

DEFINICION DE METODOLOGIA DE DESARROLLO DE SOFTWARE Y  
DOCUMENTACION DE PROCESOS EN IPSOFT S.A

LUIS ALEJANDRO VARGAS MUÑOZ

UNIVERSIDAD AUTÓNOMA DE OCCIDENTE  
FACULTAD DE INGENIERÍA  
DEPARTAMENTO DE CIENCIAS DE LA INFORMACION  
PROGRAMA DE INGENIERÍA INFORMATICA  
SANTIAGO DE CALI  
2006

DEFINICION DE METODOLOGIA DE DESARROLLO DE SOFTWARE Y  
DOCUMENTACION DE PROCESOS EN IPSOFT S.A

LUIS ALEJANDRO VARGAS MUÑOZ

PASANTÍA PARA OPTAR TITULO DE  
INGENIERO EN INFORMÁTICA

DIRECTORA  
MARY ELIZABETH RAMIREZ  
INGENIERA DE SISTEMAS

UNIVERSIDAD AUTÓNOMA DE OCCIDENTE  
FACULTAD DE INGENIERÍA  
DEPARTAMENTO DE CIENCIAS DE LA INFORMACION  
PROGRAMA DE INGENIERÍA INFORMATICA  
SANTIAGO DE CALI  
2006

Nota de Aceptación:

Aprobado por el Comité de Grado en Cumplimientos de los requisitos exigidos por la Universidad Autónoma de Occidente para optar el título de Ingeniero en Informática.

Ing. Mary Elizabeth Ramirez  
Directora Académica del Proyecto

Ciudad y fecha: Santiago de Cali, 3 de Febrero 2006

## **AGRADECIMIENTOS**

El autor expresa sus agradecimientos a:

Miguel Vargas y Rosa Muñoz por su apoyo en toda mi formación académica, a Mary Elizabeth Ramirez, Docente de la Universidad Autonoma de Occidente y coordinadora de proyecto, por sus valiosos aportes, a IPSOFT S.A por la colaboración prestada, y a todos los amigos que colaboran en mi formación académica.

## CONTENIDO

	Pág.
RESUMEN	10
INTRODUCCION	11
1. METODOLOGÍAS DE SOFTWARE EXISTENTES	13
1.1 PROGRAMACION EXTREMA	14
1.1.1 Definición	14
1.1.2 Características	14
1.1.3 Roles	15
1.1.4 Procesos	15
1.1.5 Practicas	16
1.1.6 Valores	18
1.1.7 Principios	18
1.2 PROCESO UNIFICADO RACIONAL	19
1.2.1 Definición	19
1.2.2 Características	19
1.2.3 Practicas	19
1.2.4 Roles	20
1.2.5 Fases	21
1.2.6 Procesos Generales	24
2. PROCESOS DE SOFTWARE	26
2.1 REQUERIMIENTOS	26
2.1.1 Descripción general del proceso de requerimientos	28
2.1.2 Entradas	29
2.1.3 Salidas	29
2.1.4 Productos Internos	30
2.1.5 Referencia Bibliográficas	31
2.1.6 Roles Involucrados	31
2.1.7 Actividades	31
2.1.8 Verificaciones y/o validaciones	35
2.2 ANÁLISIS Y DISEÑO	38
2.2.1 Descripción general del proceso de análisis y diseño	38
2.2.2 Entradas	38
2.2.3 Salidas	38
2.2.4 Productos Internos	40
2.2.5 Referencia Bibliográficas	40
2.2.6 Roles Involucrados	41
2.2.7 Actividades	42
2.2.8 Verificaciones y/o validaciones	45

2.3	CONSTRUCCIÓN	48
2.3.1	Descripción general del proceso de construcción	48
2.3.2	Entradas	49
2.3.3	Salidas	50
2.3.4	Productos Internos	50
2.3.5	Referencia Bibliográficas	50
2.3.6	Roles Involucrados	50
2.3.7	Actividades	51
2.3.8	Verificaciones y/o validaciones	52
2.4	PRUEBAS	54
2.4.1	Descripción general del proceso de análisis y diseño	54
2.4.2	Entradas	54
2.4.3	Salidas	55
2.4.4	Productos Internos	55
2.4.5	Referencia Bibliográficas	55
2.4.6	Roles Involucrados	56
2.4.7	Actividades	57
2.4.8	Verificaciones y/o validaciones	59
3.	CONCLUSIONES	61
4.	RECOMENDACIONES	63
	BIBLIOGRAFIA	64
	ANEXOS	65

## LISTA DE TABLAS

	<b>Pag.</b>
Tabla 1. Descripción general del proceso de requerimientos	28
Tabla 2. Entradas - requerimientos	29
Tabla 3. Salidas - requerimientos	29
Tabla 4. Productos internos - requerimientos	30
Tabla 5. Referencias bibliográficas – requerimientos	31
Tabla 6. Roles involucrados - requerimientos	31
Tabla 7. Actividades - requerimientos	31
Tabla 8. Verificación y/o validaciones - requerimientos	35
Tabla 9. Descripción general del proceso de análisis y diseño	38
Tabla 10. Entradas - análisis y diseño	38
Tabla 11. Salidas - análisis y diseño	38
Tabla 12. Productos internos - análisis y diseño	40
Tabla 13. Referencias bibliográficas – análisis y diseño	40
Tabla 14. Roles involucrados - análisis y diseño	41
Tabla 15. Actividades - análisis y diseño	42
Tabla 16. Verificación y/o validaciones - análisis y diseño	45
Tabla 17. Descripción general del proceso de construcción	48
Tabla 18. Entradas - construcción	49
Tabla 19. Salidas - construcción	50
Tabla 20. Productos internos - construcción	50
Tabla 21. Referencias bibliográficas – construcción	50
Tabla 22. Roles involucrados - construcción	50
Tabla 23. Actividades - construcción	51
Tabla 24. Verificación y/o validaciones - construcción	52
Tabla 25. Descripción general del proceso de pruebas	54
Tabla 26. Entradas - pruebas	54
Tabla 27. Salidas - pruebas	55
Tabla 28. Productos internos - pruebas	55
Tabla 29. Referencias bibliográficas – pruebas	55
Tabla 30. Roles involucrados - pruebas	56
Tabla 31. Actividades - pruebas	57
Tabla 32. Verificación y/o validaciones - pruebas	59

## LISTA DE FIGURAS

	<b>Pag.</b>
Figura 1. Comparación Grafica de XP con el Enfoque tradicional en las etapas de desarrollo (costo vs. tiempo)	17
Figura 2. Grafica del modelo cascada, iterativo y XP	18
Figura 3. Fases, Procesos e iteraciones del RUP	21
Figura 4. Diagrama de actividades del proceso de requerimientos	37
Figura 5. Diagrama de actividades del proceso de análisis y diseño	47
Figura 6. Diagrama de actividades del proceso de construcción	53
Figura 7. Diagrama de actividades del proceso de pruebas	60



## LISTA DE ANEXOS

	<b>Pag.</b>
Anexo A. Formato de guía de entrevista	65
Anexo B. Formato de detalle de procesos	66
Anexo C. Formato de especificación de requerimientos	67
Anexo D. Instructivo de guía de entrevista	68
Anexo E. Instructivo de detalle de procesos	69
Anexo F. Instructivo de especificación de requerimientos	70
Anexo G. Templates de guía de entrevista	71
Anexo H. Templates de arquitectura de software	72
Anexo I. Paper	73

## **RESUMEN**

El propósito de diseñar una Metodología de Desarrollo de Software de software en IPSOFT S.A es identificar, describir, mejorar e implantar los procesos de desarrollo de software que permita disminuir la desorganización en desarrollos de software, para nuevos productos o aplicaciones.

Para desarrollar esta metodología se comenzará con el estudio de la situación actual en que se encuentra la empresa, identificando su estructura, quienes los conforman cada uno de los departamentos, quienes conforman el equipo de desarrollo, sus funciones y cuales son los productos y/o proyectos actuales, además, se realizará una descripción de algunas metodologías de desarrollo de software existentes, tales como (Programación Extrema, Scrum, RUP, metodología de Desarrollo de Sistema Dinámico, etc).

Finalmente se realiza una descripción de los procesos de desarrollo de software propuestos: Requerimientos, Análisis, Diseño, Construcción y Pruebas, y los estadares o modelos que fueron utilizados para su desarrollo.

La Metodología podrá ser empleada en proyectos de distinto tamaño y complejidad, su aplicación tendrá como objetivo proyectos de pequeña escala y riesgo limitado. También será independiente del lenguaje o la arquitectura utilizada.

## INTRODUCCION

Dentro del ámbito de la ingeniería de software (y la informática en general), se ha observado que para cualquier empresa en el campo de software requiere definir, documentar e implantar su propia metodología de desarrollo de software y los procesos que la componen, para obtener una distribución general en los proyectos de software por medio de políticas, procedimientos, actividades, etc. Pero esto, no es suficiente para guiar y mantener un producto de software, buscando por medio de la documentación de los procesos de desarrollo de software, se utilicen las prácticas que ofrece la ingeniería de software para alcanzar sus objetivos.

Sin embargo, muchas veces no se toma en cuenta el utilizar una metodología adecuada, sobre todo cuando se trata de proyectos pequeños de dos o tres meses. Lo que se hace con este tipo de proyectos es separar rápidamente el aplicativo en procesos, cada proceso en funciones, y por cada función determinar un tiempo aproximado de desarrollo.

Cuando los proyectos que se van a desarrollar son de mayor envergadura, ahí sí toma sentido el basarse en una metodología de desarrollo, y se empieza a buscar cual sería la más apropiada para el caso. Lo cierto es que muchas veces no se encuentra la más adecuada y se termina por realizar la que se ajusta a las necesidades de la empresa.

El Grupo de Investigación de Ingeniería de Software de la Universidad Autónoma de Occidente, ofreció su apoyo en la modalidad de pasantía para el desarrollo de la metodología de software en IPSOFT S.A. La metodología es diseñada con base a los recursos y a los procesos que se llevan a cabo en el desarrollo de una nueva aplicación.

La metodología de desarrollo de software contempla el conjunto de filosofías, fases, procedimientos, herramientas, documentación y aspectos que ayuden a utilizar las buenas prácticas para el desarrollo del software, cuyo objetivo es construir mejores aplicaciones, procesos de desarrollo que se identifiquen entradas, salidas(o productos internos) para cada proceso de forma que se pueda planificar y controlar el producto. Una buena metodología se compone de actividades y tareas, procedimientos que son los que describen las actividades o tareas, productos, técnicas, herramientas de software, que ayudan a automatizar algunas actividades y/o tareas para disminuir tiempo y costos en desarrollo.

Conforme aumenta la complejidad y el tamaño del proyecto, la coordinación se dificulta debido al incremento en la comunicación entre los ingenieros de software, administradores y clientes (Fairley, 1985; Kraut y Streeter, 1995).

El proceso de desarrollo de software se puede definir como el conjunto de actividades, métodos, prácticas y transformaciones que los individuos emplean para desarrollar y mantener el software, así como los productos asociados (Paulk, M. C., Weber, C. V., Curtis, B. y Chrisis, M. B., 1995).

Un proceso definido y efectivo disminuye el esfuerzo en el desarrollo de un producto de software, y aumenta la productividad del grupo de desarrollo (Clark, 2000).

El propósito de la ingeniería de software es generar y mantener sistemas de software dentro de las restricciones de tiempo, funcionalidad y costos acordados con el cliente. Las metas de esta disciplina tecnológica son mejorar la calidad de los productos desarrollados y aumentar la productividad de los ingenieros de software. El grado de formalidad y el tiempo asignado al proyecto de software varía de acuerdo al tamaño y complejidad del producto que será desarrollado.

## 1. METODOLOGIAS DE SOFTWARE EXISTENTES

Para diseñar una metodología de software según las investigaciones realizadas se deben tener en cuenta los siguientes aspectos:

Las metodologías imponen un proceso disciplinado sobre el desarrollo de software con el fin de hacerlo más predecible y eficiente. Lo hacen desarrollando un proceso detallado con un fuerte énfasis en planificar inspirado por otras disciplinas de la ingeniería.

Las metodologías de software contemplan el conjunto de filosofías, fases, reglas, procedimientos, técnicas, herramientas, documentación y aspectos de formación para un equipo de desarrollo.

Una metodología de software tiene como objetivos, realizar mejores aplicaciones, obtener mejores procesos de desarrollo que identifique entradas, salidas(o productos intermedios) de cada fase, de forma que se pueda planificar y controlar el proyecto, además proveer un proceso estándar en la organización.

Sus componentes principales para lograr abarcar un buen proceso es necesario tener en cuentas los siguientes:

- Los procesos se descomponen hasta el nivel de actividades y tareas.
- Los procedimientos definen la forma de llevar a cabo las tareas.
- Los productos son obtenidos como resultado de seguir un procedimiento, ya sea productos internos o salientes.
- Las técnicas se utilizan para aplicar un procedimiento, estas pueden ser graficas o textuales, y determina el formato de los productos resultantes de cada actividad o tarea.
- Las herramientas de software proporcionan soporte a la aplicación de las técnicas.

Para el desarrollo de una metodología de software en una empresa implica conocer los procesos que se llevan a cabo cuando se realiza un nuevo proyecto de software, estacando el personal de desarrollo, funciones de cada uno de los integrantes y los procedimientos, productos, técnicas, herramientas de software y documentación que exista que puedan aportar para el desarrollo de la metodología de software, permitiendo identificar falencias en el desarrollo de un nuevo producto de software. Cuando los sistemas “pequeños” van creciendo, Esto

realmente funciona muy bien si el sistema es pequeño, pero conforme el sistema crece llega a ser cada vez más difícil agregar nuevos aspectos al mismo.

A principios de la década del '90, surgió un enfoque que fue bastante revolucionario para su momento ya que iba en contra de la creencia de que mediante procesos altamente definidos se iba a lograr obtener software en tiempo, costo y con la requerida calidad.

La historia de las Metodologías Ágiles y su apreciación como tales en la comunidad de la ingeniería de software tiene sus inicios en la creación de una de las metodologías utilizada como arquetipo: XP - eXtreme Programming. XP surge de la mente de Kent Beck, tomando ideas recopiladas junto a Ward Cunningham, y utilizando conceptos como el de Chief Programmer creado por IBM en la década de los '70.

Entre las metodologías ágiles más destacadas hasta el momento podemos nombrar:

- XP – Extreme Programming
- Scrum
- Crystal Clear
- DSDM – Dynamic Systems Development Method
- FDD – Feature Driven Development
- ASD – Adaptive Software Development
- XBreed
- Extreme Modeling

A continuación, se dará una descripción de las metodologías; Extreme Programming (XP) y Process Unified Racional (RUP), su definición características, prácticas, entre otros aspectos:

## 1. PROGRAMACIÓN EXTREMA

**1.1.1 Definición** La programación extrema se basa en una serie de reglas y principios que se han ido gestando a lo largo de toda la historia de la ingeniería del software. Usadas conjuntamente proporcionan una nueva metodología de desarrollo software que se puede englobar dentro de las metodologías ligeras, que son aquéllas en la que se da prioridad a las tareas que dan resultados directos y que reducen la burocracia que hay alrededor tanto como sea posible.

**1.1.2 Características** las características de la programación extrema son las siguientes:

- Permite introducir nuevos requisitos o cambiar los anteriores de un modo dinámico.

- Publica pronto versiones que implementan parte de los requisitos.
- Es adecuado para proyectos pequeños y medianos.
- También es adecuado para proyectos con alto riesgo.
- Su ciclo de vida es iterativo e incremental.

### 1.1.3 **Roles** Los roles que conforman la programación extrema

- Programador: Escribe las pruebas unitarias y produce el código del sistema.
- Cliente: Escribe las *Historias de Usuario*<sup>1</sup> y las pruebas funcionales para validar su implementación. Además, asigna la prioridad a las historias de usuario y decide cuales se implementa en cada iteración centrándose en aportar mayor valor al negocio.
- Encargado de Pruebas (Tester): Ayuda al cliente a ejecutar las pruebas funcionales, ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para pruebas.
- Encargado de Seguimiento (Tracker): Proporciona realimentación al equipo. Verifica el grado de acierto entre las estimaciones realizadas y el tiempo real dedicado, para mejorar sus propias estimaciones. Realiza el seguimiento del progreso de cada iteración.
- Entrenador (Coach): Es el responsable del proceso global. Debe proveer guías al equipo de forma que se apliquen las practicas XP y se siga el proceso adecuadamente.
- Consultor: Es un miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto.
- Gestión (Big Boss): Es el vinculo entre los clientes y los programadores, ayude a que el equipo trabaje efectivamente creando las condiciones adecuadas.

**1.1.4 Proceso** el proceso de captura de requisitos de XP gira entorno a una lista de características que el cliente desea que existan en el sistema final. Cada una de estas características recibe el nombre de historias de usuarios y su definición consta de dos fases:

*Fase 1:* El cliente describe con sus propias palabras las características y el responsable del equipo de desarrollo le informa de la dificultad técnica de cada una de ellas y por lo tanto de su coste. A través del diálogo resultante el cliente deja por escrito un conjunto de historias y las ordena en función de la prioridad

que tienen para él. En este momento ya es posible definir unos hitos y unas fechas aproximadas para ellos.

*Fase 2:* Consiste en coger las primeras historias que serán implementadas (primera iteración) y dividir las en las tareas necesarias para llevarlas a cabo. El cliente también participa, pero hay más peso del equipo de desarrollo, que dará como resultado una planificación más exacta. En cada iteración se repetirá esta segunda fase para las historias planificadas para ella.

El proceso consiste a grandes rasgos en los siguientes pasos:

- El cliente define el valor del negocio a implementar.
- El programador estima el esfuerzo necesario para su implementación.
- El cliente selecciona que construir, de acuerdo con sus prioridades y restricciones de tiempo.
- El programador construye ese valor del negocio.
- Vuelve al paso 1.

**1.1.5 Prácticas** Las prácticas se pueden agrupar en 4 categorías:

- *Retroalimentación a escala fina:* Pruebas, Programación en parejas, Planificación, cliente en el lugar.
- *Proceso continuo en lugar de por lotes:* Integración continua, refactorización, entregas pequeñas.
- *Entendimiento compartido:* Diseño simple, metáfora, propiedad colectiva del código, estándares de programación o codificación.
- *Bienestar del programador:* 40 horas por semana.

