer el Repositorio de la Configuración del Proyecto donde se almacenarán Elementos, Componentes y Unidades de la Configuración, esto incluye el medio físico de almacenamiento, el sistema de configuración de software y control de cambios, las herramientas para administrar al sistema de configuración de software y control de cambios, la base de datos de solicitudes de cambios, y la configuración de niveles de control y promoción. Esta actividad se debe realizar guiada por las definiciones de procedimientos correspondientes (incluyendo los procedimientos especiales para este proyecto)

Entradas: 46.Lista de productos y elementos adicionales de la configuración
47.Lista de productos y elementos de la configuración
26.Estimaciones del proyecto
78.Procedimientos especiales de la configuración

Salidas: 19.Descripción de niveles de promoción
20.Descripción del repositorio
92.Repositorio del proyecto

16-Revisar el establecimiento del Repositorio de la configuración de software (Administrador de la calidad)

Esta es la primera auditoría por parte de la Administración de la Calidad al Repositorio de la Configuración de Software, pero es la auditoría más importante por que de su establecimiento inicial depende el desempeño de las actividades de la Administración de la Configuración. En

esta actividad se confirma el cumplimiento con los estándares, procedimientos y políticas aplicables a la Administración de la Configuración en sus actividades de establecer el repositorio. Así mismo, se revisa que el repositorio en sí este en conformidad con Estándares, Políticas, los lineamientos de los artefactos y términos incluidos en el Glosario. Como resultado se obtiene el repositorio posiblemente corregido y reportes sobre su calidad y el desempeño de las actividades para crearlo (Reportes de Auditoría por Artefacto y Reportes de Auditoría por Actividad, respectivamente).

Entradas: 14.Definición de artefactos
16.Definición de procedimientos
24.Estándares
76.Políticas
20.Descripción del repositorio
78.Procedimientos especiales de la configuración
46.Lista de productos y elementos adicionales de la configuración
47.Lista de productos y elementos de la configuración

Salidas: 89.Reportes de auditoría por actividad
90.Reportes de auditoría por artefacto

17-Identificar herramientas necesarias y productos o servicios externos (Administrador de la configuración y Administrador de desarrollo)

Sobre la base de los artefactos e información del proyecto que se ha generado se deben identificar las tecnologías y productos necesarios para la satisfacción de los requerimientos, una vez que se hayan identificado se deben detectar aquellos que no pueden ser cubiertos por la organización de desarrollo de software con su infraestructura actual y se genera una descripción de productos y servicios que será necesario adquirir de forma externa. Esta identificación de nuevas adquisiciones de productos o servicios ayudara para estimar los costos y tiempos del proyecto.

Entradas: 78.Procedimientos especiales de la configuración
26.Estimaciones del proyecto
52.Matriz de traza de requerimientos
96.Solicitud del proyecto
23.Especificación de requerimientos
58.Modelo de casos de uso

Salidas: 44.Lista de nuevas herramientas, productos o servicios para adquirir

18-Definir criterios para evaluar proveedores potenciales (Administrador del proyecto y Administrador de desarrollo)

Se debe establecer y documentar los criterios para evaluar proveedores potenciales, en base a la naturaleza del proyecto y a la Lista de Nuevas Herramientas, productos o servicios para adquirir. Una vez definidos estos criterios, se redacta un documento donde se especifiquen los requerimientos solicitados a externos por cada adquisición, que sea compatible con los requerimientos del cliente y los estándares y políticas de la organización de desarrollo de software.

Entradas: 23.Especificación de requerimientos
44.Lista de nuevas herramientas, productos o servicios para adquirir

Salidas: 13.Criterios de evaluación de proveedores
93.Requerimientos a externos

19-Identificar proveedores potenciales y distribuirles requerimientos solicitados (Administrador del proyecto y Administrador de desarrollo)

De acuerdo a la naturaleza de las adquisiciones, a los requerimientos a externos y al Registro Histórico de Proveedores, se identifican a los proveedores potenciales y se les distribuye la solicitud de la adquisición y los Requerimientos a Externos, esperando sus propuestas de cada uno de ellos.

Entradas: 13.Criterios de evaluación de proveedores
93.Requerimientos a externos

85.Registro Histórico de Proveedores
Salidas: 45.Lista de posibles proveedores

20-Evaluar propuestas, riesgos y capacidades de cada proveedor potencial (Administrador del proyecto y Administrador de desarrollo)

Una vez que se cuente con las propuestas y ofertas de todos los proveedores candidatos, se procede a evaluar y comparar cada una de ellas y se agrega estas propuestas y su evaluación al Registro Histórico de Proveedores

Entradas: 81.Propuestas/ofertas de proveedores
85.Registro Histórico de Proveedores
93.Requerimientos a externos
Salidas: 85.Registro Histórico de Proveedores

21-Seleccionar proveedor(es) (Administrador del proyecto y Administrador de desarrollo)

En base a la evaluación y comparación de cada una de las propuestas y ofertas de los proveedores candidatos, el Administrador de Proyecto y el Administrador de Desarrollo seleccionan un Proveedor entre todos ellos como primera opción para negociaciones y acuerdos futuros durante este proyecto. También deben seleccionar una segunda y tercer opción en caso de que no se llegue a un acuerdo con la primera opción.

Dentro de la propuesta de los proveedores debe estar incluido el Plan del Proveedor para la satisfacción de la adquisición, pero en esta actividad debe ser solicitado a los proveedores seleccionados como un documento independiente y más detallado (de cara a la planeación del proyecto). Así mismo, en base a la evaluación y comparación de las propuestas de los proveedores candidatos, el Administrador de Desarrollo y el Administrador del Proyecto deben redactar los Riesgos y Capacidades de Proveedor(es) Seleccionado(s) para ser considerados en la identificación de riesgos del proyecto.

Entradas: 85.Registro Histórico de Proveedores
Salidas: 71.Plan de desarrollo de proveedor
81.Propuestas/ofertas de proveedores
94.Riesgos y capacidades de proveedor(es) seleccionado(s)

22-Elaborar la evaluación de riesgos (Administrador principal, Administrador del proyecto, Administrador de la calidad y Administrador de desarrollo)

Debido a la importancia de esta actividad, debe ser realiza por el grupo de administradores del proyecto, y a través de ella se identifican o descubren, analizan (incluyendo sus orígenes, impacto, probabilidad, etc.), clasifican y priorizan los riesgos. Después de identificar y documentar los riesgos, se debe llegar a un acuerdo entre todo el grupo de administradores en la completitud y exactitud de los riesgos identificados, para así evitar futuras tensiones.

Entradas: 94.Riesgos y capacidades de proveedor(es) seleccionado(s)
23.Especificación de requerimientos
58.Modelo de casos de uso
31.Glosario de términos
46.Lista de productos y elementos adicionales de la configuración
78.Procedimientos especiales de la configuración
63.Modelo del dominio
64.Modelo del negocio
96.Solicitud del proyecto
47.Lista de productos y elementos de la configuración
26.Estimaciones del proyecto
Salidas: 48.Lista de riesgos

23-Elaborar el Plan de desarrollo del proyecto (Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y Administrador de desarrollo)

En esta actividad el grupo de administradores establece el Plan de Desarrollo del Proyecto que será la base de la administración del proyecto, considerando para su elaboración todos los

artefactos elaborados anteriormente, principalmente las Estimaciones del Proyecto y la Especificación de Requerimientos. Una vez elaborado y establecido, el Plan de Desarrollo del Proyecto deberá ser actualizado periódicamente por el Administrador del Proyecto.
Además de crear el Plan de Desarrollo del Proyecto, también se crea el Plan de Contingencia y el Plan de Contención (para evitar los riesgos importantes y limitar sus efectos en caso de que se presenten), el Plan de Aseguramiento de la Calidad (que se deriva directamente del Plan de Desarrollo del Proyecto y dirige las actividades de la Administración de la Calidad), el Plan de la Administración de la Configuración (que se deriva directamente del Plan de Desarrollo del Proyecto y dirige las actividades de la Administración de la Configuración), y el Estimado de Recursos de Cómputo.
También se actualizan la Matriz de Traza de Requerimientos, y posiblemente el Organigrama de Ingeniería.

Entradas: 53.Mecanismos para la adquisición de conocimientos y habilidades
26.Estimaciones del proyecto
23.Especificación de requerimientos
58.Modelo de casos de uso
96.Solicitud del proyecto
66.Organigrama de ingeniería
48.Lista de riesgos
81.Propuestas/ofertas de proveedores
52.Matriz de traza de requerimientos

Salidas: 52.Matriz de traza de requerimientos
72.Plan de desarrollo del proyecto
69.Plan de contención
70.Plan de contingencia
66.Organigrama de ingeniería
27.Estimado de recursos de cómputo
68.Plan de aseguramiento de la calidad
73.Plan de la administración de la configuración

24-Calcular estimados monetarios (Administrador principal y Administrador del proyecto)

En esta actividad se define el costo monetario total de proyecto considerando las actividades, recursos, esfuerzos y riesgos para las áreas de desarrollo, aseguramiento de la calidad y administración del proyecto, todo esto documentado en los diversos artefactos elaborados hasta este momento. Se obtiene como salida el documento de Cotización del Proyecto, con el cual se inicia la negociación del contrato con el Cliente.

Entradas: 63.Modelo del dominio
64.Modelo del negocio
58.Modelo de casos de uso
69.Plan de contención
70.Plan de contingencia
48.Lista de riesgos
72.Plan de desarrollo del proyecto
26.Estimaciones del proyecto
27.Estimado de recursos de cómputo
44.Lista de nuevas herramientas, productos o servicios para adquirir
81.Propuestas/ofertas de proveedores

Salidas: 12.Cotización del proyecto

25-Revisar documentos para negociación (Administrador de la calidad)

Debido a la importancia de los documentos que se han elaborado para la negociación del contrato (Plan de Desarrollo del Proyecto, Lista de Riesgos y Cotización del proyecto), el Administrador de la Calidad revisa su conformidad con Estándares, Políticas, procedimientos definidos para su creación (Definición de Procedimiento), lineamientos de artefactos (Definición de Artefactos) y términos incluidos en el Glosario. Como resultado se pueden obtener estos documentos corregidos y reportes sobre la calidad de los artefactos y el desempeño de las

actividades para crearlo (Reportes de Auditoría por Artefacto y Reportes de Auditoría por Actividad, respectivamente).

Entradas: 16.Definición de procedimientos
76.Políticas
24.Estándares
14.Definición de artefactos
72.Plan de desarrollo del proyecto
12.Cotización del proyecto
48.Lista de riesgos

Salidas: 89.Reportes de auditoría por actividad
90.Reportes de auditoría por artefacto

26-Integrar Línea base inicial y otros artefactos iniciales al repositorio (Administrador de la configuración)

Una vez que se han creado y revisado los artefacto previos a la negociación del contrato, se debe obtener la autorización del grupo de administradores para agregarlos al Repositorio de la Configuración del Proyecto, de estos artefactos algunos (los que tiene que ver con el producto en si) forman parte de la Línea Base Inicial del Producto, por lo cual se crea la Primer versión de la Línea Base con ellos. Después se procede a documentar el contenido del Repositorio y a liberar todos los artefactos del repositorio (incluida la Línea Base inicial) para hacerlos disponibles a los grupos de roles adecuados por medio del sistema de Configuración de Software.

Entradas: 47.Lista de productos y elementos de la configuración
19.Descripción de niveles de promoción
46.Lista de productos y elementos adicionales de la configuración
20.Descripción del repositorio
78.Procedimientos especiales de la configuración

Salidas: 88.Reporte del contenido del repositorio
92.Repositorio del proyecto
41.Línea base

27-Realizar negociaciones y acuerdos (Cliente y Administrador principal)

En esta actividad se realiza la negociación con el Cliente para obtener el contrato a través del cual se satisfacerán sus requerimientos por parte de la organización de desarrollote software, exponiendo la forma en que se logrará a través de las actividades, recursos, estrategias y técnicas mostradas en el Plan de Desarrollo del Proyecto y la Cotización del Proyecto. Durante esta negociación, se pueden ajustar algunos de los artefactos. En caso de que se consiga la firma del contrato, se procede con las actividades de administración, desarrollo y aseguramiento de la calidad encaminadas a la satisfacción de los requerimientos del cliente, en caso contrario se realizará la evaluación Post-mortem del proyecto con objetivo de documentar el caso.

Entradas: 12.Cotización del proyecto
48.Lista de riesgos
58.Modelo de casos de uso
72.Plan de desarrollo del proyecto

Salidas: 11.Contrato del proyecto

28-Establecer acuerdos con proveedor (es) (Administrador del proyecto y administrador de desarrollo)

Una vez que se ha firmado el contrato entre el cliente y la Organización de Desarrollo de Software, se procede a firmar el contrato con el o los proveedores que satisfacerán los productos o servicios requeridos como externos para el desarrollo del proyecto. Durante esta negociación se obtiene formalmente la lista de artefactos que entregara el proveedor.

Entradas: 71.Plan de desarrollo de proveedor
81.Propuestas/ofertas de proveedores
94.Riesgos y capacidades de proveedor(es) seleccionado(s)

Salidas: 10.Contrato con proveedor
42.Lista de artefactos que entregará proveedor

29-Identificar nuevos Actores y Casos de uso (Analista de sistema y Cliente)

El objetivo de esta actividad es continuar con la delimitación del sistema en su entorno, definición de quién y qué Actores interactúan con el sistema y que funcionalidad se espera del sistema. Esta actividad es de las más decisivas para obtener adecuadamente los requerimientos del sistema, por lo cual es necesaria la participación del Cliente y la utilización del Modelo del Negocio y del Modelo del Negocio. De manera general la actividad consta de la identificación de nuevos Actores, identificación de nuevos Casos de Uso, descripción breve de cada nuevo Caso de Uso principal y descripción breve del Modelo de Casos de Uso.

Como resultado de esta actividad, se obtiene una versión preliminar del Modelo de Casos de Uso a manera de esbozo, para que sirva de base para las actividades de priorizar los casos de Uso, detallar los casos de uso y prototipar la interfaz de usuario

Posteriormente este esbozo del Modelo de Casos de Uso se estructura y mejora para ser tomado en su conjunto como base para los demás flujos de trabajo (análisis, diseño, construcción y pruebas).

Durante el desarrollo de esta actividad, también se actualiza la Matriz de Traza de Requerimientos y el Glosario de Términos

Entradas: 31.Glosario de términos
63.Modelo del dominio
64.Modelo del negocio
23.Especificación de requerimientos
58.Modelo de casos de uso

Salidas: 31.Glosario de términos
58.Modelo de casos de uso
52.Matriz de traza de requerimientos

30-Priorizar los nuevos Casos de uso (Administrador de desarrollo)

El propósito de esta actividad es determinar el orden de desarrollo de los Casos de Uso (incluidos los recién identificados) a través de las iteraciones. Esta información se plasma en el Modelo de Casos de Uso y en Plan de Desarrollo del Proyecto donde se indicará cuáles Casos de Uso se desarrollarán dentro de las iteraciones calendarizadas.

Entradas: 31.Glosario de términos
58.Modelo de casos de uso
23.Especificación de requerimientos

Salidas: 58.Modelo de casos de uso

31-Detallar Casos de uso (Especificador de casos de uso)

El objetivo de detallar cada Caso de Uso es describir su flujo de sucesos completo Para la realización de esta actividad es importante poder trabajar de forma interactiva con el Cliente, Usuario, o alguien que lo represente con suficiente conocimiento de los requerimientos. Como salida se obtendrán los Casos de Uso detallados.

Entradas: 23.Especificación de requerimientos
63.Modelo del dominio
64.Modelo del negocio
58.Modelo de casos de uso
31.Glosario de términos
82.Prototipo de interfaz de usuario (iteraciones anteriores)

Salidas: 4.Casos de uso

32-Prototipar la Interfaz de usuario (Especificador de casos de uso)

El objetivo de esta actividad es construir un Prototipo de Interfaz de Usuario, que permita al Usuario llevar acabo los Casos de Uso de manera eficiente. Se inicia trabajando con los Casos de Uso intentando discernir qué se necesita en las Interfaces de Usuario para habilitar los Casos de Uso para cada Actor, es decir, se crea un Diseño Lógico de la Interfaz de Usuario

para comprender las necesidades antes de intentar realizarlas. Después se crea el diseño físico de la Interfaz de Usuario, y se crea el prototipo que ilustre como se podrá usar el sistemas por cada Usuario.

Entradas: 23.Especificación de requerimientos
58.Modelo de casos de uso
31.Glosario de términos
4.Casos de uso
Salidas: 82.Prototipo de interfaz de usuario

33-Estructurar el Modelo de casos de uso (Administrador de desarrollo)

En este punto se reestructura el Modelo de Casos de Uso completo tomado como un todo, para hacer que el modelo sea más fácil de entender y de trabajar con él, para esto, el Administrador de Desarrollo debe encontrar funcionalidades compartidas, extensiones de funcionalidades adicionales y opcionales, e identificar otras relaciones entre los Casos de Uso actuales. Se debe agregar al Modelo de Casos de Uso, una descripción actual del modelo para formar parte de la descripción de la Arquitectura

Entradas: 23.Especificación de requerimientos
58.Modelo de casos de uso
4.Casos de uso
Salidas: 58.Modelo de casos de uso (estructurado o mejorado)

34-Analizar la Arquitectura (Administrador de desarrollo)

En esta actividad se esboza el Modelo del Análisis y se comienza a definir la Arquitectura del Sistema, mediante la identificación de Paquetes del Análisis (descubriendo aspectos comunes entre Paquetes, Paquetes de Servicios y dependencias entre Paquetes), identificación de Clases del Análisis evidentes e identificar requerimientos especiales comunes. Para iteraciones posteriores a la primera, se tendrá como entrada a esta actividad el Modelo del Análisis de la iteración anterior.

Entradas: 64.Modelo del negocio
63.Modelo del dominio
58.Modelo de casos de uso
23.Especificación de requerimientos
Salidas: 61.Modelo del análisis (esbozado)
67.Paquetes del análisis (esbozados)
5.Clases del análisis (esbozadas)

35-Analizar Casos de uso (Ingeniero de Casos de uso)

Se debe analizar un Caso de Uso para identificar las Clases del Análisis cuyos objetos son necesarios para llevar a cabo el flujo de sucesos del Casos de Uso, distribuir el comportamiento de los Caso de Uso entre las Clases del Análisis identificadas, identificar responsabilidades, atributos y asociaciones de las Clases del Análisis e identificar requerimientos especiales sobre la realización de un Caso de Uso particular. Para esto se debe hacer lo siguiente: completar la descripción de los Casos de Uso para entender mejor su comportamiento interno, representar este comportamiento por medio de las realizaciones donde intervengan las Clases del Análisis, tal vez encontrar nuevas Clases del Análisis (incluyendo la descripción de sus responsabilidades, atributos, asociaciones y dependencias) y capturar requerimientos especiales para la realización del Caso de Uso. Durante esta actividad se pueden utilizar Diagramas de Interacción, Diagramas de Actividades, Diagramas de Clases y Diagramas de Estados.

Entradas: 61.Modelo del análisis (esbozado)
63.Modelo del dominio
64.Modelo del negocio
58.Modelo de casos de uso
23.Especificación de requerimientos
Salidas: 83.Realizaciones de Casos de Uso – Análisis
5.Clases del análisis (esbozadas)

36-Analizar Clases (Ingeniero de Componentes)

Para analizar las clases es necesario identificar y mantener sus responsabilidades, identificar y mantener sus atributos, identificar sus asociaciones y agregaciones, identificar generalizaciones, y capturar requisitos especiales. Durante esta actividad se pueden identificar nuevas Clases del Análisis, que se deben de analizar y actualizar en los demás artefactos del Modelo de Análisis.

Entradas: 83.Realizaciones de Casos de Uso – Análisis
5.Clases del análisis (esbozadas)
Salidas: 5.Clases del análisis (terminadas)

37-Analizar Paquetes (Ingeniero de Componentes)

Los objetivos de analizar los Paquetes del análisis son: garantizar que los Paquetes del Análisis sean débilmente acoplados, que cumplan su objetivo de realizar algunos Casos de Uso, y describir las dependencias de forma que se pueda estimar el efecto de los cambios futuros. Para realizar esta actividad se debe definir y mantener las dependencias de Paquete con otros Paquetes cuyas Clases contenidas estén asociadas a él, se debe asegurar de que el Paquete contiene las Clases correctas, y se debe limitar las dependencias entre Paquetes.

Entradas: 61.Modelo del análisis (esbozado)
67.Paquetes del análisis (esbozados)
Salidas: 67.Paquetes del análisis (terminados)

38-Estructurar el Modelo del análisis (Administrador de desarrollo)

Al final del flujo de trabajo del análisis se reestructura el Modelo del Análisis completo tomado como un todo, para hacer que el modelo sea más fácil de entender y de trabajar de él, para esto el Administrador de Desarrollo debe mantener la consistencia e integridad del modelo a través de los cambios y las agregaciones.

Entradas: 61.Modelo del análisis (esbozado)
67.Paquetes del análisis (terminados)
5.Clases del análisis (terminadas)
83.Realizaciones de Casos de Uso – Análisis
Salidas: 61.Modelo del análisis (estructurado y mejorado)

39-Diseñar la Arquitectura (Administrador de desarrollo)

El objetivo esta actividad es esbozar de forma preliminar los Modelos de Diseño y Despliegue mediante la identificación de los siguientes elementos: Nodos y sus configuraciones de red, Subsistemas y sus Interfaces y dependencias, Clases del Diseño significativas para la Arquitectura, y Mecanismos de Diseño Genéricos. A lo largo de esta actividad se consideran diversas posibilidades de reutilización. Los Subsistemas, Interfaces u otros elementos de diseño resultantes se añadirán posteriormente al Modelo de Diseño. Los resultados de esta actividad son utilizados como base para la elaboración de las actividades del flujo de diseño, al final de estas, el Administrador de Desarrollo vuelve a retomar todo el Modelo del Diseño para estructurarlo, darle consistencia y mejorarlo. Para iteraciones posteriores a la primera, se tendrá como entrada de esta actividad el Modelo del Diseño de la iteración anterior.

Entradas: 61.Modelo del análisis
58.Modelo de casos de uso
23.Especificación de requerimientos
Salidas: 87Reglas específicas de diseño
21.Especificación de ambientes y herramientas
22.Especificación de bibliotecas comunes
97.Subsistemas de diseño (esbozados)
36.Interfaces diseñadas de clases (esbozadas)
37.Interfaces diseñadas de subsistemas (esbozadas)
62.Modelo del diseño (esbozado)
6.Clases del diseño (esbozadas)

40-Iniciar la estructuración del Modelo de despliegue (Administrador de desarrollo)

Durante el flujo de trabajo del diseño se cuenta ya con información suficiente (Subsistemas definidos, Subsistemas de Servicio identificados, y todo cohesivo y débilmente acoplado) para iniciar la estructuración o reestructuración del Modelo de Despliegue, que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad (Subsistemas de alto nivel y de servicios) entre los Nodos de cómputo. Durante esta actividad se puede agregar información a la Especificación de Ambientes y Herramientas. Para iteraciones posteriores a la primera, se tendrá como entrada de esta actividad el Modelo de Despliegue de la iteración anterior.

Entradas: 21.Especificación de ambientes y herramientas
79.Proceso de migración (de iteraciones anteriores)
97.Subsistemas de diseño (esbozados)
36.Interfaces diseñadas de clases (esbozadas)
37.Interfaces diseñadas de subsistemas (esbozadas)
62.Modelo del diseño (esbozado)

Salidas: 21.Especificación de ambientes y herramientas
79.Proceso de migración (actualizado)
59.Modelo de despliegue (esbozado)

41-Diseñar Casos de uso (Ingeniero de Casos de uso)

Los objetivos de esta actividad son: identificar las Clases del Diseño y/o los Subsistemas del Diseño que son necesarios para llevar a cabo el flujo de sucesos de los Casos de Uso, distribuir el comportamiento de los Casos de Uso entre los objetos del diseño y/o entre los Subsistemas participantes, definir requerimientos sobre las operaciones de las Clases del Diseño y/o sobre los Subsistemas del Diseño y sus Interfaces, y capturar los requerimientos de implementación de los Casos de Uso. Para ello, en cada Caso de Uso el Ingeniero de Casos de Uso debe identificar las Clases de Diseño participantes, describir las interacciones entre los objetos del diseño, identificar los Subsistemas de Diseño y las Interfaces participantes, describir interacciones entre Subsistemas, y capturar los requerimientos de implementación.

Entradas: 21.Especificación de ambientes y herramientas
61.Modelo del análisis
58.Modelo de casos de uso
62.Modelo del diseño (esbozado)
59.Modelo de despliegue (esbozado)
21.Especificación de ambientes y herramientas
79.Proceso de migración (actualizado)
87Reglas específicas de diseño
22.Especificación de bibliotecas comunes

Salidas: 84.Realizaciones de Casos de Uso – Diseño
6.Clases del diseño (esbozadas)
36.Interfaces diseñadas de clases (esbozadas)
97.Subsistemas de diseño (esbozados)
37.Interfaces diseñadas de subsistemas (esbozadas)
22.Especificación de bibliotecas comunes

42-Diseñar Clases (Ingeniero de componentes)

El propósito de esta actividad es crear Clases del Diseño que cumplan su papel en las Realizaciones de Casos de Uso-Diseño y en los requerimientos no funcionales involucrados. Esto incluye definir y mantener el propio diseño de la Clase, sus operaciones, sus atributos, las relaciones en que participa, sus métodos, los estados que pueda tomar, sus dependencias, los requerimientos de su implementación, y sus interfaces. Para esto se debe auxiliar con una descripción textual y diversos tipos de diagramas (de estados, de clases y de interacción). Esta actividad es de suma importancia por que a partir de la identificación de un tipo especial de Clases de Diseño, llamadas persistentes, se puede comenzar a definir el modelo de datos que nos ayudara a diseñar la Base de Datos a utilizar, en caso de requerirla. Para esto se debe relacionar las Clases de Diseño persistentes al Modelo de Datos (por medio de un Diagrama

de Clases persistentes), estimar el volumen y crecimiento de datos, optimizar el Modelo de Datos, optimizar el acceso a datos, definir características de almacenamiento, definir relaciones entre tablas, definir reglas de integridad referencial y de datos, y distribuir el comportamiento de las clases persistentes a la base de datos.

