

**SISTEMA DE INFORMACION WEB PARA PROPIEDAD HORIZONTAL  
“SIPHO”**

**Luz Adriana Bravo Castro**

**ID 000052248**

**Laura Andrea Orjuela Díaz**

**ID 000072747**

**Elsy Bonilla Figueroa**

**ID 000292064**

**UNIVERSIDAD UNIMINUTO**

**Tecnología en informática**

**Soacha Cundinamarca**

**2015**

**SISTEMA DE INFORMACION WEB PARA PROPIEDAD HORIZONTAL  
“SIPHO”**

**Luz Adriana Bravo Castro**

**ID 000052248**

**Laura Andrea Orjuela Díaz**

**ID 000072747**

**Elsy Bonilla Figueroa**

**ID 000292064**

**Trabajo de grado dirigido por**

**JULIO EDUARDO JEJEN**

**Ingeniero**

**UNIVERSIDAD UNIMINUTO**

**Tecnología en informática**

**Soacha Cundinamarca**

**2015**

NOTA DE ACEPTACIÓN

---

---

---

**Presidente del Jurado**

---

**Jurado**

---

**Jurado**

---

Soacha, 11 de Junio de 2015

## AGRADECIMIENTOS

El proyecto SIPHO es el resultado del esfuerzo conjunto de quienes participamos activamente en él. Inicialmente agradecemos a Dios por darnos las capacidades necesarias para llegar a ser personas íntegras. Al ingeniero Julio Eduardo Jején, quien con sus sabias orientaciones nos ayudó a hacer nuestro mejor esfuerzo para lograr la meta propuesta. A nuestros familiares quienes a lo largo de toda nuestra vida han apoyado y motivado nuestra formación académica y creen en nosotros en todo momento. A nuestros profesores orientadores destacados a lo largo de toda nuestra formación a quienes les debemos gran parte de nuestros conocimientos, gracias a su paciencia y enseñanza y un eterno agradecimiento a esta prestigiosa universidad la cual nos abrió sus puertas y nos brindó un ambiente de acogida y familiaridad.

Y finalmente a nuestros compañeros con quienes compartimos valiosos momentos que llevaremos grabados en lo más profundo de nuestro corazón.

## DEDICATORIA

Este proyecto se lo dedicamos principalmente a cada una de las personas que nos colaboraron de una u otra forma en su culminación, inicialmente a Dios por permitimos lograr este paso tan importante para nuestra superación personal y profesional, adicional a cada una de nuestra familias, quienes siempre estuvieron apoyándonos incondicionalmente, a nuestros tutores en cada una de las diferentes materias, ya que con su aporte nos colaboraron en la aclaración de conocimientos y aplicaciones al proyecto y finalmente a cada a una de nosotras, por nuestro esfuerzo y dedicación en el sentido de querer salir adelante, como personas y profesionales.

## RESUMEN

El desarrollo de este trabajo sirve como apoyo a la administración de propiedad horizontal. Este sistema de información cumple con la reglamentación para las organizaciones de propiedad horizontal ley 675 del 2001, se da solución a la problemática del tratamiento manual de la información y pagos. A través del planteamiento, diseño y desarrollo de un sistema de información que permita el registro, control y visualización de la información relacionada con la administración del Conjunto Residencial Quintas El Portal.

Este sistema de información SIPHO cuenta con los siguientes módulos: Registro y logueo de usuarios, una base de datos en MySQL para el ingreso de pago de cuotas, registro de copropietarios, servicios y comunicados de la administración, formulario para el pago de administración, visualización y solicitud de los servicios ofrecidos a los copropietarios por el conjunto residencial Quintas El Portal, mostrar comunicados administrativos, circulares y notas de interés a los residentes y/o copropietarios, consultas de pagos de administración y consultas de disponibilidad de servicios.

El sistema de información SIPHO está diseñado en el framework Bootstrapp, lenguaje web HTML5 y lenguaje de código abierto php.

El manejo de los documentos queda soportado por documentos electrónicos, muy similares a los de papel, facilitando su conservación y utilización. De igual manera permite acceder a la información de forma inmediata y confiable.

La integridad de los datos se sustenta en procesos que ejecuta la base de datos.

## **ABSTRACT**

The development of this work serves as support for the administration of condominiums. This information system complies with the regulations for organizations condo law 675 of 2001, it solves the problem of manual processing of information and payments. Through the approach, design and development of an information system that allows the registration, control and display information related to the administration of Residential Quintas El Portal.

SIPHO This information system has the following modules: Registration and logging of users, a MySQL database for entry fee payment, registration of co-owners, services and news management, payment form management, visualization and demand for services offered to the investors by the residential Quintas El Portal, show administrative circulars and notes of interest communicated to residents and / or co-payments consultations administration and service availability inquiries.

SIPHO information system is designed in the framework Bootstrapp, web language HTML5 and PHP open source language.

The handling of documents is supported by electronic documents, very similar to the role, facilitating conservation and use. Likewise allows access to the information immediately and reliably.

The data integrity is supported by processes running the database.

## INTRODUCCIÓN

En las empresas la información es un recurso tan importante como los recursos financieros, humanos y materiales, exige un conocimiento del entorno, del mundo cada vez más complejo y cambiante. El sistema de información, es un proceso de planificación empresarial, es un factor clave para la toma de decisiones en las mismas, el objeto es conectar a un usuario con una fuente de información que necesita para satisfacer sus necesidades, es un conjunto de componentes que interactúan entre sí para lograr un objetivo común.

Teniendo en cuenta lo anterior, y ante los constantes avances significativos en el ramo de la Tecnología, es que vimos la necesidad de implementar un Sistema de Información Web para propiedad horizontal a nivel municipal en Soacha – Cundinamarca, teniendo en cuenta la demanda de habitantes que actualmente posee el municipio.

Este Sistema de Información Web para Propiedad Horizontal permitirá que un conjunto residencial muestre su organización, comparta los diferentes servicios que ofrece y genere reportes sobre estados de pagos administrativos.



## TABLA DE CONTENIDO

	Pág.
PORTADA	1
CONTRAPORTADA	2
NOTA DE ACEPTACIÓN	3
AGRADECIMIENTO	4
DEDICACIÓN	5
RESUMEN	6
ABSTRACT	7
INTRODUCCIÓN	8
TABLA DE CONTENIDO	9
1. JUSTIFICACIÓN	12
2. MISIÓN	13
2.1. Del proyecto	13
2.2. De la empresa	13
3. VISIÓN	14
3.1. Del proyecto	14
3.2. De la empresa	14
4. OBJETIVO	15
4.1. General	15
4.2. Específicos	15
5. ESTUDIO DE CAMPO	16
5.1. Visita al terreno	16
5.2. Encuestas	18
5.3. Entrevistas	20
6. MARCO TEÓRICO	22
7. Marco Histórico	24
8. Marco Referencial	25
9. Marco Legal	27

10. Marco Conceptual	31
11. CICLOS DE VIDA DEL SISTEMA	36
11.1 Ciclo de vida clásico	37
11.2. Modelo en espiral	39
11.3. Modelo en cascada	41
11.4. Modelo orientado a objetos	44
12. METODOLOGÍAS	47
12.1. Metodología RUP	47
12.2. Metodología RAD	51
12.3. Metodología XP	53
13. UML	59
14. MODELOS	62
14.1. Casos de uso	62
14.2. Diagrama de clases	69
14.3. Diagrama de estados	72
14.4. FLUJOGRAMA DE PROCESOS	74
14.5. ORGANIGRAMA DE LA EMPRESA	76
14.6. DIAGRAMA DE FLUJO DE DATOS	77
15. MODELO DE DATOS	80
15.1. Modelo Entidad Relación	80
15.2. Modelo Relacional	81
15.3. Modelo Tabular	82
16. VIABILIDAD O FACTIBILIDAD	83
16.1. Técnica	83
16.2. Humana	84
16.3. Legal	84
16.4. Financiera	84
17. COSTOS	85
17.1. Fijos	85
17.2. Variables	85

17.3. Utilidad	86
17.4. Valor del proyecto	86
18. Matriz DOFA	87
19. DICCIONARIO DE DATOS	88
20. GLOSARIO	89
21. CIBERGRAFIA	90
ANEXOS	

## 1. JUSTIFICACIÓN

De acuerdo a los avances tecnológicos existen muchos conjuntos residenciales que cuentan con sistemas de información, pero hay otros que en la actualidad el tratamiento de los datos se hace de forma manual causando retrasos en los reportes y estados financieros de los conjuntos residenciales.

Actualmente se observa que el conjunto residencial Quintas El Portal no cuenta con ningún sistema de información adecuado para el desarrollo de los procesos básicos de la administración, generando problemas como:

- Retrasos e inconsistencias en el registro manual del pago administrativo.
- No hay un medio centralizado para el envío de los comunicados de la administración a los copropietarios o residentes, generando inasistencias y falta de respuestas asertivas a los mismos.
- Por ser conjuntos residenciales nuevos la mayoría de los copropietarios o residentes no conocen los servicios a los cuales pueden acceder o su disponibilidad y la información de este tema es deficiente.
- Las PQR (peticiones, quejas y reclamos) se registran a través de un formato físico que no es controlado, por lo que la respuesta maneja tiempos inadecuados.

Conscientes que en la actualidad el incremento del uso de la tecnología es uno de los aspectos fundamentales del crecimiento comercial y mejora en servicios, se busca implementar un software que vaya al ritmo de los actuales avances en la solución de vivienda, utilizando los conocimientos adquiridos en la Tecnología en Informática de "UNIMINUTO".

## **2. MISIÓN**

### **2.1 DEL PROYECTO**

El proyecto de Sistema de Información Web para Propiedad Horizontal “SIPHO” inspirado en la necesidad de globalización, busca brindar una valiosa herramienta de administración y organización al Conjunto Residencial Quintas El Portal, con el fin de mantener disponible a cualquier momento la información de tipo administrativo, social y otros de su interés, ofreciendo calidad, seguridad y transparencia en el desarrollo de los procesos.