Las 12 prácticas de la XP son:

*Planificación:* se realiza una comunicación entre los clientes y los programadores, el equipo técnico estima el tiempo requerido para establecer la implementación de las historias de usuario y el cliente decide sobre el ámbito y el tiempo de las entregas.

*Entregas Pequeñas:* Producir rápidamente versiones del sistema que sean operativas, aunque no cuenten con toda la funcionalidad del sistema. Una entrega no debería tardar más de 3 meses.

*Metáfora:* Es la arquitectura del sistema. El sistema es definido mediante una metáfora o conjuntos de metáforas compartidas por el cliente y el equipo de



desarrollo. Una metáfora es una metáfora compartida que describe las funciones del sistema.

*Diseño Simple:* Se debe diseñar la solución más simple para que el sistema funcione y ser implementada en un momento determinado del proyecto.

*Pruebas (Testing):* La producción de código está dirigida por las pruebas unitarias. Estas son establecidas por el cliente antes de escribirse el código y son ejecutadas constantemente antes de las modificaciones del sistema.

*Refactorización (Refactoring):* es una actividad constante de reestructuración del código con el objetivo de eliminar código duplicado, mejorar su legibilidad, simplificarlo y hacerlo más flexible sin cambiar el sistema.

*Programación en Parejas:* Toda programación debe realizarse por parejas de programadores. Esto conlleva a ventajas implícitas.

*Propiedad colectiva de código:* Cualquier programador puede cambiar cualquier parte del código en cualquier momento.

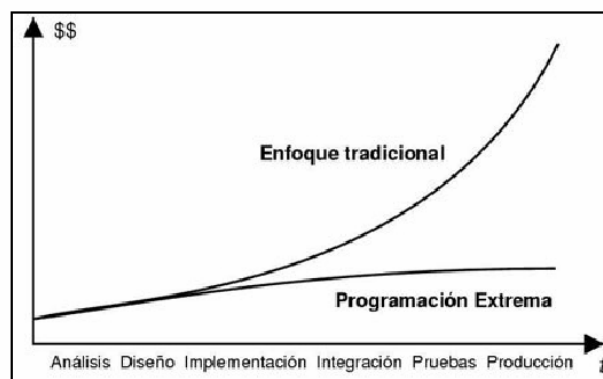
*Integración continua:* Cada pieza de código es integrada y probada en la ejecución del sistema.

*40 horas por semana:* Se deben trabajar 40 horas por semana, no se deben trabajar horas extras en dos semanas seguidas.

*Cliente en el Lugar:* El cliente debe estar presente y disponible todo el tiempo. Es uno de los principales factores de éxito del proyecto.

*Estándares de programación o codificación:* enfatiza la comunicación de los programadores a través del código.

**Figura 1. Comparación Gráfica de XP con el Enfoque tradicional en las etapas de desarrollo (costo vs. tiempo)**



En la Figura 1. Observe que con el enfoque tradicional a medida que el desarrollo del proyecto avanza los costos van aumentando cada vez más, en comparación con la XP que su aumento es poco.

**Figura 2. Grafica del modelo cascada, iterativo y XP**



En la Figura 2. se observa XP es una metodología basada en el modelo cascada de forma iterativa.

#### 1.1.6 Valores Los valores con que cuenta la Programación Extrema:

*Comunicación:* Algunos problemas en los proyectos tienen origen en la comunicación con los usuarios. XP hace casi imposible la falta de comunicación.

Para lograr una buena comunicación es necesario realizar pruebas unitarias, programación por parejas, estimación en las tareas. Estas tres prácticas llevan a que el programador, el cliente y el gerente se comuniquen.

*Simplicidad:* XP propone el principio de hacer la cosa más simple que pueda funcionar y cambiarlo mañana si es necesario, y es mejor hacer algo simple hoy, que hacerlo más complicado hoy y probablemente nunca usarlo.

*Retroalimentación:* La retroalimentación concreta y frecuente del cliente, equipo y usuarios finales, da una mayor oportunidad para dirigir eficientemente.

*Coraje:* El sistema código y pruebas comunica claramente todo lo que necesita ser comunicado en el instante mismo del desarrollo, esto significa que corren todas las pruebas y que el código fuente revela su intención.

#### 1.1.7 Principios Los principios fundamentales se apoyan en los valores y también son cuatro. Se busca una realimentación veloz, modificaciones incrementales, trabajo de calidad y asunción de simplicidad.

## 1.2 PROCESO UNIFICADO RACIONAL (RUP)

**1.2.1 Definición** RUP (Rational Unified Process) es un proceso de Ingeniería de Software cuyo objetivo es producir software de alta calidad, es decir, que cumpla con los requerimientos de los usuarios dentro de una planificación y presupuesto establecidos. Dirigido por casos de uso, centrado en la arquitectura, iterativo (mini-proyectos) e incremental (versiones).

### 1.2.2 Características Las Características Fundamentales de la RUP:

- Aumenta la productividad de los desarrolladores mediante acceso a base de conocimiento, plantillas y herramientas.
- Se centra en la producción y mantenimiento de modelos del sistema más que en producir documentos.
- RUP es una guía de cómo usar UML de la forma más efectiva.
- Existen herramientas de apoyo a todo el proceso:
  - Modelamiento visual, programación, pruebas, etc.
  - Cubre el ciclo de vida de desarrollo de software.

**1.2.3 Prácticas** RUP toma en cuenta las mejores prácticas en el modelo de desarrollo de software en particular las siguientes:

Desarrollo de software en forma iterativa (repite una acción):

- Un proceso iterativo permite una comprensión creciente de los requerimientos a la vez que se va haciendo crecer el sistema.
- RUP sigue un modelo iterativo que aborda las tareas más riesgosas primero.
- Con esto se logra reducir los riesgos del proyecto y tener un subsistema ejecutable tempranamente.

Manejo de requerimientos:

RUP describe cómo obtener los requerimientos, organizarlos, documentar requerimientos de funcionalidad y restricciones, rastrear y documentar decisiones, captar y comunicar requerimientos del negocio.

Utiliza arquitectura basada en componentes:

La arquitectura debe ser: flexible, fácil de modificar, intuitivamente comprensible, promueve la reutilización de componentes.

Modela el software visualmente (Modela con Unified Modeling Language, UML):

- Modelamiento visual de la estructura y el comportamiento de la arquitectura y los componentes.
- Bloques de construcción:
  - Ocultan detalles.
  - Permiten la comunicación en el equipo de desarrollo.
  - Permiten analizar la consistencia.
  - entre las componentes.
  - entre diseño e implementación

Verifica la calidad del software:

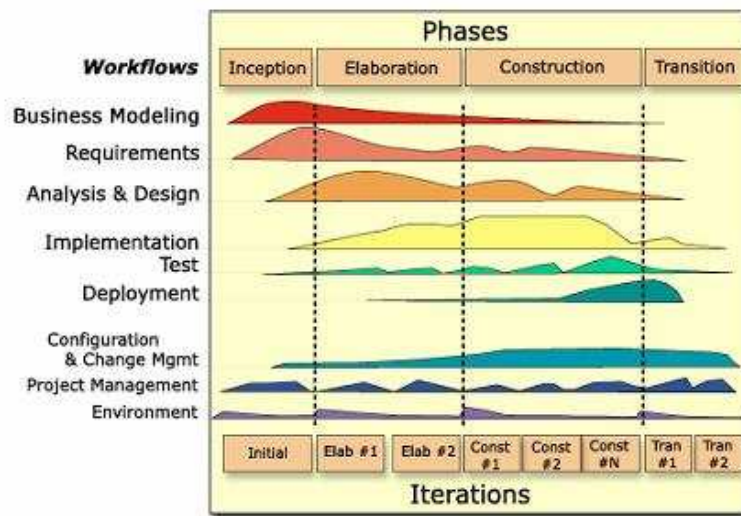
- RUP ayuda a planificar, diseñar, implementar, ejecutar y evaluar pruebas que verifiquen estas cualidades.

Controla los cambios:

- Los cambios son inevitables, pero es necesario evaluar si éstos son necesarios y rastrear su impacto.
- RUP indica como controlar, rastrear y monitorear los cambios dentro del proceso iterativo de desarrollo.

**1.2.4 Roles** RUP Propone los siguientes roles para abarcar las fases de desarrollo, los roles son: administrador de base de datos, líder del proyecto, analistas, diseñador/desarrollador, Tester, administrador de configuración, ingeniero de desempeño o pruebas.

**Figura 3. Fases, Procesos e iteraciones del RUP**



**1.2.5 Fases RUP** esta compuesto por las siguientes fases:

### **Fase de Inicio**

Está principalmente dirigida al entendimiento de los requerimientos y determinar el alcance del esfuerzo de desarrollo. Se define la idea, la visión y el alcance del proyecto. Esta fase incluye la fase de análisis y diseño.

Se identifican todas las entidades externas con las que se trata (actores) y se define la interacción a un alto nivel de abstracción:

- Identificar todos los casos de uso
- Describir algunos en detalle

La oportunidad del negocio incluye:

- Criterios de éxito
- Identificación de riesgos
- Estimación de recursos necesarios
- Plan de las fases incluyendo hitos

En cuanto al producto (Actividades):

- Un documento de visión general:
- Requerimientos generales del proyecto

- Características principales
- Restricciones
- Modelo inicial de casos de uso (10% a 20 % listos).
- Glosario.

Caso de negocio:

- Contexto
- Criterios de éxito
- Pronóstico financiero
- Identificación inicial de riesgos.
- Plan de proyecto.
- Uno o más prototipos.

Las partes interesadas deben acordar el alcance, el tiempo y el costo presupuestado.

### **Fase de elaboración**

Planificar las actividades necesarias y los recursos requeridos, especificando las características y el diseño de la arquitectura del software.

Los objetivos en esta fase son:

- Analizar el dominio del problema
- Establecer una arquitectura base sólida
- Desarrollar un plan de proyecto
- Eliminar los elementos de mayor riesgo para el desarrollo exitoso del proyecto

En cuanto al producto:

- Es la parte más crítica del proceso:
  - Al final toda la ingeniería “dura” está hecha.
  - Se puede decidir si vale la pena seguir adelante.
- A partir de aquí la arquitectura, los requerimientos y los planes de desarrollo son estables.
- Ya hay menos riesgos y se puede planificar el resto del proyecto con menor incertidumbre.
- Se construye una arquitectura ejecutable que contemple:
  - Los casos de uso críticos
  - Los riesgos identificados
- Modelo de casos de uso (80% completo) con descripciones detalladas.
- Otros requerimientos no funcionales o no asociados a casos de uso.

- Descripción de la arquitectura del Software.
- Un prototipo ejecutable de la arquitectura.
- Lista revisada de riesgos y del caso de negocio.
- Plan de desarrollo para el resto del proyecto.
- Un manual de usuario preliminar.

## **Fase de construcción**

Desarrollar el producto y evolucionar la visión, la arquitectura y los planes hasta que el producto en una primera versión esté listo para ser enviado a la comunidad de usuarios.

- En esta fase todas las componentes restantes se desarrollan e incorporan al producto.
- Puede hacerse construcción en paralelo, pero esto exige una planificación detallada y una arquitectura muy estable.

En cuanto al producto:

- El producto de software integrado y corriendo en la plataforma adecuada.
- Manuales de usuario.
- Una descripción del “release” actual.

## **Fase de transición**

Realizar la transición del producto a los usuarios, lo cual incluye: manufactura, envío, entrenamiento, soporte y mantenimiento del producto hasta que el cliente esté satisfecho. Esta fase culmina con la versión de producto, la cual a su vez concluye el ciclo.

Los objetivos de esta etapa son:

- El objetivo principal es traspasar el software desarrollado a la comunidad de usuarios.
- Obtener autosuficiencia de parte de los usuarios.
- Concordancia en los logros del producto de parte de las personas involucradas.
- Lograr el consenso cuanto antes para liberar el producto al mercado.
- Una vez instalado surgirán nuevos elementos que implicarán nuevos desarrollos (ciclos).
- Incluye:

- Pruebas Beta para validar el producto con las expectativas del cliente
- Ejecución paralela con sistemas antiguos
- Conversión de datos
- Entrenamiento de usuarios
- Distribuir el producto

**1.2.6 Procesos Generales** A continuación se describe los procesos generales que conforman la metodología RUP:

### **Requerimientos**

Los desarrolladores y clientes deben acordar qué es lo que el sistema debe hacer relevar requerimientos, documentar funcionalidad y restricciones, documentar decisiones, identificar actores, identificar casos de uso (describen la funcionalidad) y los requerimientos no funcionales se incluyen en una especificación complementaria.

### **Análisis y Diseño**

En este proceso el análisis y diseño se realiza la descripción de cómo se implementará el sistema y se debe ejecutar las tareas y funciones descritas en los casos de uso, satisfacer todos los requerimientos, y estar flexible a cambios.

El diseño se centra en la noción de arquitectura, la cual es necesaria diseñarla y validarla. El modelo de diseño consta de clases estructuradas en paquetes, diseños de subsistemas con interfaces definidas (componentes), forma de colaboración entre las clases.

### **Desarrollo**

Su propósito es producir un producto y hacerlo llegar a sus usuarios finales.

El desarrollo Incluye varias actividades, como: producir un “release”, empaquetar el software, distribuir el software, instalar el software, apoyar a los usuarios, y a veces también se incluyen, realizar pruebas beta migración de datos, aceptación formal, la mayor parte de la distribución ocurre durante la transición.



## **Implementación**

Su propósito es definir la organización del código, implementar clases y objetos en forma de componentes (fuente, ejecutables, etc.), probar las componentes desarrolladas e integrar las componentes en un sistema ejecutable.

## **Prueba**

Su propósito es verificar la interacción entre los objetos, la integración apropiada de componentes, cumplimiento de los requerimientos los requerimientos e identificar los defectos y corregirlos antes de la instalación.

## **2. PROCESOS DE SOFTWARE**

Para el desarrollo de la metodología de software con el apoyo del Grupo de Investigación de Ingeniería de Software de la Universidad Autónoma de Occidente, la metodología es diseñada con base a los recursos y a los procesos que se llevan acabo en el desarrollo de una nueva aplicación.

Es difícil encontrar una metodología que se adapta a las necesidades que la empresa requiere, y pueda abarcar procesos importantes que ayuden a la calidad del software, y es mejor diseñar una metodología que satisfaga las necesidades de la empresa.

Con el grupo de investigación se realizaron reuniones donde se propusieron alternativas para cada uno de los procesos que van a formar parte de la metodología de desarrollo de software.

La metodología es basada en los procesos de desarrollo de software, en estos se describen procedimientos, actividades, tareas, roles, formatos, instructivos, diagramas, estándares y herramientas de software que ayudan a complementar los procesos de software.

Los procesos de software que se consideran son: Requerimientos, Análisis, Diseño, Construcción y Pruebas; para la respectiva documentación de los procesos se baso en el Modelo de Procesos para la Industria del Software (MOPROSOFT), de este modelo decidimos adoptar aspectos importantes de este modelo, realizamos al documentación de los procesos de Requerimientos, Análisis y Diseño, Construcción y Pruebas, además tuvimos en cuenta los estándares tales como el CMMI, IEEE, ISO, Modelo IDEAL propuesto por CMMI, que nos proporciona una idea mas aterrizada acerca de los que se debe hacer en la documentación de los procesos.

### **2.1 REQUERIMIENTOS**

Como primera instancia el proceso que se va a atacar es de requerimientos, este proceso es muy importante debido a que brinda la información necesaria para comenzar en el desarrollo de un software, en este proceso existe una mayor interacción con el cliente.

Las entrevistas son esenciales porque en ellas se identifica procesos, requerimientos y casos de uso que son importantes para un proyecto de desarrollo de software, y además es la técnica de levantamiento de información mas utilizada y con la que se interactúa con el cliente. Para las entrevistas es encuentros con cada cliente para identificar los procesos, requerimientos y casos de uso fundamentales en el desarrollo del proyecto. En el primer encuentro con el cliente se identificarán los procesos(claves) del proyecto y algunos requerimientos y casos de uso que afectaran de manera directa o indirecta el desarrollo del proyecto, en los siguientes dos encuentros con el cliente se enfocara en recolectar los requerimientos y casos de uso de manera detallada.

Un acta de entrevista es buena idea cuando termina una entrevista, pero es mejor fusionarla en la guía de entrevista porque será un formato adicional a la entrevista, dando como solución, agregar 3 campos tales como, resumen, pendientes y resultados, donde el resumen pondría ser requerimientos, procesos o casos de uso que se identificaron en la entrevista, pendientes son actividades que se deben hacer para un mejor entendimiento de los que se va a realizar y resultados son aspectos que se obtienen durante el desarrollo de la entrevista.

La observación de procesos que es una buena técnica de levantamiento de información porque es una fuente de información que permite una mayor claridad de los procesos que se llevan actualmente en los procesos del cliente, detectando fallas e inconsistencias, que con las demás técnicas de levantamiento de información no se harían. Además realizar la observación de procesos solo aquellos que se conspiran críticos para el desarrollo del proyecto.

La revisión de documentos es una técnica sobre leer documentos acerca de Leyes, terminología u otras características para tener un entendimiento de algún proceso y/o requerimiento que se esta trabajando.

La lista de procesos es el documento donde se describen cuales son los procesos que se tienen, y cuales de los son críticos para el proyecto. Un detalle de procesos para los procesos de mayor importancia resaltados en la lista de procesos, basadas en el levantamiento de información y la experiencia del director de desarrollo porque el enfoque de la empresa esta orientado a entender los procesos de sus clientes.

En este proceso se genera la especificación de requerimientos del software (SRS) que contiene toda la descripción de requisitos del software, en este documento se contempla todo el entorno del proceso de requerimientos y sirve como entrada para el proceso de análisis y diseño para llevar a cabo esta actividad nos basamos en el estándar de la IEEE (IEEE-830-1998) este documento nos dice que tenemos que hacer en un SRS.