Entradas: 21.Especificación de ambientes y herramientas
84.Realizaciones de Casos de Uso – Diseño
6.Clases del diseño (esbozadas)
36.Interfaces diseñadas de clases (esbozadas)
5.Clases del análisis
79.Proceso de migración (actualizado)
87Reglas específicas de diseño
22.Especificación de bibliotecas comunes

Salidas: 6.Clases del diseño (terminadas)
25.Modelo de Datos
36.Interfaces diseñadas de clases (terminadas)
22.Especificación de bibliotecas comunes

43-Diseñar subsistemas (Ingeniero de componentes)

El diseño de Subsistemas tiene como objetivo detallar los Subsistemas del Diseño ya identificados, identificar nuevos Subsistemas del Diseño y mantener la integridad de todos ellos. Para ello el Ingeniero de componentes debe garantizar que un Subsistema (nuevo o ya identificado) sea tan independiente como sea posible de otros Subsistemas y/o de sus Interfaces, y que un Subsistema implemente las Interfaces correctas y de forma adecuada. También en esta actividad, se diseñan las variables y constantes que van a parametrizar al sistema

Entradas: 37.Interfaces diseñadas de subsistemas (esbozadas)
97.Subsistemas de diseño (esbozados)
62.Modelo del diseño (esbozado)
6.Clases del diseño (terminadas)
25.Modelo de Datos
36.Interfaces diseñadas de clases (terminadas)
22.Especificación de bibliotecas comunes
21.Especificación de ambientes y herramientas
79.Proceso de migración (actualizado)
87Reglas específicas de diseño

Salidas: 18.Definición de variables y constantes de configuración de subsistemas
97.Subsistemas de diseño (terminados)
25.Modelo de Datos
37.Interfaces diseñadas de subsistemas (terminadas)
22.Especificación de bibliotecas comunes

44-Diseñar pruebas (Diseñador de pruebas)

Una vez establecida parte del diseño, se procede a iniciar el diseño de las pruebas para el sistema con el propósito de identificar y describir los Casos de Prueba y los Procedimientos de Prueba para la iteración actual, guiados por los modelos hechos hasta el momento y por una detección de variables que puedan afectar el funcionamiento del sistema en el estado en que lo dejará la iteración actual.

Entradas: 23.Especificación de requerimientos
61.Modelo del análisis
62.Modelo del diseño
58.Modelo de casos de uso
60.Modelo de implementación (de iteración anterior)
59.Modelo de despliegue (esbozado)

Salidas: 3.Casos de pruebas
77.Procedimientos de prueba

45-Estructurar el Modelo de diseño (Administrador de desarrollo)

Al final del flujo de trabajo del diseño se reestructura el Modelo del Diseño completo tomado como un todo, para hacer que el modelo sea más fácil de entender y de trabajar de él, para esto el Administrador de Desarrollo debe mantener la consistencia e integridad del modelo a través de los cambios y las agregaciones.

Entradas: 18.Definición de variables y constantes de configuración de subsistemas
21.Especificación de ambientes y herramientas
59.Modelo de despliegue (esbozado)
84.Realizaciones de Casos de Uso – Diseño
6.Clases del diseño (terminadas)
36.Interfaces diseñadas de clases (terminadas)
97.Subsistemas de diseño (terminados)
37.Interfaces diseñadas de subsistemas (terminadas)
79.Proceso de migración (actualizado)
25.Modelo de Datos
87Reglas específicas de diseño
62.Modelo del diseño (esbozado)

Salidas: 62.Modelo del diseño (estructurado y mejorado)

46-Implementar la Arquitectura (Administrador de desarrollo)

La actividad de Implementar la Arquitectura tiene como objetivo identificar los componentes significativos para la Arquitectura del sistema, tales como ejecutables. Para esto se debe establecer la estructura sobre la cual residirá la implementación del sistema a partir del Modelo del Diseño, adaptar la estructura del modelo para reflejar la organización del equipo o las restricciones del lenguaje de implementación, definir dependencias entre Subsistemas, e identificar los Componentes ejecutables. Como entradas utilizara el Modelo del Diseño y el avance del Modelo de Implementación, y al finalizar esta actividad se obtendrá el Modelo de Implementación esbozado y las Reglas Especificas de Codificación. También se puede incluir ya la parte de la implementación referente al Modelo de Datos. Para iteraciones posteriores a la primera, se usará como entrada el Modelo de Implementación de la iteración anterior.

Entradas: 62.Modelo del diseño
59.Modelo de despliegue (esbozado)
25..Modelo de Datos

Salidas: 86.Reglas específicas de codificación
60.Modelo de implementación (esbozado)

47-Planear la integración en la iteración (Integrador del sistema)

En esta actividad el Integrador del Sistema identifica los Subsistemas de Implementación que se integrarán en la Construcción actual para implementar los Casos de Uso correspondientes a esta iteración, considera el impacto de implementar los requerimientos de estos subsistemas y de los sus componentes sobre la Construcción actual, evalúa si el impacto es aceptable, en caso de que el impacto sea aceptable se planifica implementar los Casos de Uso en la iteración actual, en caso contrario se pospone a iteraciones posteriores.

Entradas: 60.Modelo de implementación (esbozado)
58.Modelo de casos de uso
61.Modelo del análisis
62.Modelo del diseño
72.Plan de desarrollo del proyecto

Salidas: 74.Plan de la integración de la construcción

48-Implementar Clases (Ingeniero de componentes)

El objetivo de esta actividad es implementar una Clase del Diseño en un Componente, para esto se esboza un componente que contenga el código fuente de acuerdo a su ámbito, se genera el código fuente a partir de una Clase del Diseño y sus relaciones, se implementan las

operaciones de la Clase del Diseño como métodos, se implementan sus estados, y se comprueba que el componente proporciona las mismas interfaces que la Clase del Diseño.

Entradas: 74.Plan de la integración de la construcción
86.Reglas específicas de codificación
6.Clases del diseño
36.Interfaces diseñadas de clases
Salidas: 38.Interfaces implementadas de componentes
8.Componente

49-Realizar Pruebas de unidad (Ingeniero de componentes)

El objetivo de esta actividad es probar los componentes implementados como unidades individuales, para esto se tiene dos tipos de pruebas de unidad: las Pruebas de Especificación o Pruebas de Caja Negra (verifican el comportamiento de la unidad observable externamente) y las Pruebas de Estructura (verifica la implementación interna de la unidad), también se pueden usar sobre las unidades otras pruebas, como por ejemplo las Pruebas de Rendimiento que evalúan la utilización de memoria, carga y capacidad.

Entradas: 38.Interfaces implementadas de componentes
8.Componente
Salidas: 38.Interfaces implementadas de componentes (probadas)
8.Componente (probados)

50-Implementar e integrar subsistemas (Ingeniero de componentes)

El propósito de esta actividad es asegurar que el Subsistema cumple su papel en cada Construcción, tal y como se especifica en el Plan de la Integración de la Construcción. Para ello, se integraran los Componentes que forman parte del subsistema que ya hayan sido probados, verificando que no afectan al Subsistema completo.

Entradas: 74.Plan de la integración de la construcción
60.Modelo de implementación (esbozado)
97.Subsistemas de diseño
37.Interfaces diseñadas de subsistemas
86.Reglas específicas de codificación
38.Interfaces implementadas de componentes (probadas)
8.Componente (probados)
Salidas: 98.Subsistemas de implementación (de iteración actual)
39.Interfaces implementadas de subsistemas (de iteración actual)

51-Integrar el sistema (Integrador del sistema)

El objetivo de esta actividad es crear cada Construcción antes de someterla a Pruebas de Integración, para ello el Integrador del Sistema debe recopilar las versiones correctas de los Subsistemas de Implementación y de los Componentes, compilados y enlazados para generar una Construcción.

Entradas: 98.Subsistemas de implementación (de iteración actual)
39.Interfaces implementadas de subsistemas (de iteración actual)
74.Plan de la integración de la construcción
Salidas: 101.Construcción

52-Terminar la estructuración del Modelo de despliegue (Administrador de desarrollo)

A través del flujo de trabajo de Implementación se puede asignar los diversos Subsistemas y elementos del Modelo de Datos a nodos en las configuraciones de redes, habilitando al Administrador de Desarrollo para terminar de estructurar el Modelo de Despliegue, actualizándolo y manteniendo su consistencia.

Entradas: 59.Modelo de despliegue (esbozado)
60.Modelo de implementación (esbozado)
Salidas: 59.Modelo de despliegue (estructurado y mejorado)

53-Estructurar el modelo de implementación (Administrador de desarrollo)

Al final del flujo de trabajo de implementación se reestructura el Modelo del Implementación completo tomado como un todo, para hacer que el modelo sea más fácil de entender y de trabajar de él, para esto el Administrador de Desarrollo debe mantener la consistencia e integridad del modelo a través de los cambios y las agregaciones.

Entradas: 59.Modelo de despliegue (estructurado y mejorado)
60.Modelo de implementación (esbozado)
101.Construcción
86.Reglas específicas de codificación

Salidas: 60.Modelo de implementación (estructurado y mejorado)

54-Implementar los componentes de pruebas (Ingeniero de componentes)

El propósito de esta actividad es implementar (grabar, generar o programar) los Procedimientos de Prueba que fueron definidos en el diseño de las pruebas. La salida es una versión de computadora de los Procedimientos de Prueba (Componentes de Prueba), como por ejemplo scripts de prueba. Las técnicas que se utilizarán para ello serán de grabación (utilizar herramientas para capturar y grabar la interacción con el objeto de prueba), de programación (utilizar un ambiente de programación para programar los pasos necesarios para ejecutar y capturar el resultado de las pruebas), o de generación automatizada de pruebas (utilizar una herramienta de generación de pruebas para generar los scripts de prueba sin intervención del usuario). Además se debe identificar las clases o subsistemas que proveerán de una funcionalidad especifica para el Procedimiento de Prueba, y en caso de ser necesario generar una versión incompleta de dichas clases o subsistemas (stubs)

Entradas: 60.Modelo de implementación
3.Casos de pruebas
77.Procedimientos de prueba
101.Construcción

Salidas: 9.Componente de prueba

55-Planificar pruebas (Diseñador de pruebas)

El propósito de esta actividad es planificar los esfuerzos de aplicar las pruebas en una iteración por medio de describir la estrategia de prueba que se va a llevar, estimar los requerimientos para el esfuerzo de la prueba (por ejemplo, los recursos humanos y sistemas necesarios), y planificar el orden y tiempo en que se deben aplicar las pruebas.

Entradas: 23.Especificación de requerimientos
58.Modelo de casos de uso
61.Modelo del análisis
62.Modelo del diseño
60.Modelo de implementación
59.Modelo de despliegue

Salidas: 75.Plan de pruebas

56-Supervisar la generación o el funcionamiento del ambiente de pruebas (Diseñador de pruebas)

En esta actividad el Diseñador de Pruebas debe supervisar que el ambiente de pruebas (hardware, software y recursos humanos) este listo para iniciar las pruebas. Esta actividad es importante debido a que se deben detectar en ella posibles factores ajenos al sistema que influyan en el resultado de las pruebas, por ejemplo, Componentes de Prueba mal implementados, recursos de cómputo demasiado lentos, etc.

Entradas: 3.Casos de pruebas
77.Procedimientos de prueba
9.Componente de prueba
75.Plan de pruebas

Salidas:

57-Realizar pruebas de integración (Ingeniero de pruebas de integración)

El objetivo de estas pruebas es asegurar que el ensamble de los componentes del sistema colabore como se planeaba, así como verificar que el incremento tiene el comportamiento correcto. Se deben realizar las pruebas de integración relevantes a la construcción realizando los Procedimientos de Prueba manualmente para cada Caso de Prueba o ejecutando los componentes de prueba que automaticen los Procedimientos de Prueba, después se comparan los resultados de las pruebas con los resultados esperados y se investigan las diferencias, finalmente se notifica a los Ingeniero de Componentes responsables del Componente que contiene fallos y a los Diseñadores de Pruebas para evaluar los resultados. El ciclo de aplicar las Pruebas de Integración y corregir posibles componentes o subsistemas defectuosos continúa hasta que se pase exitosamente el total de las Pruebas de Integración dentro de un rango preestablecido.

Entradas: 3.Casos de pruebas
77.Procedimientos de prueba
9.Componente de prueba
101.Construcción

Salidas: 95.Solicitud de cambio (opcional)
34.Informe de defectos

58-Realizar pruebas del sistema (Ingeniero de pruebas de integración)

El propósito de las Pruebas del Sistema es asegurar que el sistema en su totalidad (lo que se ha construido hasta el momento incluido el incremento de funcionalidad de la iteración) funcione como se planeaba. Las Pruebas del Sistema pueden empezar cuando las Pruebas de Integración indican que el sistema satisface los objetivos de calidad de integración fijados en el Plan de Pruebas de la iteración actual. El ciclo de aplicar las Pruebas del Sistema y corregir posibles defectos continúa hasta que se pase exitosamente el total de las Pruebas del Sistema dentro de un rango preestablecido.

Entradas: 49.Manual de usuario
50.Manual técnico
3.Casos de pruebas
77.Procedimientos de prueba
9.Componente de prueba
101.Construcción

Salidas: 95.Solicitud de cambio (opcional)
34.Informe de defectos

59-Evaluar pruebas y estructurar el Modelo de pruebas (Diseñador de pruebas)

Los Diseñadores de Pruebas evalúan los resultados de las pruebas comparando los resultados obtenidos con los objetivos esbozados en el Plan de Prueba. Éstos preparan métricas (por ejemplo, completez y fiabilidad) que les permitan determinar el nivel de calidad del software. Al mismo tiempo se estructura el Modelo de Pruebas completo para mantenerlo actualizado y consistente

Entradas: 3.Casos de pruebas
77.Procedimientos de prueba
9.Componente de prueba
75.Plan de pruebas
34.Informe de defectos

Salidas: 65.Modelo de pruebas
95.Solicitud de cambio (opcional)
28.Evaluación de pruebas (para la iteración actual)

60-Realizar pruebas de aceptación (Cliente)

Es recomendable para proyectos de alto riesgo que se realicen Pruebas de Aceptación en cada iteración, para que el Cliente (o un representante de él) evalúe cada uno de los incrementos funcionales que se realicen a las Construcciones hasta alcanzar el Sistema completo. Las Pruebas de Aceptación pueden proceder una vez que en el interior de la

organización se han realizado las Pruebas del Sistema. Desde el punto de vista del Cliente, las Pruebas de aceptación se realizan evaluando las funcionalidades implementadas en la iteración actual, y por tanto guiado por los Casos de Uso que las describan. En el caso de proyectos con proveedores involucrados, el Administrador de Desarrollo de la organización de desarrollo deberá tomar el papel de cliente para aplicar las pruebas de aceptación a los productos proporcionados por el proveedor

Entradas: 49.Manual de usuario
50.Manual técnico
23.Especificación de requerimientos
101.Construcción

Salidas: 34.Informe de defectos

61-Supervisar pruebas de aceptación (Administrador de desarrollo, Diseñador de pruebas)

Es recomendable para proyectos de alto riesgo que se realicen Pruebas de Aceptación en cada iteración, para que el Cliente (o un representante de él) evalúe cada uno de los incrementos funcionales que se realicen a las Construcciones hasta alcanzar el Sistema completo. De esta manera, a la par de la realización de Pruebas de Aceptación por parte del Cliente, el Administrador de Desarrollo y el Diseñador de Pruebas supervisan que se estén aplicando correctamente las pruebas, evitando así que factores externos al sistema interfieran en los resultados. Para proyectos que cuenten con proveedores de productos o componentes de productos, se debe adaptar esta actividad para que también se apliquen las pruebas de aceptación a los productos proporcionados por el proveedor

Entradas: 65.Modelo de pruebas

Salidas: 95.Solicitud de cambio (opcional)
28.Evaluación de pruebas (de aceptación para la iteración actual)

62-Integrar Línea base (Administrador de la configuración)

El objetivo de esta actividad es asegurar que todos los artefactos desarrollados (y que sean identificados como parte de la Línea Base) sean capturados y archivados en determinados puntos de tiempo (preferentemente cada vez que se complete una iteración, lo cual nos daría lapsos de tiempo no uniformes), para ser tomados como bases para los desarrollos futuros del producto. Para esto, el Administrador de la Configuración debe obtener autorización del Administrador de Desarrollo sobre los productos que se encuentran en su versión final y se deben de incluir en la Línea Base, para posteriormente documentar el contenido de la Línea Base, y establecer los permisos correspondientes en el repositorio para hacer disponible la Línea Base a los roles correspondientes.

Entradas: 58.Modelo de casos de uso
61.Modelo del análisis
62.Modelo del diseño
60.Modelo de implementación
59.Modelo de despliegue
49.Manual de usuario
50.Manual técnico
101.Construcción

Salidas: 92.Repositorio del proyecto (actualizado)
41.Línea base (actualizada)

63-Iniciar o continuar el análisis del Material de soporte y Material de capacitación necesarios para el usuario final (Editor técnico)

Al inicio de la las iteraciones, el Editor Técnico analiza el Modelo del Dominio, el Modelo del Negocio, el Modelo de Casos de Uso en su totalidad y cada uno de los Casos de Uso de la iteración actual para determinar qué material de soporte y capacitación se debe elaborar, esto documentado en la Lista de Material de Soporte y Capacitación. Normalmente esta lista incluye al Manual de Usuario y al Manual Técnico, y puede contener otro material dependiendo de la naturaleza del producto y del Cliente, como por ejemplo diapositivas de presentación del producto, diapositivas y resúmenes para capacitación sobre el producto, etc. Para iteraciones

posteriores a la primera, esta actividad debe considerar agregar nuevo material que no se había detectado para elaborar, con su consecuente actualización de la lista.

Entradas: 64.Modelo del negocio
63.Modelo del dominio
58.Modelo de casos de uso
Salidas: 43.Lista de material de soporte y capacitación

64-Estructurar o reestructurar el Material de soporte y Material de capacitación para el usuario final (Editor técnico)

Una vez que el Editor Técnico determino qué material de soporte y capacitación se debe elaborar, debe establecer su objetivo, definir su contenido en forma de índice o tabla de contenido, y prerrequisitos para abordarlo. El contenido para los manuales puede variar de proyecto en proyecto y de tipo de Cliente a tipo de Cliente. El objetivo principal de esta actividad es estructurar la organización del contenido del material. Para iteraciones posteriores a la primera esta actividad se refiere a reestructurar el contenido del material de soporte y capacitación debido a detección de omisiones o a la actualización debido a algún cambio en el proyecto.

Entradas: 64.Modelo del negocio
63.Modelo del dominio
58.Modelo de casos de uso
43.Lista de material de soporte y capacitación
Salidas: 99.Tablas de contenido de manuales, presentaciones y cursos

65-Desarrollar el Material de soporte Material de capacitación para el usuario final (Editor técnico)

Por medio de esta actividad, el Editor Técnico crea, gradualmente a través de las iteraciones, el material que se utilizará para capacitación y los Manuales de Usuario y Técnico, tomando como base el análisis realizado anteriormente y consultando a los roles correspondientes en caso de alguna duda sobre el funcionamiento o estructura del producto, o incluso sobre el perfil de quienes utilizarán el material.

Entradas: 43.Lista de material de soporte y capacitación
99.Tablas de contenido de manuales, presentaciones y cursos
23.Especificación de requerimientos
82.Prototipo de interfaz de usuario
31.Glosario de términos
58.Modelo de casos de uso
61.Modelo del análisis
62.Modelo del diseño
59.Modelo de despliegue
60.Modelo de implementación
49.Manual de usuario (iteración anterior)
50.Manual técnico (iteración anterior)
51.Material de capacitación(iteración anterior)
Salidas: 49.Manual de usuario (mejorado)
50.Manual técnico (mejorado)
51.Material de capacitación(mejorado)

66-Monitorear avance del proyecto y colectar Métricas de productos y recursos (Administrador de desarrollo)

El Objetivo de esta actividad periódica es capturar el estado actual del desempeño de recursos (bajo su autoridad) y la elaboración de productos. En esta actividad el Administrador de Desarrollo captura métricas primitivas sobre el desempeño de los recursos y sobre la elaboración de los productos, también encuentra inconsistencias entre las actividades de los recursos y la elaboración de los productos con respecto a los requerimientos, y actualiza la Lista de Riesgos de acuerdo a lo encontrado. Todo lo descubierto durante esta actividad lo

reporta en forma puntual en las métricas de recursos y productos y lo describe textualmente en el Informe de Avance Técnico.

Entradas: 72.Plan de desarrollo del proyecto
48.Lista de riesgos
33.Informes de avance técnico (anteriores)
15.Definición de métricas
52.Matriz de traza de requerimientos
71.Plan de desarrollo de proveedor
42.Lista de artefactos que entregará proveedor

Salidas: 32.Inconsistencias con requerimientos
33.Informe de avance técnico
48.Lista de riesgos (actualizada)
55.Métricas de recursos
54.Métricas de productos

67-Revisar actividades, roles y artefactos (Administrador de la calidad)

Periódicamente el Administrador de la Calidad revisa artefactos, roles y actividades para evaluar su conformidad con Estándares, Políticas, Definición de Procedimientos, Definición de Artefactos, Definición de Roles y términos incluidos en el Glosario. Como resultado se pueden obtener estos documentos corregidos y reportes sobre la calidad de los artefactos y el desempeño de las actividades y roles que intervinieron para crearlo (Reportes de Auditoría por Artefacto y Reportes de Auditoría por Actividad y Reportes de Auditoría por Rol, respectivamente).

Entradas: 76.Políticas
24.Estándares
16.Definición de procedimientos
14.Definición de artefactos
17.Definición de roles
72.Plan de desarrollo del proyecto
33.Informe de avance técnico
55.Métricas de recursos
54.Métricas de productos
88.Reporte del contenido del repositorio
35.Informe de la revisión al repositorio
32.Inconsistencias con requerimientos

Salidas: 90.Reportes de auditoría por artefacto
91.Reportes de auditoría por rol
89.Reportes de auditoría por actividad

68-Completar métricas de recursos (Administrador del proyecto)

El objetivo de esta actividad es que el Administrador del Proyecto periódicamente tome métricas de los recursos que no están bajo la autoridad del Administrador de Desarrollo, incluyendo al propio Administrador de Desarrollo, al Administrador de la Calidad y al Administrador de la Configuración, para completar el documento de Métricas de Recursos.

Entradas: 72.Plan de desarrollo del proyecto
55.Métricas de recursos
33.Informes de avance técnico
91.Reportes de auditoría por rol
89.Reportes de auditoría por actividad
32.Inconsistencias con requerimientos
15.Definición de métricas

Salidas: 55.Métricas de recursos

69-Analizar Métricas y administrar excepciones y problemas (Administrador del proyecto)

En esta actividad periódica el Administrador del Proyecto calcula las métricas derivadas restantes referentes al progreso del proyecto y analiza los reportes de auditoría generados por

el Administrador de la Calidad, para posteriormente obtener indicadores de progreso. Después compara los indicadores contra el estado esperado del proyecto definido en el Plan de Desarrollo, evaluando si todas las tareas planeadas están completas, si todos los artefactos han sido elaborados conforme a lo planeado, si el esfuerzo para completar las tareas es coherente con el plan, si el desempeño de los recursos es el esperado, etc. También revisará los indicadores de riesgo para cada riesgo de la Lista de Riesgos, para decidir si alguna estrategia de contención o contingencia de riesgos debe ser activada en este momento. Todo lo anterior es documentado en la Interpretación de Métricas.

Entradas: 32.Inconsistencias con requerimientos
69.Plan de contención
70.Plan de contingencia
48.Lista de riesgos (actualizada)
72.Plan de desarrollo del proyecto
54.Métricas de productos
33.Informe de avance técnico
66.Organigrama de ingeniería
55.Métricas de recursos
91.Reportes de auditoría por rol
90.Reportes de auditoría por artefacto
89.Reportes de auditoría por actividad
11.Contrato del proyecto

Salidas: 40.Interpretación de métricas

70-Actualizar Plan de trabajo y Organigrama (Administrador del proyecto)

De acuerdo a lo obtenido en la Interpretación de Métricas, periódicamente el Administrador del Proyecto actualiza o modifica el Plan de Desarrollo, el Plan de Contingencia, el Plan de Contención, y/o el Organigrama de Ingeniería.

Entradas: 72.Plan de desarrollo del proyecto

Salidas: 72.Plan de desarrollo del proyecto (actualizado)
66.Organigrama de ingeniería (actualizado)
69.Plan de contención
70.Plan de contingencia

71-Recibir, registrar y convocar la evaluación de Solicitudes de cambio e informes de problemas. (Administrador de la configuración)

El propósito de esta actividad es asegurar que los cambios sean registrados y realizados de una manera consistente y que los interesados correspondientes estén informados de las solicitudes de cambios, por medio de convocar a una reunión para tal fin. Esta actividad se dispara como respuesta a una Solicitud de Cambios elaborada por cualquier rol y entregada al Administrador de la Configuración o se ejecuta periódicamente y se analizan las propuestas de cambio acumuladas en el periodo.

Entradas: 95.Solicitud de cambio

Salidas:

72-Definir impacto de cambio o de la solución al problema (Administrador de la configuración, Administrador del proyecto, Administrador de la calidad y Administrador de desarrollo)

Esta actividad se realiza en conjunto por el Administrador de la Configuración, el Administrador del Proyecto, el Administrador de la Calidad y el Administrador de Desarrollo, con el propósito de describir detalladamente el impacto del cambio o solución en la funcionalidad del sistema, en el diseño, en el código, en la documentación técnica y en la definición de las pruebas (Análisis Técnico), así como su costo total (en tiempo, dinero y esfuerzo) y si la propuesta de cambio es acorde a las políticas y estándares del proyecto y del proceso (Análisis de Impacto). Al final de esta actividad, se convocara una junta con el Administrador de la Configuración, el Administrador del Proyecto, el Administrador de la Calidad, el Administrador de Desarrollo, el Cliente y un Usuario, para definir si el cambio no es aplicado o es aplicado en la iteración o

versión actual o en posteriores. En caso de autorización, se procederá a asignar el trabajo correspondiente y darle seguimiento. En caso de rechazo se propondrá una alternativa de solución.