### **2.2 DE LA EMPRESA**

Conformar un espacio de vivienda organizado que propicie espacios de comunicación asertiva y sana convivencia; en el cual reine la armonía, la paz, el respeto y la equidad para todos los propietarios siguiendo los lineamientos establecidos en nuestro reglamento con el fin de lograr un desarrollo humano sostenible.

### **3. VISIÓN**

#### **3.1. DEL PROYECTO**

Partiendo del momento histórico de transformaciones tecnológicas para el año 2016 se contará con el sistema de información SIPHO que responda a un 90% de la gestión administrativa para conjuntos residenciales a nivel municipal en Soacha - Cundinamarca.

#### **3.2 DE LA EMPRESA**

Constituirse en el municipio de Soacha como ejemplo a seguir desde el punto de vista de organización residencial, así mismo motivo de orgullo de todos los propietarios por ser reconocidos como el mejor conjunto residencial del sector, comprometido con el progreso y caracterizado por la integridad de todos los Copropietarios.

## **4. OBJETIVOS**

### **4.1. OBJETIVOS GENERAL**

Desarrollar un Sistema de Información Web para Propiedad Horizontal “SIPHO”, que permita el registro, control y visualización de la información relacionada con la administración del Conjunto Residencial Quintas El Portal.

### **4.2. OBJETIVOS ESPECÍFICOS**

- 4.2.1.** Crear un módulo que permita el registro y logueo de usuarios.
- 4.2.2.** Diseñar e implementar una base de datos, en MySQL para el ingreso de pago de cuotas, registro de copropietarios, servicios y comunicados de la administración.
- 4.2.3.** Elaborar un formulario para el pago de Administración.
- 4.2.4.** Diseñar un módulo que permita visualizar y solicitar los servicios ofrecidos a los copropietarios por el conjunto residencial Quintas El Portal.
- 4.2.5.** Mostrar en un módulo, los diferentes comunicados administrativos, circulares y notas de interés a los residentes y/o copropietarios.
- 4.2.6.** Generar un módulo de consultas de pagos de administración.
- 4.2.7.** Generar un módulo de consultas de disponibilidad de servicios.

## 5. ESTUDIO DE CAMPO

Para llevar a cabo el proceso de reconocimiento de la problemática que vamos a tratar en el desarrollo de este proyecto, se hace necesario realizar algunas actividades, las cuales detallaremos para su mejor comprensión:

### 5.1. VISITA AL TERRENO

Se realiza reconocimiento físico al Conjunto Residencial Quintas El Portal, en donde se observa que se encuentra ubicado en Soacha Cundinamarca, dirección actual Calle 1 No. 4F-45, se encuentra en zona central del Municipio, alrededor de él se encuentra los sectores de Quintas de La Laguna, Bosques de Sapan, Lagos de Malibu.



Este conjunto consta de Cuatro Interiores de Casas, cada interior contempla 24 casas y 3 Interiores de Apartamentos, cada uno con 20 apartamentos para un total de 156 viviendas.







Adicional consta de: La Portería, dos zonas de parqueaderos (motocicletas y vehículos), un salón comunal, una capilla, una cancha deportiva y una zona de juegos para niños menores de 5 años.



## 5.2. ENCUESTAS

Se realiza otra actividad con el fin de dar a conocer el proyecto a la comunidad interesada, en donde es recibido con gran acogida, pues varios de los copropietarios se encontraron interesados en que en el Conjunto Residencial Quintas El Portal se pueda implementar este Sistema de Información Web. Diseño encuesta:

### **“SISTEMA DE INFORMACION PROPIEDAD HORIZONTAL “CONJUNTO QUINTAS DEL PORTAL”**

Cordial saludo, señor copropietario:

Con el fin de implementar un sistema de información web del Conjunto Residencial “Quintas del Portal”, realizaremos la encuesta anexa, ya que para el desarrollo de este proyecto es importante conocer sus observaciones y punto de vista.

1. ¿Está usted de acuerdo con implementar en el Conjunto Residencial “Quintas del Portal”, una página web del mismo, en donde además de dar a conocer nuestro sector nos permita indagar en varios servicios?

SI 

X
---

  
NO 

--

  
PORQUE \_\_\_\_\_

2. Una de las funcionalidades de esta página, es que a través de esta, usted pueda generar información sobre el estado actual de su deuda con la parte administrativa, adicional que una vez usted se acerque a cancelar le sea generado el recibo de pago en línea, ¿Qué opina usted?

SI  X  
NO   
PORQUE \_\_\_\_\_

3. Adicional, se implementará para control y proceso administrativos una base de datos para que el Administrador ingrese la información únicamente de pago de cuotas a través de este sistema. ¿Estaría usted de acuerdo, de que la administración del conjunto empiece a implementar herramientas tecnológicas en su proceso?

SI  X  
NO   
PORQUE \_\_\_\_\_

4. Por otra, parte se permitirá que los copropietarios puedan realizar consultas en línea desde cualquier lugar a documentos como actas, acuerdos, comunicados de servicios sociales, entre otros. ¿Qué opina usted?

SI  X  
NO   
PORQUE \_\_\_\_\_

5. Igualmente se incluye un espacio, en el cual los copropietarios, van a poder ingresar observaciones, quejas o reclamos en línea, adicional de que por este mismo medio pueda recibir la respuesta respectiva. ¿Estaría usted de acuerdo en este proceso?

SI  X

NO  
PORQUE

Atentamente:

Copropietario. **LUIS GERMAN MOJICA PORTELA**

...”

### 5.3. ENTREVISTAS

Como última gestión al reconocimiento del campo en el cual se basa el SIPHO, hemos logrado realizar una entrevista vía telefónica con la actual Administradora de Conjunto Residencial Quintas El Portal, señora SOR MARIA ALARCON ALBA, quien nos prestó su colaboración en indicarnos el por qué le parece un tema interesante el implementar un Sistema de Información Web en un conjunto residencial textualmente así:

“... Yo me desempeño como Administradora de Conjuntos Residenciales en varios sectores, uno de ellos es Quintas El Portal, en donde actualmente llevé trabajando seis años de experiencia, y sería de mucha utilidad en implementar un Sistema de Información Web, como me lo están explicando las estudiantes de la Universidad Uniminuto, ya que facilitaría muchísimo la comunicación entre los copropietarios y la mía, me llama la atención el hecho de que a través de este se puedan ingresar todos los comunicados que sean de importancia para la comunidad del conjunto, adicional el hecho de poder recibir peticiones sobre los servicios que el conjunto ofrece no únicamente a los copropietarios sino también a las personas externas del mismo, y más aún es super novedoso el hecho de que se pueda generar el recibo de pago administrativo a través de este sistema y que las copropietarios puedan consultar el estado de sus deudas pendientes. La verdad que este sistema cumpliría con la solución a un problema

administrativo, pues apoyaría en gran parte a la labor administrativa por lo cual estoy totalmente de acuerdo con ello. “

## 6. MARCO TEÓRICO

Día a día se ve un auge en la construcción de vivienda, debido a esto surge la necesidad de llevar un control adecuado de la administración de la propiedad horizontal, respondiendo a esta necesidad se ha diseñado diferentes tipos de software que realizan esta tarea, pero solo interactúa con el administrador ya que es un software local, estos software han contemplado los módulos de contabilidad, documentación y cartera, entre los cuales podemos encontrar, Milenio, DeB Propiedad Horizontal, SOFTWAREING, HelisaNIIF.

En entrevistas y encuestas realizadas a diferentes administradores y residentes de conjuntos residenciales, sobre el manejo y control de los procesos básicos de administración, se ve reflejado que la mayoría de administradores no cuentan con un SI que respondan a una óptima realización de los procesos a su responsabilidad, ya que la mayoría cuenta con un control rudimentario (excel, hojas de comunicados, recibos a mano), por otro lado los residentes se encuentran poco informados de las diferentes actividades o reuniones de su conjunto, ya que muchos salen desde muy temprano a su trabajo y llegan tarde del mismo, por lo que estar informados es más complicado. Para realizar toda la investigación de este proyecto se utilizaron las siguientes herramientas metodológicas: estudio de campo, en el cual analizamos cómo funciona administrativamente el conjunto residencial Quintas el Portal.

Utilizamos metodología RUP, casos de usos, diagrama de clases, modelado de datos., diagramas de estados, etc.

A la pregunta ¿Creen importante un Sistema de Información Web que les permita estar enterados de diferentes actividades, servicios y reuniones desde cualquier ubicación?

Tanto administradores como residentes están de acuerdo que un SI web les permitirá tener más acercamiento a su conjunto y poder participar más en eventos citados, ya que contarían con información propicia para permisos.

## 7. MARCO HISTÓRICO

Quintas El Portal surgió de un proyecto de vivienda de interés social en el año 2.003 en Soacha Cundinamarca, cuando la firma constructora AMPITEC, la fundadora de la Ciudadela Colsubsidio de la 80 en Bogotá, busco implementar un esquema similar a la construcción de casas y apartamentos en un conjunto residencial al sur de la ciudad, proyecto este que fue presentado ante las entidades respectivas del municipio de Soacha, como son Planeación y Catastro, quienes lo avalaron y dieron el visto bueno a los planos arquitectónicos presentados, logrando así la meta de iniciar las construcciones respectivas. El terreno fue avalado, y aprobado apto para la construcción de estas viviendas. Se hace la primera entrega de casas con el interior 1 en el año 2004 y así sucesivamente se lograron las construcciones de las diferentes etapas y sus entregas. Estas viviendas son de interés social nivel 3 y cuenta con los servicios necesarios.

Barrios Colindantes : **Quintas** de San Luis, **Soacha** Centro, Satélite, Quintas de La Laguna, Bosques de Sapan y Lagos de Malibu.



## 8. MARCO REFERENCIAL

Mediante entrevistas y encuestas realizadas a diferentes administradores y residentes de conjuntos residenciales, sobre el manejo y control de los procesos básicos de administración, pudimos consatarla necesidad de un sistema de información que permita tanto a administradores como copropietarios estar en permanente contacto con aspectos esenciales de su vida comunitaria.

El sistema de información SIPHO busca ser una solución que haga más armoniosa la ardua labor del administrador.