A continuación se presentara la documentación del proceso de requerimientos realizado en IPSOFT - S.A:

### 2.1.1 Definición general del proceso de requerimientos

**Tabla 1. Definición general del proceso de requerimientos**

<b>Proceso</b>	Requerimientos
<b>Categoría</b>	Desarrollo
<b>Propósito</b>	El propósito del proceso de requerimientos es obtener una lista de requerimientos y casos de uso validados, clasificados y priorizados, para conseguir un entendimiento común entre el cliente y el proyecto.
<b>Descripción</b>	<p>El proceso de requerimientos esta compuesto de las siguientes fases:</p> <ul style="list-style-type: none"> <li>○ Técnicas de Levantamiento de Información: Definición y ejecución de la(s) entrevista(s), revisión de documentos y observación de procesos.</li> <li>○ Especificación de Procesos: Identificar los procesos que componen el proyecto e identificar los procesos centrales.</li> <li>○ Detalle de Procesos: Se detallan los procesos centrales.</li> <li>○ Validación de procesos: se valida el detalle de los procesos.</li> <li>○ Especificación de Requerimientos: De cada proceso se identifica los requerimientos y de los procesos centrales, los requerimientos claves. (Priorización de requerimientos).</li> <li>○ Definición de Casos de uso: Con base en los requerimientos se definen todos los casos de uso.</li> <li>○ Detalle de Casos de uso: Se detallan los casos de uso centrales.</li> <li>○ Validación: Se validan los detalles casos de uso centrales y los requerimientos que no tienen asociado detalle de casos de uso.</li> </ul>
<b>Objetivos</b>	O1 Definir y ejecutar las técnicas de levantamiento de información (entrevistas, revisión de documentos y observación de los procesos).
	O2 Definir los procesos e identificar los procesos centrales.
	O3 Detallar los procesos centrales.
	O4 Definir, especificar, clasificar y priorizar los requerimientos.
	O5 Definir la lista de casos de uso.

	O6 Detallar los casos de uso centrales.
	O7 Validar el detalle de casos de uso centrales y los requerimientos que no tienen asociado detalle de caso de uso.
<b>Responsabilidad y Autoridad</b>	Responsable: <ul style="list-style-type: none"> <li>Analista(s).</li> </ul> Autoridad: <ul style="list-style-type: none"> <li>Director de Desarrollo.</li> </ul>
<b>Procesos Relacionados</b>	Solicitudes de Desarrollo Análisis y Diseño

### 2.1.2 Entradas

**Tabla 2. Entradas - requerimientos**

Nombre	Fuente
Equipo de trabajo	Proyecto
Cronograma	
Solicitud de Ampliación	Proceso de Solicitud de Desarrollo

### 2.1.3 Salidas

**Tabla 3. Salidas - requerimientos**

Nombre	Descripción	Destino
Lista de Procesos	Está compuesta por un identificador de especificación de proceso, el cliente, nombre del proyecto, un identificador, fuente (Numero del documento de levantamiento de información que se utilizo para obtener el proceso), nombre del proceso, objetivo(s), cargo/perfil y el tipo del proceso (Modulo-Submodulo) y un DPR ( <i>campo donde se marca si tiene asociado un detalle del proceso</i> ). (Ver Anexo 3)	Análisis
Detalle de Procesos	Detalle de Procesos esta compuesto por un identificador, un nombre del proceso, objetivo general, objetivos específicos, responsabilidad y autoridad, procesos relacionados, entradas, salidas, roles y actividades.	
Lista de Requerimientos	Está compuesta por un identificador de especificación de requerimientos, cliente, nombre del proyecto, un identificador,	

	<p>identificador del proceso al cual pertenece, número del caso de uso que lo cubre, fuente (Numero del documento de levantamiento de información que se utilizo para obtener el requerimiento) y una descripción del requerimiento.</p> <p><i>Nota: Para identificar los requerimientos de mayor prioridad se escribirán en negrilla.</i></p>	
Lista casos de uso	Está compuesta por un identificador de la lista de casos de uso, cliente, nombre del proyecto, un identificador, nombre del caso de uso y un DCU (campo donde se marca con (X) si tiene asociado un detalle de caso de uso).	
Detalle Casos de uso	Detalle de los casos de uso esta compuesto por un identificador del detalle de casos de uso, cliente, nombre del proyecto, el nombre del caso de uso, un identificador, actor participante, propósito, precondiciones, poscondiciones, referencias cruzadas, flujo principal, subflujos, flujo de excepciones, se detallan por lo menos los centrales.	Diseño
Documento de Validación	El documento de validación esta compuesto por el identificador del documento de validación de casos de uso y requerimientos, cliente, nombre del proyecto, un identificador, observaciones, la fecha en se valida el requerimiento o caso de uso y la firma del cliente.	
Especificación de Requerimiento del Software (SRS)	El SRS esta basado en el estándar de IEEE. IEEE STD 830 1998, IEEE Recommended Practice for Software Requirements Specifications.	<p>Análisis y Diseño</p> <p>Construcción</p>

#### 2.1.4 Productos Internos

**Tabla 4. Productos Internos - requerimientos**

Nombre	Descripción
Entrevista(s)	Entrevistas transcritas, apuntes o grabaciones.
Revisión de Documento(s)	Lectura de documentos que ayuden al entendimiento del proyecto.

Observación de Procesos	Observar el(los) proceso(s) central(es) para tener un mayor entendimiento del proceso.
-------------------------	--

### 2.1.5 Referencias Bibliográficas

**Tabla 5. Referencias Bibliográficas - requerimientos**

Modelo de Procesos para la industria del Software, Moprosoft v1.1, Mayo 2003.
IEEE STD 830 1998, IEEE Recommended Practice for Software Requirements Specifications.
The Capability Maturity Model Integration, CMMI <sup>SM</sup> for Systems Engineering and Software Engineering (CMMI-SE/SW, V1.1) CMMI, Product Team, December 2001.

### 2.1.6. Roles Involucrados

**Tabla 6. Roles Involucrados - requerimientos**

<b>Rol</b>	<b>Capacitación</b>
Analista	Conocimiento y experiencia en la obtención de especificación y análisis de los requerimientos.
Director de Desarrollo	Conocimiento general de los proyectos y experiencia en dirección, planeación y desarrollo de software.
Coordinador de Proyecto	Conocimiento en dirección, planeación y desarrollo de software.
Usuario Final	Conocimiento del proceso y orientación para satisfacer el objetivo de la(s) entrevista(s), revisión de documentos u observación de procesos.
Usuario Líder	Capacidad de toma de decisiones.

### 2.1.7 Actividades

**Tabla 7. Actividades - requerimientos**

<b>Rol</b>	<b>Descripción</b>
A1. Elección de la(s) Técnica(s) de levantamiento de Información (O1)	
Analista	A.1.1 Revisar Solicitud de Ampliación. A.1.2 Definir cuales de las Técnicas de levantamiento de Información (TLI) se requieren ejecutar: <ul style="list-style-type: none"> <li>○ Entrevista(s). (Ver Anexo A, Anexo D)</li> <li>○ Revisión de Documento(s).</li> <li>○ Observación de Proceso(s).</li> </ul>

A2. Realizar Entrevista(O1)	
Analista	<p>A.2.1 Definir la guía de entrevista y/o escoger una guía estándar. (Ver Anexo G)</p> <p>A.2.2 Realizar la ejecución de la entrevista.</p> <p><i>Nota: La grabación de la entrevista es opcional, si hay grabación el cassette deberá ser marcado con el número respectivo de la entrevista.</i></p> <p>A.2.3 Realizar el formato diligenciado de la entrevista.</p>
A3. Realizar Revisión de documentos(O1)	
Analista	<p>A.3.1 Leer los documentos que ayuden al entendimiento del proyecto.</p> <p>A.3.2 Realizar el documento de revisión de documentos para el entendimiento del proyecto.</p>
A4. Realizar observaciones de los procesos(O1)	
Analista	<p>A.4.1 Identificar los procesos que se requieren observar.</p> <p>A.4.2 Realizar el documento de observación de los procesos que sean relevantes para el desarrollo del proyecto.</p>
A5. Realizar la Especificación de Procesos(O2)	
Analista	<p>A.5.1 Identificar los procesos que componen el proyecto e identificar los procesos claves.</p> <p><i>Nota: Para identificar los procesos claves se escribirán en <b>negrilla</b>.</i></p>
A6. Realizar Detalle de Procesos(O3)	
Analista	<p>A.6.2 Elaborar el detalle de los procesos claves del proyecto. (Ver Anexo B, Anexo E)</p> <p><i>Nota: El detalle de proceso debe ser de lo que se quiere conseguir.</i></p>
A7. Validación de Procesos	
Usuario Lider  Analista  Director de Desarrollo	<p>A.7.1 Se validan los detalles de proceso.</p> <p>A.7.2. Se revisa cada detalle de proceso.</p> <p>A.7.3 Si la validación de los procesos es satisfactoria continúa con la actividad A8.</p> <p>A.7.4 De lo contrario, volver a la actividad A5 y revisar y/o modificar los procesos.</p>
A8. Realizar la Especificación de Requerimientos(O4)	
Analista	<p>A.8.1 Definir la lista de requerimientos e identificar los requerimientos de cada proceso.</p> <p>A.8.2 De los procesos claves identificar los requerimientos centrales. (Ver Anexo C, Anexo F)</p> <p><i>Nota: Para identificar los requerimientos de mayor</i></p>



	<p><i>prioridad se escribirán en <b>negrilla</b>.</i></p> <p>Las características de un requerimiento son sus propiedades principales. A continuación se presentan las más importantes:</p> <ul style="list-style-type: none"> <li>○ Necesario: Un requerimiento es necesario si su omisión provoca una deficiencia en el sistema a construir, y además su capacidad, características físicas o factor de calidad no pueden ser reemplazados por otras capacidades del producto o del proceso.</li> <li>○ Conciso: Un requerimiento es conciso si es fácil de leer y entender. Su redacción debe ser simple y clara para aquellos que vayan a consultarlo en un futuro.</li> <li>○ Completo: Un requerimiento está completo si no necesita ampliar detalles en su redacción, es decir, si se proporciona la información suficiente para su comprensión.</li> <li>○ Consistente: Un requerimiento es consistente si no es contradictorio con otro requerimiento.</li> <li>○ No ambiguo: Un requerimiento no es ambiguo cuando tiene una sola interpretación. El lenguaje usado en su definición, no debe causar confusiones al lector.</li> <li>○ Verificable: Un requerimiento es verificable cuando puede ser cuantificado de manera que permita hacer uso de los siguientes métodos de verificación: inspección, análisis, demostración o pruebas.</li> </ul> <p>Primero que todo, antes de especificar el requerimiento nos debemos hacer una serie de preguntas como por ejemplo ¿mi sistema puede funcionar sin este requerimiento o funcionalidad? o ¿existe alguna otra forma de hacerlo con las funcionalidades actuales del sistema?, es decir, debemos estar seguros de que el requerimiento cumpla con la primera y mas importante característica de un requerimiento, que sea necesario.</p> <p>Para redactar los requerimientos, no existe una formula milagrosa para hacerlo y los siguientes</p>
--	--

	<p>modelos de redacción tampoco son una camisa de fuerza, pero son una guía que nos puede ser muy útil.</p> <p><b>MODELO REDACCIÓN DE REQUERIMIENTOS</b>  <b>“NUEVOS”</b>  [LUGAR, TIEMPO, EVENTO, OBJETO] + “debe, deberá, no debe, no deberá” + [ACCIÓN, VERBO, SENTENCIA]+ [A QUIEN, SUJETO] + [RESULTADO, CONSECUENCIA]</p> <p><u>Algunos ejemplos para el modelo:</u>  “El sistema en el módulo de ventas debe ofrecer una opción mediante la cual el usuario pueda ver una comparación de lo presupuestado versus la venta real en un rango de fechas”</p> <p>“Cuando el tiempo de conexión exceda el valor predeterminado como máximo, el sistema debe cancelar la sección informándole a el usuario que el tiempo se agotó y que por lo tanto será desconectado”</p> <p><b>MODELO REDACCIÓN DE REQUERIMIENTOS</b>  <b>“ERRORES, MODIFICACIÓN Y/O MEJORAS”</b>  [OPCIÓN, MODULO, EVENTO, OBJETO]+ [CONDICIONES, PARAMETROS, AMBIENTE]+[RESULTADO ACTUAL, ERROR, ESTADO] + [ACCIÓN, (CORREGIR, MODIFICAR)]+ [RESULTADO ESPERADO, DESTINO]</p> <p><u>Algunos ejemplos para el modelo:</u>  “En el reporte de mercados objetivos, al aplicarle filtros por asesor y zona, muestra la información en desorden. Se debe corregir de tal forma que salga ordenado por número de cédula del asesor y agrupado por zona”</p> <p><u>Tengamos en cuenta siempre:</u></p> <ul style="list-style-type: none"> <li>○ Mantener sentencias y párrafos CORTOS.</li> <li>○ Escribir sentencias completas, con una apropiada gramática, ortografía y puntuación.</li> <li>○ No olvidar utilizar la sentencia “El sistema debe”, “deberá”, seguidas de una acción y finalmente un resultado.</li> <li>○ Para reducir ambigüedades evite términos subjetivos como: Amigable, fácil, simple, rápido, eficiente, soporte, fuerte, superior,</li> </ul>
--	---

	<p>aceptable, robusto.</p> <ul style="list-style-type: none"> <li>○ Para reducir malas interpretaciones omita palabras como: Máximo, mínimo, optimizable.</li> </ul> <p>Recuerde que los requerimientos deben cumplir con ciertas características por favor revíselas para cerciorarse de tener un requerimiento válido.</p>
A9. Definición de los Casos de Uso(O5)	
Analista Coordinador del Proyecto	A.9.1 Elaborar la lista de casos de uso basada en la especificación de requerimientos.
A10. Realizar Detalle de Casos de Uso(O6,O7)	
Analista	A.10.1 Elaborar el detalle de casos de usos centrales del proyecto.
A11. Validación de Casos de Uso y Requerimientos (O7)	
Usuario Líder  Analista  Director de Desarrollo  Coordinador de proyecto	<p>A.11.1 Validar los detalles de casos de uso centrales del proyecto. (Val1)</p> <p><i>Nota: Si el detalle de caso de uso no pasa la validación entonces se procederá a validar los requerimientos que están asociados con el caso de uso.</i></p> <p>A.11.2 Validar los requerimientos que no tiene detalle de caso de uso asociado.</p> <p>A.11.3 Se validará cada caso de uso o requerimiento con fecha y firma del cliente, siempre y cuando se apruebe.</p> <p>A.11.4 Los requerimientos que no fueron aprobados se harán las observaciones respectivas.</p> <p>A.11.5 Con este mismo documento los casos de uso o requerimientos que no fueron aprobados, se debe decidir en que actividad o actividades del proceso de levantamiento de requerimientos desea revisar para su modificación.</p> <p>A.11.6 Terminado las modificaciones, deberán ser validados nuevamente con el cliente.</p> <p><i>Nota: En el diagrama de actividades estará contemplado en el peor de los casos “realizar y ejecutar una técnica de levantamiento de información”</i></p>
12. Diligenciar el documento de Especificación de Requerimientos del	

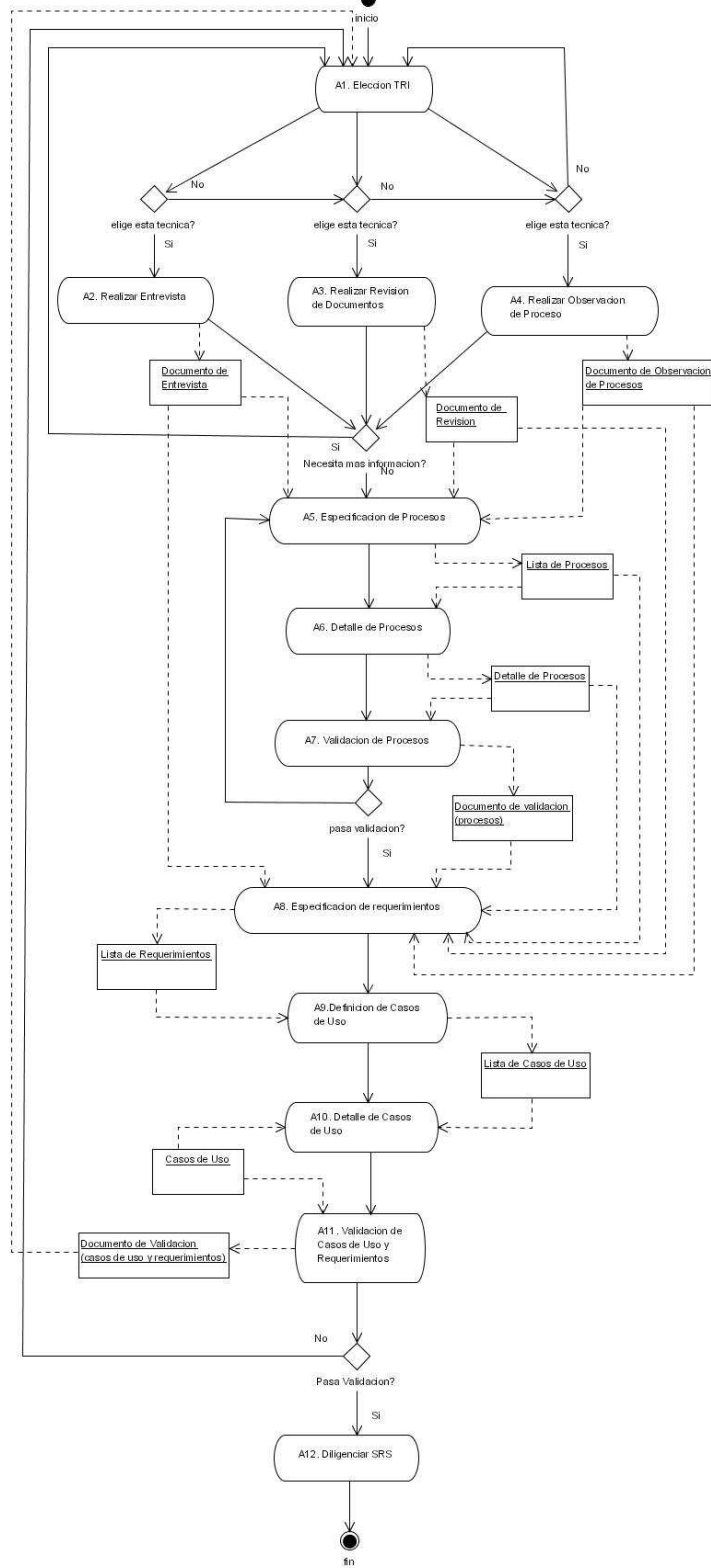
Software(SRS)	
Analista	A.12.1 Determinar si el SRS se va a diligenciar o modificar. A.12.2 Diligenciar el documento de Especificación de Requerimientos del Software (SRS).