Entradas: 95.Solicitud de cambio
Salidas: 1.Análisis del impacto del cambio

73-Controlar y dar seguimiento a los cambios o soluciones autorizadas (Administrador de la configuración)

El Administrador de la Configuración debe dar seguimiento a las actividades involucradas con la autorización y aplicación de un cambio o solución de problema, para que sean realizadas de acuerdo a lo concluido en la junta que la autorizo, incluye actividades como seguimiento de actividades y cambios realizados a productos, así como la verificación de las correcciones.

Entradas:
Salidas:

74-Integrar artefactos del Repositorio (Administrador de la configuración)

El objetivo de esta actividad es asegurar que los artefactos desarrollados sean capturados y archivados en puntos de tiempo periódicos y preestablecidos, incluyendo aquellos que forman la Línea Base controlados de manera más estricta y que servirán para continuar el desarrollo técnico. Por medio de los procedimientos definidos para esta actividad, el Administrador de la Configuración colectará los artefactos, y actualizará el Repositorio del Proyecto en su contenido. Finalmente genera un Reporte del Contenido del Repositorio y los promoverá entre los involucrados con el proyecto dentro de la organización de desarrollo.

Entradas: 47.Lista de productos y elementos de la configuración
19.Descripción de niveles de promoción
46.Lista de productos y elementos adicionales de la configuración
20.Descripción del repositorio
78.Procedimientos especiales de la configuración
Salidas: 88.Reporte del contenido del repositorio
92.Repositorio del proyecto (actualizado)

75-Revisar la integridad del Repositorio (Administrador de la configuración)

Por medio de esta actividad el Administrador de la Configuración periódicamente comprueba que la estructura y el contenido del Repositorio sean los mostrados en el reporte de su contenido y verifica el estado en que se encuentran los artefactos para actualizar los reportes e informar de esto a los interesados. En esta actividad debe colectar e informar cualquier incongruencia, error o descubrimiento encontrado.

Entradas:
Salidas: 35.Informe de la revisión al repositorio

76-Convocar junta interna de revisión e informe general de avance. (Administrador del proyecto)

Periódicamente el Administrador del Proyecto convoca a los administradores a una junta de informe interna a la organización de desarrollo, donde se da a conocer los avances del proyecto, junto con los problemas encontrados, así como las acciones tomadas para su solución o contención, y las adecuaciones al Plan de Desarrollo y otros artefactos. Todo lo discutido y acordado debe ser documentado y archivado en las minutas, para ser retomadas en las posteriores juntas internas.

Entradas: 92.Repositorio del proyecto (actualizado)
41.Línea base (actualizada)
Salidas: 102.Minuta

77-Colectar métricas del proyecto (Administrador del proyecto y Administrador de desarrollo)

Al final de cada iteración el Administrador del Proyecto y el Administrador de Desarrollo analizan las métricas de productos, las métricas de recursos y los reportes de auditoría

obtenidos hasta el momento para calcular métricas del proyecto, para esto se debe caracterizar el proyecto para limitar los rangos de las métricas. Todo esto es registrado en el documento Métricas del Proyecto.

Entradas: 76.Políticas
24.Estándares
16.Definición de procedimientos
72.Plan de desarrollo del proyecto
33.Informes de avance técnico
89.Reportes de auditoría por actividad
15.Definición de métricas

Salidas: 57.Métricas del proyecto

78-Convocar junta externa de revisión e informe general de avance (Administrador del proyecto)

Una vez que el Administrador del Proyecto cuenta con las métricas de recursos, las métricas de productos, las métricas del proyecto, los reportes de auditoría de la iteración actual y además de haber analizado todo esto en una junta interna con los participantes del proyecto, al final de cada iteración convoca a una junta externa con el Cliente para revisar avances y desviaciones con el Plan de Desarrollo. El resultado de esta revisión es documentado en una minuta.

Entradas: 92.Repositorio del proyecto (actualizado)
41.Línea base (actualizada)
57.Métricas del proyecto

Salidas: 102.Minuta

79-Informar del avance de proyecto al Administrador principal (Administrador del proyecto, Administrador principal)

Al final de cada iteración, y después de la junta externa de revisión, el Administrador del Proyecto informa del estado del proyecto al Administrador Principal.

Entradas:

Salidas:

80-Producir Unidades de despliegue (Administrador de la configuración)

Al final de la última iteración, es decir una vez que se han cubierto todas las funcionalidades del producto, el Administrador de la Configuración crea las unidades de despliegue y el entorno de distribución que permitirán instalar y ejecutar el producto final.

Entradas:

Salidas: 100.Unidades de despliegue

81-Coordinar (realizar) prueba final de aceptación en sitio (Usuario, Cliente, Administrador del proyecto y Administrador de desarrollo)

En esta actividad el Cliente y el Usuario final prueban el producto software del proyecto, desde el funcionamiento de las Unidades de Despliegue, el producto software en sí y la documentación, dentro de las propias instalaciones del Cliente. Mientras el Administrador del Proyecto y el Administrador de Desarrollo supervisan la correcta aplicación de las pruebas detectando posibles omisiones, errores o defectos, lo cual podría generar una Solicitud de Cambio, con el correspondiente flujo de eventos.

Entradas: 100.Unidades de despliegue

Salidas: 95.Solicitud de cambio (opcional)

82-Entrega formal del producto al cliente (Cliente, Administrador Principal, Administrador del Proyecto y Administrador de desarrollo)

Un vez que se ha pasado las pruebas de aceptación con éxito, se realiza la entrega formal del producto al cliente por parte del Administrador del Principal, el Administrador del Proyecto y el Administrador de Desarrollo, incluyendo las unidades de despliegue, los manuales y el Material para capacitación. Como salida se obtiene la Carta de Liberación firmada por el Cliente

Entradas: 100.Unidades de despliegue
Salidas: 7.Comentarios y observaciones del producto final
2.Carta de liberación

83-Proporcionar capacitación a usuarios finales (Editor técnico y Administrador de desarrollo)

En esta actividad el Editor Técnico imparte capacitación sobre las funcionalidades del producto a los usuarios finales, con ayuda o supervisión del Administrador de Desarrollo. En esta actividad a si mismo se pueden generar comentarios y observaciones relevantes sobre el producto final del proyecto, los cuales serán documentados para su posterior análisis.

Entradas: 49.Manual de usuario (terminado)
50.Manual técnico (terminado)
51.Material de capacitación (terminado)
Salidas: 7.Comentarios y observaciones del producto final

84-Monitorear retroalimentación de uso operacional (Administrador del proyecto y Administrador de desarrollo)

El Administrador del Proyecto y el Administrador de Desarrollo mantienen comunicación constante con el Cliente y los usuarios, recibiendo cualquier comentario o sugerencia surgidos durante el uso operacional del proyecto, y documentándolos en los comentarios y Observaciones del Producto Final.

Entradas:
Salidas: 7.Comentarios y observaciones del producto final

85-Preparar propuestas de respuesta a las retroalimentaciones (Administrador del proyecto y Administrador de desarrollo)

De acuerdo a la naturaleza de las observaciones y de los comentarios, el Administrador de Desarrollo y el Administrador del Proyecto toman como base el contrato y la especificación de requerimientos para proponer posibles mejoras detectadas, ya sea incluidas en este mismo contrato o bajo un nuevo contrato.

Entradas: 7.Comentarios y observaciones del producto final
Salidas: 80.Propuesta de respuesta a retroalimentaciones

86-Revisar propuestas de respuesta a las retroalimentaciones (Administrador de la calidad)

En esta actividad se revisa que las Propuestas de Respuesta a las Retroalimentaciones estén en conformidad con Estándares, Políticas, los procedimientos definidos para su creación (Definición de Procedimiento), los lineamientos de los artefactos que incluyen (Definición de Artefactos) y términos incluidos en el Glosario. Como resultado se obtiene el documento posiblemente corregido y un reporte sobre la calidad del artefacto. El objetivo de esta revisión es que se cuente con el documento lo suficientemente correcto de cara a una próxima negociación con el Cliente.

Entradas: 76.Políticas
24.Estándares
16.Definición de procedimientos
14.Definición de artefactos
80.Propuesta de respuesta a retroalimentaciones
Salidas: 90.Reportes de auditoría por artefacto

87-Presentar respuestas a retroalimentaciones (Administrador del proyecto)

En esta actividad, el Administrador del Proyecto le propone al Cliente mejoras del producto, pudiendo obtener una nueva especificación de requerimientos, que generará un nuevo proyecto.

Entradas: 80.Propuesta de respuesta a retroalimentaciones
Salidas: 23 Especificación de requerimientos

88-Promover evaluación post-mortem (Administrador del proyecto)

En esta actividad el Administrador del Proyecto promueve que los participantes en el proyecto, perteneciente a la organización de desarrollo evalúen el proyecto desde sus puntos de vista, con el objetivo de asegurar la evaluación de cada uno de ellos

Entradas:

Salidas:

89-Evaluación Post-mortem del proyecto (Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y administrador de desarrollo)

Todos los roles participantes en el proyecto generan su evaluación de acuerdo a su perspectiva, incluyendo deficiencias detectadas en procesos, actividades y recursos, así como también propuestas de mejora. También el Administrador de Desarrollo genera la evaluación de Evaluación de Proveedor(es).

Entradas:

Salidas: 30.Evaluaciones parciales del proyecto
103. Evaluación de proveedor(es)

90-Colectar métricas del producto terminado, generales de recursos, del proceso y del proyecto. (Administrador del proyecto)

En esta actividad el Administrador del Proyecto en base a las métricas parciales referentes a productos, recursos y del proyecto colectadas durante el desarrollo y a las evaluaciones parciales del proyecto elaboradas por los participantes, genera las métricas del producto terminado, las métricas generales de recursos y proyecto, y las métricas del proceso; todo esto a manera de indicadores que resuman las medidas de cada aspecto de este proyecto y el proceso utilizado en él para ayudar en futuras mejoras de la metodología.

Entradas: 30.Evaluaciones parciales del proyecto
57.Métricas del proyecto
55.Métricas de recursos
54.Métricas de productos
15.Definición de métricas

Salidas: 29.Evaluación general del proyecto
56.Métricas del proceso

91-Cerrar repositorio del proyecto (Administrador de la configuración)

Finalmente el Administrador de la Configuración de Software se encarga de guardar los últimos artefactos (incluidos los de Línea Base) dentro del Repositorio del Proyecto, realiza una última revisión, genera un reporte, y cierra el repositorio.

Entradas: 30.Evaluaciones parciales del proyecto
29.Evaluación general del proyecto
54.Métricas de productos
55.Métricas de recursos
56.Métricas del proceso
57.Métricas del proyecto
102.Minuta
2.Carta de liberación
80.Propuesta de respuesta a retroalimentaciones
7.Comentarios y observaciones del producto final
100.Unidad de despliegue

Salidas: 92.Repositorio del proyecto (cerrado)
41.Línea base (cerrada)

En este capítulo se ha descrito el tercer bloque de construcción de la metodología propuesta, que tiene que ver con las actividades, listándolas, indicando su organización y otorgando una breve descripción de las mismas

8

Propuestas sobre el Flujo de Trabajo

Este capítulo describe el flujo de trabajo en el que se combinarán los tres bloques de construcción básicos de la metodología propuesta, tomando en cuanta las bases teóricas de UML, del Proceso Unificado y de CMMI.

8.1. Introducción

Tomando en consideración los bloques de construcción de la metodología propuesta, descritos hasta ahora, en este capítulo se combinarán dentro de un flujo de trabajo que mantenga las características principales del Proceso Unificado (basado en Casos de Uso, centrado en la arquitectura, e iterativo e incremental), y al mismo tiempo que cumpla con el nivel de madurez 2 de CMMI para Ingeniería de software.

Este flujo de trabajo será representado a través de un diagrama de actividades, sin embargo debido al tamaño y complejidad del mismo, se organizará por medio de paquetes de UML. Antes de iniciar con la descripción del flujo de trabajo, se darán algunos prerrequisitos necesarios para aplicarlo.

8.2. Prerrequisitos

El flujo de trabajo que se propone se basa en algunos aspectos que se consideran ya establecidos antes de ejecutarlo. Estos aspectos abarcan compromisos, actividades y artefactos que son considerados como las bases sobre las que se va a aplicar la metodología propuesta.

- Compromisos
  Para poder aplicar la metodología propuesta, y en general cualquier metodología, es necesario que todos los miembros de la organización de desarrollo de software se comprometan en un cambio de modo de trabajo, incluyendo al grupo de ingeniería y al grupo de administradores (principalmente al Administrador Principal). Los compromisos deben ser adquiridos también por los proveedores con los que se tiene más interacción, e incluso hasta cierto punto al Cliente.

- Recursos
  Muy ligado con los compromisos, están los recursos que se asignen para este cambio y la adopción de la metodología. Es importante que en los niveles directivos de la organización de desarrollo de software se tenga conciencia de los recursos humanos y materiales que se deben facilitar para la adopción de la metodología.

- Capacitación en aspectos técnicos y en Áreas de Procesos de CMMI
  Es importante que antes de aplicar la metodologías e capacite a los recursos humanos sobre la misma, sobre todo en aspectos de UML, Proceso Unificado, CMMI, y la propia metodología en sí, haciendo evidentes los beneficios que se obtendrán en los diversos niveles de la organización.

- Políticas
  Es necesario que dentro de la Organización de Desarrollo de Software se definan las políticas que gobiernen los diversos aspectos del desarrollo de software, para que en base a estas políticas, se definan estándares, roles, procedimientos y artefactos.

- Estándares
  Los dos estándares marcados (estándar de documentación y estándar de programación) deben estar definidos aun antes de roles y procedimientos. Es

importante tener en cuenta que debido a que estos estándares cubren aspectos técnicos y de formato, deben ser revisarlos frecuentemente para que no se vuelvan obsoletos.

- Definiciones de roles
  Para cada uno de los roles que participan en la metodología se deben definir objetivo, funciones, responsabilidades, interacción con otros roles y nivel de autoridad de acuerdo a las políticas y estándares, para que en base a ello se definan los procedimientos y artefactos en los que interviene.

- Definiciones de artefactos
  Cada uno de los artefactos que se marcan en la metodología debe ser definido en su documento de definición correspondiente donde se especificará a detalle el contenido que debe incluir, así como su formato.

- Definiciones de procedimientos
  Para cada una de las actividades de la metodología se debe definir su procedimiento correspondiente que guíe en su realización, indicando secuencia de ejecución y responsables.

- Definición de métricas
  En algunas actividades de la metodología se menciona la necesidad de colectar métricas, mismas que deben estar especificadas y definidas antes de aplicar la metodología, donde se deben indicar los aspectos que se deben medir del proceso, del proyecto, de los recursos, etc., así como la manera de medirlos..

Una vez que estos prerrequisitos han sido cubiertos, una organización de desarrollo de software puede proceder a iniciar la aplicación de la metodología en forma gradual, o por medio de un proyecto piloto.

8.3. Organización del Flujo de Trabajo

Los artefactos y actividades mencionados anteriormente deben estar organizados y relacionados de acuerdo a un flujo de trabajo plasmado en un diagrama de actividades de UML. El diagrama de actividades tiene un carril para cada rol de la metodología, donde se encontrarán las actividades que debe realizar cada rol. Cada una de estas actividades se representa por medio de un rectángulo de esquinas redondeadas que tiene un nombre y número asignados, además de documentos de entrada y de salida, colocados encima y debajo de la actividad respectivamente.

Sin embargo, como se mencionó anteriormente este diagrama de actividades puede resultar complejo debido a los 16 roles, 103 artefactos y 91 actividades de la metodología, por lo que se organiza en etapas representadas por paquetes de UML.

Iniciaremos dividiendo el diagrama de actividades en tres etapas principales:

- Etapa de Preparación de Proyecto: Esta etapa establece lo necesario para la ejecución del proyecto. Inicia con la recepción de la Solicitud del Proyecto, y finaliza con las negociaciones de firma del Contrato.

- Etapa Iterativa: Esta etapa procede solo si se llego a un acuerdo en la etapa anterior, y se plasmó esto en un Contrato, en caso contrario se salta hasta el cierre del proyecto. En esta etapa se construye el sistema software a través de interacciones incrementales, heredadas directamente del Proceso Unificado.

- Etapa de Cierre de Proyecto: Se puede llegar a esta etapa desde la etapa iterativa o directamente de la etapa de preparación, según se haya firmado o no el contrato, respectivamente. En caso de llegar a ella desde la etapa iterativa, inicia entregando el producto final al Cliente y termina documentando la experiencia durante el proyecto. En caso de llegar a ella directamente desde la Etapa de Preparación, solo se procede a documentar la experiencia y motivos de no llegar a acuerdo alguno con el Cliente.

La Figura 8-1 muestra en forma de diagrama la secuencia de estas etapas.

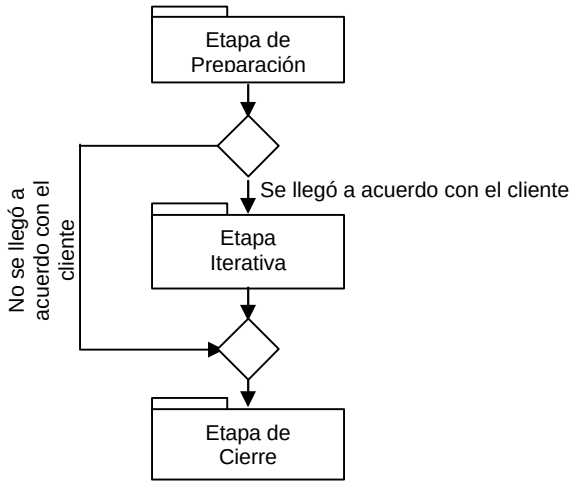


Figura 8-1: Etapas principales del Flujo de Trabajo de la metodología propuesta

A continuación se describirá cada una de las etapas principales, detallando en un diagrama de actividades el contenido de sus sub-etapas.

8.3.1. Etapa de Preparación

En esta etapa se solicita el proyecto y se establece lo necesario para iniciarlo. Termina con la negociación entre el Cliente y la organización de desarrollo de software para la firma del Contrato que formaliza los acuerdos entre ambas partes e inicia el proyecto pasando a la etapa iterativa. En caso de no llegar a cuerdo con el Cliente, se procede con la etapa de cierre. Dentro de esta etapa se encuentran las actividades de la 1 a la 28.

Esta etapa se compone de 6 sub-etapas que se ejecutan de manera secuencial, a saber:

- *Solicitud del proyecto:* En esta sub-etapa se encuentran 5 actividades que tienen como objetivo principal recibir del cliente la Solicitud del Proyecto y la Especificación de Requerimientos, una vez recibidas son revisadas por el Administrador principal y en base a esto se selecciona al equipo de administradores. El diagrama de actividades que detalla esta etapa no requiere de mayores explicaciones, pero cabe hacer notar el paralelismo de sus dos últimas actividades que indica que estas actividades se pueden realizar al mismo tiempo.

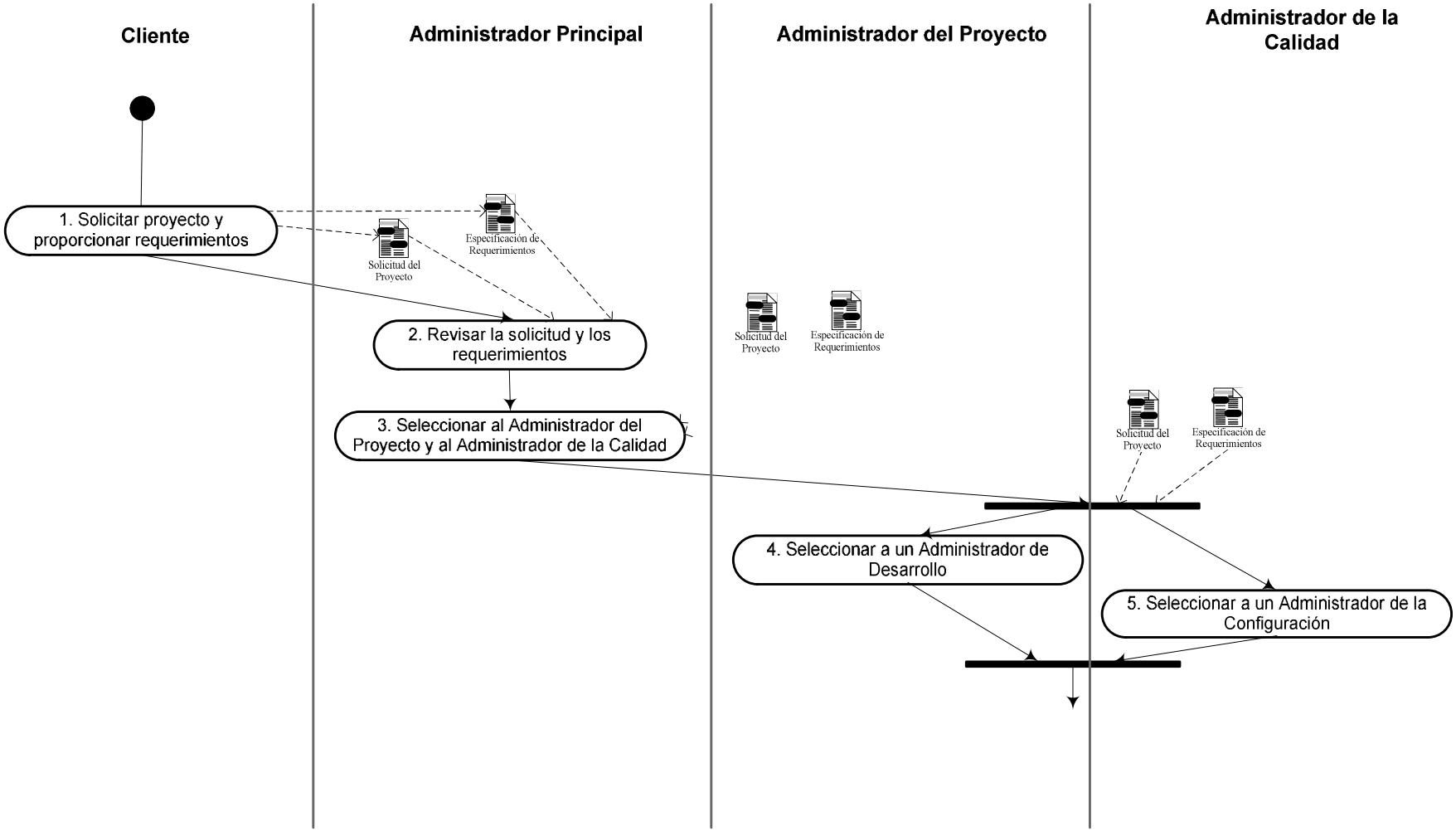


Figura 8-2: Sub-etapa de "Solicitud del proyecto" de la Etapa de Preparación

- *Estimación del proyecto:* Sub-etapa formada por 8 actividades que tienen como objetivo principal la generación de estimaciones del proyecto, por medio de análisis y adecuación de la Especificación de Requerimientos y de la creación del Modelo del Dominio y/o Modelo del Negocio, así como de un preliminar del Modelo de Casos de Uso. En esta sub-etapa también se forma el Grupo de Ingeniería. Su diagrama de actividades inicia con una actividad que es realizada por seis roles de manera conjunta para analizar y ajustar los requerimientos y la Solicitud del Proyecto. También se realizan por más de un rol las actividades que crean los Modelos del Dominio y del Negocio, y la identificación de Actores.

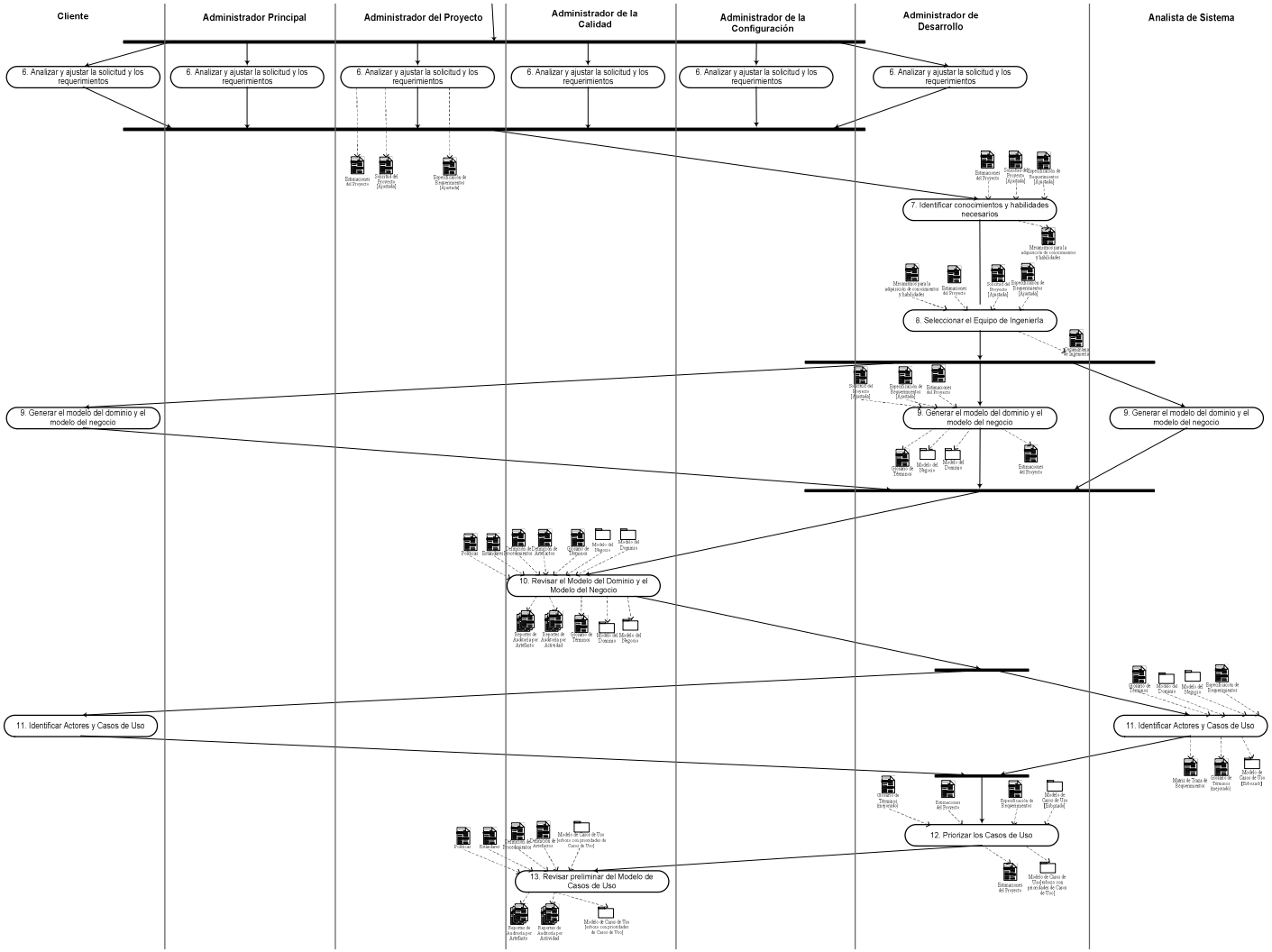


Figura 8-3: Sub-etapa de "Estimación del Proyecto" de la Etapa de Preparación

- *Establecimiento del repositorio*: A través de 3 actividades esta sub-etapa tiene como objetivo establecer el Repositorio de la Configuración de Software del proyecto. Su diagrama de actividades es por demás sencillo y no requiere explicación alguna.