En el conjunto residencial quintas del portal, viven personas dedicadas en la mayor parte de su tiempo a sus actividades laborales, por este motivo con poco o nada de tiempo para las actividades comunitarias que exige la convivencia de un conjunto residencial.

El proyecto sistema de información SIPHO será una herramienta eficaz para mantener intercomunicados a los copropietarios con el administrador, además tener una comunicación en tiempo real de aspectos como: servicios ofrecidos por el conjunto residencial, prestamos de salón comunal y capilla, disponibilidad de canchas deportivas, además de circulares y en general aspectos relevantes para su bienestar.

El sistema de información SIPHO es una herramienta de fácil manejo y amigable en todos sus aspectos. Contará con los módulos de logueo de usuarios, pago de cuotas de administración, servicios que presta el conjunto, comunicados administrativos, generación de consulta de pagos y generación de recibos de pago.

Este sistema de información tendrá un impacto positivo en la comunidad circundante del municipio de Soacha y así llegar a ser una solución a otros conjuntos residenciales.

## 9. MARCO LEGAL

La propiedad horizontal en Colombia está regida por la ley 675 del 2001, cuyo objeto es: “La presente ley regula la forma especial de dominio, denominado propiedad horizontal, en la que concurren derechos de propiedad exclusiva sobre bienes privados y derechos de copropiedad sobre el terreno y los demás bienes comunes, con el fin de garantizar la seguridad y la convivencia pacífica en los inmuebles sometidos a ella, así como la función social de la propiedad. Texto declarado EXEQUIBLE por la Corte Constitucional mediante Sentencia C-318 de 2002”.

Son principios orientadores de la presente ley: “1. Función social y ecológica de la propiedad. Los reglamentos de propiedad horizontal deberán respetar la función social y ecológica de la propiedad, y por ende, deberán ajustarse a lo dispuesto en la normatividad urbanística vigente.

2. Convivencia pacífica y solidaridad social. Los reglamentos de propiedad horizontal deberán propender al establecimiento de relaciones pacíficas de cooperación y solidaridad social entre los copropietarios o tenedores.

3. Respeto de la dignidad humana. El respeto de la dignidad humana debe inspirar las actuaciones de los integrantes de los órganos de administración de la copropiedad, así como las de los copropietarios para el ejercicio de los derechos y obligaciones derivados de la ley.

4. Libre iniciativa empresarial. Atendiendo las disposiciones urbanísticas vigentes, los reglamentos de propiedad horizontal de los edificios o conjuntos de uso comercial o mixto, así como los integrantes de los órganos de administración correspondientes, deberán respetar el desarrollo de la libre iniciativa privada dentro de los límites del bien común.

5. Derecho al debido proceso. Las actuaciones de la asamblea o del consejo de administración, tendientes a la imposición de sanciones por incumplimiento de

obligaciones no pecuniarias, deberán consultar el debido proceso, el derecho de defensa, contradicción e impugnación.”

Algunas leyes y decretos que reglamentan la actividad de propiedad horizontal son:

➤ Ley 182 de 1948 “Sobre régimen de la propiedad de pisos y departamentos de un mismo edificio”.

➤ Ley 66 de 1968 “El Gobierno Nacional, a través del Superintendente Bancario ejercerá la inspección y vigilancia de las actividades relacionadas con la enajenación de inmuebles destinados a vivienda y sobre el otorgamiento de créditos para la adquisición de lotes o viviendas o para la construcción de las mismas. La transferencia del dominio a título oneroso de viviendas en unidades independientes o sometidas al régimen de propiedad horizontal.”

➤ Decreto 1380 de 1972 “Por el cual se dictan unas medidas reglamentarias sobre inscripción de parcelaciones y urbanizaciones en el folio de matrícula inmobiliaria de que trata el Decreto Ley número 1250 de 1970, en desarrollo de lo dispuesto en los artículos 49 y 50 de dicha norma y en el artículo 5° de la Ley 66 de 1968”

➤ 1644 de 1978 “Los Notarios deberán autorizar y los Registradores deberán inscribir en el registro, las escrituras de enajenación de inmuebles que formen parte de un plan o programa de vivienda en los términos de la Ley 665 de 1968...”

➤ Ley 16 de 1985 “La llamada propiedad horizontal, que se rige por las normas de la Ley 182 de 1948 y del presente estatuto, es una forma de dominio que hace objeto de propiedad exclusiva o particular determinadas partes de un inmueble y sujeta las áreas de éste destinadas al uso o servicio común de todos o parte de los propietarios de aquéllas al dominio de la persona jurídica que nace conforme con las disposiciones de esta Ley.”

➤ 1941 de 1986 “Asignase al Ministerio de Desarrollo Económico las funciones de vigilancia y control que le fueron otorgadas a la Superintendencia Bancaria, sobre las siguientes personas e instituciones: Personas naturales y jurídicas

dedicadas a la actividad de enajenación de inmuebles destinados a vivienda, de conformidad con lo dispuesto en la Ley 66 de 1968 y el Decreto 2610 de 1979...”

➤ Decreto reglamentario N° 1365 de noviembre 28 de 1986: “Podrán someterse al régimen de propiedad horizontal que establecen las Leyes 182 de 1948 y 16 de 1985, tanto los edificios de uno o varios pisos, como los grupos de edificios que constituyan un conjunto, construido o por construirse, sobre el mismo terreno, que sean susceptibles de división en unidades privadas independientes con salida directa a la vía pública o por áreas destinadas al uso común.”

➤ La Ley 810 del 2003 “Por medio de la cual se modifica la Ley 388 de 1997 en materia de sanciones urbanísticas y algunas actuaciones de los curadores urbanos y se dictan otras disposiciones.”

➤ Acuerdo 79 del código de policía agosto 13 del 2003 “ Este Código comprende las reglas mínimas que deben respetar y cumplir todas las personas en el Distrito Capital para propender por una sana convivencia ciudadana.”

## **ORGANOS DE DIRECCIÓN Y ADMINISTRACIÓN DE LA PROPIEDAD HORIZONTALES SEGÚN LA LEY 675 DE 2001.**

**Asamblea General de copropietarios:** Asamblea General es la máxima autoridad de la copropiedad, y está conformada por los copropietarios “o sus representantes” reunidos en la forma en que se indica en la ley y en el reglamento de Propiedad Horizontal, las decisiones tomadas por la Asamblea son obligatorias para todos los copropietarios, incluso ausentes y disidentes, para el Administrador y revisor Fiscal, etc.

La Asamblea debe reunirse por lo menos una vez al año, en la fecha que señale el reglamento para aprobar balances, estados financieros y el presupuesto la cual permitirá fijar la cuota de Administración para el año.

**Junta directiva:** Es el organismo representativo de los copropietarios, autorizados para planificar, coordinar, organizar y ejecutar aquellos asuntos y trabajos que surjan dentro de la comunidad, para bien de la misma y conservación del conjunto.

**Revisor fiscal:** No se limita únicamente al análisis de cuentas y arqueos, sino que debe cerciorarse de que todas, las operaciones realizadas en el conjunto residencial, comercial o mixto, estén ajustadas a las leyes, reglamento de propiedad horizontal.

**Administrador:** Persona natural o jurídica elegida por la asamblea general de copropietarios en edificios o conjuntos residenciales para la representación legal de una persona o empresa, Los actos y contratos que celebre en ejercicio de sus funciones, se radican en cabeza de la persona jurídica, siempre y cuando se ajusten a las normas legales y reglamentarias.

## 10. MARCO CONCEPTUAL

Las definiciones más relevantes del presente trabajo son:

**Régimen de Propiedad Horizontal:** Sistema jurídico que regula el sometimiento a propiedad horizontal de un edificio o conjunto, construido o por construirse.

**Reglamento de Propiedad Horizontal:** Estatuto que regula los derechos y obligaciones específicas de los copropietarios de un edificio o conjunto sometido al régimen de propiedad horizontal.

**Edificio:** Construcción de uno o varios pisos levantados sobre un lote o terreno, cuya estructura comprende un número plural de unidades independientes, aptas para ser usadas de acuerdo con su destino natural o convencional, además de áreas y servicios de uso y utilidad general. Una vez sometido al régimen de propiedad horizontal, se conforma por bienes privados o de dominio particular y por bienes comunes.

**Conjunto:** Desarrollo inmobiliario conformado por varios edificios levantados sobre uno o varios lotes de terreno, que comparten, áreas y servicios de uso y utilidad general, como vías internas, estacionamientos, zonas verdes, muros de cerramiento, porterías, entre otros. Puede conformarse también por varias unidades de vivienda, comercio o industria, estructuralmente independientes.

**Edificio o conjunto de uso residencial:** Inmuebles cuyos bienes de dominio particular se encuentran destinados a la vivienda de personas, de acuerdo con la normatividad urbanística vigente.

**Edificio o conjunto de uso comercial:** Inmuebles cuyos bienes de dominio particular se encuentran destinados al desarrollo de actividades mercantiles, de conformidad con la normatividad urbanística vigente.

Edificio o conjunto de uso mixto: Inmuebles cuyos bienes de dominio particular tienen diversas destinaciones, tales como vivienda, comercio, industria u oficinas, de conformidad con la normatividad urbanística vigente.

Bienes privados o de dominio particular: Inmuebles debidamente delimitados, funcionalmente independientes, de propiedad y aprovechamiento exclusivo, integrantes de un edificio o conjunto sometido al régimen de propiedad horizontal, con salida a la vía pública directamente o por pasaje común.

Bienes comunes: Partes del edificio o conjunto sometido al régimen de propiedad horizontal pertenecientes en proindiviso a todos los propietarios de bienes privados, que por su naturaleza o destinación permiten o facilitan la existencia, estabilidad, funcionamiento, conservación, seguridad, uso, goce o explotación de los bienes de dominio particular.

Bienes comunes esenciales: Bienes indispensables para la existencia, estabilidad, conservación y seguridad del edificio o conjunto, así como los imprescindibles para el uso y disfrute de los bienes de dominio particular. Los demás tendrán el carácter de bienes comunes no esenciales. Se reputan bienes comunes esenciales, el terreno sobre o bajo el cual existan construcciones o instalaciones de servicios públicos básicos, los cimientos, la estructura, las circulaciones indispensables para aprovechamiento de bienes privados, las instalaciones generales de servicios públicos, las fachadas y los techos o losas que sirven de cubiertas a cualquier nivel.