### 2.1.8 Verificaciones y/o Validaciones

**Tabla 8. Verificaciones y/o Validaciones - requerimientos**

Actividad	Producto	Rol	Descripción
A7(Val1)	Documento de Validación de procesos	Usuario Líder Analista Director de Desarrollo Director de Proyecto	Validar que el detalle de los procesos cumple con las necesidades y expectativas del usuario líder y el usuario final.
A11(Val1)	Documento de Validación de casos de uso y requerimientos	Usuario Líder Analista Director de Desarrollo Director de Proyecto	Validar que el detalle de casos de uso centrales y los requerimientos que no tienen detalle de casos de uso, cumple con las necesidades y expectativas acordadas por el usuario líder y el usuario final.

**Figura 4. Diagrama de actividades del proceso de requerimientos**



## 2.2 ANÁLISIS Y DISEÑO

En el proceso de análisis y diseño es fundamental analizar los diagramas que brinda UML para la documentación del software, y además seleccionar que tipo de diagramas se puedan utilizar y ayuden verdaderamente para el desarrollo del software. Es importante destacar la utilización de estándares (CMMI, ISO, IEEE) que aporte una orientación para el diseño de un buen proceso de análisis y diseño.

El diseño de la arquitectura de software y el modelo de datos es muy importante verificar ya que son elementos esenciales que deben ser utilizados en casi todos los desarrollos de software.

La verificación y validación de los productos en este proceso permite una confiabilidad para continuar con el siguiente proceso.

### 2.2.1 Definición general del proceso de Análisis y Diseño

**Tabla 9. Definición general del proceso de análisis y diseño**

<b>Proceso</b>	Análisis y Diseño
<b>Categoría</b>	Operación
<b>Propósito</b>	El propósito del proceso de análisis y diseño es modelar y definir los componentes del sistema para obtener un entendimiento del sistema y consistencia entre el proceso de levantamiento de requerimientos y el proceso de análisis y diseño.
<b>Descripción</b>	<p>El proceso de análisis y diseño está compuesto de las siguientes fases:</p> <ul style="list-style-type: none"><li>○ Análisis: Se analiza la especificación de requerimientos y los casos de uso a ser diseñados.<ul style="list-style-type: none"><li>▪ Modelo de negocio: Se realiza el diagrama de objetos y el diagrama de casos de uso para entender el modelo de negocio.</li></ul></li><li>○ Pool de Arquitecturas: Se escoge una alternativa de arquitectura a utilizar y se modifica de acuerdo a la especificación de requerimientos y</li></ul>

	<p>el modelo de negocio.</p> <ul style="list-style-type: none"> <li>○ Arquitectura: Se realizan y/o modifican los diagramas de despliegue, diagramas de paquetes y diagrama de componentes, identificando sus componentes a utilizar.</li> <li>○ Diseño detallado: Se realizan y/o modifican el diagrama de clases detallado a partir del diagrama de clases del modelo de negocio y los diagramas de secuencia.</li> <li>○ Modelo de datos: Se realiza y/o modifica el modelo de datos de acuerdo al modelo del negocio. El modelo de datos esta compuesto por Modelo Entidad Relación (MER), Modelo Relacional de Datos (MRD) y Componentes de Base de Datos.</li> <li>○ Verificación y Validación: Se verifican y validan los documentos del proceso de análisis y diseño.</li> </ul>
<b>Objetivos</b>	O1 Definir el conjunto de requerimientos y casos de uso que serán considerados en el análisis y diseño.
	O2 Realizar y/o modificar los diagramas que componen el modelo de negocio.
	O3 Seleccionar y/o refinar la arquitectura de software.
	O4 Realizar los diagramas que componen la arquitectura de software.
	O5 Realizar y/o modificar los diagramas de diseño detallado de acuerdo al conjunto de requerimientos y casos de uso definidos.
	O6 Realizar y/o modificar el modelo de datos.
	O7 Realizar y/o modificar el diseño de los componentes de base de datos.
	O8 Realizar y/o modificar el documento de verificación y validación.
<b>Responsabilidad y Autoridad</b>	<p>Responsable:</p> <ul style="list-style-type: none"> <li>• Analista(s).</li> </ul> <p>Autoridad:</p> <ul style="list-style-type: none"> <li>• Director de Desarrollo.</li> </ul>
<b>Procesos</b>	Requerimientos

<b>Relacionados</b>	Construcción y Mantenimiento Pruebas
---------------------	---

### 2.2.2 Entradas

**Tabla 10 Entradas – análisis y diseño**

<b>Nombre</b>	<b>Fuente</b>
Listado Requerimientos	Proceso de Requerimientos
Listado Casos de Uso	
Detalle de Casos de Uso	
SRS	
Productos de trabajo de análisis y diseño existentes.	Proceso de Análisis y Diseño

### 2.2.3 Salidas

**Tabla 11. Salidas – análisis y diseño**

<b>Nombre</b>	<b>Descripción</b>	<b>Destino</b>
Modelo de Negocio	El documento del Modelo de Negocio está compuesto por el id, fecha, cliente, proyecto, autor. Información del conjunto de requerimientos y casos de uso que son, identificación del documento de requerimientos, numero del requerimiento, identificación del documento de casos de uso, numero del caso de uso y la identificación del documento del SRS. El diagrama de objetos del modelo de negocio y el diagrama de casos de uso de acuerdo al conjunto de requerimientos y casos de uso definidos.	Proceso Construcción
Documento de Arquitectura de Software	El documento de arquitectura de software está compuesto por el id, fecha, id_pool (identificador del pool de arquitecturas, si se requiere), cliente, proyecto, autor, el diagrama de paquetes, diagrama de despliegue y el diagrama de componentes. (Ver Anexo H)	
Documento de Diseño Detallado	El documento de diseño detallado está compuesto por el id, fecha, cliente, proyecto, autor, el diagrama de clases detallado de acuerdo a la arquitectura de	



	software, al conjunto de requerimientos y casos de uso definidos y los diagramas de secuencia por cada caso de uso.	
Modelo de Datos	<p>El Modelo de Datos contiene:</p> <ul style="list-style-type: none"> <li>• Modelo Entidad Relación (MER) (entidades, atributos y relaciones).</li> <li>• Modelo Relacional de datos (MRD), basado en el diagrama de clases detallado y los componentes de la base de datos (triggers, procedimientos almacenados).</li> </ul>	

#### 2.2.4 Productos Internos

**Tabla 12. Productos Internos – análisis y diseño**

Nombre	Descripción
Pool de Arquitecturas	Modelos de arquitecturas con sus componentes ya definidos para ser utilizados en el proceso de análisis y diseño.
Documento de Verificación y Validación	Se verifican y validan los documentos del proceso de análisis y diseño de acuerdo al documento especificado.
Documento de Archivos de Configuración o de Interfase	Se especifican las características de los archivos que son necesarios para el buen funcionamiento del software.

#### 2.2.5 Referencias Bibliográficas

**Tabla 13. Referencias Bibliográficas – análisis y diseño**

Modelo de Procesos para la industria del Software, Moprosoft v1.1, Mayo 2003.
The Capability Maturity Model Integration, CMMI <sup>SM</sup> for Systems Engineering and Software Engineering (CMMI-SE/SW, V1.1) CMMI, Product Team, December 2001.

#### 2.2.6 Roles Involucrados

**Tabla 14. Roles Involucrados – análisis y diseño**

Rol	Capacitación
Analista	Conocimiento y experiencia en ingeniería de

	software, elaboración de diagramas UML y Modelo Entidad Relación.
Director de Desarrollo	Conocimiento general de los proyectos y experiencia en dirección, planeación y desarrollo de software.

## 2.2.7 Actividades

**Tabla 15. Actividades – análisis y diseño**

<b>Rol</b>	<b>Descripción</b>
<b>A1. Análisis(O1,O2)</b>	
Analista	<p>A.1.1 Revisar la especificación de requerimientos.</p> <p>A.1.2 Revisar los casos de uso.</p> <p>A.1.3 Definir el conjunto de requerimientos y casos de uso que serán considerados en el diseño.</p> <p>A.1.4 Realizar el documento del modelo del negocio:</p> <p style="padding-left: 40px;">A.1.4.1 Realizar el diagrama de casos de uso. Los casos de uso se van a asociar en paquetes los cuales serán módulos del sistema.</p> <p style="padding-left: 40px;">A.1.4.2 Realizar el diagrama de objetos del modelo de negocio.</p>
<b>A2. Realizar y/o modificar la Arquitectura de Software (O3,O4)</b>	
Analista	<p>A.2.1 Seleccione un modelo de arquitectura de software del pool de arquitecturas de acuerdo a los siguiente criterios:</p> <ul style="list-style-type: none"> <li><input checked="" type="checkbox"/> Modular</li> <li><input checked="" type="checkbox"/> Claro</li> <li><input checked="" type="checkbox"/> Simple</li> <li><input checked="" type="checkbox"/> Mantenible</li> <li><input checked="" type="checkbox"/> Verificable</li> <li><input checked="" type="checkbox"/> Portable</li> <li><input checked="" type="checkbox"/> Confiable</li> <li><input checked="" type="checkbox"/> Exacto</li> <li><input checked="" type="checkbox"/> Seguro</li> <li><input checked="" type="checkbox"/> Escalable</li> <li><input checked="" type="checkbox"/> Utilizable</li> </ul> <p><i>(Ver Anexo H)</i></p> <p>A.2.2 Si se selecciona un modelo de arquitectura del pool de arquitecturas, identifique en el formato en el campo (<i>ID_POOL</i>) cual modelo de arquitectura selecciono de acuerdo a su numero de identificación</p>

	<p>y ajústelo al modelo de negocio:</p> <p>A.2.2.1 Ajuste el diagrama de paquetes.</p> <p>A.2.2.2 Ajuste el diagrama de despliegue.</p> <p>A.2.2.3 Ajuste el diagrama de componentes.</p> <p>A.2.3 De lo contrario, debe realizar el documento de arquitectura de software:</p> <p>A.2.2.1 Realizar y/o modificar el diagrama de paquetes.</p> <p>A.2.2.2 Realizar y/o modificar el diagrama de despliegue.</p> <p>A.2.2.3 Realizar y/o modificar el diagrama de componentes.</p> <p>A.2.4 Anexar el nuevo documento de arquitectura generado al pool de arquitecturas de software y renombrarlo como un Anexo Técnico del proceso.</p>
<b>A3. Verificar el Diseño de Arquitectura de Software(O8)</b>	
Analista	<p>A.3.1 Verificar el documento de Arquitectura del software. (Ver1)</p> <p>A.3.2 Realizar y/o modificar Documento de Verificación y Validación de acuerdo al listado de criterios definidos.</p> <p>Si la verificación es satisfactoria se realiza A4, de lo contrario vuelve a A2.</p>
<b>A4. Validar el Diseño de Arquitectura de Software(O8)</b>	
Director de Desarrollo	<p>A.4.1 Validar el documento de Arquitectura software. (Val1)</p> <p>A.4.2 Realizar y/o modificar Documento de Verificación y Validación de acuerdo al listado de criterios definidos.</p> <p>Si la validación es satisfactoria se realiza A5, de lo contrario vuelve a A2.</p>
<b>A5. Realizar y/o modificar el Diseño Detallado(O5)</b>	
Analista	<p>A.5.1 Realizar y/o modificar el diagrama de clases detallado a partir del diagrama de clases del modelo de negocio y la arquitectura de software definida.</p> <p>A.5.2 Realizar y/o modificar los diagramas de secuencia basado en el diagrama de clases detallado para identificar los objetos que interactúan entre ellos por cada detalle de caso de uso.</p> <p>A.5.2.1 Ajustar el diagrama de clases detallado basado en los diagramas de secuencia.</p> <p>A.5.3 Definir si se necesita archivos de configuración o de interfase.</p>

	<p>A.5.3.1 Realizar y/o modificar el diseño de los archivos de configuración o de interfase definidos.</p> <p>A.5.3.2 Anexar el documento de archivos de configuración o de interfase al documento de diseño detallado.</p>
<b>A6. Verificar el Diseño Detallado(O8)</b>	
Analista	<p>A.6.1 Verificar el documento de diseño detallado. (Ver2)</p> <p>A.6.2 Realizar y/o modificar Documento de Verificación y Validación de acuerdo al listado de criterios definidos.</p> <p>Si la verificación es satisfactoria se realiza A7, de lo contrario vuelve a A5.</p>
<b>A7. Validar el Diseño Detallado(O8)</b>	
Director de Desarrollo	<p>A.7.1 Validar el documento de diseño detallado. (Val2)</p> <p>A.7.2 Realizar y/o modificar Documento de Verificación y Validación de acuerdo al listado de criterios definidos.</p> <p>Si la validación es satisfactoria se realiza A5, de lo contrario vuelve a A5.</p>
<b>A8. Realizar y/o modificar el Modelo de Datos(O6,O7)</b>	
Analista	<p>A.8.1 Revisar el diagrama de clases del modelo de negocio.</p> <p>A.8.2 Identificar las clases persistentes del diagrama de clases del modelo de negocio.</p> <p>A.8.3 Realizar y/o modificar el modelo de datos con sus respectivas entidades, atributos y relaciones.</p> <p>A.8.4 Realizar y/o modificar el MRD de acuerdo al MER para cada entidad.</p> <p>A.8.4.1 Identificar la entidad.</p> <p>A.8.4.2 Definir la tipo y longitud, cardinalidad, dominio y claves de cada atributo para cada entidad.</p> <p>A.8.5 Realizar y/o modificar el diseño de los Componentes de Bases de Datos (O6)</p> <p>A.8.5.1 Revisar el MER y MRD</p> <p>A.8.5.2 Identificar los elementos (tablas y</p>

	<p>atributos) en el cual se realizara el Triggrrer o Procedimientos Almacenados.</p> <p>A.8.5.3 Realizar y/o modificar Triggers basados en los elementos identificados anteriormente.</p> <p>A.8.5.4 Realizar y/o modificar procedimientos almacenados basados en los elementos identificados anteriormente.</p>
<b>A9. Verificar el Modelo de Datos(O8)</b>	
Director de Desarrollo	<p>A.9.1 Verificar el Modelo de Datos (Ver3)</p> <p>A.9.2 Realizar y/o modificar Documento de Verificación y Validación de acuerdo al listado de criterios definidos.</p> <p>Si la verificación es satisfactoria se realiza A10, de lo contrario vuelve a A8.</p>
<b>A10. Validar el Modelo de Datos(O8)</b>	
Director de Desarrollo	<p>A.10.1 Validar el Modelo de Datos (Val3)</p> <p>A.10.2 Realizar y/o modificar Documento de Verificación y Validación de acuerdo al listado de criterios definidos.</p> <p>Si la validación es satisfactoria, termina el proceso, de lo contrario vuelve a A8.</p>

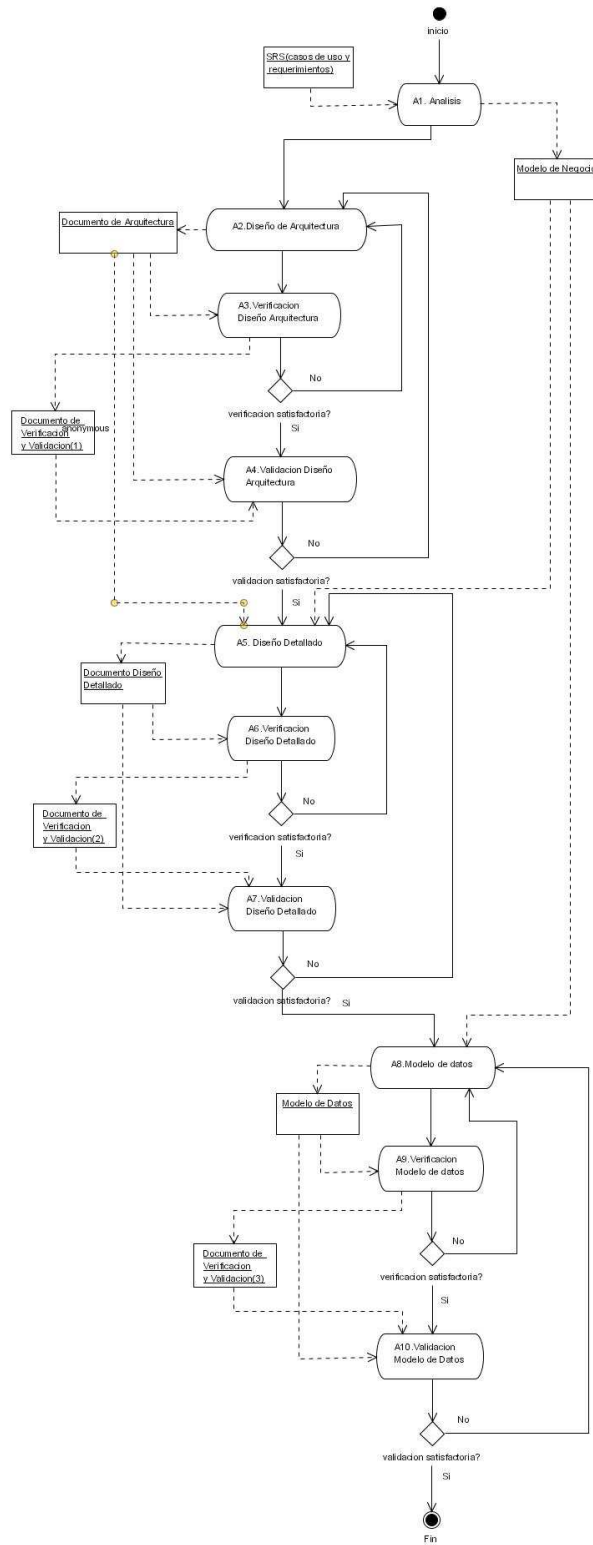
## 2.2.8 Verificaciones y/o Validaciones

**Tabla 16. Verificaciones y/o Validaciones – análisis y diseño**

<b>Actividad</b>	<b>Producto</b>	<b>Rol</b>	<b>Descripción</b>
A6(Ver1)	Documento de Arquitectura	Analista	Verificar que el documento de Arquitectura de software cubre todo el conjunto de requerimientos y casos de uso definidos, y el modelo de negocio.
A7(Val1)	Documento de Arquitectura	Director de Desarrollo	Validar que el documento de Arquitectura de software cumple con todo el conjunto de requerimientos y casos de uso definidos, y el modelo de

			negocio.
A9(Ver2)	Documento de Diseño Detallado	Analista	Verificar que el documento diseño detallado cubre todo el conjunto de requerimientos y casos de uso definidos.
A10(Val2)	Documento de Diseño Detallado	Director de Desarrollo	Validar que el documento diseño detallado cumple con todo el conjunto de requerimientos y casos de uso definidos.
A3(Ver3)	Modelo de Datos	Analista	Verificar el Modelo de datos cubre todo el conjunto de requerimientos y casos de uso definidos, y el modelo de negocio.
A4(Val3)	Modelo de Datos	Director de Desarrollo	Validar que el MER cumple con todo el conjunto de requerimientos y casos de uso definidos, y el modelo de negocio.