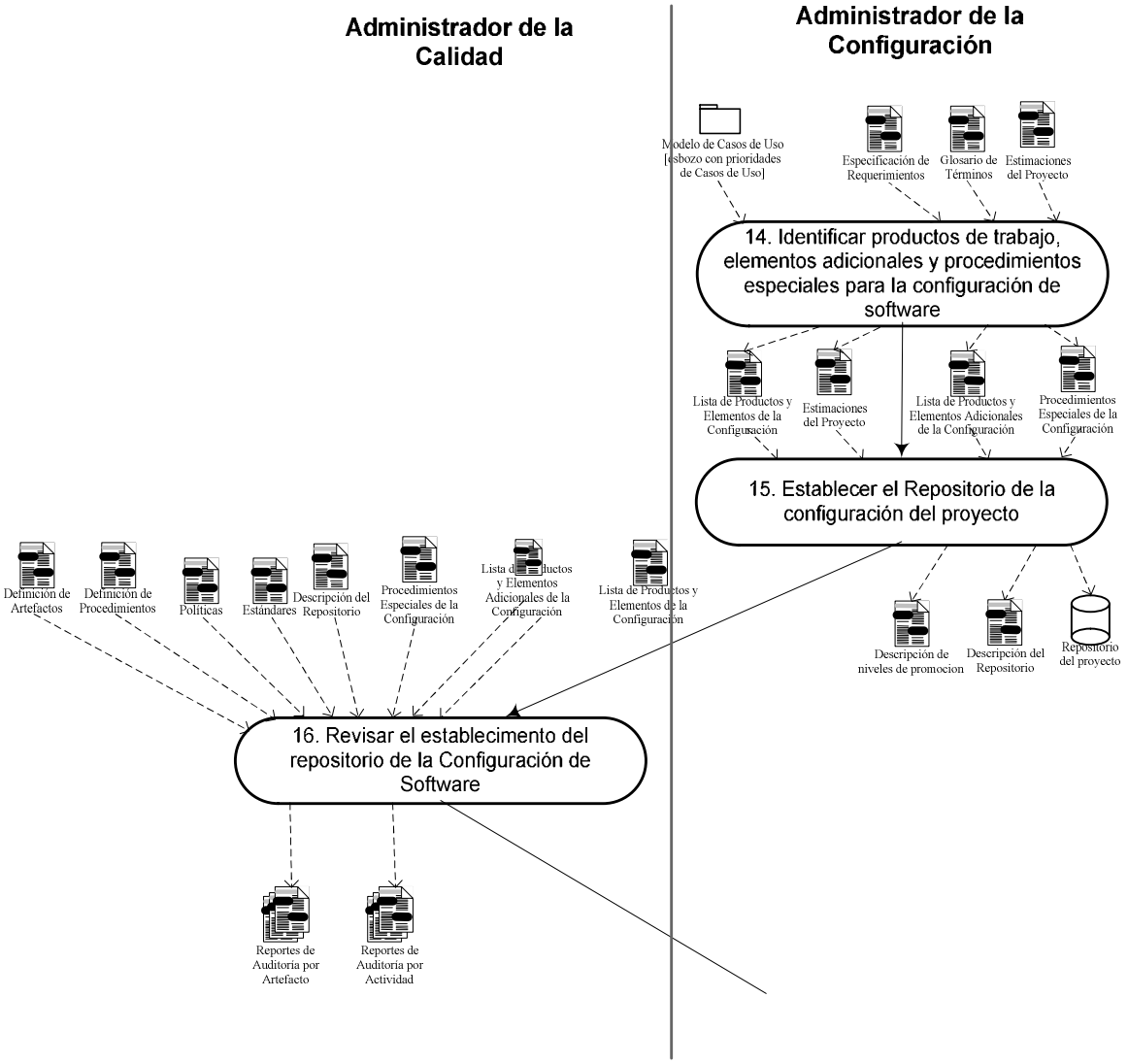


Figura 8-4: Sub-etapa de "Establecimiento del repositorio" de la Etapa de Preparación

- *Inicio de administración de proveedores:* En esta sub-etapa se determina si es necesario para el proyecto la intervención de Proveedores, en caso de ser así, el objetivo de ésta es seleccionar al o a los Proveedores. Su diagrama de actividades contiene también actividades en paralelo y actividades que se realizan por mas de un rol, pero ahora se incluye una condición (iniciada y finalizada por rombos) que deja opcionales algunas actividades de acuerdo a qué si se requieren o no adquisiciones externas que involucren Proveedores.

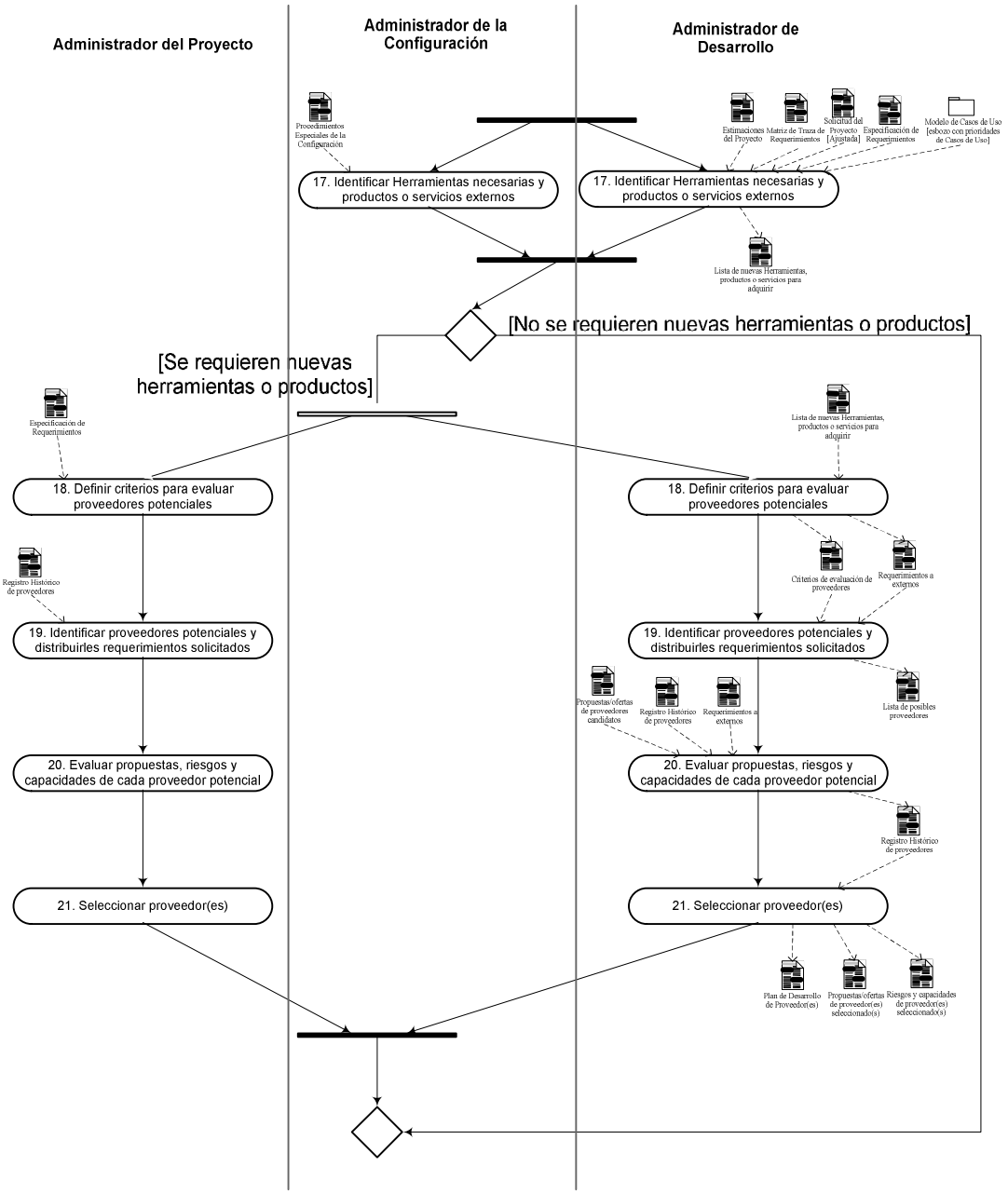


Figura 8-5: Sub-etapa de "inicio de la administración de proveedores" de la Etapa de Preparación

- *Elaboración del plan de desarrollo y Estimación de riesgos:* Con los datos obtenidos hasta el momento, en esta sub-etapa se elabora el Plan de Desarrollo del Proyecto y la Evaluación de Riesgo. Otros objetivos son, la obtención de la Cotización del Proyecto, e integrar en el repositorio todos los artefactos que se han elaborado hasta el momento. Su diagrama de actividades es sencillo y solo cabe hacer un comentario sobre la actividad que lo finaliza, la cual es referente a la integración dentro del Repositorio de la Configuración de Software de todos los documentos que se han generado hasta el momento, antes de iniciar la negociación con Cliente y Proveedores. Hasta esta etapa aun no se ha firmado contrato alguno con el Cliente, lo cual aparenta que se ha realizado trabajo innecesario, pero precisamente el trabajo realizado hasta el momento es el que puede brindar más seguridad en la obtención del contrato. En caso de no obtener el contrato, lo elaborado hasta el momento sirve para brindar experiencia en futuras negociaciones, por lo cual se debe almacenar la información en el repositorio y posteriormente analizarla en la Etapa de Cierre del Proyecto.

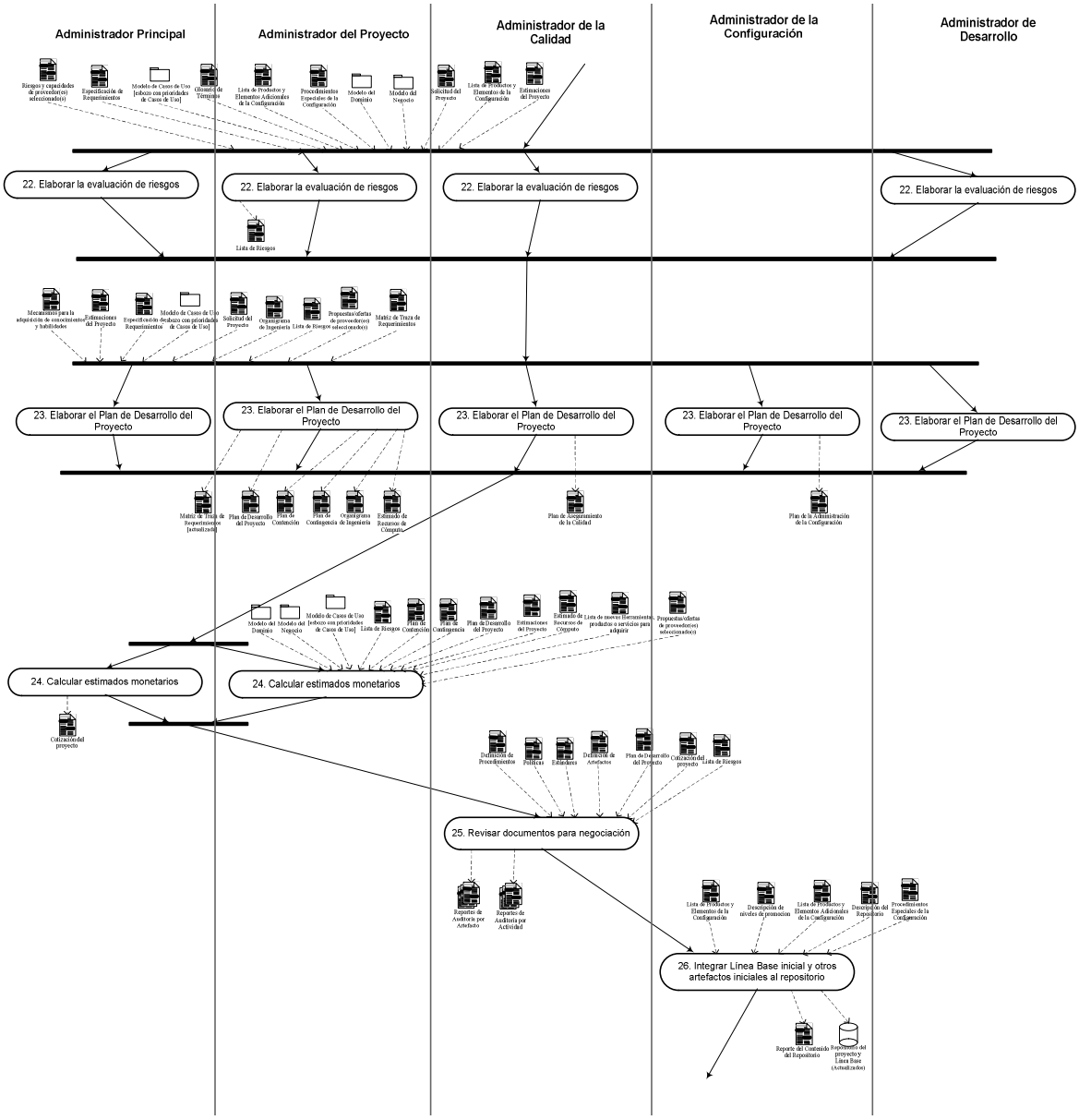


Figura 8-6: Sub-etapa de "Elaboración del plan de desarrollo y estimación de riesgos" de la Etapa de Preparación

- *Negociación y acuerdos con el Cliente y Proveedores:* En la sub-etapa final se realizan negociaciones con el Cliente para formalizar el arranque del proyecto, y en caso de obtener el contrato, se procede a formalizar acuerdo con los Proveedores antes seleccionados. En este diagrama de actividades se realiza una actividad conjunta entre el Cliente y el Administrador Principal para negociar la firma del Contrato, y de acuerdo a su obtención se entra en una condición que permitirá realizar acuerdos con Proveedores si es que se obtuvo el contrato y se requiere de ellos; posteriormente se procede a las actividades de la etapa iterativa. En caso de no obtener el contrato, se envía el flujo directamente a la Etapa de Cierre de Proyecto.

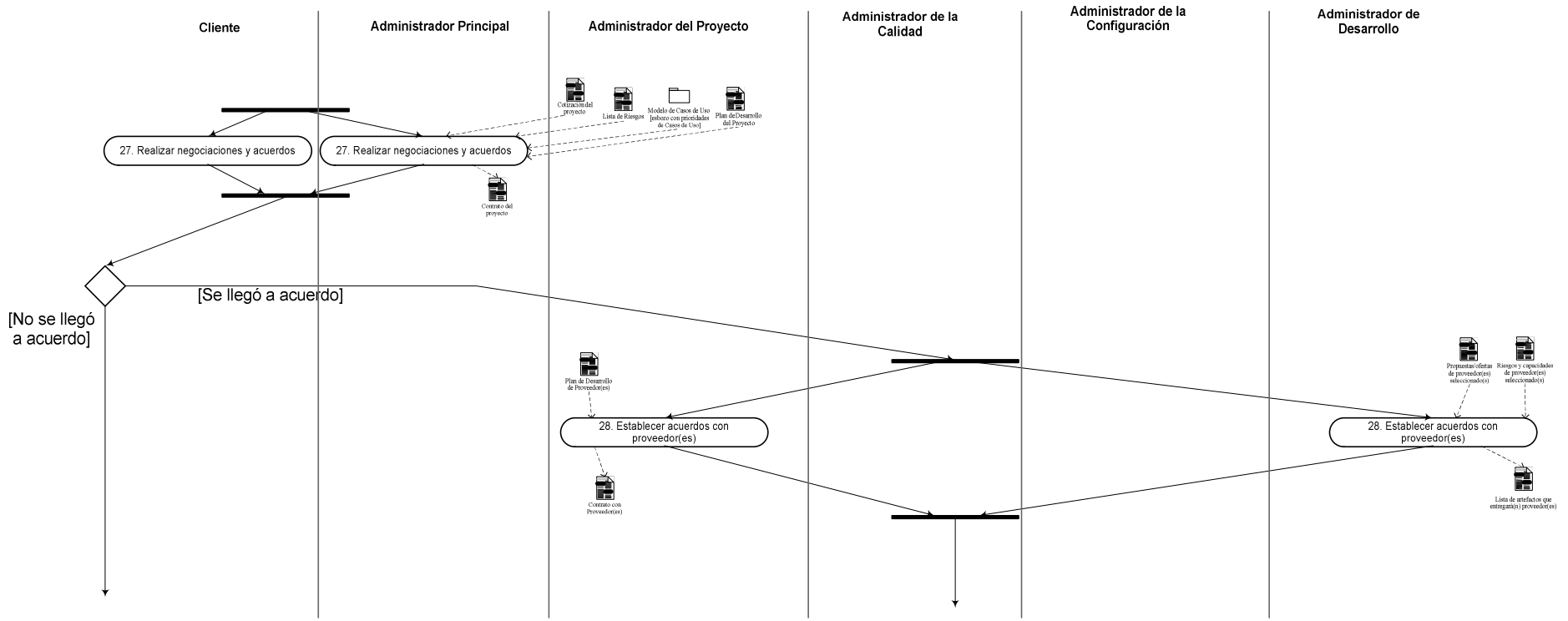


Figura 8-7: Sub-etapa de "Negociación y acuerdos con el Cliente y proveedores" de la Etapa de Preparación

Como se menciono líneas arriba las sub-etapas de la Etapa de Preparación se ejecutan en orden secuencial, y cada una de ellas contendrá cierto número de actividades. En el diagrama siguiente se muestra el orden de ejecución.

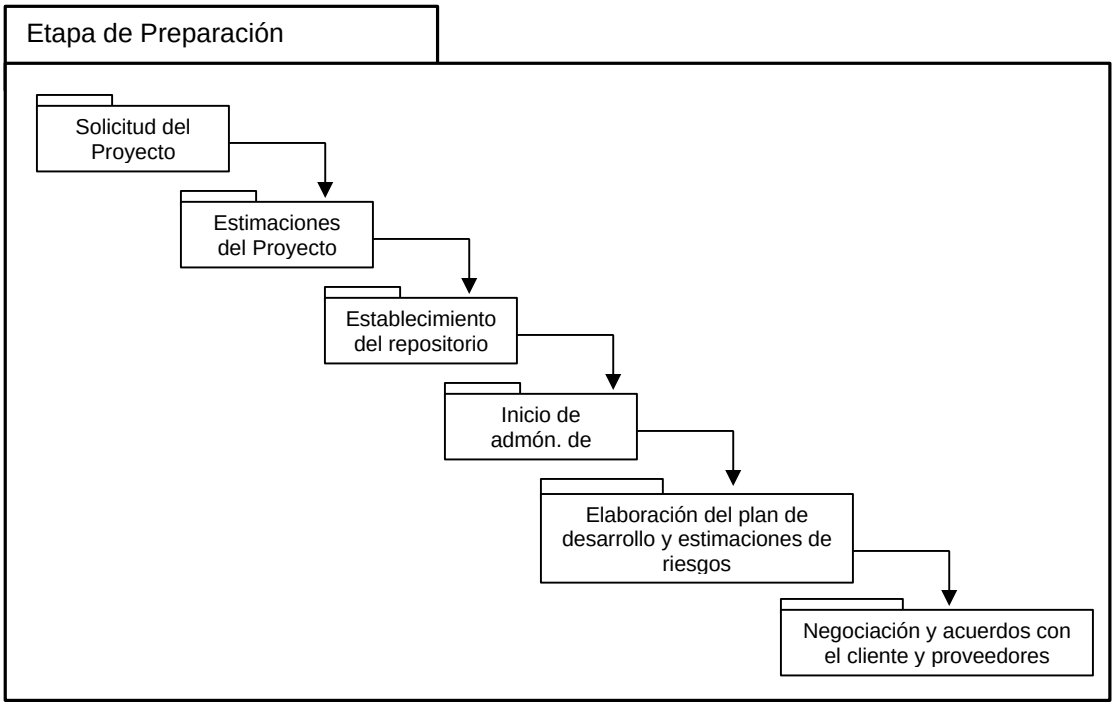


Figura 8-8: Orden de ejecución de sub-etapas de la Etapa de Preparación

8.3.2. Etapa Iterativa

Una vez que se ha llegado a un acuerdo con el Cliente y el Proveedor la realización del proyecto, se inicia el cuerpo principal del desarrollo, contenido en la Etapa Iterativa. Esta etapa se compone de 5 sub-etapas secuénciales y 2 etapas paralelas a estas, todas ellas dentro de iteraciones que construyen cada uno de los requerimientos funcionales del sistema.

Dentro de esta etapa se contiene las actividades de la 29 a la 79 organizadas en las siguiente sub etapas:

- *Requerimientos:* Esta primer sub-etapa secuencial tiene por objetivo la identificación y pre-análisis de los requerimientos seleccionados del sistema para una iteración específica, modelándolos como Casos de Uso y en lenguaje comprensible al Cliente, obteniendo un Modelo de Casos de Uso del Sistema actualizado hasta la iteración actual. Su diagrama de actividades, al ser el primero de la Etapa Iterativa, muestra el cierre de las iteraciones (la flecha que llega al rombo y que cierra una condición, la cual se verá posteriormente) y continúa con una división del flujo de trabajo hacia las actividades administrativas y las actividades técnicas. Precisamente las actividades que se muestran en este diagrama son parte de las actividades técnicas. Al final, también se muestra otra división del flujo o bifurcación, la flecha de la derecha prosigue con las actividades de construcción del sistema, mientras que la de la derecha nos envía hacia las actividades que crean el Material de Soporte y Capacitación.