Expensas comunes necesarias: Erogaciones necesarias causadas por la administración y la prestación de los servicios comunes esenciales requeridos para la existencia, seguridad y conservación de los bienes comunes del edificio o conjunto. Para estos efectos se entenderán esenciales los servicios necesarios, para el mantenimiento, reparación, reposición, reconstrucción y vigilancia de los bienes comunes, así como los servicios públicos esenciales relacionados con estos.



Coeficientes de copropiedad: Índices que establecen la participación porcentual de cada uno de los propietarios de bienes de dominio particular en los bienes comunes del edificio o conjunto sometido al régimen de propiedad horizontal. Definen además su participación en la asamblea de propietarios y la proporción con que cada uno contribuirá en las expensas comunes del edificio o conjunto, sin perjuicio de las que se determinen por módulos de contribución, en edificios o conjuntos de uso comercial o mixto.

Propietario inicial: Titular del derecho de dominio sobre un inmueble determinado, que por medio de manifestación de voluntad contenida en escritura pública, lo somete al régimen de propiedad horizontal.

Área privada construida: Extensión superficiaria cubierta de cada bien privado, excluyendo los bienes comunes localizados dentro de sus linderos, de conformidad con las normas legales.

Área privada libre: Extensión superficiaria privada semidescubierta o descubierta, excluyendo los bienes comunes localizados dentro de sus linderos, de conformidad con las normas legales.

Sistema de información: Un sistema de información es un conjunto de elementos que interactúan entre sí con el fin de apoyar las actividades de una empresa o negocio.

El equipo computacional: el hardware necesario para que el sistema de información pueda operar.

El recurso humano que interactúa con el Sistema de Información, el cual está formado por las personas que utilizan el sistema.

Un sistema de información realiza cuatro actividades básicas: entrada, almacenamiento, procesamiento y salida de información.

**Entrada de información:** Es el proceso mediante el cual el Sistema de Información toma los datos que requiere para procesar la información. Las entradas pueden ser manuales o automáticas. Las manuales son aquellas que se proporcionan en forma directa por el usuario, mientras que las automáticas son datos o información que provienen o son tomados de otros sistemas o módulos. Esto último se denomina interfaces automáticas.

Las unidades típicas de entrada de datos a las computadoras son las terminales, las cintas magnéticas, las unidades de diskette, los códigos de barras, los escáneres, la voz, los monitores sensibles al tacto, el teclado y el mouse, entre otras.

**Almacenamiento de información:** El almacenamiento es una de las actividades o capacidades más importantes que tiene una computadora, ya que a través de esta propiedad el sistema puede recordar la información guardada en la sección o proceso anterior. Esta información suele ser almacenada en estructuras de información denominadas archivos. La unidad típica de almacenamiento son los discos magnéticos o discos duros, los discos flexibles o diskettes y los discos compactos (CD-ROM).

**Procesamiento de Información:** Es la capacidad del Sistema de Información para efectuar cálculos de acuerdo con una secuencia de operaciones preestablecida. Estos cálculos pueden efectuarse con datos introducidos recientemente en el sistema o bien con datos que están almacenados. Esta característica de los sistemas permite la transformación de datos fuente en información que puede ser utilizada para la toma de decisiones, lo que hace posible, entre otras cosas, que un tomador de decisiones genere una proyección financiera a partir de los datos que contiene un estado de resultados o un balance general de un año base.

**Salida de Información:** La salida es la capacidad de un Sistema de Información para sacar la información procesada o bien datos de entrada al exterior. Las unidades típicas de salida son las impresoras, terminales, diskettes, cintas magnéticas, la voz, los

graficadores y los plotters, entre otros. Es importante aclarar que la salida de un Sistema de Información puede constituir la entrada a otro Sistema de Información o módulo. En este caso, también existe una interface automática de salida. Por ejemplo, el Sistema de Control de Clientes tiene una interface automática de salida con el Sistema de Contabilidad, ya que genera las pólizas contables de los movimientos procesales de los clientes.

## 11. CICLOS DE VIDA DEL SISTEMA

Los sistemas de información basados en computadoras, actualmente, son el corazón de la vida cotidiana en toda organización empresarial y principalmente en la toma de decisiones de las mismas. Este sistema de información debe cumplir dos objetivos principales: que sean un sistema correcto y que este correcto el sistema, deben adaptarse a su población de clientes, dar respuesta a sus necesidades, generar informes precisos y dar soporte a las actividades del negocio.

El ciclo de vida es un enfoque por fases del análisis y diseño que sostiene que los sistemas son desarrollados de mejor manera mediante el uso de un ciclo específico de actividades del analista y del usuario.

Según James Senn, existen tres estrategias para el desarrollo de sistemas: el método clásico del ciclo de vida de desarrollos de sistemas, el método de desarrollo por análisis estructurado y el método de construcción de prototipos de sistemas. Cada una de estas estrategias tiene un uso amplio en las empresas y resultan efectivas si son aplicadas correctamente.

## 11.1. CICLO DE VIDA CLÁSICO

El método de ciclo de vida para el desarrollo de sistemas es el conjunto de actividades que los analistas, diseñadores y usuarios realizan para desarrollar e implantar un sistema de información. El método del ciclo de vida para el desarrollo de sistemas consta de 6 fases:

1) Investigación preliminar: La solicitud para recibir ayuda de un sistema de información puede originarse por varias razones; el proceso se inicia siempre con la petición de una persona.

2) Determinación de los requerimientos del sistema: El aspecto fundamental del análisis de sistemas es comprender todas las facetas importantes de la parte de la empresa que se encuentra bajo estudio. Los analistas, al trabajar con los empleados y administradores, deben estudiar los procesos de una empresa para dar respuesta a las siguientes preguntas clave:

¿Qué es lo que hace? ¿Cómo se hace? ¿Con que frecuencia se presenta? ¿Qué tan grande es el volumen de transacciones o decisiones? ¿Cuál es el grado de eficiencia con el que se efectúan las tareas? ¿Existe algún problema? ¿Qué tan serio es? ¿Cuál es la causa que lo origina?

3) Diseño del sistema: El diseño de un sistema de información produce los detalles que establecen la forma en la que el sistema cumplirá con los requerimientos identificados durante la fase de análisis. Los especialistas en sistemas se refieren, con frecuencia, a esta etapa como diseño lógico en contraste con la del desarrollo del software, a la que denominan diseño físico.

4) Desarrollo del software: Los encargados de desarrollar software pueden instalar software comprobando a terceros o escribir programas diseñados a la medida del solicitante. La elección depende del costo de cada alternativa, del tiempo disponible para escribir el software y de la disponibilidad de los programadores.

5) Prueba de sistemas: Durante la prueba de sistemas, el sistema se emplea de manera experimental para asegurarse de que el software no tenga fallas, es decir, que funciona de acuerdo con las especificaciones y en la forma en que los usuarios esperan que lo haga.

Se alimentan como entradas conjunto de datos de prueba para su procesamiento y después se examinan los resultados.

6) Implantación y evaluación: La implantación es el proceso de verificar e instalar nuevo equipo, entrenar a los usuarios, instalar la aplicación y construir todos los archivos de datos necesarios para utilizarla. Una vez instaladas, las aplicaciones se emplean durante muchos años. Sin embargo, las organizaciones y los usuarios cambian con el paso del tiempo, incluso el ambiente es diferente con el paso de las semanas y los meses.

Por consiguiente, es indudable que debe darse mantenimiento a las aplicaciones. La evaluación de un sistema se lleva a cabo para identificar puntos débiles y fuertes. La evaluación ocurre a lo largo de cualquiera de las siguientes dimensiones:

\*Evaluación operacional: Valoración de la forma en que funciona el sistema, incluyendo su facilidad de uso, tiempo de respuesta, lo adecuado de los formatos de información, confiabilidad global y nivel de utilización.

\*Impacto organizacional: Identificación y medición de los beneficios para la organización en áreas tales como finanzas, eficiencia operacional e impacto competitivo. También se incluye el impacto sobre el flujo de información externo e interno.

\*Opinión de los administradores: evaluación de las actividades de directivos y administradores dentro de la organización así como de los usuarios finales.

\*Desempeño del desarrollo: La evaluación de proceso de desarrollo de acuerdo con criterios tales como tiempo y esfuerzo de desarrollo, concuerdan con presupuestos y estándares, y otros criterios de administración de proyectos. También se incluye la valoración de los métodos y herramientas utilizados en el desarrollo.



## 11.2. MODELO EN ESPIRAL

Es un modelo de ciclo de vida del software definido por primera vez por Barry Boehm en 1986. Las actividades de este modelo se conforman en una espiral, en la que cada bucle o interacción representa un conjunto de actividades. Las actividades no están fijadas a ninguna prioridad, sino que las siguientes se eligen en función del análisis de riesgo, comenzando por el bucle interior. Es un proceso de software evolutivo que acompaña la naturaleza evolutiva con los aspectos controlados y sistemáticos del ciclo de vida tradicional. Proporciona el potencial para el desarrollo rápido de versiones incrementales del software. En este *modelo*, el sistema se desarrolla en una serie de versiones incrementales. Durante las primeras iteraciones, la versión incremental podría ser un modelo en papel o un prototipo. Durante las últimas iteraciones se producen versiones cada vez más completas de ingeniería del sistema.

El Modelo en Espiral se divide en un número de actividades estructurales, también llamadas "regiones de tareas". Generalmente existen entre tres y seis regiones de tareas:

**Comunicación con el cliente.-** Las tareas requeridas para establecer comunicación entre el desarrollador y el cliente, sea revisar especificaciones, plantear necesidades, etc.