**Figura 5. Diagrama de actividades del proceso de análisis y diseño**



## 2.3 CONSTRUCCIÓN

En el proceso de construcción se observa el entorno en el que se va a desarrollar la aplicación, y la distribución de las tareas dentro del equipo de desarrollo y la utilización de herramientas de software para el desarrollo de la aplicación, estos son aspectos importantes que se debe tener en cuenta antes de comenzar a construir el software.

El desarrollo de los scripts de la base de datos y los componentes de software constituyen la parte principal de este proceso, que es el que permite transformar los requerimientos, casos de usos y diagramas (UML) en unidades de código, además, es fundamental hacer pruebas unitarias ya que permite realizar una verificación del código que se ha construido.

Algo importante es especificar la utilización de herramientas que permita el control de las versiones de los componentes de software desarrollados o modificados (si existen) para que no haya confusiones en la implantación del software.

A continuación se presenta la documentación del proceso de construcción en IPSOFT S.A:

### 2.3.1 Definición general del proceso de construcción

**Tabla 17. Definición general del proceso de construcción**

<b>Proceso</b>	Construcción y Mantenimiento
<b>Categoría</b>	Operación
<b>Propósito</b>	El propósito del proceso de construcción es desarrollar una aplicación estable, funcional, mantenible a las necesidades del cliente, basado en los productos del proceso de análisis y diseño.
<b>Descripción</b>	<p>El proceso de construcción contará con las siguientes fases para su desarrollo:</p> <ul style="list-style-type: none"><li>○ Entorno de trabajo: Revisar que los componentes físicos descritos en la arquitectura de software, estén disponibles para el desarrollo de la aplicación.</li><li>○ Distribución del trabajo: Revisar la distribución del trabajo de acuerdo al plan de desarrollo.</li><li>○ Base de datos: Realizar y/o modificar la base de datos de acuerdo al MRD y generar scripts.</li><li>○ Componentes de Software: se construyen y/o modifican los componentes de software basado en la <i>documentación de la política de</i></li></ul>



	<p><i>desarrollo.</i></p> <ul style="list-style-type: none"> <li>○ Pruebas unitarias: realizar y aplicar pruebas unitarias a los componentes de software desarrollados y/o modificados.</li> <li>○ Documentación técnica: se realiza la documentación correspondiente a cada componente de software.</li> <li>○ Sistema de Control de Versiones (CVS): Entregar los componentes desarrollados y/o modificados al CVS.</li> <li>○ Reporte de actividades: Reportar las actividades realizadas.</li> </ul>
<b>Objetivos</b>	O1 Revisar el ambiente de trabajo para la aplicación.
	O2 Revisar el Plan de Desarrollo para la distribución del trabajo.
	O3 Realizar y/o modificar la base de datos.
	O4 Construir y/ o modificar componentes de software.
	O5 Realizar y aplicar pruebas unitarias.
	O6 Realizar y/o modificar documentación técnica.
	O7 Actualizar los componentes de software en el CVS.
	O8 Reportar las actividades realizadas.
	O9 Actualizar las actividades en el <i>Plan de Desarrollo</i> .
<b>Responsabilidad y Autoridad</b>	Responsable: <ul style="list-style-type: none"> <li>• Analista Programador.</li> </ul> Autoridad: <ul style="list-style-type: none"> <li>• Director de Desarrollo.</li> </ul>
<b>Procesos Relacionados</b>	Requerimientos Análisis y Diseño Pruebas

### 2.3.2 Entradas

Tabla 18. Entradas – construcción

Nombre	Fuente
Ambiente de Trabajo	Administración de Proyectos
	Administración de Recursos
SRS(Casos de Uso y Requerimientos)	Proceso de Requerimientos
Arquitectura de Software	Proceso de Análisis y Diseño
Diseño Detallado	
MRD	

### 2.3.3 Salidas

**Tabla 19. Salidas - construcción**

<b>Nombre</b>	<b>Descripción</b>	<b>Destino</b>
Componentes de Software	Componentes (archivos fuente, ejecutables, clases) necesarios para la ejecución de la aplicación.	Proceso de Pruebas
Scripts de Base de Datos	Archivos que contienen sentencias SQL para modificar la BD (tablas, procedimientos, restricciones, vistas, funciones, triggers, secuencias)	

### 2.3.4 Productos Internos

**Tabla 20. Productos Internos – construcción**

<b>Nombre</b>	<b>Descripción</b>
Templates de clases	Plantillas estándares utilizados en el desarrollo de la aplicación.
Pruebas Unitarias	Unidades de código (clases) realizadas para probar los componentes de software.
Pruebas de Integración	Unidades de código (clases) realizadas para probar los componentes de software.
Documentación técnica	Documentación técnica de los componentes de software.

### 2.3.5 Referencias Bibliográficas

**Tabla 21. Referencias Bibliográficas - construcción**

Modelo de Procesos para la industria del Software, Moprosoft v1.1, Mayo 2003.
The Capability Maturity Model Integration, CMMI <sup>SM</sup> for Systems Engineering and Software Engineering (CMMI-SE/SW, V1.1) CMMI, Product Team, December 2001.

### 2.3.6 Roles Involucrados

**Tabla 22. Roles Involucrados - construcción**

<b>Rol</b>	<b>Capacitación</b>
Analista	Conocimiento y experiencia en ingeniería de software y elaboración de diagramas UML.
Analista Programador	Conocimiento y/o experiencia en programación de software, integración y pruebas unitarias.

Director de Desarrollo	Conocimiento general de los proyectos y experiencia en dirección, planeación y desarrollo de software.
Tester	Conocimiento y experiencia en la elaboración y ejecución en pruebas de software.

### 2.3.7 Actividades

**Tabla 23. Actividades - construcción**

<b>Rol</b>	<b>Descripción</b>
<b>A1. Revisar entorno de trabajo (O1)</b>	
Director de Desarrollo	<p>A.1.1 Verificar el espacio físico donde será desarrollada la aplicación y que esté acorde con el documento de arquitectura especificado.</p> <p>A.1.2 Si la verificación NO es satisfactoria, se debe adecuar el ambiente de trabajo.</p>
<b>A2. Realizar la distribución del trabajo(O2)</b>	
Director de Desarrollo	A.2.1 Distribuir el trabajo al personal de desarrollo de acuerdo al Plan de Desarrollo.
<b>A3. Realizar y/o modificar la base de datos(O3)</b>	
Analista Programador	<p>A.3.1 Identificar SQLs (Tablas, Triggers, Procedimientos, Vistas, Secuencias) existentes en la base de datos.</p> <p>A.3.2 Realizar y/o modificar SQLs (Tablas, Triggers, Procedimientos, Vistas, Secuencias) de la base de datos.</p> <p>A.3.3 Entregar Scripts SQLs al administrador de la base de datos de desarrollo.</p>
<b>A4. Construir y/o modificar Componente(s) de Software(O4)</b>	
Analista Programador	A.4.1 Construir y/o modificar Componente(s) de software con base en el documento de Análisis y Diseño (diagramas de secuencia) y la SRS (casos de uso y requerimientos).
<b>A5. Pruebas Unitarias(O5)</b>	
Analista Programador	A.5.1 Definir, construir y/o modificar pruebas unitarias para verificar el correcto funcionamiento de cada componente.
<b>A6. Ejecutar Pruebas unitarias(O5)</b>	
Analista Programador	<p>A.6.1 Ejecutar la pruebas unitarias a los componentes de software.</p> <p>A.6.2 Realizar y/o modificar el reporte de pruebas</p>

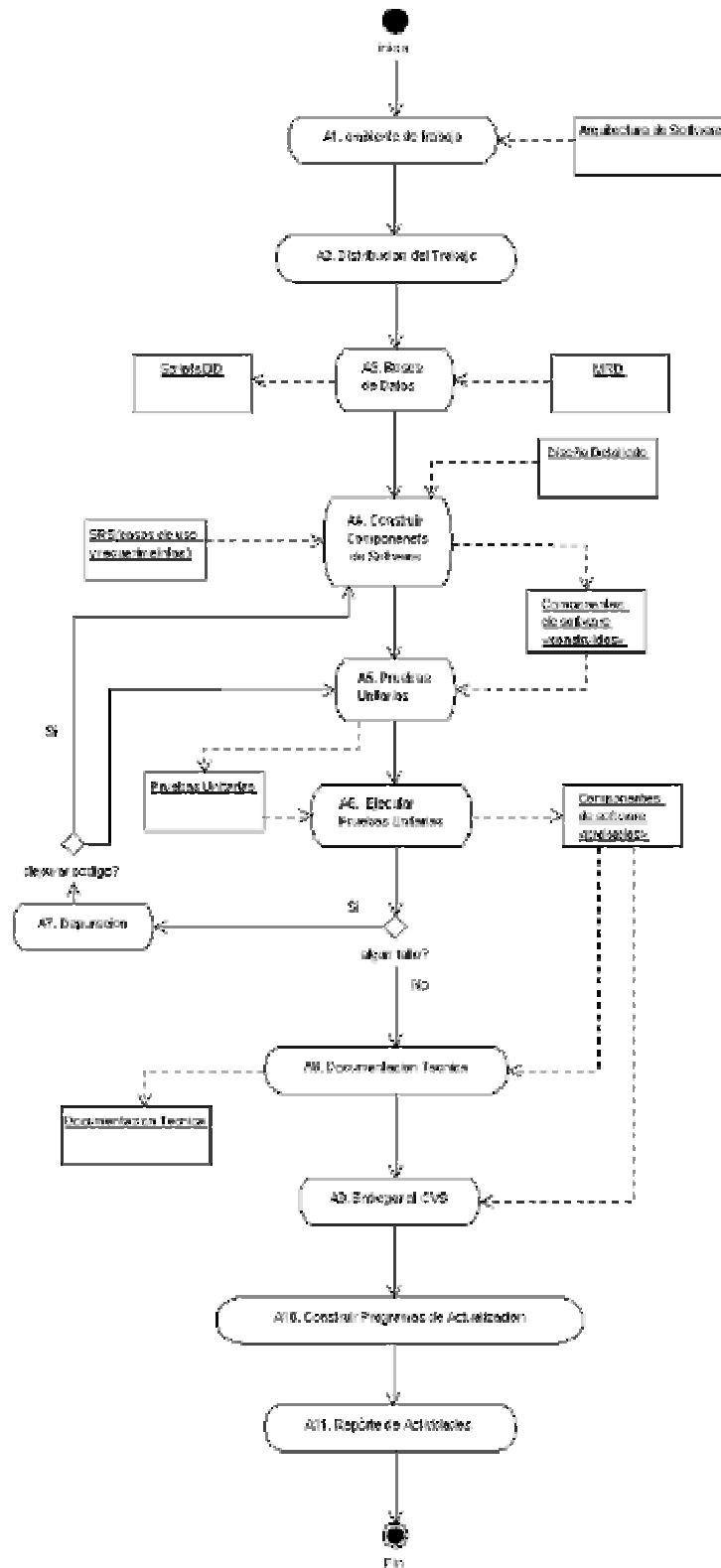
	unitarias.  A.6.3 De acuerdo al reporte de la ejecución de la prueba unitaria afinar el componente de software hasta que pase la prueba.
A7. Depuración(O5)	
Tester	A.7.1 Localizar la causa del error: <ul style="list-style-type: none"> <li>○ En la prueba</li> <li>○ En el código</li> </ul> A.7.2 Analizar el error. A.7.3 Corregir el error.
A8. Documentación técnica(O6)	
Analista Programador	A.8.1 Realizar y/o modificar la documentación técnica de los componentes de software.
A9. Entregar al CVS(O7)	
Analista Programador	A.9.1 Entregar al CVS los nuevos componentes de software desarrollados y/o modificados.
A10. Construir Programas de Actualización(O4)	
Analista Programador	A.10.1 Construir programas de instalación y/o de actualización de aplicación y base de datos. A.10.2 Entregar programas de instalación y/o actualización al administrador de aplicaciones.
A11. Reporte de actividades(O8,O9)	
Analista Programador	A.11.1 Reportar las actividades realizadas utilizadas en la construcción de los componentes de software. A.11.2 Actualizar las actividades realizadas de los componentes de software en el <i>Plan de Desarrollo</i> .

### 2.3.8 Verificaciones y/o Validaciones

**Tabla 24. Verificaciones y/o Validaciones - construcción**

Actividad	Producto	Rol	Descripción
A1(Ver1)	Documento de Verificación del Ambiente de Trabajo	Director de Desarrollo	Verificar que el ambiente de trabajo este acorde con los requerimientos técnicos (servidor de base de datos, servidor de aplicaciones, sistema operativo) y la arquitectura de software.

Figura 6. Diagrama de actividades del proceso de construcción



## 2.4 PRUEBAS

A continuación se presenta la documentación del proceso de pruebas en IPSOFT S.A:

### 2.4.1 Descripción General del Proceso de Pruebas

Tabla 25. Descripción General del Proceso de Pruebas

<b>Proceso</b>	Pruebas
<b>Categoría</b>	Operación
<b>Propósito</b>	El propósito del proceso de pruebas es encontrar la mayor cantidad posible de errores en los componentes del software.
<b>Descripción</b>	<p>El proceso de pruebas básicamente está compuesto por las siguientes fases:</p> <ul style="list-style-type: none"><li>○ Técnicas de diseño de casos de prueba: se utilizan para obtener una confianza aceptable en que se detectarán los defectos existentes.<ul style="list-style-type: none"><li>○ Pruebas de Aceptación: Se realiza los casos de prueba que el cliente requiere en el software.</li><li>○ Pruebas Funcionales o de caja negra: Se identifican las clases de equivalencia y se crean los casos de prueba correspondientes.</li></ul></li><li>○ Diseño de Pruebas<ul style="list-style-type: none"><li>○ Plan de Pruebas</li><li>○ Especificación de los casos de prueba</li><li>○ Especificación de los procedimientos de prueba.</li></ul></li><li>○ Ejecución de las pruebas<ul style="list-style-type: none"><li>○ Histórico de Pruebas</li><li>○ Informe resumen</li></ul></li></ul>
<b>Objetivos</b>	O1 Realizar las técnicas de Diseño de Casos de Prueba.
	O2 Realizar y/o modificar el Plan de Pruebas
	O3 Realizar y/o modificar la Especificación de los Casos de Prueba.
	O4 Realizar y/o modificar la Especificación de los Procedimientos de Prueba.
	O5 Realizar la Ejecución de las Pruebas.
<b>Responsabilidad y Autoridad</b>	<p>Responsabilidad:</p> <ul style="list-style-type: none"><li>• Tester</li></ul> <p>Autoridad:</p> <ul style="list-style-type: none"><li>• Director Desarrollo</li></ul>
<b>Procesos</b>	Requerimientos

<b>Relacionados</b>	Análisis y Diseño Construcción
---------------------	-----------------------------------

### 2.4.2 Entradas

**Tabla 26. Entradas - pruebas**

<b>Nombre</b>	<b>Fuente</b>
SRS(Requerimientos y Casos de Uso)	Requerimientos
Componentes de Software	Construcción
Scripts BD	

### 2.4.3 Salidas

**Tabla 27. Salidas - pruebas**

<b>Nombre</b>	<b>Descripción</b>	<b>Destino</b>
Documento de la ejecución de las pruebas	<p>El documento esta compuesto por los siguientes ítems:</p> <ul style="list-style-type: none"> <li>○ <i>Histórico de pruebas:</i> Documenta todos los hechos relevantes ocurridos durante la ejecución de las pruebas.</li> <li>○ <i>Informe resumen:</i> Resume los resultados de las actividades de prueba (las reseñadas en el propio informe) y aporta una evaluación del software, basada en dichos resultados.</li> </ul>	Implementación

### 2.4.4 Productos Internos

**Tabla 28. Productos Internos - pruebas**

<b>Nombre</b>	<b>Descripción</b>
Plan de Pruebas	<p>Se enfoca en detallar el modelo de prueba para los componentes de software, se debe señalar:</p> <ul style="list-style-type: none"> <li>○ El enfoque</li> <li>○ Los recursos</li> <li>○ El esquema de actividades de prueba</li> </ul>

	<ul style="list-style-type: none"> <li>○ Los elementos a probar</li> <li>○ Las características</li> <li>○ Las actividades de prueba</li> <li>○ El personal responsable</li> <li>○ Los riesgos asociados</li> </ul>
Especificación de Casos de Prueba	Define uno de los casos de prueba identificado por una especificación del diseño de las pruebas.
Especificación de Procedimientos de Prueba	Especifica los pasos para la ejecución de un conjunto de casos de prueba o, más generalmente, los pasos utilizados para analizar un elemento software con el propósito de evaluar un conjunto de características del mismo.

## 2.4.5 Referencias Bibliográficas

**Tabla 29. Referencias Bibliográficas - pruebas**

Modelo de Procesos para la industria del Software, Moprosoft v1.1, Mayo 2003.
The Capability Maturity Model Integration, CMMI <sup>SM</sup> for Systems Engineering and Software Engineering (CMMI-SE/SW, V1.1) CMMI, Product Team, December 2001.
IEEE STD 829 1998, IEEE Standard for Software Test Documentation
Pruebas de software, Universidad Pontificia de Madrid, Ingeniera de Software, 2005 <a href="http://tdi.eui.upm.es/Is/descargas/Tema5_PruebasSoftware.pdf">http://tdi.eui.upm.es/Is/descargas/Tema5_PruebasSoftware.pdf</a>
Pruebas de Software, Universidad Rey Juan Carlos, Ingeniería de Software I <a href="http://kybele.escet.urjc.es/Documentos/ISI/Pruebas%20de%20Software.pdf">http://kybele.escet.urjc.es/Documentos/ ISI/Pruebas%20de%20Software.pdf</a>

## 2.4.6 Roles Involucrados

**Tabla 30. Referencias Bibliográficas - pruebas**

<b>Rol</b>	<b>Capacitación</b>
Tester	Conocimiento y experiencia en ingeniería de software, elaboración y ejecución de las pruebas de software.
Director de Implementación	Conocimiento y/o experiencia en la programación, integración y pruebas de software.
Director de Desarrollo	Conocimiento general de los proyectos y experiencia en dirección, planeación y desarrollo de software.
Usuario Líder	Conocimiento del proceso y capacidad de toma de decisiones.