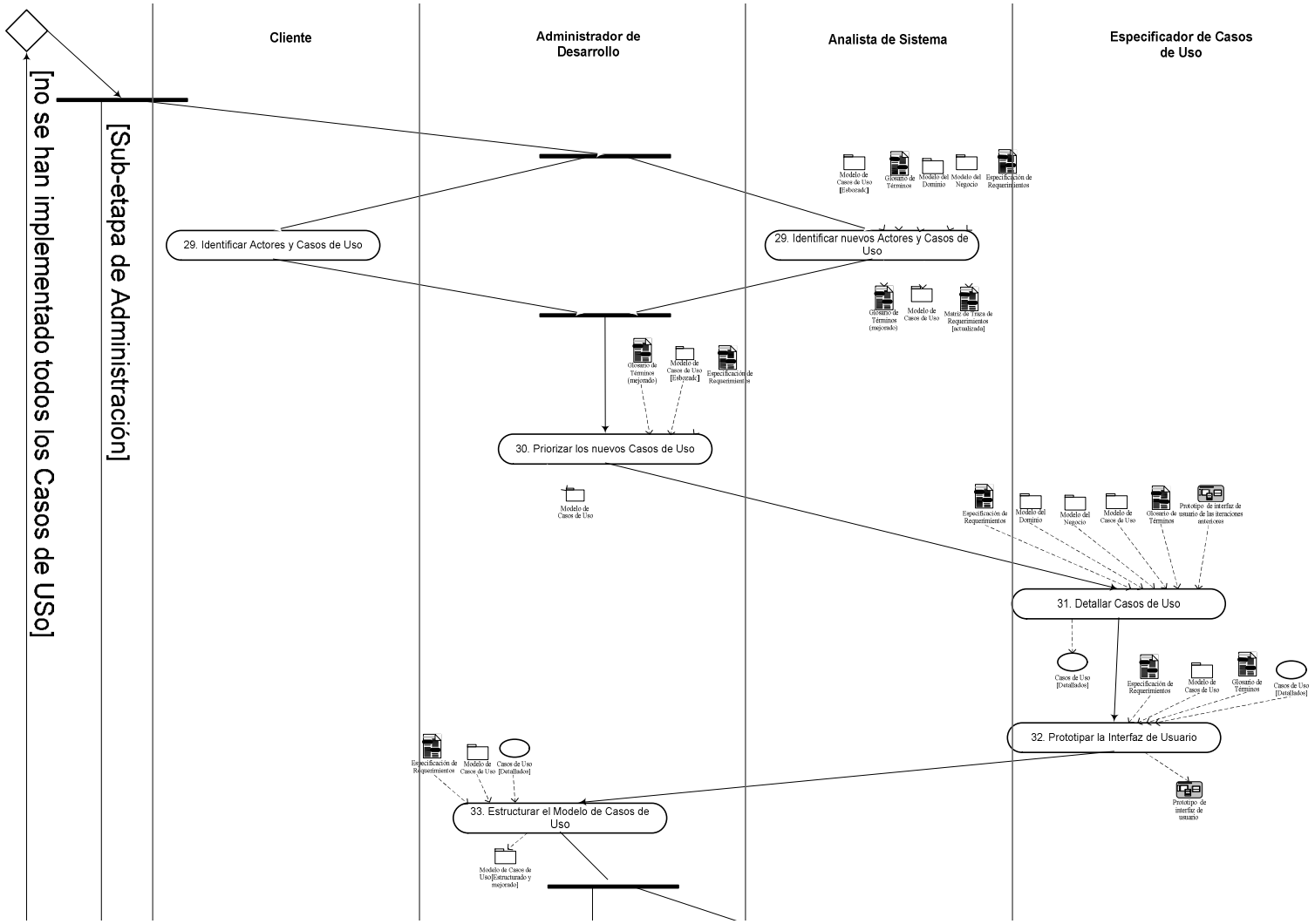
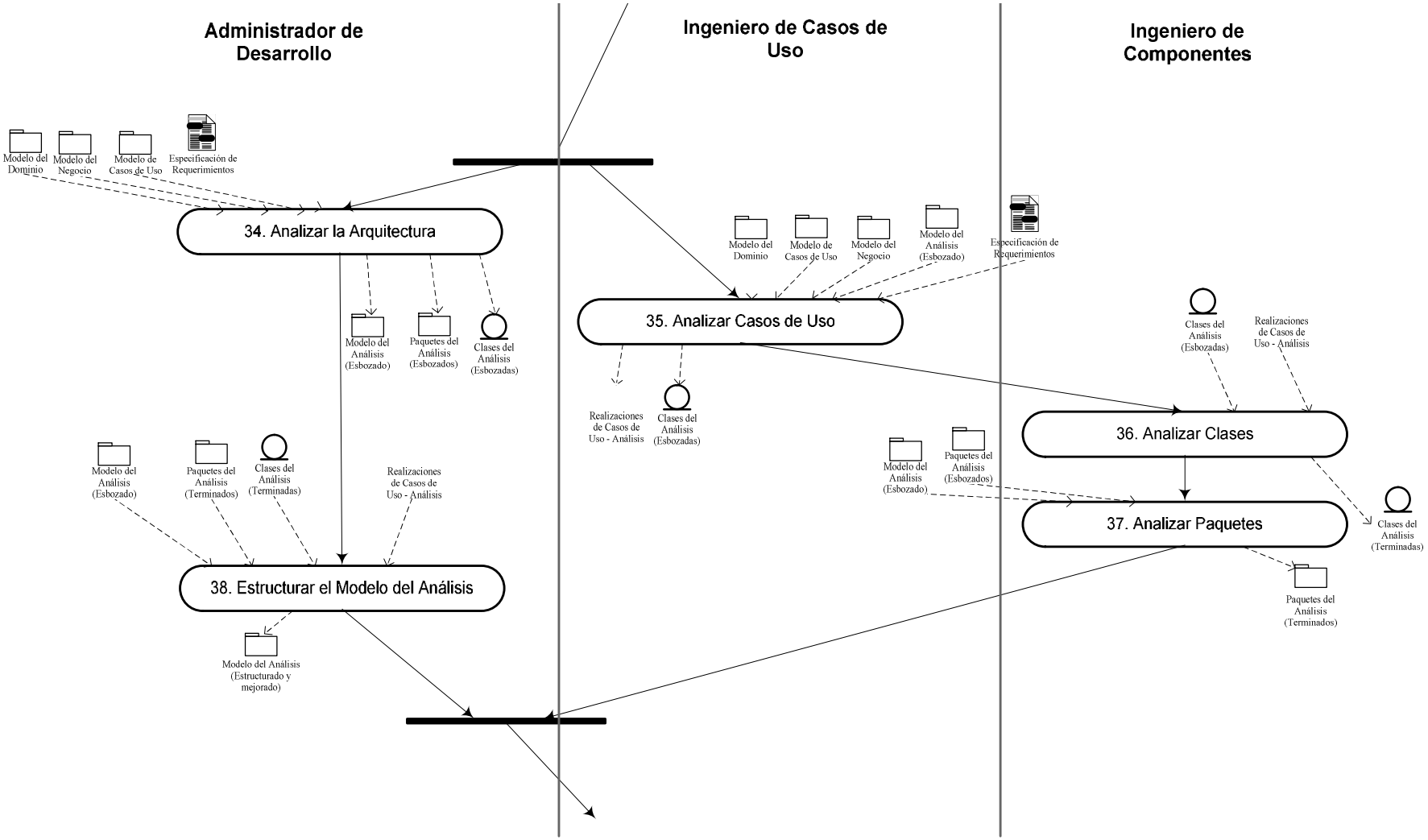


Figura 8-9: Sub-etapa de "Requerimientos" de la Etapa Iterativa

- *Análisis:* Una vez que los Casos de Uso para la iteración actual se han desarrollado, se procede a realizar un análisis mas profundo dentro de esta sub-etapa secuencial con el objetivo de obtener el Modelo de Análisis actualizado. Su diagrama de actividades muestra otra bifurcación en el flujo de trabajo, una referente a la creación y estructuración del Modelo del Análisis por parte del Administrador de Desarrollo, y otra que en paralelo realiza el análisis de Casos de Uso, Clase y Paquetes que se integraran progresivamente al Modelo del Análisis que esta estructurando el Administrador de Desarrollo


Administrador de Desarrollo
Ingeniero de Casos de Uso
Ingeniero de Componentes
Modelo del Dominio
Modelo del Negocio
Modelo de Casos de Uso
Especificación de Requerimientos
34. Analizar la Arquitectura
Modelo del Análisis (Esbozado)
Paquetes del Análisis (Esbozados)
Clases del Análisis (Esbozadas)
35. Analizar Casos de Uso
Realizaciones de Casos de Uso - Análisis
Clases del Análisis (Esbozadas)
36. Analizar Clases
Clases del Análisis (Esbozadas)
Realizaciones de Casos de Uso - Análisis
37. Analizar Paquetes
Paquetes del Análisis (Esbozados)
Clases del Análisis (Terminadas)
Paquetes del Análisis (Terminados)
38. Estructurar el Modelo del Análisis
Paquetes del Análisis (Terminados)
Clases del Análisis (Terminadas)
Modelo del Análisis (Estructurado y mejorado)

- *Diseño:* Dentro de esta sub-etapa secuencial a través del Modelo del Análisis se elabora o actualiza el Modelo del Diseño, realizando un análisis en términos de tecnología de implementación para los Casos de Uso del Análisis. Su diagrama de actividades es muy similar al anterior, y solo cabe anotar que la bifurcación involucra la creación y estructuración del Modelo del Diseño.

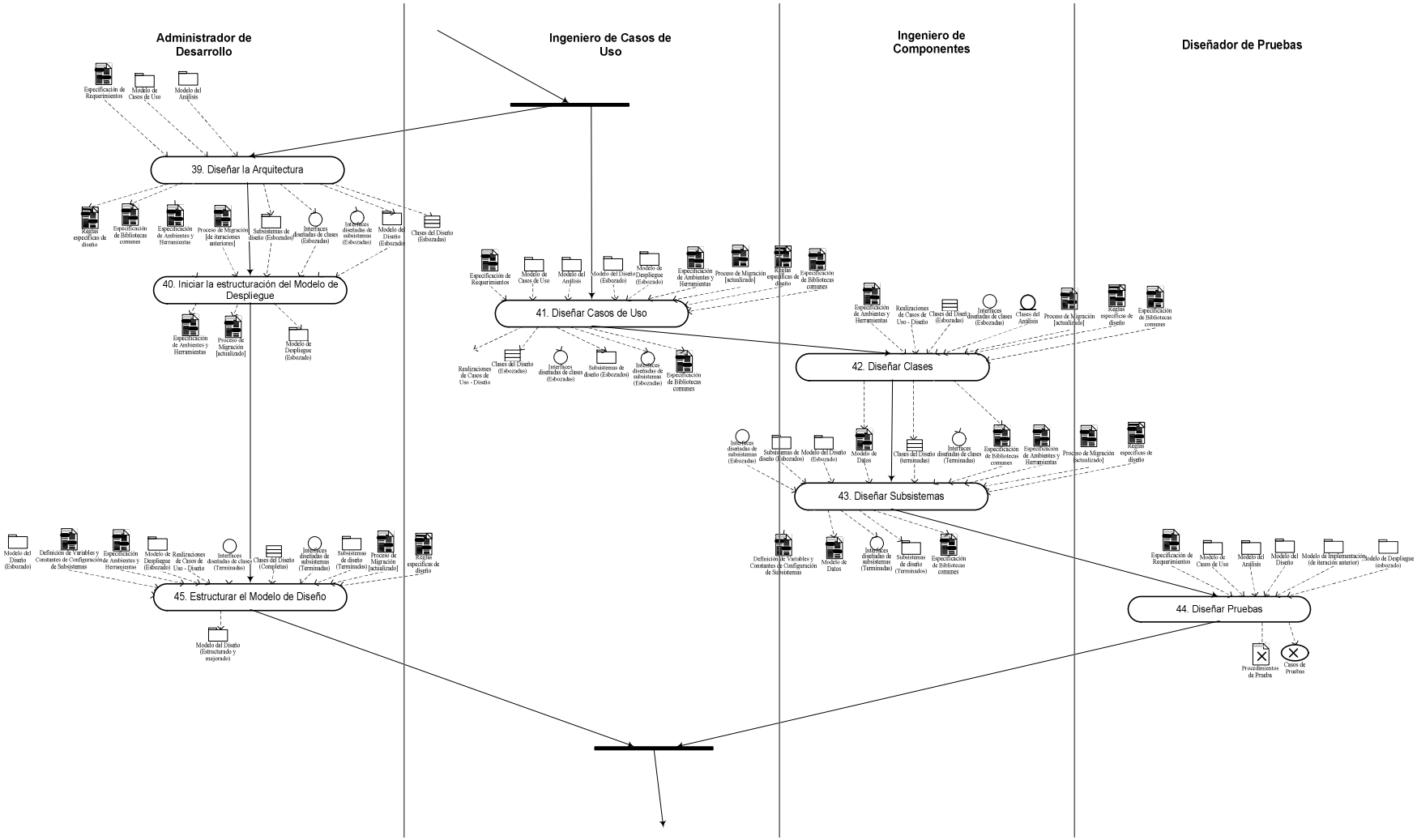


Figura 8-11: Sub-etapa de "Diseño" de la Etapa Iterativa

- *Implementación:* En esta sub-etapa secuencial se realiza la creación del software como tal, que implementará los Casos de Uso de la iteración actual. El objetivo aquí es obtener las Construcciones que se planearon para la iteración actual y los Modelos de Despliegue e Implementación actualizados. Su diagrama de actividades también involucra una bifurcación, pero además tres decisiones anidadas que tienen que ver con la implementación de los Componentes, Subsistemas y Construcciones correspondientes a la iteración actual, las cuales se anidan de acuerdo a la naturaleza de lo implementado. Se debe notar que el trabajo relacionado a las pruebas se ha adelantado ya, incluso desde la Sub-etapa de Diseño, y aquí toca crear los componentes que ayudarán a probar las construcciones realizadas hasta esta iteración.

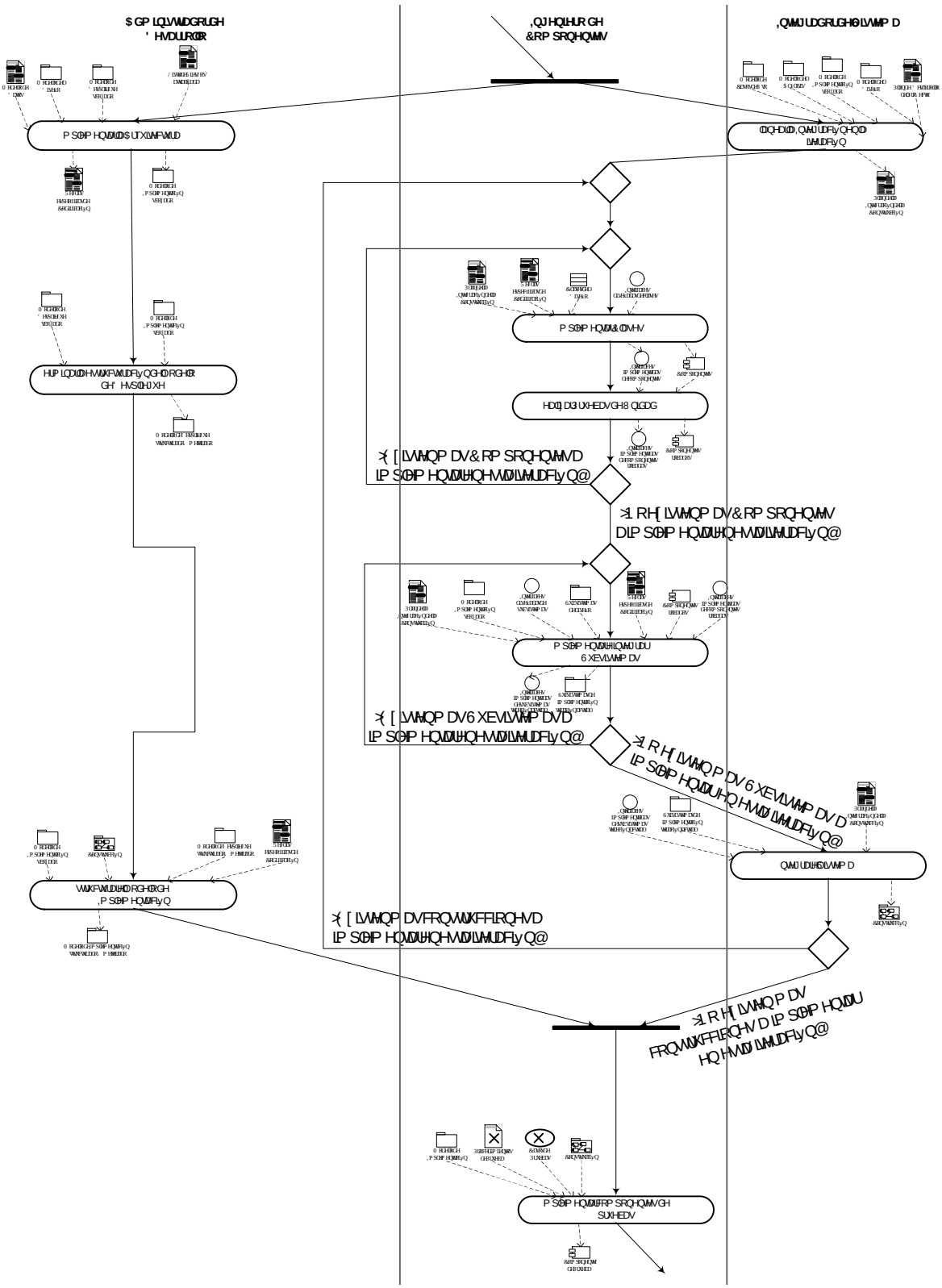


Figura 8-12: Sub-etapa de "Implementación" de la Etapa Iterativa

- *Pruebas:* Esta es la última sub-etapa secuencial se construye el Modelo de Pruebas que ayudará para probar diversos aspectos de las construcciones obtenidas hasta la iteración actual. Finalmente, y una vez que se hayan concluido satisfactoriamente las diversas pruebas, se continua con la siguiente iteración y con sus Casos de Uso correspondientes a través de las sub-etapas secuénciales de requerimientos, análisis, diseño, implementación y pruebas. Su diagrama de actividades no muestra algo extraordinario pues contiene básicamente iteraciones que se realizan de acuerdo a los resultados de las diversas pruebas. Sin embargo, podemos detallar dos aspectos importantes. Primero, se incluye ya las pruebas de aceptación dentro de las iteraciones, esto con el propósito de involucrar al Cliente (o a algún representante de él) en cada construcción que se vaya realizando, obteniéndose una responsabilidad compartida y una detección temprana de posibles desviaciones de los requerimientos. En segundo lugar, al ser la última sub-etapa secuencial de la etapa iterativa, la flecha con la que se finaliza, es la que nos va a volver a unir con las actividades que crean el material de soporte, con las actividades administrativas y posteriormente nos enviará a la siguiente iteración o al cierre del proyecto.

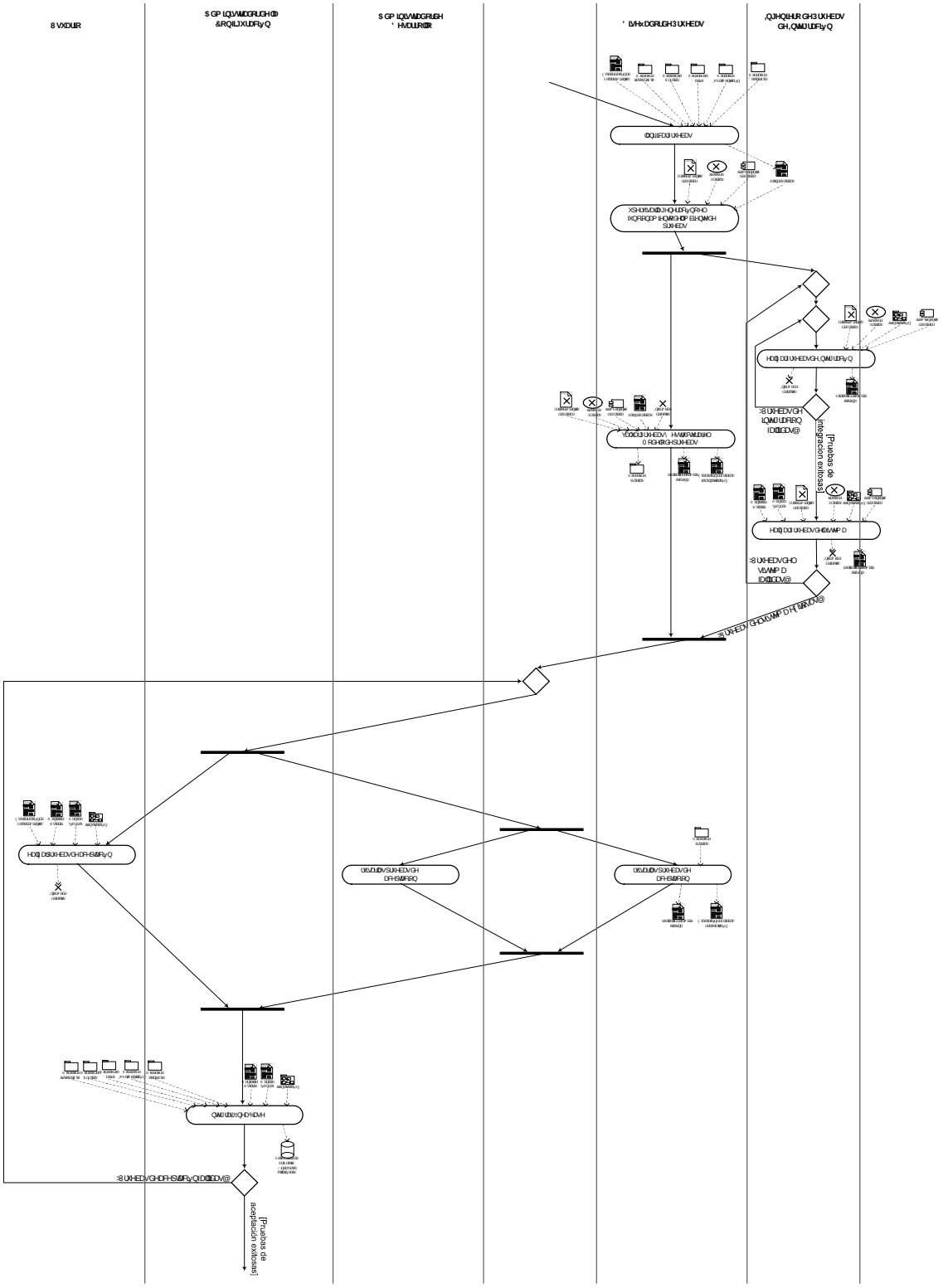


Figura 8-13: Sub--etapa de "Pruebas" de la Etapa Iterativa

- *Administración:* Esta sub-etapa se realiza en forma paralela a la ejecución de las sub-etapas secuenciales y contiene tareas de administración de diversas Áreas de Procesos de CMMI, como monitoreo y control, aseguramiento de la calidad, configuración de software, medición y análisis, etc. En su diagrama de actividades se muestran los rombos de la decisión que controla las iteraciones, condicionadas por la implementación de todos los Casos de Uso. También muestra dos flechas en el extremo derecho, que nos conecta con las sub-etapas secuenciales de esta etapa (requerimientos, análisis diseño, implementación, pruebas).

  En este punto es importante hacer algunas aclaraciones con respecto a los diferentes tipos de métricas y las iteraciones. Las iteraciones siempre son apoyadas por las actividades administrativas, pero mientras que una iteración puede tardar incluso hasta un mes, la cadena de actividades administrativas puede llevarse una semana, es decir, pueden ser más frecuentes que una iteración[1]. Debido a lo anterior, las métricas de productos y recursos deben ser tomadas constantemente, guiadas por la frecuencia de las actividades administrativas, en el caso descrito anteriormente se tomarían cuatro mediciones de métricas de recursos y productos por iteración (una cada semana). Por otro lado, las métricas del proyecto actual deben ser colectadas siempre al final de una iteración, punto en el cual se nos permite medir aspectos más generales que tienen que ver más con el proyecto que con el producto. Finalmente las métricas del proceso se dejan para ser recolectadas al final del proyecto, momento en el cual se pueden tomar las mediciones generales del proceso utilizado en el proyecto.

  También es importante notar que la recolección de métricas de recursos se realiza en dos actividades y por dos roles diferentes. Primero, el Administrador de Desarrollo recolecta métricas relacionadas a los recursos del grupo de ingeniería, posteriormente, el Administrador del Proyecto recolecta métricas relacionadas al grupo de administradores. Esto es debido a que no se recomienda que el Administrador de Desarrollo recolecte métricas sobre él mismo o sobre roles que se encuentran en la misma jerarquía de él, siendo permitido esto para el Administrador del Proyecto.

[1] Si se da el caso de que una iteración sea demasiado rápida, por ejemplo de un día de duración, la frecuencia de las actividades administrativas debe ser ajustada a la de las iteraciones. Se debe evitar que la frecuencia de las iteraciones sea más rápida que la frecuencia de las actividades administrativas, en el peor de los casos deben ser iguales.

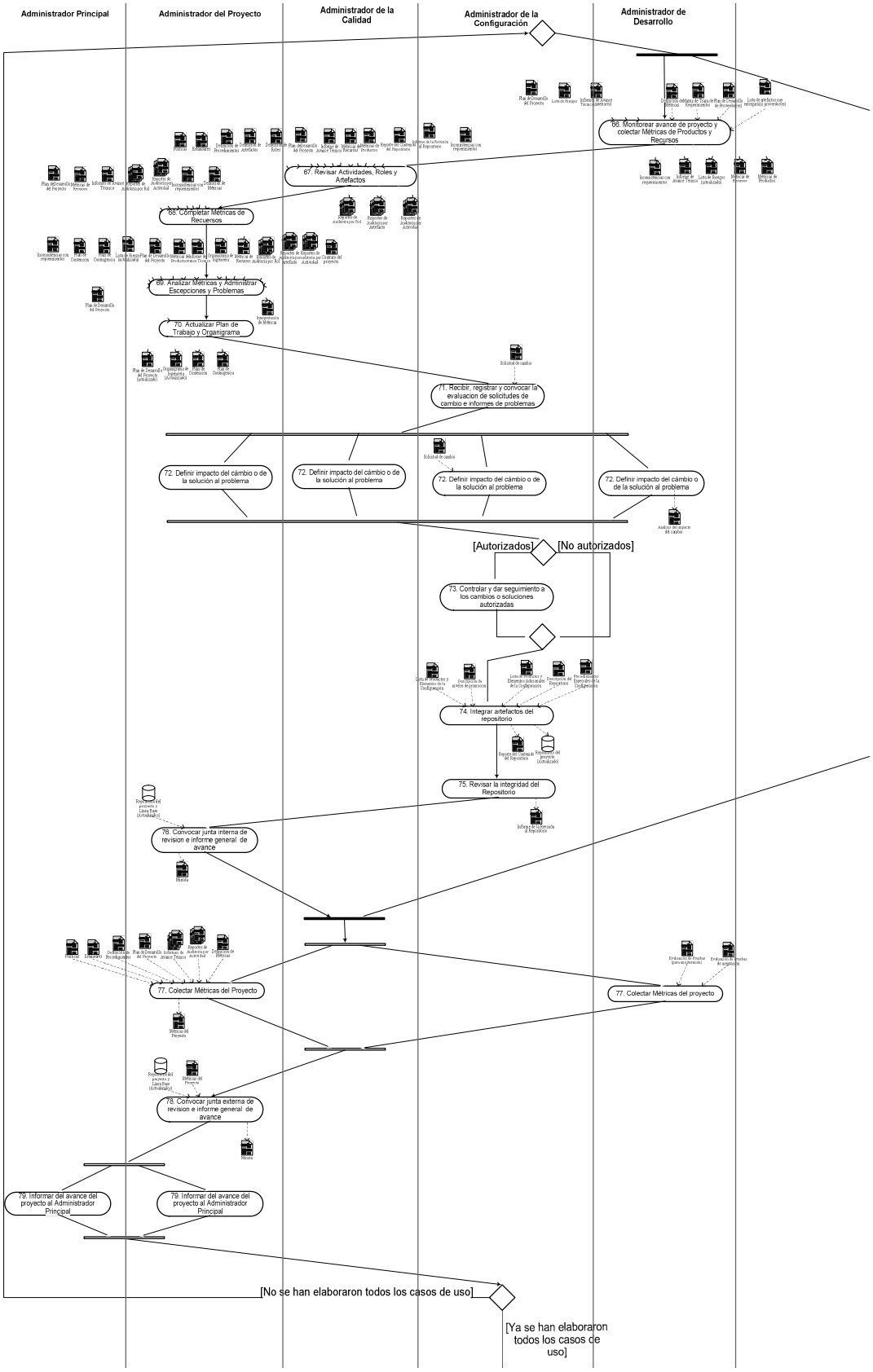


Figura 8-14: Sub-etapa de "Administración" de la Etapa Iterativa

- *Elaboración de material de soporte y capacitación:* Se trata de otra sub-etapa paralela a la realización de las secuénciales y tiene como objetivo crear el Material de Soporte y Capacitación del sistema que se esta construyendo. El diagrama de actividades de esta sub-etapa muestra tres actividades secuénciales, y cabe hacer notar las flechas que se encuentran en el extremo izquierdo, y que nos conectan con el flujo de trabajo de la construcción del producto.