**Planificación.-** Las tareas requeridas para definir recursos, tiempos e información relacionada con el proyecto.

**Análisis de riesgos.-** Las tareas requeridas para evaluar riesgos técnicos y de gestión.

**Ingeniería.-** Las tareas requeridas para construir una o más representaciones de la aplicación

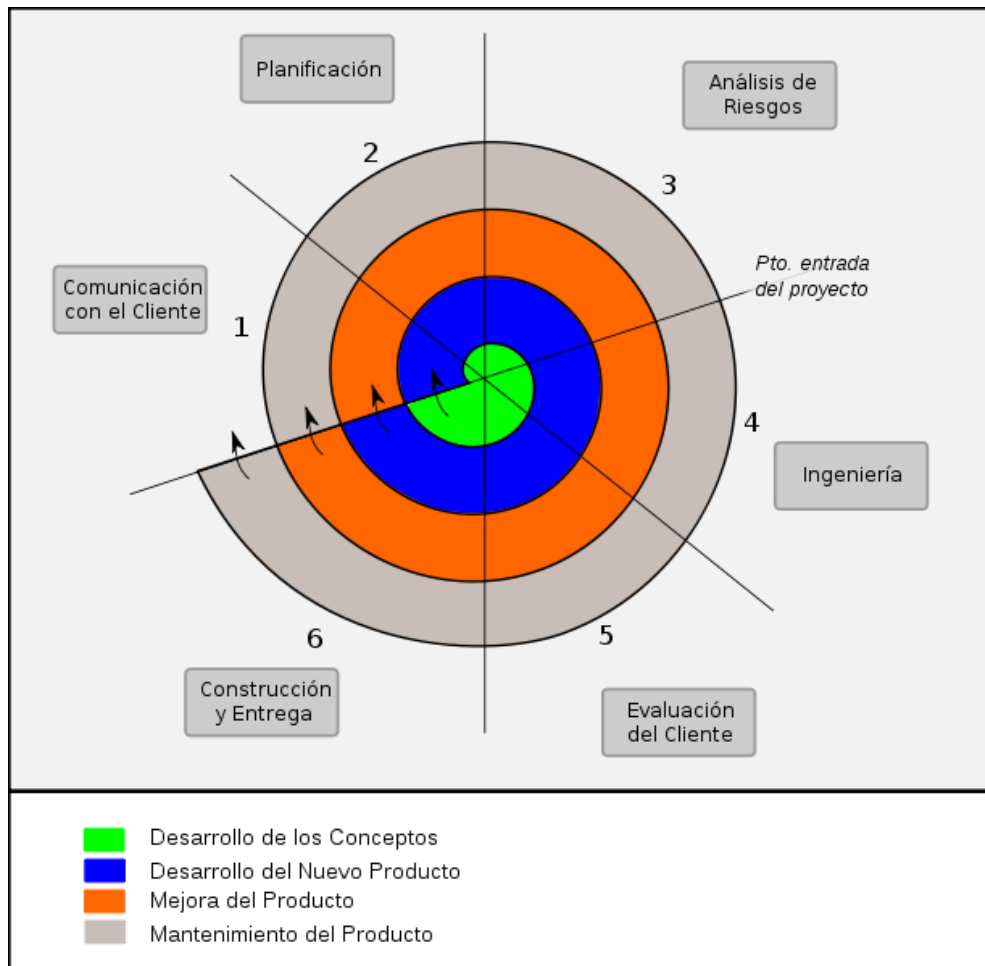
**Construcción y adaptación.-** Las tareas requeridas para construir, probar, instalar y proporcionar soporte al usuario.

**Evaluación del cliente.-** Las tareas requeridas para obtener la reacción del cliente, según la evaluación de las representaciones del software creadas durante la etapa de ingeniería e implementada durante la etapa de instalación

Cada una de las regiones está poblada por una serie de tareas que se adaptan a las características del proyecto que va a emprenderse. Para proyectos pequeños el número de tareas y su formalidad es bajo, para proyectos mayores y más críticos, cada región contiene tareas que se definen para lograr un nivel más alto de formalidad.

Cuando empieza este proceso evolutivo, el equipo de trabajo gira alrededor de las agujas del reloj, comenzando por el centro. El primer circuito de la espiral produce el desarrollo de una especificación de productos, los pasos siguientes en la espiral se podrían utilizar para desarrollar un prototipo y progresivamente versiones más sofisticadas del software. Cada paso de la región de planificación produce ajustes en el plan del proyecto. . El coste y la planificación se ajustan en función de la evaluación del cliente. Además, el gestor del proyecto ajusta el número planificado de iteraciones requeridas para completar el proyecto o el producto software de que se trate.





### 11.3. MODELO EN CASCADA

El modelo en cascada (denominado así por la posición de las fases en el desarrollo de esta, que parecen caer en cascada “por gravedad” hacia las siguientes fases), es el enfoque metodológico que ordena rigurosamente las etapas del proceso para el desarrollo del software, de tal forma que el inicio de cada etapa debe esperar a la finalización de la etapa anterior. Al final de cada etapa, el modelo está diseñado para

llevar a cabo una revisión final, que se encarga de determinar si el proyecto está listo para avanzar a la siguiente fase. Este modelo fue el primero en originarse y es la base de todos los demás modelos de ciclo de vida.

La versión original fue propuesta por Winston W. Royce en 1970 y posteriormente revisada por Barry Boehm en 1980 e Ian Sommerville en 1985.

Un ejemplo de una metodología de desarrollo en cascada es:

**Análisis de requisitos:** En esta fase se analizan las necesidades de los usuarios finales del software para determinar qué objetivos debe cubrir. De esta fase surge una memoria llamada SRD (documento de especificación de requisitos), que contiene la especificación completa de lo que debe hacer el sistema sin entrar en detalles internos. Es importante señalar que en esta etapa se debe consensuar todo lo que se requiere del sistema y será aquello lo que seguirá en las siguientes etapas, no pudiéndose requerir nuevos resultados a mitad del proceso de elaboración del software de una manera.

**Diseño del Sistema:** Descompone y organiza el sistema en elementos que puedan elaborarse por separado, aprovechando las ventajas del desarrollo en equipo. Como resultado surge el SDD (Documento de Diseño del Software), que contiene la descripción de la estructura relacional global del sistema y la especificación de lo que debe hacer cada una de sus partes, así como la manera en que se combinan unas con otras.

Es conveniente distinguir entre diseño de alto nivel o arquitectónico y diseño detallado. El primero de ellos tiene como objetivo definir la estructura de la solución (una vez que la fase de análisis ha descrito el problema) identificando grandes módulos (conjuntos de funciones que van a estar asociadas) y sus relaciones. Con ello se define la arquitectura de la solución elegida. El segundo define los algoritmos empleados y la organización del código para comenzar la implementación.

**Diseño del programa:** Es la fase en donde se realizan los algoritmos necesarios para el cumplimiento de los requerimientos del usuario así como también los análisis necesarios para saber qué herramientas usar en la etapa de Codificación.

**Codificación:** Es la fase donde se implementa el código fuente, haciendo uso de prototipos así como de pruebas y ensayos para corregir errores. Dependiendo del

lenguaje de programación y su versión se crean las bibliotecas y componentes reutilizables dentro del mismo proyecto para hacer que la programación sea un proceso mucho más rápido.

**Pruebas:** Los elementos, ya programados, se ensamblan para componer el sistema y se comprueba que funciona correctamente y que cumple con los requisitos, antes de ser entregado al usuario final.

**Verificación:** Es la fase en donde el usuario final ejecuta el sistema, para ello el o los programadores ya realizaron exhaustivas pruebas para comprobar que el sistema no falle. En la creación de desarrollo de cascada se implementan los códigos de investigación y pruebas del mismo.

**Mantenimiento:** Una de las etapas más críticas, ya que se destina un 75% de los recursos, es el mantenimiento del software ya que al utilizarlo como usuario final puede ser que no cumpla con todas nuestras expectativas.

De esta forma, cualquier error de diseño detectado en la etapa de prueba conduce necesariamente al rediseño y nueva programación del código afectado, aumentando los costos del desarrollo. La palabra cascada sugiere, mediante la metáfora de la fuerza de la gravedad, el esfuerzo necesario para introducir un cambio en las fases más avanzadas de un proyecto.



#### 11.4. MODELO ORIENTADO A OBJETOS

El MOO es la construcción de modelos de un sistema por medio de la identificación y especificación de un conjunto de objetos relacionados, que se comportan y colaboran entre sí de acuerdo a los requerimientos establecidos para el sistema de objetos.

La definición anterior ya sugiere la distinción, dentro del proceso de MOO, de tres dimensiones o perspectivas relativamente ortogonales para describir un sistema de objetos:

- Dimensión estructural de los objetos: Se centra en las propiedades estáticas o pasivas de los sistemas. Está relacionada con la estructura estática del sistema de objetos.

- Dimensión dinámica del comportamiento: Se centra en las propiedades activas y describe el comportamiento individual y la colaboración entre los objetos que constituyen el sistema.

- Dimensión funcional de los requerimientos: Son consideradas las propiedades relativas a la función de transformación del sistema de objetos, es decir, los procesos de conversión de entradas en salidas.

El proceso de MOO puede ser dividido en un conjunto mínimo de actividades. La lista a seguir muestra estas actividades sin ninguna secuencia específica:

- Identificar las clases, objetos y atributos: Se determinan cuáles son las clases, objetos y atributos que deben incluirse en el modelo.

- Asociar estáticamente los objetos: Es la configuración de una estructura estática que exprese relaciones dependientes del dominio del problema.

- Describir el comportamiento de los objetos: Es la especificación del comportamiento de los objetos sobre la base de los conceptos básicos de estado, regla de transición, evento y acción.

- Definir la colaboración del comportamiento de los objetos: Busca reflejar la interacción o colaboración entre los objetos, considerando el flujo de eventos o mensajes entre los mismos.

- Organizar las clases en jerarquías de herencia: Se propone organizar las clases de tal forma a maximizar la compartición de propiedades comunes entre ellas.