## 2.4.7 Actividades

Tabla 31. Actividades - pruebas

Rol	Descripción
A1. Técnicas de diseño de casos de pruebas(O1)	
Tester  Usuario Líder	<p>A.1.1 Realizar las Pruebas de Aceptación  A.1.1.1 El cliente debe realizar los casos de prueba en conjunto con el Tester.</p> <p>A.1.2 Realizar las pruebas funcionales o de caja negra.  A.1.2.1 Identificar las clases de equivalencia.  A.1.2.1.1 Identificar las restricciones al formato y contenido de los datos de las entradas.  A.1.2.1.2 Identificar las clases de equivalencia:  <ul style="list-style-type: none"> <li>• De datos Válidos</li> <li>• De Datos no Válidos o Erróneos</li> </ul> </p> <p>Existen algunas reglas que ayudan a identificar clases:</p> <ul style="list-style-type: none"> <li>• Si se especifica un rango de valores para los datos de entrada, se creará una clase válida y dos clases no válidas.</li> <li>• Si se especifica un número de valores, se creará una clase válida y dos no válidas.</li> <li>• Si se especifica una situación del tipo «debe ser» o booleana (por ejemplo, «el primer carácter debe ser una letra»), se identifican una clase válida («es una letra») y una no válida («no es una letra»).</li> <li>• Si se especifica un conjunto de valores admitidos y se sabe que el programa trata de forma diferente cada uno de ellos, se identifica una clase válida por cada valor y una no válida.</li> <li>• En cualquier caso, si se sospecha que ciertos elementos de una clase no se tratan igual que el resto de la misma, deben dividirse en clases menores.</li> </ul> <p>A.1.2.2 Crear los casos de prueba correspondientes:  A.1.2.2.1 Este proceso consta de los siguientes pasos:  <ul style="list-style-type: none"> <li>• Asignación de un número único a cada clase de equivalencia.</li> </ul> </p>

	<ul style="list-style-type: none"> <li>• Hasta que todas las clases de equivalencia hayan sido cubiertas por (incorporadas a) casos de prueba, se tratará de escribir un caso que cubra tantas clases válidas no incorporadas como sea posible.</li> <li>• Hasta que todas las clases de equivalencia no válidas hayan sido cubiertas por casos de prueba, escribir un caso para una única clase no válida sin cubrir.</li> </ul>
A2. Realizar y/o modificar el plan de pruebas(O2)	
Tester  Director de Desarrollo	A.2.1 Realizar y/o modificar el documento de plan de pruebas. A.2.1.1 Describir una estrategia de prueba. A.2.1.2 Estimar los requisitos para el esfuerzo de la prueba (recursos humanos y sistemas necesarios). A.2.1.3 Planificar el esfuerzo de la prueba. A.2.1.4 Señalar el enfoque, recursos, esquema de actividades de prueba, los elementos a probar, las características, las actividades de prueba, el personal responsable, los riesgos asociados del plan de pruebas.
A3. Realizar y/o modificar especificación de los casos de prueba(O3)	
Tester  Director de Implementación	A.4.1 Realizar y/o modificar especificación de un caso de prueba. A.4.1.1 Describir los casos de prueba para los casos de uso. A.4.1.2 Describir las entradas para cada caso de prueba. A.4.1.3 Describir las salidas o resultados para cada caso de prueba. A.4.1.4 Describir las condiciones relevantes para cada caso de prueba.
A4. Realizar y/o modificar la especificación de los procedimientos de prueba(O4)	
Tester Director de Implementación	A.5.1 Realizar y/o modificar especificación de un procedimiento de prueba. A.5.1 Numerar y Describir los pasos de cada

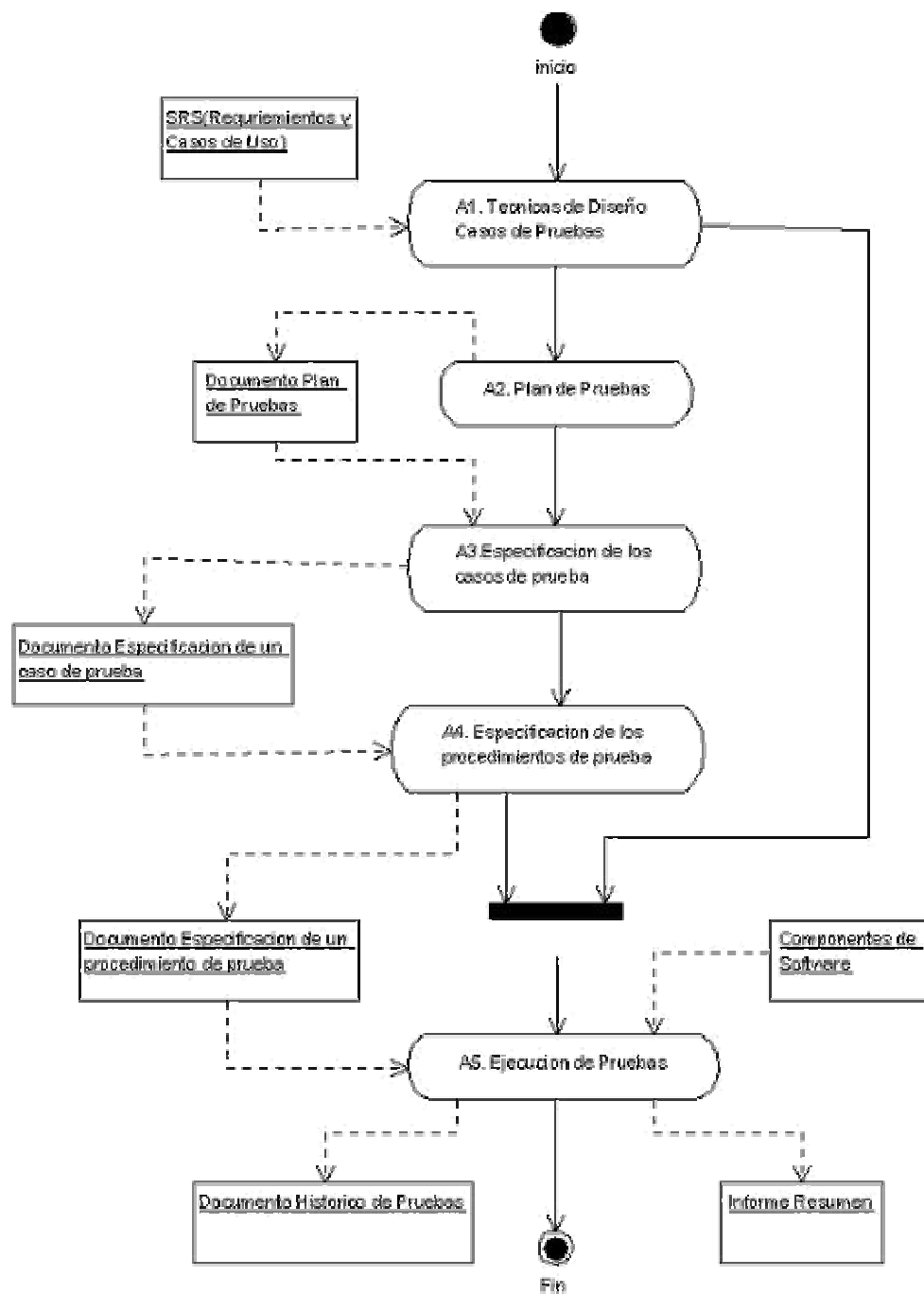
	procedimiento de prueba.
<b>A5. Realizar y/o modificar el documento de ejecución de pruebas(O5)</b>	
Tester  Director de Implementación	<p>A.6.1 Realizar ejecución de la pruebas basado en el documento de especificación de casos de prueba, en la especificación de los procedimientos de prueba y en los componentes de software que se van a probar.</p> <p>A.6.2 Realizar y/o modificar el documento de ejecución de las pruebas que contienen los siguientes fases:</p> <p style="padding-left: 40px;">A.6.2.1 Realizar y/o modificar el histórico de pruebas.</p> <p style="padding-left: 80px;">A.6.1.2.1 Documentar todos los hechos relevantes en la ejecución de las pruebas.</p> <p style="padding-left: 40px;">A.6.2.2 Realizar y/o modificar informe resumen.</p> <p style="padding-left: 80px;">A.6.2.2.1 Resumir los resultados de las actividades de prueba basado en el histórico de pruebas.</p> <p style="padding-left: 80px;">A.6.2.2.1 Aportar una evaluación del software basada en dichos resultados.</p>

#### 2.4.9 Verificaciones y/o Validaciones

**Tabla 32. Verificaciones y/o Validaciones - pruebas**

Actividad	Producto	Rol	Descripción
Ninguna...	-	-	-

Figura 7. Diagrama de actividades del proceso de Pruebas



### 3. CONCLUSIONES

- Las metodologías de software son fundamentales en una empresa de desarrollo de software permitiendo definir reglas, procedimientos, actividades, tareas, herramientas de software, etc. Para cada etapa de desarrollo de un nuevo proyecto, además permite que las personas trabajen mejor y los proyectos finalicen exitosamente.
- No es fácil aplicar una nueva metodología en un equipo de desarrollo ya que obliga a aprender una nueva forma de trabajar. También obliga a abandonar cómo se hacían las cosas antes, que aunque no fuera la mejor forma posible ya se conocía. XP ha sido adoptado por un gran número de equipos en los últimos años y de sus experiencias se ha extraído una conclusión sencilla: es mejor empezar a hacer XP gradualmente.
- Este tipo de desarrollos eran en general de creación de software a la medida del cliente y hay numerosas opiniones que relatan el éxito de esta metodología en este ámbito. Queda por ver si es posible aplicar sus ideas también en procesos de desarrollo muy diferentes, como el software libre.
- RUP contempla más rigurosidad en cuanto al desarrollo de software y permite establecer los requerimientos y casos de uso de manera iterativa, aunque algunas personas utilizan el RUP con el modelo cascada, pero la verdad RUP puede adaptarse a ser una metodología ágil, produciendo código y haciendo pruebas, además la RUP contempla procesos generales y procesos de soporte que otras metodologías no lo aplican.
- Para la empresa adaptarse a los procesos desarrollados va a llevar tiempo y capacitación para las personas que conforman el equipo de desarrollo, y se necesitara el apoyo de los altos directivos para realizarlo.
- En el proceso de requerimientos este proceso es fundamental porque aquí se produce la información necesaria para comenzar el desarrollo de software, en esta etapa evaluamos de la mejor manera la captura de los requerimientos haciendo posible la trazabilidad es este proceso, y comprobamos que es la etapa para la empresa ya que de eso depende de que su construcción sea lo mejor realizada posible. Los casos de uso forman parte especial porque determinan el procedimiento que se va a construir y realizan parte fundamental

de la pruebas para el software. Se probó el proceso de requerimientos dando como resultado un proceso exitoso.

- Análisis y Diseño es uno de los procesos que no se realizaban en la empresa, por este motivo el equipo de desarrollo le tomara tiempo adaptarse a los nuevos procedimientos que se deben realizar cuando se este desarrollo un nuevo aplicativo. Para este proceso se involucraron diagramas que permiten tener una interpretacion grafica de los que seria el software, su principal fuente esta en el modelo de datos.
- Construcción es el proceso que se considera más adaptable a las necesidades actuales, debido a que se centra en la programación, y gracias a la implantación de herramientas como *CVS*, *MANTIS*, *dotproject*. (Herramientas de software).
- El proceso de pruebas se encuentra es su estado mejora y debe realizarse un estudio mas exhaustivo de esta proceso.

#### **4. RECOMENDACIONES**

- No es fácil aplicar una nueva metodología en un equipo de desarrollo ya que obliga a aprender una nueva forma de trabajar. También obliga a abandonar cómo se hacían las cosas antes, que aunque no fuera la mejor forma posible ya se conocía. XP ha sido adoptado por un gran número de equipos en los últimos años y de sus experiencias se ha extraído una conclusión sencilla: es mejor empezar a hacer XP gradualmente.
- Este tipo de desarrollos eran en general de creación de software a la medida del cliente y hay numerosas opiniones que relatan el éxito de esta metodología en este ámbito. Queda por ver si es posible aplicar sus ideas también en procesos de desarrollo muy diferentes, como el software libre.

## BIBLIOGRAFIA

BOOCH, Grady. RAMBAUGH, James. JACOBSON, Ivar. El Proceso Unificado de Desarrollo de Software: Requerimientos. 1 ed. Mexico: Addison Wesley, 1999. 425 p.

BOOCH, Grady. El Lenguaje Unificado de Modelado: UML. 1 ed. Mexico: Addison Wesley, 1999. 385 p.

JACOBSON, Ivar. El Proceso Unificado de Modelado. 1 ed. Madrid: Addison Wesley, 1999. 486 p.

KENDALL k., KENDALL, Julie. Análisis y Diseño de sistemas. 2 ed. Madrid: Prentice-Hall, 1999. 467 p.

LARMAN, Craig. UML y Patrones Una introducción al Análisis y Diseño Orientado a Objetos y al Proceso Unificado. 2 ed. Madrid: Prentice Hall, 2002. 356 p.

Modelo de Procesos para la industria del Software [en línea]. México: Moprosoft, 2003. [consultado 15 julio, 2005]. Disponible en internet: <http://www.lania.mx/biblioteca/manuales/moprosoft/V%201.1%20DocumentoBase.pdf>

La Nueva Metodología [en línea]. New York: Martín Fowler, 1999. [consultado 3 Mayo, 2005]. Disponible en Internet: <http://www.martinfowler.com>



## ANEXO A. Formato guía de entrevista

### GUÍA ENTREVISTA

No		FECHA	DD	MM	AAAA
CLIENTE					
PROYECTO					

ENTREVISTADO(S)	
PERFIL	
ENTREVISTADOR(ES)	

OBJETIVOS GENERALES
---------------------

OBJETIVOS ESPECIFICOS
-----------------------

GUIA ENTREVISTA
-----------------

RESUMEN
---------

PENDIENTES
------------

RESULTADO
-----------

## ANEXO B. Formato guía de entrevista

### DETALLE DE PROCESOS

#### Descripción General del Proceso

No		FECHA	DD	MM	AAAA
Cliente					
Proyecto					
Proceso					
Objetivo General					
Objetivos Específicos					
Responsabilidad y Autoridad					
Procesos Relacionados					

#### Entradas

Nombre	Fuente

#### Salidas

Nombre	Descripción	Destino

#### Roles Involucrados

Rol	Cargo y/o Profesión

#### Actividades

Rol	Descripción

**Diagrama de Flujo de Actividades (OPCIONAL):** Diagrama de Actividades UML donde se especifican las actividades y los productos.

## ANEXO C. Formato de especificación de requerimientos

## ESPECIFICACIÓN DE REQUERIMIENTOS

<b>No</b>		<b>FECHA</b>	DD	MM	AAAA
<b>CLIENTE</b>					
<b>PROYECTO</b>					

[illegible]

## ANEXO D. Instructivo de Guía de Entrevista

### GUÍA ENTREVISTA

Para realizar el documento de entrevista se plantea la siguiente información:

CAMPO	DESCRIPCIÓN
<b>No.</b>	Identificador de la guía de entrevista.
<b>FECHA</b>	Fecha de diligenciamiento del documento en formato (dd/mm/aaaa)
<b>CLIENTE</b>	Nombre del cliente.
<b>PROYECTO</b>	Nombre del proyecto.
<b>OBJETIVO GENERAL</b>	Define el propósito para la cual se va a realizar la entrevista.
<b>OBJETIVOS ESPECÍFICOS</b>	Definen los ítems para cumplir con el objetivo general.
<b>GUÍA DE ENTREVISTA</b>	<ul style="list-style-type: none"><li>• Si se va a realizar la primera entrevista, se debe elegir la guía de entrevista estándar.</li><li>• Para realizar una guía de entrevista, se debe realizar de la siguiente manera:<ul style="list-style-type: none"><li>○ Plantear el objetivo general.</li><li>○ Plantear los objetivos específicos de acuerdo al objetivo general.</li><li>○ Formular las preguntas para lograr los objetivos específicos.</li></ul>A partir de los objetivos específicos se realiza la preguntas necesarias para abarcar el objetivo general de la entrevista, por lo general se debe realizar al menos una pregunta por cada objetivo específico.</li></ul>
<b>ENTREVISTADO(S)</b>	Nombre(s) de la persona(s) entrevistada(s).
<b>PERFIL</b>	Perfil de la persona entrevistada.
<b>ENTREVISTADOR(ES)</b>	Nombre(s) de la persona(s) que realiza(n) la entrevista.
<b>FECHA</b>	Fecha en que se realiza la entrevista.
<b>RESUMEN</b>	Descripción de Procesos, requerimientos o casos de uso que se detectaron en la información recolectada de la entrevista.
<b>PENDIENTES</b>	Descripción de actividades que quedaron pendientes por realizar.
<b>RESULTADO</b>	Observaciones acerca de la información recolectada de la entrevista.

## ANEXO E. Instructivo del detalle de procesos

### DETALLE DE PROCESOS

#### Descripción General del Proceso

<b>No</b>	Identificador del proceso ( <i>el identificador esta definido por DPROXXX, donde XXX es un número consecutivo, el identificador debe existir en la Lista de Procesos</i> )
<b>FECHA</b>	Fecha de diligenciamiento del documento en formato (dd/mm/aaaa)
<b>Cliente</b>	Nombre del Cliente.
<b>Proyecto</b>	Nombre del Proyecto.
<b>Proceso</b>	Nombre del Proceso.
<b>Objetivo General</b>	Objetivos generales medibles y resultados esperados de la implantación del proceso
<b>Objetivos Específicos</b>	Objetivos específicos cuya finalidad es asegurar el cumplimiento del propósito. Los objetivos se identifican como O1,O2, etc.
<b>Responsabilidad y Autoridad</b>	Responsabilidad: es el rol principal por la ejecución del proceso. Autoridad: es el rol responsable por validar la ejecución del proceso y el cumplimiento del propósito.
<b>Procesos Relacionados</b>	Nombre de los Procesos Relacionados, entre paréntesis se especifica el identificador del proceso.