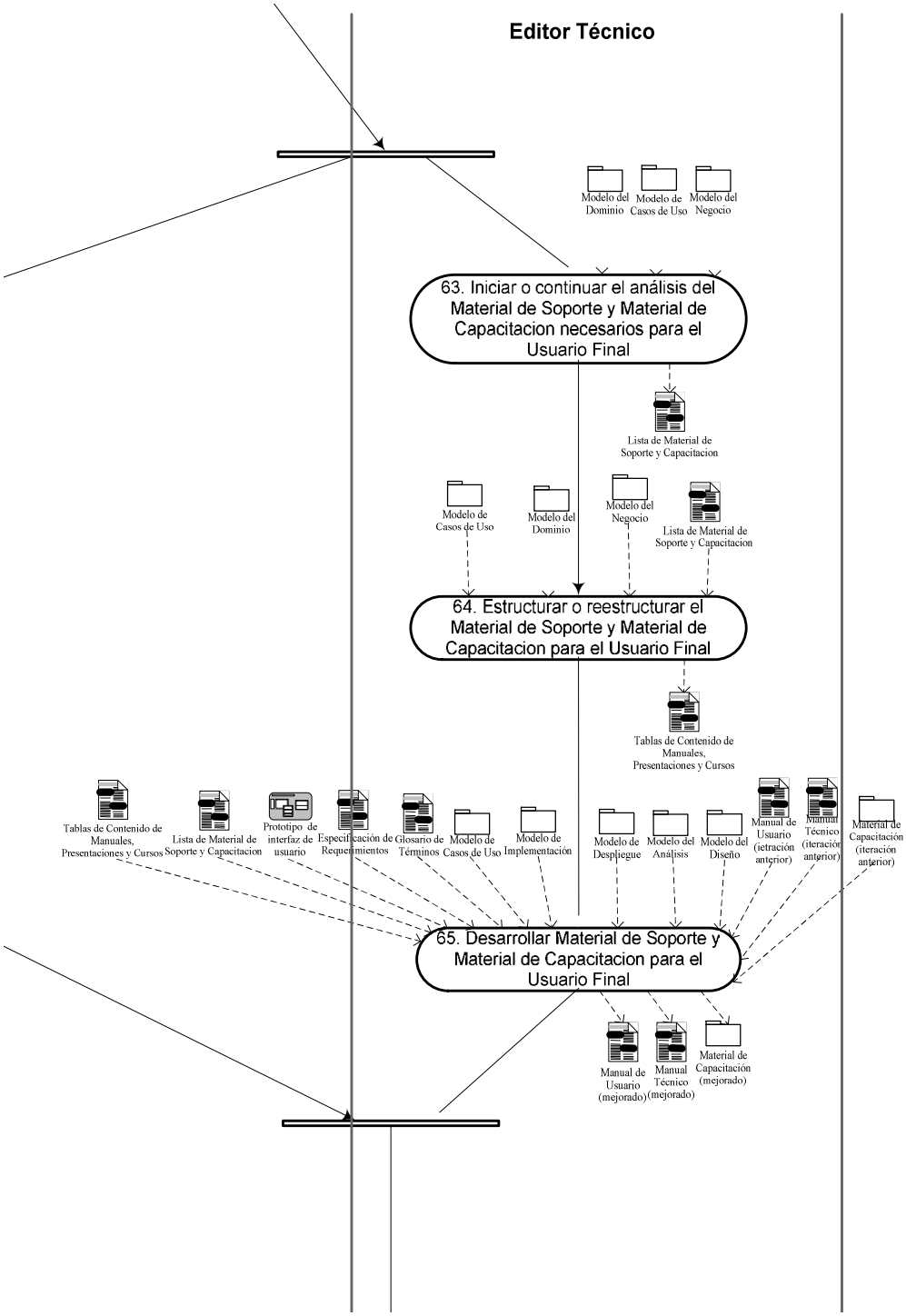


Figura 8-15: Sub-etapa de "Elaboración de material de soporte y capacitación" de la Etapa Iterativa

A continuación se muestra de manera gráfica la relación de ejecución de las sub-etapas de la etapa iterativa:

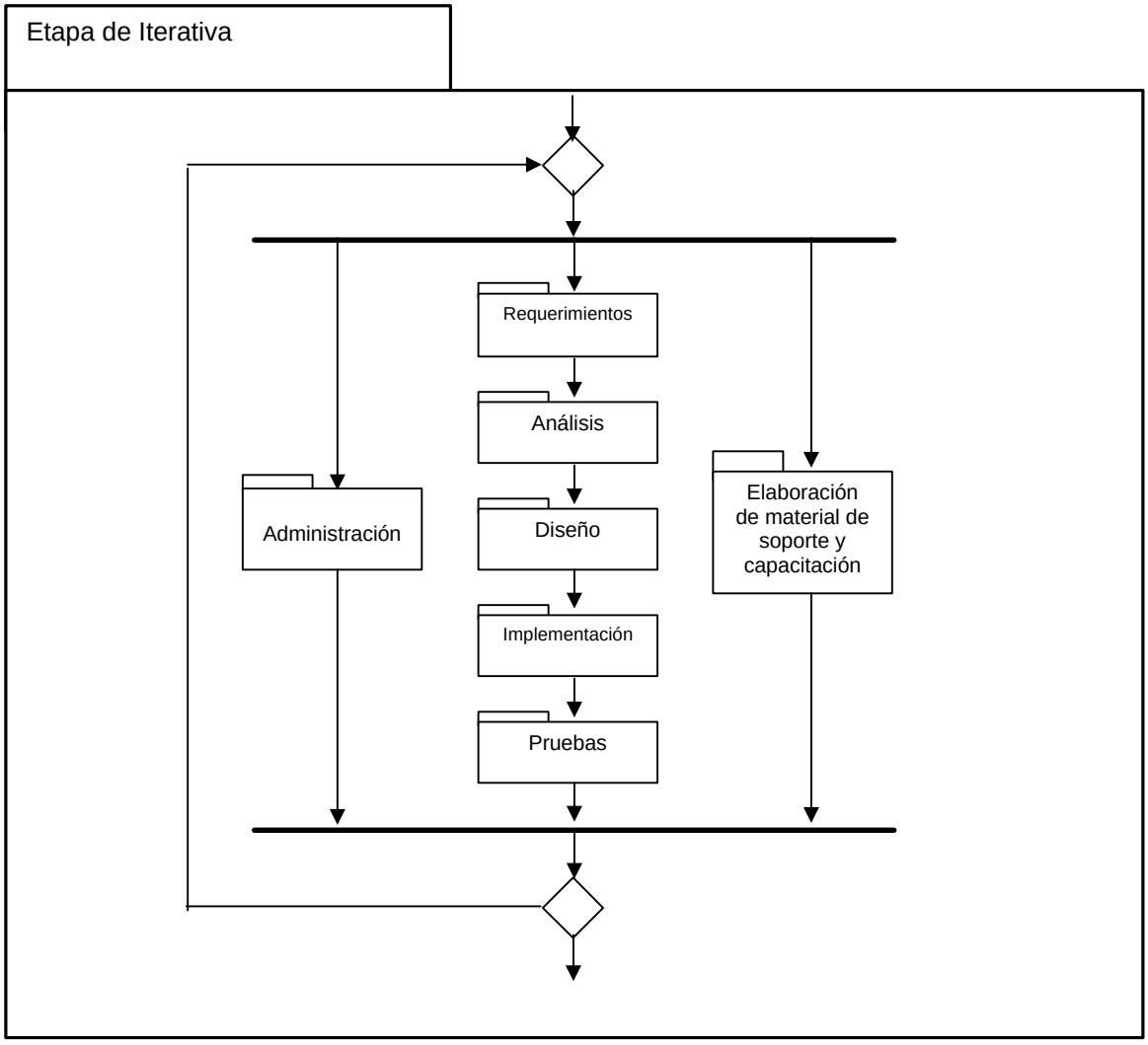


Figura 8-16: Orden de ejecución de sub-etapas de la Etapa Iterativa

8.3.3. Etapa de Cierre

La etapa de cierre consta de 3 sub-etapas secuénciales, sin embargo es necesario aclarara que las primeras dos etapas se ejecutaran siempre y cuando se haya realizado el proyecto. En caso de que no se haya realizado el proyecto por no obtener el contrato, solo se realizará la tercer sub-etapa. La etapa de cierre contiene las actividades de la 80 a la 91, y se organizan a través de las siguientes sub-etapas:

- *Despliegue:* Siempre y cuando se haya realizado el proyecto y construido el sistema, se realiza esta sub-etapa tiene como objetivo distribuir el sistema al cliente a través de las unidades de despliegue y realizar las pruebas finales en sitio para obtener, por parte del cliente, la carta de liberación del producto. Su diagrama de actividades es simple y solo muestra actividades que se realizan por mas de un rol, las cuales no requieren de comentarios adicionales.

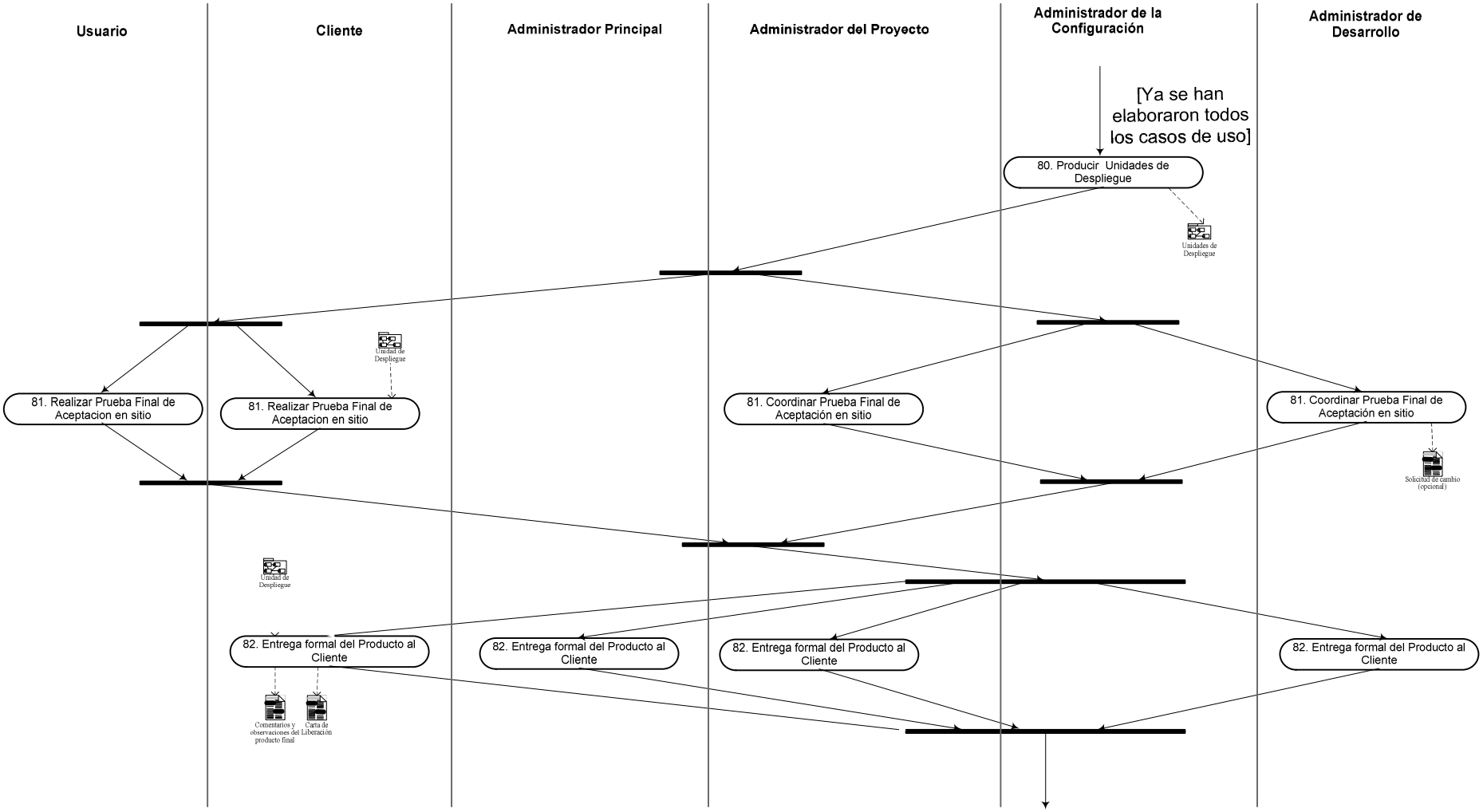


Figura 8-17: Sub-etapa de "Despliegue" de la Etapa de Cierre

- *Capacitación y retroalimentación:* Una vez que se haya realizado la entrega formal del producto, se procede a capacitar a los Usuarios en el nuevo sistema, utilizando para ello el Material de Soporte y Capacitación, y captando la retroalimentación de los Usuarios generada en el uso del producto o durante la capacitación. Al igual que el diagrama de actividades anterior, éste diagrama es sencillo y solo conviene resaltar la flecha que se encuentra en el extremo izquierdo y que proviene de la no obtención del contrato, conectándonos solo hacia las actividades de la siguiente sub-etapa.

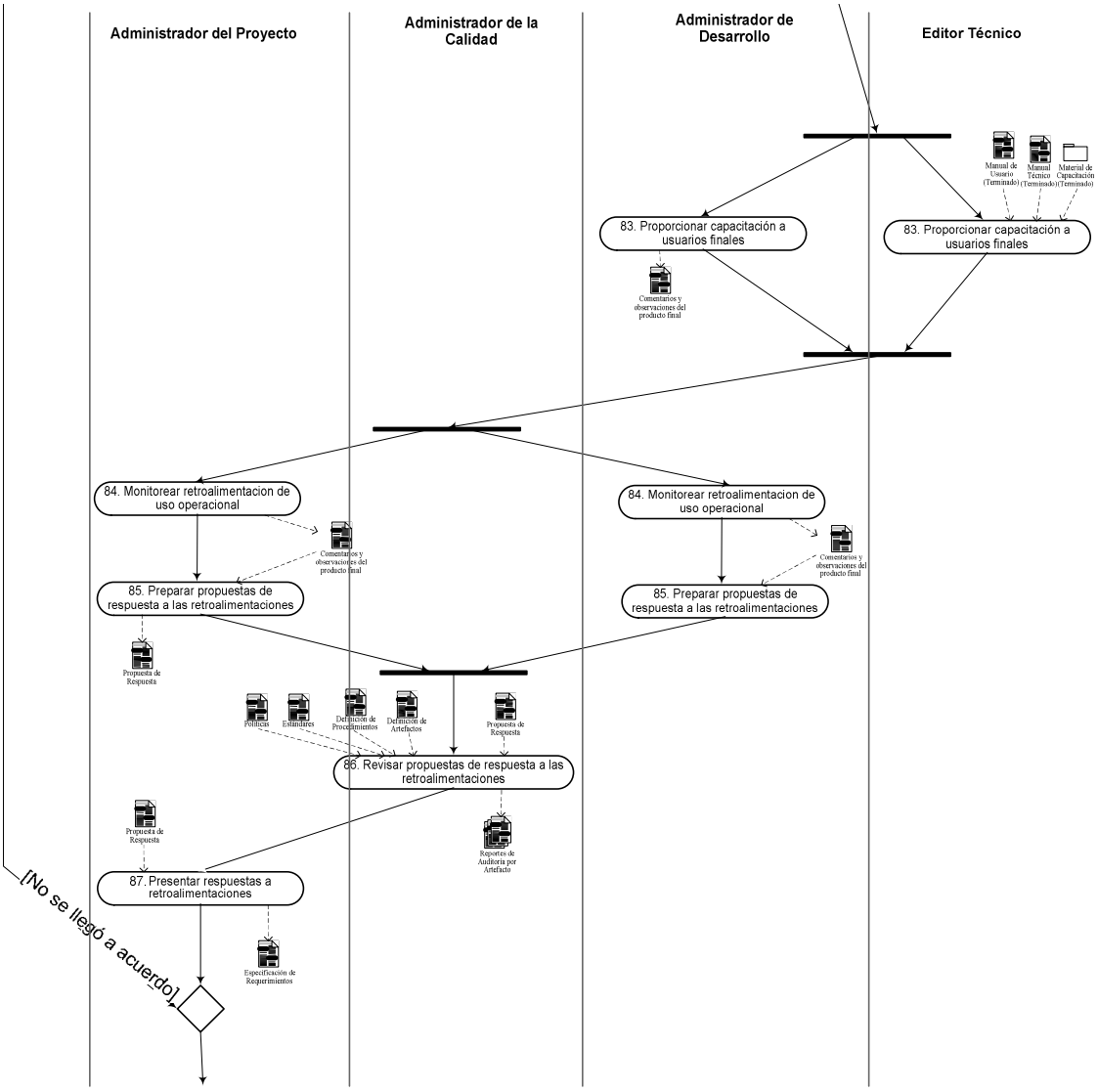


Figura 8-18; Sub-etapa de "Capacitación y retroalimentacion" de la Etapa de Cierre

- *Evaluación post-mortem:* Esta sub-etapa se realiza aunque no se haya concretado el proyecto, en este caso el grupo de administradores documentará la experiencia obtenida a partir de esta situación con el fin de utilizarla en el futuro. Cuando es el caso de haber conseguido el proyecto, el grupo de administradores evaluará todos los aspectos positivos y negativos del desarrollo y los documentarán para utilizar esta experiencia en el futuro. Su diagrama de actividades muestra la realización de la evaluación post-mortem, que se realiza en conjunto por el grupo de administradores. Al final se encuentra la actividad de cierre del repositorio del proyecto con la que se finaliza la metodología.

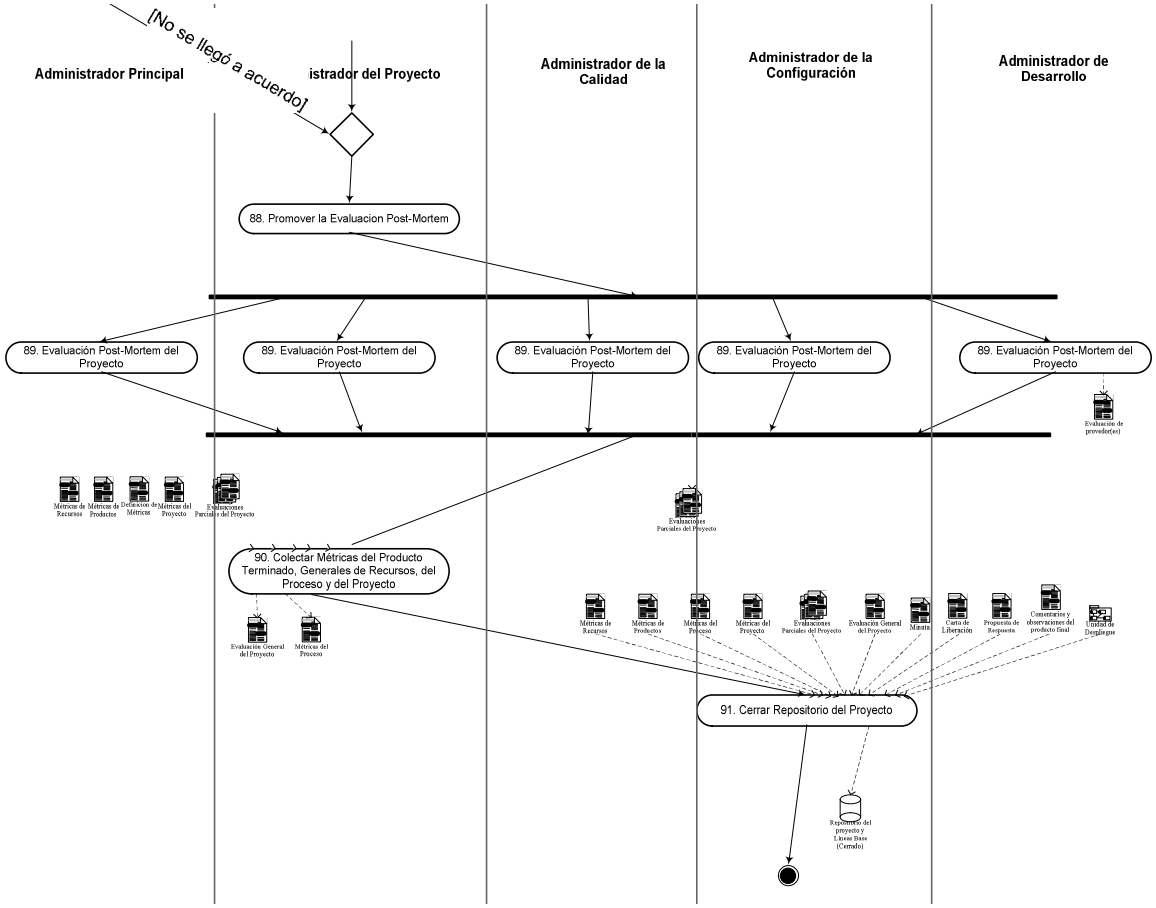


Figura 8-19: Sub-etapa de "Evaluación post-mortem" de la Etapa de Cierre

A continuación se muestra de forma esquemática la realización de las sub-etapas de la etapa de cierre:

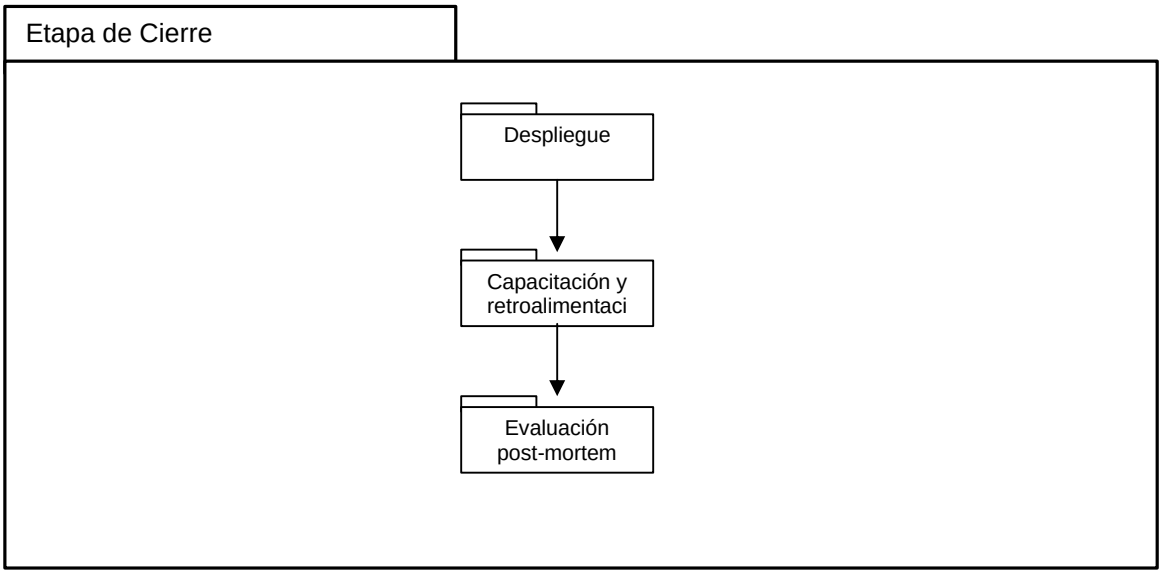


Figura 8-20: Orden de ejecución de sub-etapas de la Etapa de Cierre

---

En este capítulo se describen las etapas y sub-etapas que conforman el flujo de trabajo de la metodología propuesta.

9

Conclusiones y Recomendaciones

Este capítulo se enumera las conclusiones obtenidas al final de este trabajo, así mismo se dan una serie de recomendaciones sobre su aplicación y mejora.

9.1. Conclusiones

En este trabajo se ha expuesto una metodología que auxilie a las organizaciones en el desarrollo de sistemas de software orientado a objetos basándose en Proceso Unificado y CMMI. Sin embargo, y aunque al principio se planteo construir una metodología completa como tal, se encontró que debido a la gran variedad en los tipos de organizaciones de desarrollo de software y a la variedad de proyectos que se puede tener, era conveniente dejar algunos puntos abiertos para que sean complementados de acuerdo a la organización que desee aplicar la metodología, por ejemplo, la definición de artefactos y procedimientos, la definición de métricas, y el dictamen de políticas y estándares. A pesar de esto, se proporcionan descripciones que sirven de guía para completar estos puntos. A partir de lo anterior se hace deseable construir una metodología a la medida para una organización de software específica, que contenga todos los puntos que se han dejado para adaptación, e incluso que automatice por medio de herramientas de software lo que se ha expuesto en papel.

Al elegir CMMI en el nivel 2 de madurez como marco de referencia para la calidad de los procesos de software se ha logrado en la metodología propuesta un acercamiento y una compatibilidad heredada con las normas ISO, específicamente al ISO 15504 (SPICE).

Así mismo, se puede concluir que debido principalmente a las prácticas que marca CMMI para el nivel 2 de madurez, esta propuesta no es recomendable para proyectos pequeños, es decir, proyectos de corta duración, un grupo de trabajo pequeño y/o pocas funcionalidades que implementar; pues la carga de trabajo administrativo seria demasiada en relación a la carga de trabajo técnico para implementar el producto final. A raíz de lo anterior, es tentador construir una metodología rápida que sea adecuada a proyectos pequeños de desarrollo de software, pero que cumpla de la manera más resumida posible los requerimientos básicos de CMMI para su segundo nivel de madurez.

Finalmente, cabe mencionar que la elaboración de este trabajo me ha brindado la oportunidad de afianzar los conocimientos adquiridos durante la licenciatura y ampliar los conocimientos específicos en la ingeniería de software, abriendo puertas para futuros desarrollos en este campo durante mi vida profesional.

9.2. Recomendaciones

Sobre la base del desarrollo presentado en este trabajo se recomienda lo siguiente:

Para todo aquel que desee mejorar este trabajo

- Se recomienda mantenerlo actualizado con las versiones mas reciente de UML, Proceso Unificado y CMMI.
- Migrar a nivel de madurez 3 de CMMI solo hasta que se haya aplicado y mejorado en diversas organizaciones de desarrollo de software.
- Acotar el rango de aplicación de la metodología a cierto tipo de organización de desarrollo y de proyectos.
- Las mejoras sugeridas abarcan definiciones de formatos, artefactos, procedimientos, políticas y estándares
- Se sugiere también la creación de una herramienta software que ayude en la automatización de la metodología y simplificarla para hacerla mas manejable.
- Para enriquecer más el trabajo, es recomendable incluir cocimiento de otras disciplinas, por ejemplo, estadística, ingeniería económica, e incluso psicología.

Para las organizaciones que deseen aplicar esta metodología

- Es necesario que se establezcan primero los requerimientos mencionados en el capítulo 8, de acuerdo a su naturaleza.
- Establecer compromisos respecto a la metodología en todos los niveles de la organización, desde programadores hasta administradores de primer nivel.
- Crear un grupo dentro de la organización dedicado a la mejora continua de la metodología.