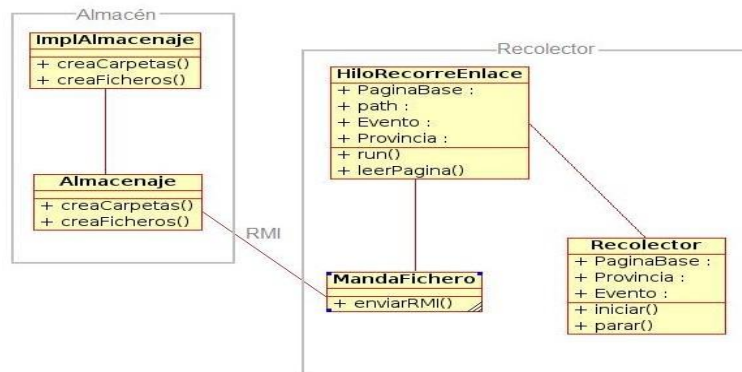
- Agregar y/o particionar las clases por niveles de abstracción: Jerarquiza el modelo por niveles de complejidad haciendo uso de mecanismos de particionamiento o de agregación.

Las seis actividades descritas en la sección anterior pueden ser ordenadas de diversas maneras, de acuerdo a los diferentes procedimientos propuestos por las técnicas de MOO. Sin embargo, estos procedimientos pueden ser categorizados en tres estrategias generales asociadas a las dimensiones del MOO. Estas estrategias son:

- Dirigida por datos: En esta estrategia el MOO es dirigido principalmente por la información estática sobre el dominio del problema. La idea central que sustenta esta estrategia es que la estructura de los objetos del sistema es determinante para el comportamiento que éste presenta. La gran mayoría de las técnicas de MOO siguen esta estrategia, con algunas pequeñas variaciones.

- Dirigida por comportamiento: La estrategia dirigida por comportamiento se basa en información sobre el comportamiento esperado del sistema de objetos. El MOO bajo esta estrategia comienza con la definición del comportamiento global del sistema, para seguir después con la definición de la colaboración de algún tipo de componentes al interior del sistema. Estos componentes o entidades serán candidatos a objetos. La hipótesis básica es que la estructura de los objetos del sistema es derivada del comportamiento que éste debe presentar. La cantidad de procedimientos de propuestas que siguen esta estrategia es bastante menor que la que adopta la estrategia dirigida por datos.

➤ Dirigida por procesos: En esta estrategia el MOO es dirigido por la información sobre las tareas (procesos, responsabilidades o funciones) que deben ser desempeñadas por el sistema y sus componentes. Esencialmente esta estrategia utiliza las facilidades de los modelos funcionales (o de procesos) para derivar (o asociar) los objetos y su estructura. Los procedimientos de las propuestas que siguen esta estrategia están entre los primeros que fueron publicados.



## 12. METODOLOGÍAS

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 ingenieriles han estado presentes durante mucho tiempo. No se han distinguido precisamente por ser muy exitosas. Aún menos por su popularidad. La crítica más frecuente a estas metodologías es que son burocráticas. Hay tanto que hacer para seguir la metodología que el ritmo entero del desarrollo se retarda. Hoy en día existen numerosas propuestas metodológicas que inciden en distintas dimensiones del proceso de desarrollo. Sin embargo la experiencia ha demostrado que las metodologías tradicionales no ofrecen una buena solución para proyectos donde el entorno es volátil y donde los requisitos no se conocen con exactitud, porque no están pensadas para trabajar con incertidumbre. Aplicar metodologías tradicionales nos obliga a forzar a nuestro cliente a que tome la mayoría de las decisiones al principio.

### 12.1. METODOLOGÍA RUP

El proceso unificado de desarrollo (RUP) es una metodología para la ingeniería de software que va más allá del mero análisis y diseño orientado a objetos para proporcionar una familia de técnicas que soportan el ciclo completo de desarrollo de software. El resultado es un proceso basado en componentes, dirigido por los casos de uso, centrado en la arquitectura, iterativo e incremental.

#### **Características principales de RUP:**

**Centrado en los modelos:** Los diagramas son un vehículo de comunicación más expresivo que las descripciones en lenguaje natural. Se trata de minimizar el uso de descripciones y especificaciones textuales del sistema.

**Guiado por los Casos de Uso:** Los Casos de Uso son el instrumento para validar la arquitectura del software y extraer los casos de prueba.

**Centrado en la arquitectura:** Los modelos son proyecciones del análisis y el diseño constituye la arquitectura del producto a desarrollar.

**Iterativo e incremental:** Durante todo el proceso de desarrollo se producen versiones incrementales (que se acercan al producto terminado) del producto en desarrollo.

### **Beneficios que aporta RUP:**

Permite desarrollar aplicaciones sacando el máximo provecho de las nuevas tecnologías, mejorando la calidad, el rendimiento, la reutilización, la seguridad y el mantenimiento del software mediante una gestión sistemática de los riesgos.

Permite la producción de software que cumpla con las necesidades de los usuarios, a través de la especificación de los requisitos, con una agenda y costo predecible.

Enriquece la productividad en equipo y proporciona prácticas óptimas de software a todos sus miembros.

Permite llevar a cabo el proceso de desarrollo práctico, brindando amplias guías, plantillas y ejemplos para todas las actividades críticas.

Proporciona guías explícitas para áreas tales como modelado de negocios, arquitectura Web, pruebas y calidad. También se proporciona guías para desarrollar en plataformas IBM WebSphere y Microsoft Web solution para acelerar el desarrollo de los productos.

Se integra estrechamente con herramientas Rational, permitiendo a los equipos de desarrollo aprovechar todas las ventajas de las características de



los productos Rational, el Lenguaje de Modelado Unificado (UML) y otras prácticas óptimas de la industria.

Unifica todo el equipo de desarrollo de software y mejora la comunicación al brindar a cada miembro del mismo una base de conocimientos, un lenguaje de modelado y un punto de vista de cómo desarrollar software.

Optimiza la productividad de cada miembro del equipo al poner al alcance la experiencia derivada de miles de proyectos y muchos líderes de la industria.

No solo garantiza que los proyectos abordados serán ejecutados íntegramente sino que además evita desviaciones importantes respecto a los plazos.

Permite una definición acertada del sistema en un inicio para hacer innecesarias las reconstrucciones parciales posteriores.

Para la elaboración de este proyecto utilizamos la metodología RUP la cual divide el proceso en 4 fases, dentro de las cuales se realizan varias iteraciones en número variable según el proyecto y en las que se hace un mayor o menor hincapié en los distintas actividades.

### **Inicio**

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

*“De acuerdo al problema encontrado en el Conjunto residencial Quintas El Portal, por su manejo rudimentario de la información, se decide el diseño, desarrollo e implementación de SI web “SIPHO”, para lograr que el administrador y copropietarios puedan acceder de manera más fácil y rápida a la información”*

## **Elaboración:**

En la fase de elaboración se seleccionan los casos de uso que permiten definir la arquitectura base del sistema y se desarrollaran en esta fase, se realiza la especificación de los casos de uso seleccionados y el primer análisis del dominio del problema, se diseña la solución preliminar.

*“De acuerdo a la etapa de análisis de este proyecto, se desarrollan los modelos:*

*Casos de uso*

*Diagrama de clases*

*Diagrama de estados*

*Diagrama de secuencias*

*Flujo grama de procesos”*

## **Construcción**

El propósito de esta fase es completar la funcionalidad del sistema, para ello se deben clarificar los requisitos pendientes, administrar los cambios de acuerdo a las evaluaciones realizados por los usuarios y se realizan las mejoras para el proyecto.

*“El SI web se desarrollara de acuerdo a las siguientes fases:*

***Fase 1:*** *Recolección de información y análisis de requerimientos.*

***Fase2:*** *Story boards, prototipos funcionales, creación de la base de datos en XAMPP y creación de formularios, creación de plantillas HTML, modelo de datos, módulos administrativos, programación de cada módulo.*

***Fase 3:*** *Implementación del SI en el conjunto residencial.*

***Fase 4:*** *Pruebas técnicas y pruebas piloto. “*

## **Transición**

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

*“Para finalizar se realizara el testeo e implementación del software con los usuarios finales“*

## 12.2. METODOLOGÍA RAD

El desarrollo rápido de aplicaciones o RAD (acrónimo en inglés de rapid application development) es un proceso de desarrollo de software, desarrollado inicialmente por James Martin en 1980. El método comprende el desarrollo interactivo, la construcción de prototipos y el uso de utilidades CASE (Computer Aided Software Engineering). Tradicionalmente, el desarrollo rápido de aplicaciones tiende a englobar también la usabilidad, utilidad y la rapidez de ejecución. Hoy en día se suele utilizar para referirnos al desarrollo rápido de interfaces gráficas de usuario tales como Glade, o entornos de desarrollo integrado completos. Algunas de las plataformas más conocidas son Visual Studio, Lazarus, Gambas, Delphi, Foxpro, Anjuta, Game Maker, Velneo o Clarion. En el área de la autoría multimedia, software como Neosoft Neoboo y MediaChance Multimedia Builder proveen plataformas de desarrollo rápido de aplicaciones, dentro de ciertos límites.

Fases:

**Modelado de gestión:** el flujo de información entre las funciones de gestión se modela de forma que responda a las siguientes preguntas: ¿Qué información conduce el proceso de gestión? ¿Qué información se genera? ¿Quién la genera? ¿A dónde va la información? ¿Quién la proceso?

**Modelado de datos:** el flujo de información definido como parte de la fase de modelado de gestión se refina como un conjunto de objetos de datos necesarios para apoyar la empresa. Se definen las características (llamadas atributos) de cada uno de los objetos y las relaciones entre estos objetos.

**Modelado de proceso:** los objetos de datos definidos en la fase de modelado de datos quedan transformados para lograr el flujo de información necesario para implementar

una función de gestión. Las descripciones del proceso se crean para añadir, modificar, suprimir, o recuperar un objeto de datos. Es la comunicación entre los objetos.

**Generación de aplicaciones:** El DRA asume la utilización de técnicas de cuarta generación. En lugar de crear software con lenguajes de programación de tercera generación, el proceso DRA trabaja para volver a utilizar componentes de programas ya existentes (cuando es posible) o a crear componentes reutilizables (cuando sea necesario). En todos los casos se utilizan herramientas automáticas para facilitar la construcción del software.