#### Entradas

<b>Nombre</b>	<b>Fuente</b>
Nombre del producto o recurso	Referencia al origen del producto o recurso.

#### Salidas

<b>Nombre</b>	<b>Descripción</b>	<b>Destino</b>
Nombre del producto o recurso	Descripción de las características del producto o recurso.	Referencia al destinatario del producto o recurso.


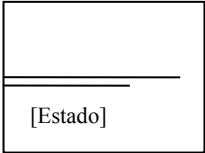

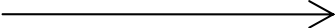
#### Roles Involucrados

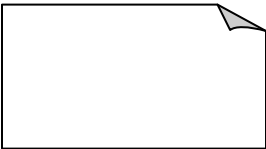
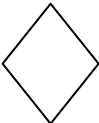

<b>Rol</b>	<b>Cargo y/o Profesión</b>
Nombre del Rol	Cargo y/o profesión del rol que ejecuta el proceso.



### Actividades

Rol	Descripción
A1. Nombre de la actividad(O1,O2, etc)	
Nombre del rol	A.1.1 Descripción de la tarea1 A.1.2 Descripción de la tarea2
A2. Nombre de la actividad(O1,O2, etc)	
Nombre del rol	A.2.1 Descripción de la tarea1 A.2.2 Descripción de la tarea2

**Diagrama de Flujo de Actividades (OPCIONAL):** Diagrama de Actividades UML donde se especifican las actividades y los productos.

Elemento	Descripción
	Actividad a realizar en un proceso. En su nombre se agrega la identificación de la actividad que representa.
	Producto generado en una actividad. En algunos casos, se indica el estado en el que se encuentra el producto.
	Transición hacia la misma actividad
	Transición entre actividades

	Flujo de información. Indican a partir de qué actividad se genera un producto, y en algunos casos su destino.
-----	Asociación entre comentarios y actividades
	Comentarios
	Bifurcación. Permite modificar la transición de una actividad de acuerdo a alguna condición.
	Barra de sincronización. Indica las actividades que deben concluirse antes de iniciar otra(s).

	Inicio del proceso
	Fin del proceso



## ANEXO F. Instructivo especificación de requerimientos

### ESPECIFICACION DE REQUERIMIENTOS

CAMPO	DESCRIPCION
<b>No</b>	Identificación de la especificación de requerimientos.
<b>FECHA</b>	Fecha de diligenciamiento del documento en formato (dd/mm/aaaa)
<b>CLIENTE</b>	Nombre del Cliente.
<b>PROYECTO</b>	Nombre del Proyecto.
<b>RQ</b>	Identificador del requerimiento <i>(el identificador esta definido por RQXXX, donde XXX es un número consecutivo)</i> .
<b>PR</b>	Identificador del proceso.
<b>DESCRIPCIÓN</b>	Descripción del Requerimiento.
<b>REEM POR</b>	Identificador del requerimiento por el cual se va reemplazar.
<b>FUENTE</b>	Identificador del documento de levantamiento de información o un producto del proceso, que se utilizo para obtener el requerimiento.
<b>CU</b>	Identificador del caso de uso al que pertenece el requerimiento.

Este es un formato donde se basara básicamente en la definición de los procesos, es decir, primero hay que identificar los procesos que se realizan, y luego se procede a la clasificación de los requerimientos por cada proceso que se identifique. Además, tanto el requerimiento como los procesos pueden ser priorizados, donde al requerimiento se le da una mayor prioridad cuando esta escrita en **negrilla**.

Tips para elaborar unos buenos requerimientos, estas son reglas sencillas pero importantes para una correcta especificación de requerimientos.

1. Si un requerimiento tiene 3 o mas signos de puntuación, este se necesita redactarlo, si este es el caso tal vez aplique hacer más de un requerimiento.
2. Si para un requerimiento piensas en más de una prueba, quizá se estén mezclando más de un requerimiento en esa especificación.
3. Use una gramática y puntuación correcta.

4. Pon atención en las conjunciones "y" "o", al usar estas conjunciones podríamos estar hablando de mas de un requerimiento.
5. Nunca usar palabras "como", "etc".
6. Los requerimientos deben poder ser medidos y probados.

## ANEXO G. Template de la guía de entrevista

### GUÍA ENTREVISTA

No		FECHA	DD	MM	AAAA
CLIENTE					
PROYECTO					

ENTREVISTADO(S)	
PERFIL	
ENTREVISTADOR(ES)	

#### OBJETIVOS GENERALES

- Obtener una visión global de los procesos que directa o indirectamente serán afectados por el proyecto.

#### OBJETIVOS ESPECIFICOS

- Identificar los procesos involucrados en el proyecto.
- Identificar los inconvenientes que se presentan en los procesos.
- Conocer los sistemas de información actuales.
- Identificar los procesos claves.

#### GUIA ENTREVISTA

1. ¿Cuál es su cargo dentro de la empresa?
2. ¿Cuánto tiempo de experiencia tiene usted en este cargo?
3. ¿Cuales son sus responsabilidades y/o actividades dentro de la empresa?
4. ¿Qué sistemas de información utiliza actualmente para cubrir estas responsabilidades y/o actividades?

*Nota: Si no maneja sistemas de información pase a la pregunta 6.*

5. ¿Qué opinión tiene acerca de estos sistemas de información?
6. ¿Cuáles son los procesos que están involucrados en el proyecto?
7. ¿Cuál es el proceso central?

8. ¿Qué información (documentos, formatos, datos, etc.) ingresa a este proceso?
9. ¿Cuántas personas intervienen en este proceso?
10. ¿Qué información produce este proceso?
11. ¿Cuáles son los problemas que actualmente se presentan?
12. ¿Por qué cree que se le presentan estos problemas?
13. ¿Ha pensado en alguna forma de solucionarlos?

## ANEXO H. Template de Arquitectura de Software

### ARTQUITECTURA DE SOFTWARE

ID		ID_POOL		FECHA	DD	MM	AAAA
CLIENTE							
PROYECTO							
AUTOR							

DIAGRAMA DE DESPLIEGUE			
ARCHIVO		UBICACION	

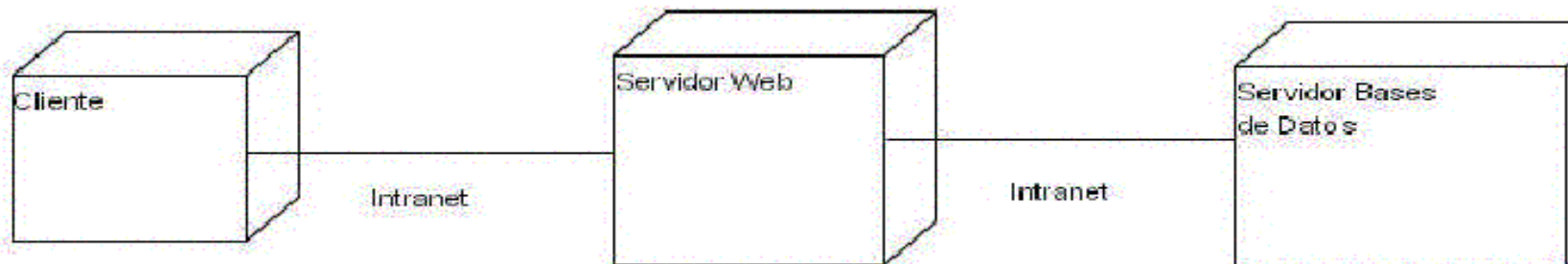
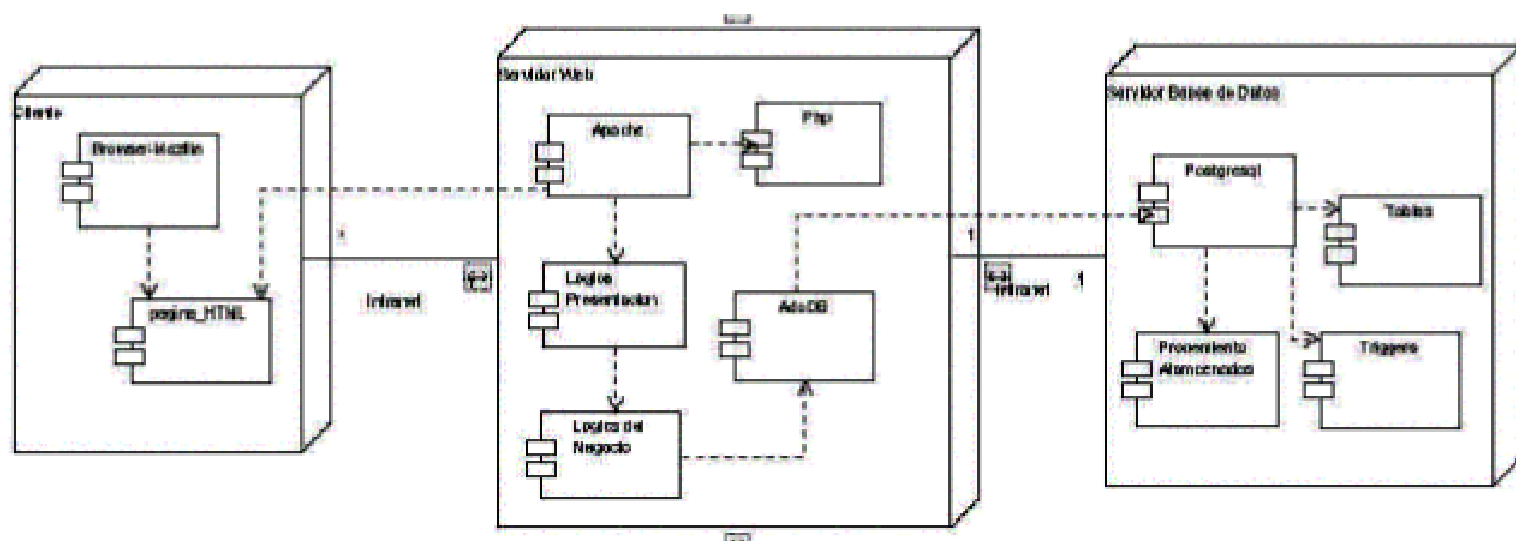




DIAGRAMA DE COMPONENTES		
ARCHIVO	UBICACION	



## ANEXO I. PAPER

### DEFINICION DE METODOLOGIA DE DESARROLLO DE SOFTWARE Y DOCUMENTACION DE PORCESOS EN IPSOFT S.A

**luis alejandro vargas muñoz**

*Universidad Autónoma de Occidente, Calle 25 No. 115-85 Kilómetro 2 vía Cali –  
Jamundí, luisvarmu@telesat.com.co , Santiago de Cali*

**Abstract:** El propósito de realizar la documentación de la metodología de desarrollo de software en IPSOFT S.A es identificar, describir, mejorar e implantar los procesos de desarrollo de software que permita disminuir la desorganización en desarrollos de software posteriores. Para desarrollar la metodología, se realizó un estudio de la situación actual e investigación de las metodologías de desarrollo de software existentes, y finalmente la documentación de los procesos de desarrollo de software: Requerimientos, Análisis, Diseño, Construcción y Pruebas, con el apoyo de Estándares (IEEE, CMMI, ISO) y Modelos(MOPROSOFT, IDEAL)

**Keywords:** Metodología de Software, Procesos de Software, Requerimientos, Análisis, Diseño, Construcción, Pruebas.

#### 1. INTRODUCCIÓN

Dentro del ámbito de la ingeniería de software (y la informática en general), se ha observado que para cualquier empresa en el campo de software se requiere definir, documentar e implantar una metodología de desarrollo de software y los procesos que la componen adaptados al contexto de la empresa, para obtener una distribución general en los proyectos de software por medio de políticas, procedimientos, actividades, etc. Pero esto, no es suficiente para guiar y mantener un producto de software, buscando por medio de la documentación de los procesos de desarrollo de software, se utilicen las prácticas que ofrece la ingeniería de software para alcanzar sus objetivos.

Sin embargo, muchas veces no se toma en cuenta el utilizar una metodología adecuada, sobre todo cuando se trata de proyectos pequeños de dos o tres meses. Lo que se hace con este tipo de proyectos es separar rápidamente el aplicativo en procesos, cada proceso en

funciones, y por cada función determinar un tiempo aproximado de desarrollo.

Cuando los proyectos que se van a desarrollar son de mayor envergadura, ahí sí toma sentido el basarse en una metodología de desarrollo robusta, y se empieza a buscar cuál sería la más apropiada para el caso. Lo cierto es difícil encontrar una metodología que se ajuste al contexto de la empresa.

El Grupo de Investigación de Ingeniería de Software de la Universidad Autónoma de Occidente, ofreció su apoyo, en la modalidad de pasantía, para elaborar la propuesta cuyo tema es el desarrollo de la metodología de software en IPSOFT S.A. La metodología es diseñada bajo el contexto propio de trabajo en IPSOFT S.A.

La metodología de desarrollo de software contempla el conjunto de procedimientos, herramientas,



documentación y aspectos que ayuden a utilizar las buenas prácticas para el desarrollo del software, cuyo objetivo es construir mejores aplicaciones, procesos de desarrollo, donde se identifiquen entradas, salidas (o productos internos) para cada uno, de forma que se pueda planificar y controlar el producto. Una buena metodología se compone de actividades, tareas y procedimientos, productos, técnicas y herramientas de software que permite llevar un proceso formal en la elaboración de un software.

## 2. METODOLOGIAS DE DESARROLLO DE SOFTWARE

La elaboración de una metodología de software implica varios aspectos entre los cuales un estudio de la situación actual de la empresa, que servirá como punto de partida ya que proporciona la información necesaria de su estructura organizacional, el equipo de desarrollo, herramientas de desarrollo de software, Sistemas Operativos y otros aplicativos, procesos y proyectos que se llevan a cabo.

De acuerdo a lo anterior, uno de los aspectos a destacar en la metodología es que sea lo más flexible posible, debido a la escasa utilización de las prácticas que brinda la ingeniería de software para el desarrollo de software.

Se realiza una investigación acerca de metodologías de desarrollo de software existentes, para analizar cuál será la más conveniente, adaptándose a las necesidades actuales de la empresa.

Entre las metodologías encontradas están la Programación Extrema (XP), Scrum, Método de Desarrollo de Sistema Dinámico (DSDM), Rational Unified Process (RUP), entre otras, para el caso se enfatizó en las metodologías XP y el RUP, por que son las que se ajustan al contexto de la empresa:

### 2.1 Programación Extrema (XP)

La programación extrema se basa en una serie de reglas y principios que se han ido gestando a lo largo de toda la historia de la ingeniería del software. Usadas conjuntamente proporcionan una nueva metodología de desarrollo de software que se puede englobar dentro de las metodologías ligeras, que son aquellas en las que se da prioridad a las tareas que dan resultados directos y que reducen la burocracia que hay alrededor tanto como sea posible (Fowler, 2003).

De todas las metodologías ágiles, ésta es la que ha recibido mayor acogida. Esto se debe en parte a la notable habilidad de los líderes XP, en particular Kent

Beck, para llamar la atención. También se debe a la habilidad de Kent Beck de atraer a las personas a este acercamiento, y tomar un papel principal en él, sin embargo, la popularidad de XP se ha vuelto un problema, pues ha acaparado la atención fuera de las otras metodologías y sus valiosas ideas. (Fowler, 2003).

La metodología XP tiene como propósito que el cliente sea parte inicial del desarrollo de software redactando los requerimientos a su manera, estos requerimientos en la metodología XP se denominan *Historias de Usuario*.

Las características que contempla la metodología XP es que permite introducir nuevos requisitos o cambiar los anteriores de un modo dinámico, publica pronto las versiones que implementan parte de los requisitos, adecuado para proyectos pequeños y medianos, con alto riesgo y su ciclo de vida es iterativo e incremental [Fig 1.]

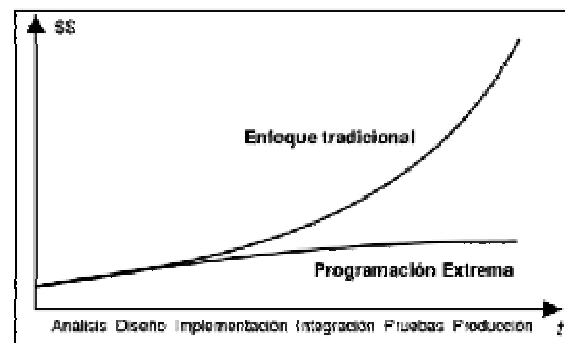


Fig 1. Comparación XP con Enfoque Tradicional Costos vs Tiempo

El proceso consiste a grandes rasgos en los siguientes pasos:

1. El cliente define el valor del negocio a implementar.
2. El programador estima el esfuerzo necesario para su implementación.
3. El cliente selecciona que construir, de acuerdo con sus prioridades y restricciones de tiempo.
4. El programador construye ese valor del negocio.
5. Vuelve al paso 1.

### 2.2 Rational Unified Process (RUP)

Su objetivo es producir software de alta calidad, es decir, que cumpla con los requerimientos de los usuarios dentro de una planificación y presupuesto establecidos. Dirigido por casos de uso, centrado en la

arquitectura, iterativo (mini-proyectos) e incremental (versiones).

RUP involucra una serie de practicas para proveer las fases que propone en el desarrollo, roles, fases e iteraciones [Fig 2.] que permiten utilizar UML de la forma mas efectiva, y además en la producción y en el mantenimientos mas que en producir documentos.

RUP es un armazón de proceso y como tal puede acomodar una gran variedad de procesos. De hecho ésta es mi crítica principal al RUP - como puede ser cualquier cosa acaba siendo nada. Yo prefiero un proceso que dice qué hacer en lugar de dar opciones infinitas.

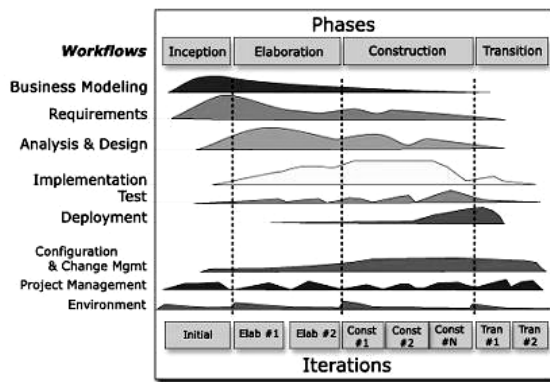


Fig 2. Fases, Procesos e Iteraciones del RUP.