ANEXO

Para su nivel de madurez 2, CMMI marca un conjunto de Metas Especificas y Practicas Especificas de cada una de las Áreas de Proceso que incluye. A continuación se validará la metodología propuesta con cada una de las Prácticas Específicas, listando las actividades y artefactos que ayudan a cubrirlas.

AP: Administración de Requerimientos

ME1: Administrar requerimientos

Se refiere a que se deben administrar los requerimientos e identificar las inconsistencias entre el plan del proyecto y los productos de trabajo, lo que se logra mediante las siguientes prácticas:

PE 1.1 Obtener una comprensión de requerimientos

Se refiere a desarrollar junto con el proveedor de requerimientos indicado (en este caso el Cliente es quien proveerá de sus requerimientos) una comprensión de los mismos y sobre el producto final. En la propuesta se logra esta práctica por medio del análisis y ajuste de la Especificación de Requerimientos y de la Solicitud del Proyecto en conjunto con el Cliente.

Actividades: 6-Analizar y ajustar la solicitud y los requerimientos (Cliente, Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y Administrador de desarrollo)

Artefactos: 23.Especificación de requerimientos

PE 1.2 Obtener un acuerdo sobre los requerimientos

Se refiere a obtener acuerdos de los requerimientos entre los involucrados en el proyecto. La Práctica Específica anterior se refería a establecer acuerdos con el proveedor de requerimientos, mientras que ésta práctica se enfoca a establecer un acuerdo entre los involucrados en el proyecto dentro de la organización de desarrollo. Esta práctica se implementa en la metodología propuesta por medio de la construcción y posterior acuerdo en la correctez del Modelo de Casos de Uso.

Actividades: 6-Analizar y ajustar la solicitud y los requerimientos (Cliente, Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y Administrador de desarrollo)
9-Generar el Modelo del dominio y el Modelo del negocio (Cliente, Administrador de desarrollo y Analista del sistema)
11-Identificar actores y casos de uso (Cliente y Analista de sistema)
12-Priorizar los casos de uso (Administrador de desarrollo)
29-Identificar nuevos Actores y Casos de uso (Analista de sistema y Cliente)
30-Priorizar los nuevos Casos de uso (Administrador de desarrollo)
33-Estructurar el Modelo de casos de uso (Administrador de desarrollo)

Artefactos: 23.Especificación de requerimientos
63.Modelo del dominio
64.Modelo del negocio
58.Modelo de casos de uso
31.Glosario de términos

PE 1.3 Administrar cambios a los requerimientos

Se refiere a administrar los cambios en los requerimientos cuando van evolucionando durante el proyecto. En la metodología propuesta esto se consigue al administrar los cambios en las actividades administrativas dentro de la etapa iterativa, así como al manejar nuevos Actores y Casos de Uso (requerimientos) que se encuentren en las diversas iteraciones.

Actividades: 29-Identificar nuevos Actores y Casos de uso (Analista de sistema y Cliente)
30-Priorizar los nuevos Casos de uso (Administrador de desarrollo)
33-Estructurar el Modelo de casos de uso (Administrador de desarrollo)
71-Recibir, registrar y convocar la evaluación de Solicitudes de cambio e informes de problemas. (Administrador de la configuración)
72-Definir impacto de cambio o de la solución al problema (Administrador de la configuración, Administrador del proyecto, Administrador de la calidad y Administrador de desarrollo)
73-Controlar y dar seguimiento a los cambios o soluciones autorizadas (Administrador de la configuración)
Artefactos: 23.Especificación de requerimientos
58.Modelo de casos de uso
31.Glosario de términos
95.Solicitud de cambio
1.Análisis del impacto del cambio

PE 1.4 Mantener una trazabilidad bidireccional de los requerimientos

Se refiere a mantener una trazabilidad bidireccional entre los requerimientos, los planes del proyecto y productos de trabajo. El principal artefacto que ayuda en su implementación dentro de la metodología propuesta es la Matriz de Traza de Requerimientos, sin embrago también se logra a través de la construcción y mantenimiento del Modelo de Casos de Uso, que plasma los requerimientos funcionales del Cliente, y del Plan de Desarrollo del Proyecto que indica como se cubrirán estos requerimientos.

Actividades: 11-Identificar actores y casos de uso (Cliente y Analista de sistema)
29-Identificar nuevos Actores y Casos de uso (Analista de sistema y Cliente)
23-Elaborar el Plan de desarrollo del proyecto (Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y Administrador de desarrollo)
70-Actualizar Plan de trabajo y Organigrama (Administrador del proyecto)
Artefactos: 58.Modelo de casos de uso
52.Matriz de traza de requerimientos
72.Plan de desarrollo del proyecto

PE 1.5 Identificar inconsistencias entre el trabajo del proyecto y los requerimientos

Se refiere a identificar inconsistencias entre los requerimientos y los planes y productos de trabajo, e iniciar las acciones correctivas pertinentes. Esto se consigue a través de las actividades de revisión del Administrador de Desarrollo, del Administrador de la Calidad y del Administrador del Proyecto sobre lo realizado en el proyecto con respecto a los requerimientos.

Actividades: 66-Monitorear avance del proyecto y colectar Métricas de productos y recursos (Administrador de desarrollo)
67-Revisar actividades, roles y artefactos (Administrador de la calidad)
70-Actualizar Plan de trabajo y Organigrama (Administrador del proyecto)
71-Recibir, registrar y convocar la evaluación de Solicitudes de cambio e informes de problemas. (Administrador de la configuración)
Artefactos: 33.Informes de avance técnico
32.Inconsistencias con requerimientos
72.Plan de desarrollo del proyecto
90.Reportes de auditoría por artefacto
91.Reportes de auditoría por rol
89.Reportes de auditoría por actividad
95.Solicitud de cambio

AP: Planeación del Proyecto

ME1: Establecer estimaciones

Se refiere a que se deben establecer y mantener los parámetros estimados de planeación del proyecto, lo que se consigue a través de las siguientes prácticas

PE 1.1 Estimar el alcance del proyecto

Se refiere a establecer y mantener estimaciones de parámetros de planeación del proyecto. Las actividades incluidas dentro de la sub-etapa de estimación del proyecto de la Etapa de Preparación de la metodología propuesta tratan esta práctica.

Actividades: 6-Analizar y ajustar la solicitud y los requerimientos (Cliente, Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y Administrador de desarrollo)

7-Identificar conocimientos y habilidades necesarios (Administrador de desarrollo)

14-Identificar productos de trabajo, elementos adicionales y procedimientos especiales para la configuración de software (Administrador de la configuración)

17-Identificar herramientas necesarias y productos o servicios externos (Administrador de la configuración y Administrador de desarrollo)

22-Elaborar la evaluación de riesgos (Administrador principal, Administrador del proyecto, Administrador de la calidad y Administrador de desarrollo)

Artefactos: 23.Especificación de requerimientos

26.Estimaciones del proyecto

46.Lista de productos y elementos adicionales de la configuración

47.Lista de productos y elementos de la configuración

44.Lista de nuevas herramientas, productos o servicios para adquirir

48.Lista de riesgos

27.Estimado de recursos de cómputo

PE 1.2 Establecer estimaciones del producto de trabajo y de las características de las tareas

Se refiere a establecer y mantener estimados de productos de trabajo y tareas. Las siguientes actividades y artefactos cubren esta práctica dentro de la metodología propuesta.

Actividades: 6-Analizar y ajustar la solicitud y los requerimientos (Cliente, Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y Administrador de desarrollo)

7-Identificar conocimientos y habilidades necesarios (Administrador de desarrollo)

14-Identificar productos de trabajo, elementos adicionales y procedimientos especiales para la configuración de software (Administrador de la configuración)

17-Identificar herramientas necesarias y productos o servicios externos (Administrador de la configuración y Administrador de desarrollo)

Artefactos: 23.Especificación de requerimientos

26.Estimaciones del proyecto

46.Lista de productos y elementos adicionales de la configuración

47.Lista de productos y elementos de la configuración

44.Lista de nuevas herramientas, productos o servicios para adquirir

27.Estimado de recursos de cómputo

PE 1.3 Definir el Ciclo de Vida del Proyecto

Se refiere a definir las fases del ciclo de vida del proyecto sobre el alcance del esfuerzo planeado para proveer periodos de evaluación y toma de decisiones. En esta práctica cabe hacer una observación con respecto a las fases: dado que la metodología propuesta esta basada en el Proceso Unificado de Desarrollo de Software, las fases en el ciclo de vida del proyecto están definidas sobre niveles de elaboración progresivos del producto final, tomando incrementos de funcionalidad que en ultima instancia son los Casos de Uso, lo anterior quiere decir que en el momento que definimos prioridades de elaboración de los Casos de Uso, estamos definiendo las fases que va a tener el proyecto. Por esta razón las actividades y artefactos que aquí se mencionan son relacionados a la priorización de los Casos de Uso

Actividades: 12-Priorizar los casos de uso (Administrador de desarrollo)

23-Elaborar el Plan de desarrollo del proyecto (Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y Administrador de desarrollo)

30-Priorizar los nuevos Casos de uso (Administrador de desarrollo)

Artefactos: 58.Modelo de casos de uso

72.Plan de desarrollo del proyecto

PE 1.4 Determinar estimaciones de esfuerzo y costo

Se refiere a estimar el costo y el esfuerzo del proyecto para productos de trabajo y tareas basándose en una estimación racional. Específicamente se trata esta práctica con el documento de Estimaciones del Proyecto, y en general se trata en la sub-etapa de estimación del proyecto de la Etapa de Preparación.

Actividades: 6-Analizar y ajustar la solicitud y los requerimientos (Cliente, Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y Administrador de desarrollo)

7-Identificar conocimientos y habilidades necesarios (Administrador de desarrollo)

11-Identificar actores y casos de uso (Cliente y Analista de sistema)

14-Identificar productos de trabajo, elementos adicionales y procedimientos especiales para la configuración de software (Administrador de la configuración)

17-Identificar herramientas necesarias y productos o servicios externos (Administrador de la configuración y Administrador de desarrollo)

24-Calcular estimados monetarios (Administrador principal y Administrador del proyecto)

Artefactos: 26.Estimaciones del proyecto

53.Mecanismos para la adquisición de conocimientos y habilidades

46.Lista de productos y elementos adicionales de la configuración

47.Lista de productos y elementos de la configuración

44.Lista de nuevas herramientas, productos o servicios para adquirir

78.Procedimientos especiales de la configuración

12.Cotización del proyecto

27.Estimado de recursos de cómputo

ME2: Desarrollar un Plan del Proyecto

Se refiere a que se debe establecer y mantener un plan del proyecto que sirva de base para administrar el proyecto, lo que se logra a través de las siguientes prácticas:

PE 2.1 Establecer el presupuesto y el calendario

Se refiere a establecer el calendario y presupuesto del proyecto. Su implementación en la metodología resulta transparente a través de:

Actividades: 23-Elaborar el Plan de desarrollo del proyecto (Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y Administrador de desarrollo)
24-Calcular estimados monetarios (Administrador principal y Administrador del proyecto)
27-Realizar negociaciones y acuerdos (Cliente y Administrador principal)
Artefactos: 72.Plan de desarrollo del proyecto
12.Cotización del proyecto
11.Contrato del proyecto

PE 2.2 Identificar los riesgos del proyecto

Se refiere a que se deben identificar y analizar los riesgos del proyecto para apoyar en la planeación del proyecto, y esta actividad debe ser extendida a todos los planes que afectan al proyecto, lo cual se cumple en la metodología prepuesta al estipular que todos los planes dependan del Plan de Desarrollo del Proyecto, el cual ya contempla una identificación de riesgos.

Actividades: 22-Elaborar la evaluación de riesgos (Administrador principal, Administrador del proyecto, Administrador de la calidad y Administrador de desarrollo)
Artefactos: 94.Riesgos y capacidades de proveedor(es) seleccionado(s)
48.Lista de riesgos
69.Plan de contención
70.Plan de contingencia

PE 2.3 Plan para la Administración de Datos

Se refiere a que se debe de planear la administración de datos del proyecto. Esto es parte de la Administración de la Configuración y es cubierto al participar el Administrador de la Configuración en la elaboración del Plan de Desarrollo del Proyecto, mismo plan que lo guiará para sus propias actividades y desarrollo de su propio plan.

Actividades: 6-Analizar y ajustar la solicitud y los requerimientos (Cliente, Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y Administrador de desarrollo)
14-Identificar productos de trabajo, elementos adicionales y procedimientos especiales para la configuración de software (Administrador de la configuración)
22-Elaborar la evaluación de riesgos (Administrador principal, Administrador del proyecto, Administrador de la calidad y Administrador de desarrollo)
Artefactos: 72.Plan de desarrollo del proyecto
73.Plan de la administración de la configuración
78.Procedimientos especiales de la configuración
46.Lista de productos y elementos adicionales de la configuración
47.Lista de productos y elementos de la configuración

PE 2.4 Plan para los Recursos del Proyecto

Se refiere a que se debe planear los recursos necesarios para la realización del proyecto, definiéndolos y cuantificándolos.

Actividades: 3-Seleccionar al Administrador del proyecto y al Administrador de la calidad (Administrador principal)
4-Seleccionar al Administrador de desarrollo (Administrador del proyecto)
5-Seleccionar al Administrador de la configuración (Administrador de la calidad)
8-Seleccionar el equipo de ingeniería (Administrador de desarrollo)
17-Identificar herramientas necesarias y productos o servicios externos (Administrador de la configuración y Administrador de desarrollo)
21-Seleccionar proveedor(es) (Administrador del proyecto y Administrador de desarrollo)

23-Elaborar el Plan de desarrollo del proyecto (Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y Administrador de desarrollo)
24-Calcular estimados monetarios (Administrador principal y Administrador del proyecto)

Artefactos: 72.Plan de desarrollo del proyecto
12.Cotización del proyecto
12.Cotización del proyecto

PE 2.5 Plan para las Habilidades y Conocimientos Necesarios

Se refiere a que se debe planear las habilidades y conocimientos necesarios para realizar el proyecto.

Actividades: 7-Identificar conocimientos y habilidades necesarios (Administrador de desarrollo)

Artefactos: 53.Mecanismos para la adquisición de conocimientos y habilidades

PE 2.6 Plan para Implicar a los Interesados

Se refiere a que se debe planear la implicación de los interesados identificados. En la metodología propuesta se asignaran responsabilidades y actividades a los interesados, a través del Plan de Desarrollo del Proyecto y la Definición de Roles.

Actividades: 3-Seleccionar al Administrador del proyecto y al Administrador de la calidad (Administrador principal)
4-Seleccionar al Administrador de desarrollo (Administrador del proyecto)
5-Seleccionar al Administrador de la configuración (Administrador de la calidad)
8-Seleccionar el equipo de ingeniería (Administrador de desarrollo)
21-Seleccionar proveedor(es) (Administrador del proyecto y Administrador de desarrollo)
23-Elaborar el Plan de desarrollo del proyecto (Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y Administrador de desarrollo)

Artefactos: 72.Plan de desarrollo del proyecto
66.Organigrama de ingeniería
17.Definición de Roles

PE 2.7 Establecer un Plan del Proyecto

Se refiere a establecer y mantener el contenido del plan global del proyecto

Actividades: 23-Elaborar el Plan de desarrollo del proyecto (Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y Administrador de desarrollo)
70-Actualizar Plan de trabajo y Organigrama (Administrador del proyecto)

Artefactos: 72.Plan de desarrollo del proyecto

ME3: Obtener acuerdos para el Plan

Se refiere a mantener y establecer los acuerdos sobre el plan del proyecto con los responsables de implementar y soportar el plan, lo que se logra a través de las siguientes prácticas:

PE 3.1: Revisar planes que afecten al proyecto

Se refiere a revisar todos los planes que afecten el proyecto para la comprensión de los acuerdos. En la metodología propuesta esto se implementa de manera automática, ya que el Plan de Desarrollo del Proyecto se establece como la guía de todos los demás planes, mismos que al momento de elaborarse pueden generar una solicitud de cambio al Plan de Desarrollo del Proyecto y activar toda la maquinaria de cambios involucrada.

Actividades: 23-Elaborar el Plan de desarrollo del proyecto (Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y Administrador de desarrollo)
70-Actualizar Plan de trabajo y Organigrama (Administrador del proyecto)
Artefactos: 72.Plan de desarrollo del proyecto
69.Plan de contención
70.Plan de contingencia
68.Plan de aseguramiento de la calidad
73.Plan de la administración de la configuración

PE 3.2: Reconciliar los niveles de recursos y de trabajo

Se debe reconciliar el plan del proyecto para reflejar los recursos disponibles y estimados detectados a través de las métricas e indicadores.

Actividades: 23-Elaborar el Plan de desarrollo del proyecto (Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y Administrador de desarrollo)
27-Realizar negociaciones y acuerdos (Cliente y Administrador principal)
28-Establecer acuerdos con proveedor (es) (Administrador del proyecto y administrador de desarrollo)
70-Actualizar Plan de trabajo y Organigrama (Administrador del proyecto)
Artefactos: 72.Plan de desarrollo del proyecto
66.Organigrama de ingeniería
17.Definición de Roles

PE 3.3: Obtener compromisos con el plan

Se refiere a obtener acuerdos con los interesados principales responsables de realizar y soportar la ejecución del plan. Para los involucrados principales dentro de la organización de desarrollo, el compromiso se obtiene en el momento de participar en la elaboración del Plan de Desarrollo del Proyecto; mientras que con los externos se obtienen por medio de contratos.

Actividades: 23-Elaborar el Plan de desarrollo del proyecto (Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y Administrador de desarrollo)
27-Realizar negociaciones y acuerdos (Cliente y Administrador principal)
28-Establecer acuerdos con proveedor (es) (Administrador del proyecto y administrador de desarrollo)
Artefactos: 72.Plan de desarrollo del proyecto
11.Contrato del proyecto
10.Contrato con proveedor

AP: Monitoreo y Control del Proyecto

ME1: Monitorear el proyecto contra el plan

Se refiere a que la ejecución actual y el progreso del proyecto son monitoreados contra el plan del proyecto, lo que se logra a través de las siguientes prácticas:

PE 1.1: Monitorear los parámetros planeados del proyecto

Monitorear los valores de los parámetros del proyecto contra el plan del proyecto.

Actividades: 66-Monitorear avance del proyecto y colectar Métricas de productos y recursos (Administrador de desarrollo)
68-Completar métricas de recursos (Administrador del proyecto)
69-Analizar Métricas y administrar excepciones y problemas (Administrador del proyecto)
77-Colectar métricas del proyecto (Administrador del proyecto y Administrador de desarrollo)

Artefactos: 72.Plan de desarrollo del proyecto
55.Métricas de recursos
54.Métricas de productos
57.Métricas del proyecto
40.Interpretación de métricas
32.Inconsistencias con requerimientos

PE 1.2: Monitorear acuerdos

Se refiere a monitorear acuerdos contra lo identificado en el plan del proyecto.

Actividades: 66-Monitorear avance del proyecto y colectar Métricas de productos y recursos (Administrador de desarrollo)
76-Convocar junta interna de revisión e informe general de avance. (Administrador del proyecto)
78-Convocar junta externa de revisión e informe general de avance (Administrador del proyecto)

Artefactos: 72.Plan de desarrollo del proyecto
55.Métricas de recursos
54.Métricas de productos
40.Interpretación de métricas
11.Contrato del proyecto
10.Contrato con proveedor
72.Plan de desarrollo del proyecto
40.Interpretación de métricas
32.Inconsistencias con requerimientos

PE 1.3: Monitorear riesgos del proyecto

Se refiere a monitorear los riesgos contra los identificados en el plan del proyecto

Actividades: 66-Monitorear avance del proyecto y colectar Métricas de productos y recursos (Administrador de desarrollo)
68-Completar métricas de recursos (Administrador del proyecto)
69-Analizar Métricas y administrar excepciones y problemas (Administrador del proyecto)

Artefactos: 72.Plan de desarrollo del proyecto
32.Inconsistencias con requerimientos
55.Métricas de recursos
54.Métricas de productos
33.Informe de avance técnico
40.Interpretación de métricas
48.Lista de riesgos

PE 1.4: Monitorear administración de datos

Se refiere a monitorear la administración de datos del proyecto contra el plan del proyecto.

Actividades: 67-Revisar actividades, roles y artefactos (Administrador de la calidad)
68-Completar métricas de recursos (Administrador del proyecto)
75-Revisar la integridad del Repositorio (Administrador de la configuración)

Artefactos: 90.Reportes de auditoría por artefacto
91.Reportes de auditoría por rol
89.Reportes de auditoría por actividad
55.Métricas de recursos
54.Métricas de productos
40.Interpretación de métricas
35.Informe de la revisión al repositorio

PE 1.5: Monitorear la implicación de los interesados

Se refiere a monitorear la implicación de los interesados contra el plan del proyecto

Actividades: 66-Monitorear avance del proyecto y colectar Métricas de productos y recursos (Administrador de desarrollo)
67-Revisar actividades, roles y artefactos (Administrador de la calidad)
68-Completar métricas de recursos (Administrador del proyecto)
69-Analizar Métricas y administrar excepciones y problemas (Administrador del proyecto)

Artefactos: 33.Informe de avance técnico
55.Métricas de recursos
40.Interpretación de métricas
90.Reportes de auditoría por artefacto
72.Plan de desarrollo del proyecto
11.Contrato del proyecto
10.Contrato con proveedor

PE 1.6: Conducir revisiones al proyecto

Se refiere a revisar periódicamente el progreso, desempeño y cuestiones del proyecto

Actividades: 77-Colectar métricas del proyecto (Administrador del proyecto y Administrador de desarrollo)
66-Monitorear avance del proyecto y colectar Métricas de productos y recursos (Administrador de desarrollo)

Artefactos: 57.Métricas del proyecto
33.Informe de avance técnico
40.Interpretación de métricas
32.Inconsistencias con requerimientos

PE 1.7: Conducir revisiones en milestones

Se refiere a revisar el cumplimiento y resultados del proyecto en milestones seleccionados del proyecto. En la metodología propuesta esto se implementa mediante las juntas de revisión externa que se realizan cada vez que termina una iteración.

Actividades: 78-Convocar junta externa de revisión e informe general de avance (Administrador del proyecto)

Artefactos: 90.Reportes de auditoría por artefacto
72.Plan de desarrollo del proyecto
11.Contrato del proyecto
10.Contrato con proveedor

ME2: Administrar las acciones correctivas

Se refiere a que las acciones correctivas sean administradas cuando el desempeño o resultado del proyecto se desvían significativamente del plan, lo que se logra a través de las siguientes prácticas:

PE 2.1: Analizar el caso

Colectar y analizar los casos de desviaciones y determinar las acciones correctivas necesarias para solucionarlo.

Actividades: 71-Recibir, registrar y convocar la evaluación de Solicitudes de cambio e informes de problemas. (Administrador de la configuración)

72-Definir impacto de cambio o de la solución al problema (Administrador de la configuración, Administrador del proyecto, Administrador de la calidad y Administrador de desarrollo)

73-Controlar y dar seguimiento a los cambios o soluciones autorizadas (Administrador de la configuración)

Artefactos: 33.Informe de avance técnico

55.Métricas de recursos

40.Interpretación de métricas

54.Métricas de productos

57.Métricas del proyecto

32.Inconsistencias con requerimientos

95.Solicitud de cambio

1.Análisis del impacto del cambio

40.Interpretación de métricas

PE 2.2: Tomar una acción correctiva

Se refiere a tomar acciones correctivas sobre los casos de desviación identificados.

Actividades: 72-Definir impacto de cambio o de la solución al problema (Administrador de la configuración, Administrador del proyecto, Administrador de la calidad y Administrador de desarrollo)

73-Controlar y dar seguimiento a los cambios o soluciones autorizadas (Administrador de la configuración)

Artefactos: 95.Solicitud de cambio

1.Análisis del impacto del cambio

PE 2.3: Administrara la acción correctiva

Se refiere a administrar las acciones correctivas para completarse.

Actividades: 73-Controlar y dar seguimiento a los cambios o soluciones autorizadas (Administrador de la configuración)

Artefactos: 95.Solicitud de cambio

1.Análisis del impacto del cambio

AP: Administrador de acuerdos con proveedores

ME1: Establecer acuerdos con proveedores

Se refiere a establecer y mantener los acuerdos con los proveedores, lo que se logra a través de las siguientes prácticas:

PE 1.1: Determinar el tipo de adquisición

Se refiere a que se debe determinar el tipo de adquisición para cada producto o componente de producto a ser adquirido

Actividades: 17-Identificar herramientas necesarias y productos o servicios externos (Administrador de la configuración y Administrador de desarrollo)

Artefactos: 44.Lista de nuevas herramientas, productos o servicios para adquirir

PE 1.2: Seleccionar proveedor

Se refiere a que se debe seleccionar el proveedor basándose en una evaluación de su habilidad para satisfacer el requerimiento especificado, y establecer el criterio

Actividades: 18-Definir criterios para evaluar proveedores potenciales (Administrador del proyecto y Administrador de desarrollo)

19-Identificar proveedores potenciales y distribuirles requerimientos solicitados (Administrador del proyecto y Administrador de desarrollo)

Artefactos: 13.Criterios de evaluación de proveedores

93.Requerimientos a externos

45.Lista de posibles proveedores

PE 1.3: Establecer acuerdos con el proveedor

Se refiere a establecer y mantener acuerdos con los proveedores.

Actividades: 21-Seleccionar proveedor(es) (Administrador del proyecto y Administrador de desarrollo)

28-Establecer acuerdos con proveedor (es) (Administrador del proyecto y administrador de desarrollo)

Artefactos: 71.Plan de desarrollo de proveedor

94.Riesgos y capacidades de proveedor(es) seleccionado(s)

10.Contrato con proveedor

42.Lista de artefactos que entregará proveedor

ME2: Satisfacer acuerdos con proveedores

Se refiere a que los acuerdos con los proveedores deben ser satisfechos, tanto por la organización de desarrollo y el proveedor, lo que se logra a través de las siguientes prácticas:

PE 2.1: Revisar ofertas de productos

Se refiere a revisar los productos de los proveedores para asegurar que satisfacen los requerimientos que se describen en los Acuerdos con el Proveedor. Para esto, la metodología propuesta debe tratar dichos productos como elaborados por el propio Grupo de Ingeniería, por lo cual pasaran a través de las auditorías de calidad.