**Pruebas de entrega:** Como el proceso DRA enfatiza la reutilización, ya se han comprobado muchos de los componentes de los programas. Esto reduce tiempo de pruebas. Sin embargo, se deben probar todos los componentes nuevos y se deben ejercitar todas las interfaces a fondo.

**Características de RAD:**

Equipos Híbridos: Equipos compuestos por alrededor de seis personas, incluyendo desarrolladores y usuarios de tiempo completo del sistema así como aquellas personas involucradas con los requisitos. Los desarrolladores de RAD deben ser "renacentistas": analistas, diseñadores y programadores en uno.

Herramientas Especializadas: Desarrollo "visual", Creación de prototipos falsos (simulación pura), creación de prototipos funcionales, múltiples lenguajes, calendario grupal, herramientas colaborativas y de trabajo en equipo, componentes reusables e interfaces estándares (API).

Timeboxing: Las funciones secundarias son eliminadas como sea necesario para cumplir con el calendario.

Prototipos Iterativos y Evolucionarios: reunión JAD (Joint Application Development), se reúnen los usuarios finales y los desarrolladores, lluvia de ideas para obtener un borrador inicial de los requisitos, iterar hasta acabar, los desarrolladores construyen y depuran el prototipo basado en los requisitos actuales, los diseñadores revisan el prototipo, los clientes prueban el prototipo, depuran los requisitos, los clientes

y desarrolladores se reúnen para revisar juntos el producto, refinar los requisitos y generar solicitudes de cambios, los cambios para los que no hay tiempo no se realizan, los requisitos secundarios se eliminan si es necesario para cumplir el calendario.

### 12.3. METODOLOGÍA XP

La programación extrema o *extreme programming* (de ahora en adelante, XP) es una metodología de desarrollo de la ingeniería de software formulada por Kent Beck, autor del primer libro sobre la materia, *Extreme Programming Explained: Embrace Change* (1999). Es el más destacado de los procesos ágiles de desarrollo de software. Al igual que éstos, la programación extrema se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad. Los defensores de la XP consideran que los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos. Creen que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos. Se puede considerar la programación extrema como la adopción de las mejores metodologías de desarrollo de acuerdo a lo que se pretende llevar a cabo con el proyecto, y aplicarlo de manera dinámica durante el ciclo de vida del software.

**Diseño simple:** se basa en la filosofía de que el mayor valor de negocio es entregado por el programa más sencillo que cumpla los requerimientos. *Simple Design* se enfoca en proporcionar un sistema que cubra las necesidades inmediatas del cliente, ni más ni menos. Este proceso permite eliminar redundancias y rejuvenecer los diseños obsoletos de forma sencilla.

**Metáfora:** desarrollada por los programadores al inicio del proyecto, define una historia de cómo funciona el sistema completo. XP estimula historias, que son breves descripciones de un trabajo de un sistema en lugar de los tradicionales diagramas y modelos UML (*Unified Modeling Language*). La metáfora expresa la visión evolutiva

del proyecto que define el alcance y propósito del sistema. Las tarjetas CRC (Clase, Responsabilidad y Colaboración) también ayudarán al equipo a definir actividades durante el diseño del sistema. Cada tarjeta representa una clase en la programación orientada a objetos y define sus responsabilidades (lo que ha de hacer) y las colaboraciones con las otras clases (cómo se comunica con ellas).

**Propiedad colectiva del código:** un código con propiedad compartida. Nadie es el propietario de nada, todos son el propietario de todo. Este método difiere en mucho a los métodos tradicionales en los que un simple programador posee un conjunto de código. Los defensores de XP argumentan que mientras haya más gente trabajando en una pieza, menos errores aparecerán.

**Estándar de codificación:** define la propiedad del código compartido así como las reglas para escribir y documentar el código y la comunicación entre diferentes piezas de código desarrolladas por diferentes equipos. Los programadores las han de seguir de tal manera que el código en el sistema se vea como si hubiera estado escrito por una sola persona.

**PROCESO DE DESARROLLO:** La programación extrema parte del caso habitual de una compañía que desarrolla software normalmente a medida, en la que hay diferentes roles: un equipo de gestión (o diseño), uno de desarrollo y los clientes finales. La relación entre el equipo de diseño, los que desarrollan el software y clientes es totalmente diferente al que se ha producido en las metodologías tradicionales, que se basaba en una fase de captura de los requisitos previa al desarrollo, y de una fase de validación posterior al mismo:

1. Interacción con el cliente: En este tipo de programación el cliente pasa a ser parte implicada en el equipo de desarrollo. Su importancia es máxima en el momento de tratar con los usuarios y en efectuar las reuniones de planificación. Tiene un papel importante de interacción con el equipo de programadores, sobre todo después de cada cambio, y de cada posible problema localizado, mostrando las prioridades, expresando sus sensaciones... En este tipo de programación existirán pruebas de aceptación de la programación que ayudarán a que su labor sea lo más provechosa posible. Al fin y al

cabo, el cliente se encuentra mucho más cerca del proceso de desarrollo. Se elimina la fase inicial de recopilación de requerimientos, y se permite que éstos se vayan cogiendo a lo largo del proyecto, de una manera ordenada. De esta forma se posibilita que el cliente pueda ir cambiando de opinión sobre la marcha, pero a cambio han de estar siempre disponibles para solucionar las dudas del equipo de desarrollo.

En XP aparece un nuevo concepto llamado “*Historia de usuario*”. Se trata de una lista de características que el cliente necesita que existan en el producto final. Estas constan de dos fases: En la primera fase, el cliente describe con sus propias palabras las características y, es el responsable del equipo, el encargado de informarlo de las dificultades técnicas de cada una de ellas y de su costo. A consecuencia de este diálogo, el cliente deja por escrito un conjunto de historias y las ordena en función de la prioridad que tienen para él. De esta manera ya es posible definir unas fechas aproximadas para ellos.

En la segunda fase, el cliente cogerá las primeras historias a implementar y las dividirá en trabajos a realizar. El cliente también participa, pero hay más peso por parte del equipo de desarrolladores, esto dará como resultado una planificación más exacta. Este método se repetirá para cada historia. A diferencia de otras técnicas, como puede ser UML, en el caso de XP, se exige que sea el cliente el encargado de escribir los documentos con las especificaciones de lo que realmente quiere, como un documento de requisitos de usuario. En esta fase, el equipo técnico será el encargado de catalogar las historias del cliente y asignarles una duración. La norma es que cada historia de usuario tiene que poder ser realizable en un espacio entre una y tres semanas de programación. Las que requieran menos tiempo serán agrupadas, y las que necesiten más serán modificadas o divididas.

Finalmente decir que las historias de los usuarios serán escritas en tarjetas, lo que facilitará que el cliente pueda especificar la importancia relativa entre las diferentes historias de usuario, así como el trabajo de los programadores que podrán catalogarlas correctamente. Este formato también es muy útil en el momento de las pruebas de aceptación.

**PLANIFICACIÓN DEL PROYECTO:** En este punto se tendrá que elaborar la planificación por etapas, donde se aplicarán diferentes iteraciones. Para hacerlo será necesaria la existencia de reglas que se han de seguir por las partes implicadas en el proyecto para que todas las partes tengan voz y se sientan realmente partícipes de la decisión tomada. Las entregas se tienen que hacer cuanto antes mejor, y con cada iteración, el cliente ha de recibir una nueva versión. Cuanto más tiempo se tarde en introducir una parte esencial, menos tiempo se tendrá para trabajar con ella después. Se aconseja muchas entregas y muy frecuentes. De esta manera un error en la parte inicial del sistema tiene más posibilidades de detectarse rápidamente.

Una de las máximas a aplicar es, los cambios, no han de suponer más horas de programación para el programador, ya que el que no se termina en un día, se deja para el día siguiente. Se ha de tener asumido que en el proceso de planificación habrán errores, es más, serán comunes, y por esto esta metodología ya los tiene previstos, por lo tanto se establecerán mecanismos de revisión. Cada tres o cinco iteraciones es normal revisar las historias de los usuarios, y renegociar la planificación. Cada iteración necesita también ser planificada, es lo que se llama *planificación iterativa*, en la que se anotarán las historias de usuarios que se consideren esenciales y las que no han pasado las pruebas de aceptación. Estas planificaciones también se harán en tarjetas, en las que se escribirán los trabajos que durarán entre uno y tres días. Es por esto que el diseño se puede clasificar como continuo. Añade agilidad al proceso de desarrollo y evita que se mire demasiado hacia delante, desarrollando trabajos que aún no han estado programados. Este tipo de planificación en iteraciones y el diseño iterativo, hace que aparezca una práctica que no existía en la programación tradicional. Se trata de las discusiones diarias informales, para fomentar la comunicación, y para hacer que los desarrolladores tengan tiempo de hablar de los problemas a los que se enfrentan y de ver cómo van con sus trabajos.

**DISEÑO, DESARROLLO Y PRUEBAS:** El desarrollo es la parte más importante en el proceso de la programación extrema. Todos los trabajos tienen como objetivo que se



programen lo más rápidamente posible, sin interrupciones y en dirección correcta. También es muy importante el diseño, y se establecen los mecanismos, para que éste sea revisado y mejorado de manera continuada a lo largo del proyecto, según se van añadiendo funcionalidades al mismo.

La clave del proceso de desarrollar XP es la comunicación. La mayoría de los problemas en los proyectos son por falta de comunicación en el equipo. En XP, aparece un nuevo concepto llamado Metáfora. Su principal objetivo es mejorar la comunicación entre todos los integrantes del equipo, al crear una visión global y común de lo que se quiere desarrollar. La metáfora tiene que ser expresada en términos conocidos por los integrantes del equipo, por ejemplo comparando el sistema que se desarrollará con alguna cosa de la vida real. Antes de empezar a codificar se tienen que hacer pruebas unitarias, es decir:

Cada vez que se quiere implementar una parte de código, en XP, se tiene que escribir una prueba sencilla, y después escribir el código para que la pase. Una vez pasada se amplía y se continúa. En XP hay una máxima que dice "Todo el código que puede fallar tiene que tener una prueba".