Una de las cosas clave que necesita el RUP es que los líderes del RUP en la industria enfaticen su acercamiento al desarrollo de software. Más de una vez he oído a la gente que usa el RUP que están usando un proceso de desarrollo estilo cascada. (Fowler, 2003).

### 3. PROCESOS DE DESARROLLO DE SOFTWARE

Es difícil encontrar una metodología que se adapta a todo el contexto de la empresa, y pueda diseñar procesos que ayuden a realizar seguimientos del software, por eso es conveniente diseñar una metodología que satisfaga las necesidades de la empresa.

Con el grupo de investigación se realizaron reuniones donde se propusieron alternativas para cada uno de los procesos que van a formar parte de la metodología de desarrollo de software.

La metodología es basada en los procesos de desarrollo de software, en estos se describen procedimientos, actividades, tareas, roles, formatos, instructivos, diagramas, estándares y herramientas de software que ayudan a complementar los procesos de software.

Los procesos de software que se consideraron son: Requerimientos, Análisis, Diseño, Construcción y Pruebas; para la respectiva documentación de los procesos se baso en el Modelo de Procesos para la Industria del Software (MOPROSOFT v1.1).

Para cada uno de los procesos se diseñaron formatos e instructivos que soportan las actividades y tarea de cada proceso, estos debe ser flexible y ajustables a los procesos de la empresa.

#### 3.1 Requerimientos

Como primera instancia el proceso que se va a atacar es de requerimientos, este proceso es muy importante debido a que brinda la información necesaria para comenzar en el desarrollo de un software, en este proceso existe una mayor interacción con el cliente.

En el proceso de requerimientos se toma en cuenta cuales son las actividades que son importantes para el desarrollo de la metodología: analizar las técnicas que ayuden a un buen levantamiento de la información, cuales son los aspectos que se deben atacar, por ejemplo, si se van a detallar todos los casos de uso, la validación será por requerimientos o por casos de uso, etc, para determinar el enfoque que se le va a dar en el transcurso de la metodología.

La validación consistía en listar los casos de uso centrales y los requerimientos que no tienen asociado un caso de uso, luego se chequea cada caso de uso o requerimiento que haya sido validado por el cliente y al final la firma del cliente y la firma de la empresa. Pero esta idea se descarto porque si faltan requerimientos o casos de uso por validar el cliente no firmaría hasta que todos los casos de uso o requerimientos queden validados.

Los formatos e instructivos son propuestos con el fin de adaptarse al contexto de la empresa, por ejemplo, el formato de entrevista que esta conformada por su *numero de identificación*, para ser identificadas y diferenciarlas de las demás entrevistas, el nombre del *cliente* para identificar a que clínica se realizara la entrevista y el proyecto para identificar el proyecto en cual se desarrolla, el *perfil* de la persona entrevistada para orientar las preguntas de la entrevista en términos del entrevistado. Se define el *objetivo general* para determinar el alcance de la entrevista, *objetivos*

*específicos* para determinar cuales son los items que se deben realizar para cumplir con el objetivo general y las preguntas se realizaran de acuerdo al perfil y objetivos específicos planteados.

A continuación, el propósito y descripción general del proceso en IPSOFT-S.A:

El propósito del proceso de requerimientos es obtener una lista de requerimientos y casos de uso validados, clasificados y priorizados, para conseguir un entendimiento común entre el cliente y el proyecto.

El proceso de requerimientos esta compuesto de las siguientes fases:

1. Técnicas de Levantamiento de Información: Definición y ejecución de la(s) entrevista(s), revisión de documentos y observación de procesos.
2. Especificación de Procesos: Identificar los procesos que componen el proyecto e identificar los procesos centrales.
3. Detalle de Procesos: Se detallan los procesos centrales.
4. Validación de procesos: se valida el detalle de los procesos.
5. Especificación de Requerimientos: De cada proceso se identifica los requerimientos y de los procesos centrales, los requerimientos claves. (Priorización de requerimientos).
6. Definición de Casos de uso: Con base en los requerimientos se definen todos los casos de uso.
7. Detalle de Casos de uso: Se detallan los casos de uso centrales.
8. Validación: Se validan los detalles casos de uso centrales y los requerimientos que no tienen asociado detalle de casos de uso.
9. SRS: se plantea con base en el estándar (IEEE STD 830 1998, IEEE Recommended Practice for Software Requirements Specifications), este es el producto final que aportara como entrada a los procesos siguientes.

La autoridad del proceso es el Director de Desarrollo y el Responsable es el Analista.

Se realizó la prueba para el proceso de requerimientos con el fin de mejorar las actividades, tareas, procedimientos y productos (formatos, instructivos, templates, etc.), para lograrlo se propusieron las siguientes métricas:

- Tiempo que toma cada una de las actividades ejecutadas.

- Número de veces que en cada proceso hubo la necesidad de devolverse o repetirlo por inconsistencias o falta de claridad y cada vez que esto suceda la causa real que originó este retraso.
- Cuantos veces y por que motivo no se siguieron las actividades sugeridas en el proceso.
- Cuantos errores o inconsistencias encuentran en cada uno de los documentos que se elaboraron.
- Cuantos errores o inconsistencias encuentran en cada actividad dentro del proceso.
- Cuantos datos involucrados en el proceso no fueron útiles y el motivo por el que se consideró así.
- En las métricas que apliquen, se debe incluir la o las personas que participaron, opinaron o se equivocaron.

### 3.2 Análisis y Diseño

En el proceso de análisis y diseño es fundamental analizar los diagramas que brinda UML para la documentación del software, y además seleccionar que tipo de diagramas se puedan utilizar y ayuden verdaderamente para el desarrollo del software. Es importante destacar la utilización de estándares (CMMI, ISO, IEEE) que aporte una orientación para el diseño de un buen proceso de análisis y diseño. El diseño de la arquitectura de software y el modelo de datos es muy importante verificar ya que son elementos esenciales que deben ser utilizados en casi todos los desarrollos de software.

La verificación y validación de los productos en esta actividad permite una confiabilidad para continuar con el siguiente proceso, y además, un formato ya que permite tener constancia por escrito de lo que se realizó en los documentos de arquitectura de software, diseño detallado y modelo de datos.

En este proceso se diseñaron formatos e instructivo para llevar a cabo los productos de cada actividad, por ejemplo, el formato de diseño detallado que contiene los diagramas de secuencia (por caso de uso) y el diagrama de clases detallado (por caso de uso) de acuerdo a la arquitectura de software, al conjunto de requerimientos y casos de uso definidos.

El formato de Modelo Relacional de Datos (MRD), es importante por que ayuda a identificar cual es la composición de cada atributo de cada entidad, ahorrando tiempo en el desarrollo de las tablas que componen la base de datos.

A continuación, el propósito y descripción general del proceso en IPSOFT-S.A:

El propósito del proceso de análisis y diseño es modelar y definir los componentes del sistema para obtener un entendimiento del sistema y consistencia entre el proceso de levantamiento de requerimientos y el proceso de análisis y diseño.

El proceso de análisis y diseño está compuesto de las siguientes fases:

1. Análisis: Se analiza la especificación de requerimientos y los casos de uso a ser diseñados.
2. Modelo de negocio: Se realiza el diagrama de objetos y el diagrama de casos de uso para entender el modelo de negocio.
3. Pool de Arquitecturas: Se escoge una alternativa de arquitectura a utilizar y se modifica de acuerdo a la especificación de requerimientos, el modelo de negocio y criterios establecidos.
4. Arquitectura: Se realizan y/o modifican los diagramas de despliegue, diagramas de paquetes y diagrama de componentes, identificando sus componentes a utilizar.
5. Diseño detallado: Se realizan y/o modifican el diagrama de clases detallado a partir del diagrama de clases del modelo de negocio y los diagramas de secuencia.
6. Modelo de datos: Se realiza y/o modifica el modelo de datos de acuerdo al modelo del negocio. El modelo de datos esta compuesto por Modelo Entidad Relación (MER), Modelo Relacional de Datos (MRD) y Componentes de Base de Datos.
7. Verificación y Validación: Se verifican y validan los documentos del proceso de análisis y diseño.

La autoridad del proceso es el Director de Desarrollo y el Responsable es el Analista.

### 3.3 Construcción

El proceso de construcción es diseñado bajo políticas definidos por la empresa, es decir, los procesos de trabajo realizados por la empresa hasta el momento.

En el proceso de construcción se observa el entorno en el que se va a desarrollar la aplicación, y la distribución de las tareas dentro del equipo de desarrollo y la utilización de herramientas de software para el desarrollo de la aplicación, estos son aspectos importantes que se debe tener en cuenta antes de comenzar a construir el software.

El desarrollo de los scripts de la base de datos y los componentes de software constituyen la parte principal de este proceso, que es el que permite transformar los

requerimientos, casos de usos y diagramas (UML) en unidades de código, además, es fundamental hacer pruebas unitarias ya que permite realizar una verificación del código que se ha construido.

Algo importante es especificar la utilización de herramientas que permita el control de las versiones de los componentes de software desarrollados o modificados (si existen) para que no haya confusiones en la implantación del software.

En este proceso se realizaron formatos e instructivos para verificar el ambiente de trabajo para el desarrollo de la aplicación y el reporte de pruebas unitarias.

A continuación, el propósito y descripción general del proceso en IPSOFT-S.A:

El propósito del proceso de construcción es desarrollar una aplicación estable, funcional, mantenible a las necesidades del cliente, basado en los productos del proceso de análisis y diseño.

El proceso de construcción contará con las siguientes fases para su desarrollo:

- 1 Entorno de trabajo: Se revisa que los componentes físicos descritos en la arquitectura de software y estén disponibles para el desarrollo de la aplicación.
- 2 Distribución del trabajo: se revisar la distribución del trabajo de acuerdo al plan de desarrollo.
- 3 Base de datos: realizar y/o modificar la base de datos de acuerdo al MRD y generar scripts.
- 4 Componentes de Software: se construyen y/o modifican los componentes de software basado en la *documentación de la política de desarrollo*.
- 5 Pruebas unitarias: se realiza y aplican las pruebas unitarias a los componentes de software desarrollados y/o modificados.
- 6 Documentación técnica: se realiza la documentación correspondiente a cada componente de software.
- 7 Sistema de Control de Versiones (CVS): Se entrega los componentes desarrollados y/o modificados al CVS.
- 8 Reporte de actividades: Reportar las actividades realizadas.

### 3.4 Pruebas

El propósito del proceso de pruebas es encontrar la mayor cantidad posible de errores en los componentes del software.

En este proceso se realizaron formatos e instructivos para realizar el plan, los casos y los procedimientos de prueba, donde contiene la información necesaria para realizar la documentación de las pruebas.

El proceso de pruebas básicamente está compuesto por las siguientes fases:

- 1 Técnicas de diseño de casos de prueba: se utilizan para obtener una confianza aceptable en que se detectarán los defectos existentes.
  - 1.1 Pruebas de Aceptación: Se realiza los casos de prueba que el cliente requiere en el software.
  - 1.2 Pruebas Funcionales o de caja negra: Se identifican las clases de equivalencia y se crean los casos de prueba correspondientes.
- 2 Diseño de Pruebas
  - 2.1 Plan de Pruebas: Se enfoca en detallar el modelo de prueba para los componentes de software.
  - 2.2 Especificación de los casos de prueba: Define uno de los casos de prueba identificado por una especificación del diseño de las pruebas.
  - 2.3 Especificación de los procedimientos de prueba: Especifica los pasos para la ejecución de un conjunto de casos de prueba o, más generalmente, los pasos utilizados para analizar un elemento software con el propósito de evaluar un conjunto de características del mismo.
- 3 Ejecución de las pruebas
  - 3.1 Histórico de Pruebas: Documenta todos los hechos relevantes ocurridos durante la ejecución de las pruebas.
  - 3.2 Informe resumen: Resume los resultados de las actividades de prueba (las reseñadas en el propio informe) y aporta una evaluación del software, basada en dichos resultados.

#### 4. CONCLUSIONES

- 1 Las metodologías de software son fundamentales en una empresa de desarrollo de software permitiendo definir reglas, procedimientos, actividades, tareas, y herramientas de software, para cada etapa de desarrollo de nuevos proyectos, y además permite que las personas trabajen mejor y sea propende de errores.
- 2 No es fácil aplicar una nueva metodología en un equipo de desarrollo ya que obliga a aprender una

nueva forma de trabajar. Además, obliga a abandonar cómo se hacían las cosas antes, que aunque no fuera la mejor forma posible, es necesario adaptarse al cambio.

- 3 El tipo de desarrollos en general es creación de software a la medida del cliente, y hay numerosas opiniones que relatan el éxito de una metodología en este ámbito. Queda por ver si es posible aplicar estas ideas también en procesos de desarrollo muy diferentes, como el software libre.
- 4 RUP contempla más rigurosidad en cuanto al desarrollo de software y permite establecer los requerimientos y casos de uso de manera iterativa, aunque algunas personas utilizan el RUP con el modelo cascada, pero la ahora existe UP ágil, donde se produce código y se hacen pruebas, contemplando procesos generales y procesos de soporte que otras metodologías ágiles no lo aplican.
- 5 Para la empresa adaptarse a los procesos desarrollados va a llevar tiempo y capacitación para las personas que conforman el equipo de desarrollo, y se necesitara el apoyo de los altos directivos para realizarlo.
- 6 En el proceso de requerimientos este proceso es fundamental porque aquí se produce la información necesaria para comenzar el desarrollo de software, en esta etapa evaluamos de la mejor manera la captura de los requerimientos haciendo posible la trazabilidad de este proceso, y comprobamos que es la etapa fundamental para la empresa ya que de eso depende que su construcción sea lo mejor realizada posible. Los casos de uso forman parte especial porque determinan el procedimiento que se va a construir, y las pruebas para el software.
- 7 La validación del proceso de requerimientos que determinada por validar cada caso de uso y requerimiento que el cliente solicitó, esta validación se realizará por medio de la firma del cliente y la fecha.
- 8 Evaluando las métricas para el proceso de requerimientos arrojo como resultado un proceso adaptable para el contexto de la empresa, aunque con pequeñas fallas en los campos de los formatos e instructivos que no fueron útiles, pero sin mayor importancia.
- 9 Análisis y Diseño es uno de los procesos que no se realizaban en la empresa, por este motivo el

equipo de desarrollo le tomara tiempo adaptarse a los nuevos procedimientos que se deben realizar cuando se este desarrollo un nuevo aplicativo. Para este proceso se involucraron diagramas que permiten tener una interpretacion grafica de las características del software, posicionando como su principal fuente el modelo de datos.

8. Construcción es el proceso que se considera más adaptable a las necesidades actuales, debido a que se centra los componentes de software, la base de datos, y a la implantación de herramientas de software que ayudan a gestionar el proceso.
9. El proceso de pruebas se encuentra es su estado de mejora y debe realizarse un estudio mas exhaustivo de este proceso.

#### REFERENCIAS

Booch, Grady. Rambaugh, James. Jacobson, Ivar. *El Proceso Unificado de Desarrollo de Software*.

Booch, Grady. *El lenguaje Unificado de Modelado*, Primera Edición, Editorial Addison Wesley, 1999.

Larman, Craig. *UML y Patrones Una introducción al Análisis y Diseño Orientado a Objetos y al Proceso Unificado*, Segunda Edición, Editorial Prentice Hall, 2002.

Jacobson, Ivar. *El Proceso Unificado de Modelado*, Primera Edición, Editorial Addison Wesley, 1999.

The Capability Maturity Model Integration, CMMI<sup>SM</sup> for Systems Engineering and Software Engineering (CMMI-SE/SW, V1.1) CMMI, Product Team, December 2001.

IEEE STD 830 1998, IEEE Recommended Practice for Software Requirements Specifications.

Modelo de Procesos para la industria del Software, Moprosoft v1.1, Mayo 2003.

Modelo IDEAL

<http://www.sei.org>

Martin Fowler

<http://www.martinfowler.com>

Programación eXtrema y Software Libre

<http://ultimaorbita.com/raciel/x-ezine/x2/2x010-XP.html>

TodoAgil

[www.willydev.net/descargas/prev/TodoAgil.Pdf](http://www.willydev.net/descargas/prev/TodoAgil.Pdf)

[Otros Links](#)

<http://www.fing.edu.uy/inco/cursos/gestsoft/ppts/XP.ppt>

<http://www.willydev.net/descargas/Articulos/General/IntroXP.PDF>

<http://www.dcc.uchile.cl/~lugerrecc61jrecursos/clase2.ppt>

Santiago de Cali, 27 de Enero de 2006

Doctor  
Jaime Campo Rodríguez  
Director Programa de Ingeniería Informática  
UAO.

Con la presente me permito comunicarle que el siguiente informe final de Pasantia, titulada "Definición de la metodología de desarrollo de software y documentación de los procesos que la componen en IPSOFT-S.A.", del cual soy directora académica del proyecto, desarrollado por el estudiante LUIS ALEJANDRO VARGAS MUÑOZ - CÓDIGO: 1007329, en la Empresa: IPSOFT-S.A, con una duración de cuatro meses; cumple satisfactoriamente en contenido y forma con lo planteado inicialmente en el anteproyecto.

Considerando lo anterior, ratifico que este proyecto ha sido revisado y aprobado por cumplir con los estándares de un proyecto de opción de grado.

Atentamente,

-----  
Mary Elizabeth Ramirez  
Directora Académica del Proyecto  
Teléfono: 3188000 ext. 11359

Santiago de Cali, 27 de Enero de 2006

Doctor

**Jaime Campo Rodríguez**

Director Programa de Ingeniería Informática  
UAO.

Con la presente me permito comunicarle que el siguiente informe final de la pasantía titulada "Definición de la metodología de desarrollo de software y documentación de los procesos que la componen en IPSOFT-S.A.", desarrollado por el estudiante LUIS ALEJANDRO VARGAS MUÑOZ - CÓDIGO: 1007329, en la Empresa: IPSOFT S.A, con una duración de cuatro meses; de la cual soy el asesor empresarial, cumple satisfactoriamente en contenido inicialmente estipulado para el desarrollo del proyecto en la empresa.

Cordialmente,

-----  
Gustavo Adolfo Rojas

Subgerente

Teléfono: 6603000 ext. 132