Actividades: 67-Revisar actividades, roles y artefactos (Administrador de la calidad)

Artefactos: 93.Requerimientos a externos

42.Lista de artefactos que entregará proveedor

95.Solicitud de cambio (opcional)

89.Reportes de auditoría por actividad

90.Reportes de auditoría por artefacto

91.Reportes de auditoría por rol

PE 2.2: Ejecutar acuerdos con proveedores

Se refiere a realizar actividades con el proveedor como se especifica en los acuerdos con el proveedor. Para la metodología propuesta las actividades involucradas varían de proyecto en proyecto dependiendo de la naturaleza del producto o servicio que se

solicite al proveedor, pero se pueden incluir el monitoreo de las actividades del proveedor dentro del monitoreo de las actividades de la propia organización de desarrollo.

Actividades: 66-Monitorear avance del proyecto y colectar Métricas de productos y recursos (Administrador de desarrollo)
68-Completar métricas de recursos (Administrador del proyecto)
69-Analizar Métricas y administrar excepciones y problemas (Administrador del proyecto)

Artefactos: 71.Plan de desarrollo de proveedor
10.Contrato con proveedor
42.Lista de artefactos que entregará proveedor
55.Métricas de recursos
54.Métricas de productos
40.Interpretación de métricas
32.Inconsistencias con requerimientos
33.Informe de avance técnico

PE 2.3: Aceptar el producto adquirido

Se refiere a que se debe asegurar que los acuerdos con el proveedor sean satisfechos antes de aceptar los productos adquiridos. Esto se implementa en la metodología propuesta por medio de aplicar pruebas de aceptación a los productos generados por los proveedores, como si se tratase de productos creados por la propia organización de desarrollo.

Actividades: 60-Realizar pruebas de aceptación (Administrador de desarrollo)
61-Supervisar pruebas de aceptación (Administrador de desarrollo, Diseñador de pruebas)

Artefactos: 93.Requerimientos a externos
34.Informe de defectos
95.Solicitud de cambio (opcional)
28.Evaluación de pruebas (de aceptación)

PE 2.4: Transición del producto

Se refiere a la transición de los productos adquiridos al proveedor hacia el proyecto. Esto se implementa en la metodología propuesta al tratar los productos del proveedor como elaborados por el grupo de ingeniería de la organización de desarrollo, e integrándolos al proyecto durante el flujo de trabajo de implementación, específicamente durante la integración de las construcciones; pero esto puede variar de acuerdo a la naturaleza de la adquisición.

Actividades: 47-Planear la integración en la iteración (Integrador del sistema)
51-Integrar el sistema (Integrador del sistema)

Artefactos: 98.Subsistemas de implementación
39.Interfaces implementadas de subsistemas
74.Plan de la integración de la construcción
93.Requerimientos a externos
42.Lista de artefactos que entregará proveedor

AP: Medición y Análisis

ME1: Alinear medidas y actividades de análisis

Se refiere a establecer objetivos y actividades de medidas respecto a las necesidades de información identificadas y los objetivos, lo que se logra a través de las siguientes prácticas:

PE 1.1: Establecer objetivos de medidas

Se refiere a establecer y mantener objetivos de las medidas que deriven de identificar necesidades de información y objetivos. Esta práctica se considera dentro de la definición de métricas, en la metodología propuesta se menciona que las métricas deben ser definidas antes de intentar dirigir un proyecto con la metodología.

Actividades:

Artefactos: 15.Definición de métricas

PE 1.2: Especificar medidas

Se refiere a especificar las medidas que cubrirán los objetivos de medición. En la metodología propuesta esto esta también incluido dentro de la definición de métricas, como un prerrequisito.

Actividades:

Artefactos: 15.Definición de métricas

PE 1.3: Especificar colección de datos y procedimientos de almacenamiento

Se refiere a cómo los datos de medición serán obtenidos y almacenados. En la metodología propuesta esto se debe implementar en la definición de los procedimientos de las actividades relacionadas con la colección de métricas, y en la definición de métricas.

Actividades: 66-Monitorear avance del proyecto y colectar Métricas de productos y recursos (Administrador de desarrollo)
68-Completar métricas de recursos (Administrador del proyecto)
77-Colectar métricas del proyecto (Administrador del proyecto y Administrador de desarrollo)
90-Colectar métricas del producto terminado, generales de recursos, del proceso y del proyecto. (Administrador del proyecto)

Artefactos: 15.Definición de métricas
16.Definición de procedimientos

PE 1.4: Especificar procedimientos de análisis

Se refiere a especificar cómo los datos medidos deben ser analizados y reportados. En la metodología propuesta esto se debe implementar en la definición de los procedimientos de las actividades relacionadas con el análisis de métricas, y en la definición de las métricas y su interpretación.

Actividades: 69-Analizar Métricas y administrar excepciones y problemas (Administrador del proyecto)

Artefactos: 15.Definición de métricas
40.Interpretación de métricas
16.Definición de procedimientos

ME2: Proveer resultados de medidas

Se refiere a proveer los resultados de medidas que cubrirán las necesidades identificadas de información y objetivos, lo que se logra a través de las siguientes prácticas:

PE 2.1: Colectar datos de medidas

Se refiere a obtener los resultados especificados de medidas. En la metodología propuesta esto se implementa mediante las actividades como tal (ya no los procedimientos) de colección de métricas y las métricas en sí (ya no su definición).

Actividades: 66-Monitorear avance del proyecto y colectar Métricas de productos y recursos (Administrador de desarrollo)
68-Completar métricas de recursos (Administrador del proyecto)
77-Colectar métricas del proyecto (Administrador del proyecto y Administrador de desarrollo)
90-Colectar métricas del producto terminado, generales de recursos, del proceso y del proyecto. (Administrador del proyecto)
Artefactos: 54.Métricas de productos
55.Métricas de recursos
56.Métricas del proceso
57.Métricas del proyecto

PE 2.2: Analizar datos de medidas

Se refiere a analizar e interpretar los datos medidos. En la metodología propuesta esto se implementa mediante las actividades como tal (ya no los procedimientos) de análisis de métricas.

Actividades: 69-Analizar Métricas y administrar excepciones y problemas (Administrador del proyecto)
Artefactos: 54.Métricas de productos
55.Métricas de recursos
56.Métricas del proceso
57.Métricas del proyecto
29.Evaluación general del proyecto
40.Interpretación de métricas

PE 2.3: Almacenar datos y resultados

Se refiere a administrar y almacenar los datos medidos, especificación de medidas y análisis de resultados. En la metodología propuesta esto se implementa al plasmar los datos medidos y su análisis en los documentos correspondientes y almacenar estos documentos en el Repositorio de la Configuración.

Actividades: 74-Integrar artefactos del Repositorio (Administrador de la configuración)
Artefactos: 54.Métricas de productos
55.Métricas de recursos
56.Métricas del proceso
57.Métricas del proyecto
29.Evaluación general del proyecto
40.Interpretación de métricas

PE 2.4: Comunicar datos

Se refiere a reportar los resultados de las actividades de medición y análisis a los interesados relevantes. Esto se implementa en la metodología propuesta en las juntas de revisión internas y externas.

Actividades: 76-Convocar junta interna de revisión e informe general de avance. (Administrador del proyecto)
78-Convocar junta externa de revisión e informe general de avance (Administrador del proyecto)
79-Informar del avance de proyecto al Administrador principal (Administrador del proyecto, Administrador principal)
Artefactos: 54.Métricas de productos
55.Métricas de recursos
56.Métricas del proceso
57.Métricas del proyecto
29.Evaluación general del proyecto
40.Interpretación de métricas

AP: Aseguramiento de la calidad de procesos y productos

ME1: Evaluar objetivamente procesos y productos de trabajo

Se refiere a evaluar objetivamente el apego de la ejecución de los procesos y productos de trabajo asociados y servicios para aplicar procesos, estándares y procedimientos, lo que se logra a través de las siguientes prácticas:

PE 1.1: Evaluar objetivamente procesos

Se refiere a evaluar objetivo los procesos señalados realizados contra las descripciones, los estándares, y los procedimientos de proceso aplicables. En la metodología propuesta esto se implementa mediante las auditorías por parte de la Administración de la Calidad y la generación de los Reportes de Auditoría por Actividad.

Actividades: 10-Revisar el Modelo del dominio y el Modelo del negocio (Administrador de la calidad)
13-Revisar preliminar del Modelo de casos de uso (Administrador de la calidad)
16-Revisar el establecimiento del Repositorio de la configuración de software (Administrador de la calidad)
25-Revisar documentos para negociación (Administrador de la calidad)
67-Revisar actividades, roles y artefactos (Administrador de la calidad)
86-Revisar propuestas de respuesta a las retroalimentaciones (Administrador de la calidad)

Artefactos: 89.Reportes de auditoría por actividad
76.Políticas
24.Estándares
16.Definición de procedimientos

PE 1.2: Evaluar objetivamente productos de trabajo y servicios

Se refiere a evaluar objetivamente los productos de trabajo y los servicios señalados contra las descripciones, los estándares y los procedimientos del proceso aplicables. Esto se implementa en la metodología propuesta mediante las auditorías por parte de la Administración de la Calidad y la generación de los Reportes de Auditoría por Artefacto.

Actividades: 10-Revisar el Modelo del dominio y el Modelo del negocio (Administrador de la calidad)
13-Revisar preliminar del Modelo de casos de uso (Administrador de la calidad)
16-Revisar el establecimiento del Repositorio de la configuración de software (Administrador de la calidad)
25-Revisar documentos para negociación (Administrador de la calidad)
67-Revisar actividades, roles y artefactos (Administrador de la calidad)
86-Revisar propuestas de respuesta a las retroalimentaciones (Administrador de la calidad)

Artefactos: 90.Reportes de auditoría por artefacto
76.Políticas
24.Estándares
16.Definición de procedimientos

ME2: Proveer una visión objetiva

Se refiere a dar seguimiento y comunicar los casos de incumplimiento, y asegurar su resolución, lo que se logra a través de las siguientes prácticas:

PE 2.1: Comunicar y asegurar la resolución de casos que no cumplieron

Comunicar cuestiones de seguridad y asegurar la resolución casos de incumplimiento con el staff y administradores. En la metodología propuesta se realizan auditorías por parte del Administrador de Calidad, lo que da lugar a la creación de los Reportes de Auditoría por Rol, por Artefacto y por Actividad, y al detectar inconsistencias de calidad

se debe generar la respectiva Solicitud de Cambio. Si una solicitud de cambio es disparada por el Administrador de Calidad, entonces ‘el junto con el Administrador de la Configuración se encargaran de dar el seguimiento correspondiente, en caso de que se autorice el cambio.

Actividades: 10-Revisar el Modelo del dominio y el Modelo del negocio (Administrador de la calidad)
13-Revisar preliminar del Modelo de casos de uso (Administrador de la calidad)
16-Revisar el establecimiento del Repositorio de la configuración de software (Administrador de la calidad)
25-Revisar documentos para negociación (Administrador de la calidad)
67-Revisar actividades, roles y artefactos (Administrador de la calidad)
86-Revisar propuestas de respuesta a las retroalimentaciones (Administrador de la calidad)
73-Controlar y dar seguimiento a los cambios o soluciones autorizadas (Administrador de la configuración)

Artefactos: 89.Reportes de auditoría por actividad
90.Reportes de auditoría por artefacto
91.Reportes de auditoría por rol
95.Solicitud de cambio

PE 2.2: Establecer registros

Se refiere a establecer y mantener registros sobre las actividades de aseguramiento de la calidad. Esto se implementa por medio de los reportes de auditoría.

Actividades:

Artefactos: 89.Reportes de auditoría por actividad
90.Reportes de auditoría por artefacto
91.Reportes de auditoría por rol

AP: Administración de la Configuración.

ME1: Establecer Líneas Base

Se refiere a establecer las Líneas Base de los productos de trabajo, lo que se logra a través de las siguientes prácticas:

PE 1.1: Identificar objetos de la configuración

Se refiere a identificar los elementos, componentes y productos de trabajo relacionados de la configuración, que serán puestos bajo la Administración de la Configuración.

Actividades: 14-Identificar productos de trabajo, elementos adicionales y procedimientos especiales para la configuración de software (Administrador de la configuración)

Artefactos: 46.Lista de productos y elementos adicionales de la configuración
47.Lista de productos y elementos de la configuración

PE 1.2: Establecer un sistema de administración de configuración

Se refiere a establecer y mantener un Sistema de Administración de la Configuración y administración de cambios, para controlar los productos de trabajo.

Actividades: 15-Establecer el Repositorio de la configuración del proyecto (Administrador de la configuración)

Artefactos: 92.Repositorio del proyecto
19.Descripción de niveles de promoción
20.Descripción del repositorio

PE 1.3: Crear y liberar Líneas Base

Se refiere a crear y liberar las Líneas Base para uso interno y para entrega al Cliente

Actividades: 62-Integrar Línea base (Administrador de la configuración)
80-Producir Unidades de despliegue (Administrador de la configuración)

Artefactos: 41.Línea base
100.Unidades de despliegue

ME 2: Seguimiento y control de cambios

Se refiere a dar seguimiento y controlar los cambios de los productos de trabajo bajo la administración de la configuración, lo que se logra a través de las siguientes prácticas:

PE 2.1: Seguimiento de solicitudes de cambio

Se refiere a dar seguimiento a las solicitudes de cambio de los elementos de la configuración

Actividades: 73-Controlar y dar seguimiento a los cambios o soluciones autorizadas (Administrador de la configuración)

Artefactos: 95.Solicitud de cambio

PE 2.2: Control de objetos de configuración

Se refiere a controlar los cambios a los elementos de la configuración.

Actividades: 73-Controlar y dar seguimiento a los cambios o soluciones autorizadas (Administrador de la configuración)
74-Integrar artefactos del Repositorio (Administrador de la configuración)

Artefactos: 95.Solicitud de cambio
92.Repositorio del proyecto (actualizado)
19.Descripción de niveles de promoción

ME3: Establecer integridad

Se refiere a establecer y mantener la integridad de las Líneas Base, lo que se logra a través de las siguientes prácticas:

PE 3.1: Establecer registros de administración de configuración

Se refiere a establecer y mantener registros que describan lo elementos de la configuración

Actividades: 26-Integrar Línea base inicial y otros artefactos iniciales al repositorio (Administrador de la configuración)

75-Revisar la integridad del Repositorio (Administrador de la configuración)

Artefactos: 20.Descripción del repositorio

88.Reporte del contenido del repositorio

35.Informe de la revisión al repositorio

PE 3.2: Realizar auditorías a la configuración

Se refiere a realizar auditorias a la configuración para mantener la integridad de las Líneas Base de la configuración.

Actividades: 75-Revisar la integridad del Repositorio (Administrador de la configuración)

Artefactos: 35.Informe de la revisión al repositorio

88.Reporte del contenido del repositorio

Las prácticas anteriores se refieren a Áreas de Proceso específicas dentro de CMMI en su nivel de madurez 2, pero a todas estas áreas se les aplican una serie de metas y practicas genéricas con el objetivo de institucionalizar el proceso de manera administrada dentro de la organización de desarrollo:

MG 2: Institucionalizar un proceso administrado

Se refiere a que el proceso se debe institucionalizar como un proceso administrado, lo que se logra a través de las siguientes prácticas genéricas para todas las Áreas de Proceso del nivel de madurez 2:

PG 2.1 Establecer una política organizacional (AR1)

Se refiere a establecer y mantener una política organizacional para planear y realizar los procesos de cada una de las Áreas de Proceso. Para la metodología propuesta se recomienda que se elaboren políticas específicas para cada una de las Áreas de Proceso.

Actividades:

Artefactos: 76.Políticas (Políticas de la Administración de requerimientos, Políticas de la Planeación del proyecto, Políticas del Monitoreo y control del proyecto, Políticas de la Administración de proveedores, Políticas de la Medición y el análisis, Políticas de Aseguramiento de la calidad, Políticas de la Administración de la configuración de software)

PG 2.2 Planear el proceso (HS1)

Se refiere a establecer y mantener un plan para realizar los procesos relacionados con cada una de las Áreas de Procesos. Para la metodología propuesta, se establece un Plan de Desarrollo del proyecto general, en el cual se incluirán o derivaran planes específicos para las Áreas de Proceso.

Actividades: 23-Elaborar el Plan de desarrollo del proyecto (Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y Administrador de desarrollo)

Artefactos: 72.Plan de Desarrollo del Proyecto (Plan para la Administración de Requerimientos, Plan para los recursos del proyecto, Plan para las habilidades y conocimientos necesarios, Plan para implicar a los interesados, Plan para mantenimiento del Plan de Desarrollo del Proyecto, Plan para Monitoreo y control del Proyecto, Plan para la Administración de Acuerdos con Proveedores, Plan para la Medición y Análisis)

68.Plan de Aseguramiento de la Calidad

73.Plan de la Administración de la Configuración

PG 2.3 Proveer recursos (HS2)

Se refiere a proveer los recursos adecuados para realizar los procesos, desarrollar los productos de trabajo y proveer los servicios para procesos de cada una de las Áreas de Proceso. En la metodología propuesta se identifican los recursos para cada Área de Proceso cuando se realiza el Estimado de Recursos de Cómputo, se descubre la necesidad de las herramientas adicionales, servicios externos y nuevos conocimientos, y se calculan los estimados monetarios. El proveer cada uno de estos recursos se contempla en los Mecanismos para la adquisición de conocimientos y habilidades, en la Cotización y en el Plan de Desarrollo del Proyecto.

Actividades: 7-Identificar conocimientos y habilidades necesarios (Administrador de desarrollo)

17-Identificar herramientas necesarias y productos o servicios externos (Administrador de la configuración y Administrador de desarrollo)

23-Elaborar el Plan de desarrollo del proyecto (Administrador principal, Administrador del proyecto, Administrador de la calidad, Administrador de la configuración y Administrador de desarrollo)

24-Calcular estimados monetarios (Administrador principal y Administrador del proyecto)

Artefactos: 53.Mecanismos para la adquisición de conocimientos y habilidades
44.Lista de nuevas herramientas, productos o servicios para adquirir
27.Estimado de recursos de cómputo
72.Plan de Desarrollo del Proyecto (Plan para la Administración de Requerimientos, Plan para los recursos del proyecto, Plan para las habilidades y conocimientos necesarios, Plan para implicar a los interesados, Plan para mantenimiento del Plan de Desarrollo del Proyecto, Plan para Monitoreo y control del Proyecto, Plan para la Administración de Acuerdos con Proveedores, Plan para la Medición y Análisis)
12.Cotización del proyecto

PG 2.4 Asignar Responsabilidades (HS3)

Se refiere a asignar responsabilidades y autoridad para realizar los procesos, desarrollar los productos de trabajo y proveer los servicios de cada una de las Áreas de Proceso. En la metodología propuesta se cubre esta práctica genérica a través de la selección del personal que va a cubrir los roles de las Áreas de Proceso, a través de la creación del Organigrama de Ingeniería (que plasma cierto nivel de autoridad) y de los planes (donde se asignan las responsabilidades).

Actividades: 3-Seleccionar al Administrador del proyecto y al Administrador de la calidad (Administrador principal)
4-Seleccionar al Administrador de desarrollo (Administrador del proyecto)
5-Seleccionar al Administrador de la configuración (Administrador de la calidad)
8-Seleccionar el equipo de ingeniería (Administrador de desarrollo)
21-Seleccionar proveedor(es) (Administrador del proyecto y Administrador de desarrollo)

Artefactos: 66.Organigrama de Ingeniería
72.Plan de Desarrollo del Proyecto (Plan para la Administración de Requerimientos, Plan para los recursos del proyecto, Plan para las habilidades y conocimientos necesarios, Plan para implicar a los interesados, Plan para mantenimiento del Plan de Desarrollo del Proyecto, Plan para Monitoreo y control del Proyecto, Plan para la Administración de Acuerdos con Proveedores, Plan para la Medición y Análisis)

PG 2.5 Capacitar Personal (HS4)

Se refiere a capacitar el personal para realizar o soportar, como se requiera, a los procesos de cada una de las Áreas de Proceso. Dentro de la metodología propuesta, en la actividad de identificar nuevos conocimientos y habilidades se contempla capacitar a recursos humanos sobre la realización de sus actividades en cierta Área de Procesos, solo aplicándose en casos de promoción de personal o de integrar nuevo personal a la organización.

Actividades: 7-Identificar conocimientos y habilidades necesarios (Administrador de desarrollo)

Artefactos: 53.Mecanismos para la adquisición de conocimientos y habilidades

PG 2.6 Administrar Configuraciones (DI1)

Se refiere a colocar los productos de trabajo señalados de cada Área de Proceso bajo el nivel apropiado de la administración de la configuración. Dentro de la metodología propuesta esto se contempla cada vez que un recurso humano (de cualquier Área de Procesos) genera un artefacto que está bajo la Administración de Configuración de Software.

Actividades: 74-Integrar artefactos del Repositorio (Administrador de la configuración)

Artefactos: 47.Lista de productos y elementos de la configuración
19.Descripción de niveles de promoción

PG 2.7 Identificar e implicar a todos los interesados (DI2)

Se refiere a identificar e involucrar a todos los interesados de cada una de las Áreas de Proceso como se planeó. Esto se implementa en la metodología propuesta en la selección de recursos humanos para cubrir cada uno de los roles, asignar autoridad y responsabilidades a través del Organigrama de Ingeniería y del Plan de Desarrollo del Proyecto, y al convocar juntas con los involucrados en el proyecto

Actividades: 3-Seleccionar al Administrador del proyecto y al Administrador de la calidad (Administrador principal)
4-Seleccionar al Administrador de desarrollo (Administrador del proyecto)
5-Seleccionar al Administrador de la configuración (Administrador de la calidad)
8-Seleccionar el equipo de ingeniería (Administrador de desarrollo)
21-Seleccionar proveedor(es) (Administrador del proyecto y Administrador de desarrollo)
76-Convocar junta interna de revisión e informe general de avance. (Administrador del proyecto)
78-Convocar junta externa de revisión e informe general de avance (Administrador del proyecto)

Artefactos: 66.Organigrama de Ingeniería
72.Plan de Desarrollo del Proyecto (Plan para la Administración de Requerimientos, Plan para los recursos del proyecto, Plan para las habilidades y conocimientos necesarios, Plan para implicar a los interesados, Plan para mantenimiento del Plan de Desarrollo del Proyecto, Plan para Monitoreo y control del Proyecto, Plan para la Administración de Acuerdos con Proveedores, Plan para la Medición y Análisis)

PG 2.8 Monitorear y controlar el proceso (DI3)

Se refiere a monitorear y controlar los procesos de cada una de las Áreas de Procesos contra el plan, para realizarlos y tomar acciones correctivas apropiadas. En la metodología propuesta se aplica a través de las diversas actividades y artefactos de monitoreo, medición, análisis de resultados e informes.

Actividades: 66-Monitorear avance del proyecto y colectar Métricas de productos y recursos (Administrador de desarrollo)
68-Completar métricas de recursos (Administrador del proyecto)
69-Analizar métricas y administrar excepciones y problemas (Administrador del proyecto)
79-Informar del avance de proyecto al Administrador principal (Administrador del proyecto, Administrador principal)

Artefactos: 33.Informe de avance técnico
54.Métricas de productos
55.Métricas de recursos
56.Métricas del proceso
57.Métricas del proyecto

PG 2.9 Evaluar objetivamente el apego al proceso (VI1)

Se refiere a evaluar de manera objetiva el apego de los procesos de cada una de las Áreas de Proceso contra su definición, estándares y procedimientos, y así como rastrear los incumplimientos. En la metodología propuesta esto se lleva a cabo por medio de las actividades del Administrador de la Calidad.

Actividades: 67-Revisar actividades, roles y artefactos (Administrador de la calidad)

Artefactos: 91.Reportes de auditoría por rol
90.Reportes de auditoría por artefacto
89.Reportes de auditoría por actividad

PG 2.10 Revisar el status con administradores de alto nivel (VI2)

Se refiere a revisar las actividades, estado y resultado del proceso de cada una de las Áreas de Proceso con administradores de alto nivel y resolver casos especiales. Esto se lleva a cabo por medio de las juntas de revisión internas y los informes del Administrador del Proyecto para el Administrador Principal.

Actividades: 76-Convocar junta interna de revisión e informe general de avance. (Administrador del proyecto)

79-Informar del avance de proyecto al Administrador principal (Administrador del proyecto, Administrador principal)

Artefactos: 102.Minuta

BIBLIOGRAFÍA

[1] Jacobson, I.; Booch, G.; Rumbaugh J.; “El Proceso Unificado de Desarrollo de Software”; Pearson Educación S.A., Madrid 2000

[2] Booch, G.; Rumbaugh, J.; Jacobson, I.; “El Lenguaje Unificado de Modelado”; Adisson Wesley Iberoamericana, Madrid 1999.

[3] Rumbaugh, J.; Jacobson, I.; Booch, G.; “El Lenguaje Unificado de Modelado, Manual de Referencia”; Pearson Educacion S.A., Madrid 2000.

[4] Krishnan, M.S.; Mukhopadhyay, Tridas; Zubrow, Dave; “Software Process Models and Project Performance”, *Information Systems Frontiers,* Vol 1, No 3, 267-277, October 1999.

[5] Paulk, Mark C.; Curtis, Bill; Chrissis, Mary Beth; Weber, Charles V.; “Capability Maturity Model for Software, Version 1.1”, Technical Report CMU/SEI-93-TR-024 ESC-TR-93-177 February 1993.

[6] “Capability Maturity Model Integration (CMMI), Version 1.1”. CMMI for Software Engineering. (CMMI-SW, V1.1) Staged Representation. CMMI Product Team. CMU/SEI-2002-TR-029 ESC-TR-*2002-029. August 2002.*

[7] “Capability Maturity Model Integration (CMMI), Version 1.1”. CMMI for Software Engineering. (CMMI-SW, V1.1) Continuous Representation. CMMI Product Team. CMU/SEI-2002-TR-028 ESC-TR-*2002-028. August 2002.*

[8] Bowen, Seth; Jiang, Li; “Capability Maturity Model Integration”; University of Calgary. Winter 2002.

[9] Donaldson, Scott E.; Siegel, Stanley G.; “Successful Software Devleopment” Prentice Hall Ptr, United States Of America, 2001.