Con estas normas se obtiene un código simple y funcional de manera bastante rápida. Por esto es importante pasar las pruebas al 100%. Respecto a la integración del soft, en XP se ha de hacer una integración continua, es decir, cada vez se tienen que ir integrando pequeños fragmentos de código, para evitar que al finalizar el proyecto se tenga que invertir grandes esfuerzos en la integración final. En todo buen proyecto de XP, tendría que existir una versión al día integrada, de manera que los cambios siempre se realicen en esta última versión. Otra peculiaridad de XP es que cada programador puede trabajar en cualquier parte del programa. De esta manera se evita que haya partes "propietarias de cada programador". Por esto es tan importante la integración diaria. Para terminar, otra peculiaridad que tiene la XP. La de fomentar la programación en parejas, es decir, hacer que los programadores no trabajen en solitario, sino que siempre estarán con otra persona. Una pareja de programadores ha de compartir el teclado, el monitor y el ratón. El principio fundamental de este hecho es realizar de manera continua y sin parar el desarrollo de código. Las parejas tienen que ir cambiando de

manera periódica, para hacer que el conocimiento se difunda en el grupo. Está demostrado que de esta manera el trabajo es más eficaz y también se consigue más y mejor código.

### 13. UML

UML es un lenguaje para hacer modelos y es independiente de los métodos de análisis y diseño. Existen diferencias importantes entre un método y un lenguaje de modelado. Un método es una manera explícita de estructurar el pensamiento y las acciones de cada individuo. Además, el método le dice al usuario qué hacer, cómo hacerlo, cuándo hacerlo y por qué hacerlo; mientras que el lenguaje de modelado carece de estas instrucciones. Los métodos contienen modelos y esos modelos son utilizados para describir algo y comunicar los resultados del uso del método.

Un modelo es expresado en un lenguaje de modelado. Un lenguaje de modelado consiste de vistas, diagramas, elementos de modelo, los símbolos utilizados en los modelos y un conjunto de mecanismos generales o reglas que indican cómo utilizar los elementos.

**Vistas:** Las vistas muestran diferentes aspectos del sistema modelado. Una vista no es una gráfica, pero sí una abstracción que consiste en un número de diagramas y todos esos diagramas juntos muestran una "fotografía" completa del sistema. Las vistas también ligan el lenguaje de modelado a los métodos o procesos elegidos para el desarrollo. Las diferentes vistas que UML tiene son: Vista Use-Case: Una vista que muestra la funcionalidad del sistema como la perciben los actores externos.

Vista Lógica: Muestra cómo se diseña la funcionalidad dentro del sistema, en términos de la estructura estática y la conducta dinámica del sistema.

Vista de Componentes: Muestra la organización de los componentes de código.

Vista Concurrente: Muestra la concurrencia en el sistema, direccionando los problemas con la comunicación y sincronización que están presentes en un sistema concurrente.

Vista de Distribución: muestra la distribución del sistema en la arquitectura física con computadoras y dispositivos llamados *nodos*.

**Diagramas:** Los diagramas son las gráficas que describen el contenido de una vista. UML tiene nueve tipos de diagramas que son utilizados en combinación para proveer todas las vistas de un sistema: diagramas de caso de uso, de clases, de objetos, de estados, de secuencia, de colaboración, de actividad, de componentes y de distribución.

**Símbolos o Elementos de modelo:** Los conceptos utilizados en los diagramas son los elementos de modelo que representan conceptos comunes orientados a objetos, tales como clases, objetos y mensajes, y las relaciones entre estos conceptos incluyendo la asociación, dependencia y generalización. Un elemento de modelo es utilizado en varios diagramas diferentes, pero siempre tiene el mismo significado y simbología.

**Reglas o Mecanismos generales:** Proveen comentarios extras, información o semántica acerca del elemento de modelo; además proveen mecanismos de extensión para adaptar o extender UML a un método o proceso específico, organización o usuario.

## **FASES DEL DESARROLLO DE UN SISTEMA**

Las fases del desarrollo de sistemas que soporta UML son:

*Análisis de requerimientos, análisis, diseño, programación y pruebas.*

**Análisis de Requerimientos:** UML tiene casos de uso (use-cases) para capturar los requerimientos del cliente. A través del modelado de casos de uso, los actores externos que tienen interés en el sistema son modelados con la funcionalidad que ellos requieren del sistema (los casos de uso). Los actores y los casos de uso son modelados con relaciones y tienen asociaciones entre ellos o éstas son divididas en jerarquías. Los actores y casos de uso son descritos en un diagrama use-case. Cada use-case es descrito en texto y especifica los requerimientos del cliente: lo que él (o ella) espera del sistema sin considerar la funcionalidad que se implementará. Un análisis de requerimientos

puede ser realizado también para procesos de negocios, no solamente para sistemas de software.

**Análisis:** La fase de análisis abarca las abstracciones primarias (clases y objetos) y mecanismos que están presentes en el dominio del problema. Las clases que se modelan son identificadas, con sus relaciones y descritas en un diagrama de clases. Las colaboraciones entre las clases para ejecutar los casos de uso también se consideran en esta fase a través de los modelos dinámicos en UML. Es importante notar que sólo se consideran clases que están en el dominio del problema (conceptos del mundo real) y todavía no se consideran clases que definen detalles y soluciones en el sistema de software, tales como clases para interfaces de usuario, bases de datos, comunicaciones, concurrencia, etc.

**Diseño:** En la fase de diseño, el resultado del análisis es expandido a una solución técnica. Se agregan nuevas clases que proveen de la infraestructura técnica: interfaces de usuario, manejo de bases de datos para almacenar objetos en una base de datos, comunicaciones con otros sistemas, etc. Las clases de dominio del problema del análisis son agregadas en esta fase. El diseño resulta en especificaciones detalladas para la fase de programación.

**Programación:** En esta fase las clases del diseño son convertidas a código en un lenguaje de programación orientado a objetos. Cuando se crean los modelos de análisis y diseño en UML, lo más aconsejable es trasladar mentalmente esos modelos a código.

**Pruebas:** Normalmente, un sistema es tratado en pruebas de unidades, pruebas de integración, pruebas de sistema, pruebas de aceptación, etc. Las pruebas de unidades se realizan a clases individuales o a un grupo de clases y son típicamente ejecutadas por el programador. Las pruebas de integración integran componentes y clases en orden para verificar que se ejecutan como se especificó. Las pruebas de sistema ven al sistema como una "caja negra" y validan que el sistema tenga la funcionalidad final que le usuario final espera. Las pruebas de aceptación conducidas por el cliente verifican que el sistema satisface los requerimientos y son similares a las pruebas de sistema.

## 14. MODELOS


### 14.1. CASOS DE USO


El diagrama de casos de uso representa la forma en como un Cliente (Actor) opera con el sistema en desarrollo, además de la forma, tipo y orden en como los elementos interactúan (operaciones o casos de uso).

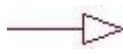
Un diagrama de casos de uso consta de los siguientes elementos:

**Actor:** Es un rol que un usuario juega con respecto al sistema. Es importante destacar el uso de la palabra rol, pues con esto se especifica que un Actor no necesariamente representa a una persona en particular, sino más bien la labor que realiza frente al sistema.

**Casos de uso:** Es una operación/tarea específica que se realiza tras una orden de algún agente externo, sea desde una petición de un actor o bien desde la invocación desde otro caso de uso.

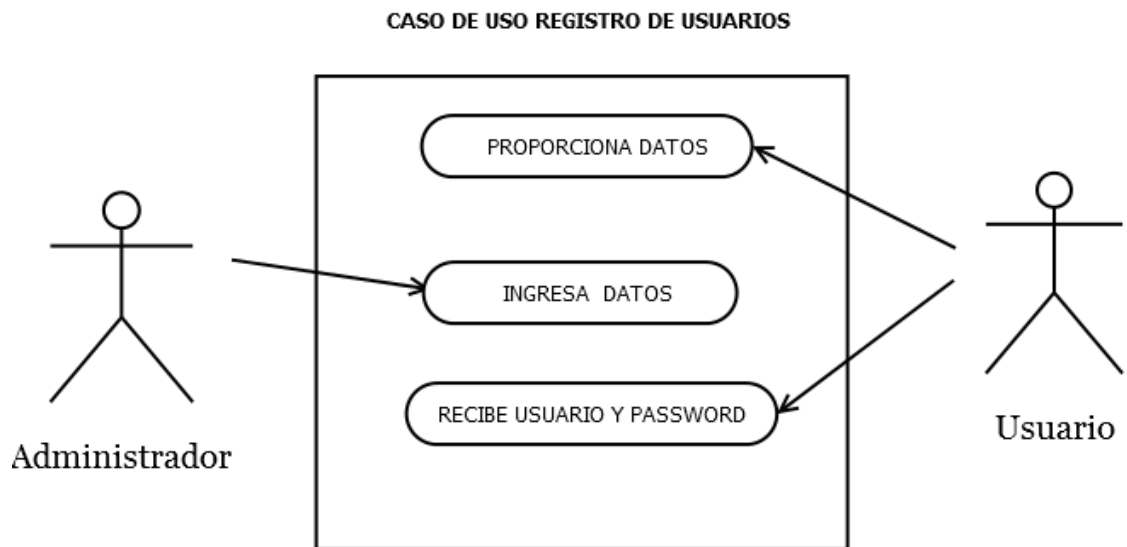
Relaciones de uso, herencia y comunicación: **Asociación:**  Es el tipo de relación más básica que indica la invocación desde un actor o caso de uso a otra operación (caso de uso). Dicha relación se denota con una flecha simple.

**Dependencia o Instanciación:**  Es una forma muy particular de relación entre clases, en la cual una clase depende de otra, es decir, se instancia (se crea). Dicha relación se denota con una flecha punteada.

**Generalización:**  Este tipo de relación es uno de los más utilizados, cumple una doble función dependiendo de su estereotipo, que puede ser de **Uso** (<<uses>>) o de **Herencia** (<<extends>>). Este tipo de relación es orientada exclusivamente para casos de uso (y no para actores). **extends:** Se recomienda utilizar cuando un caso de uso es similar a otro (características). **uses:** Se recomienda utilizar

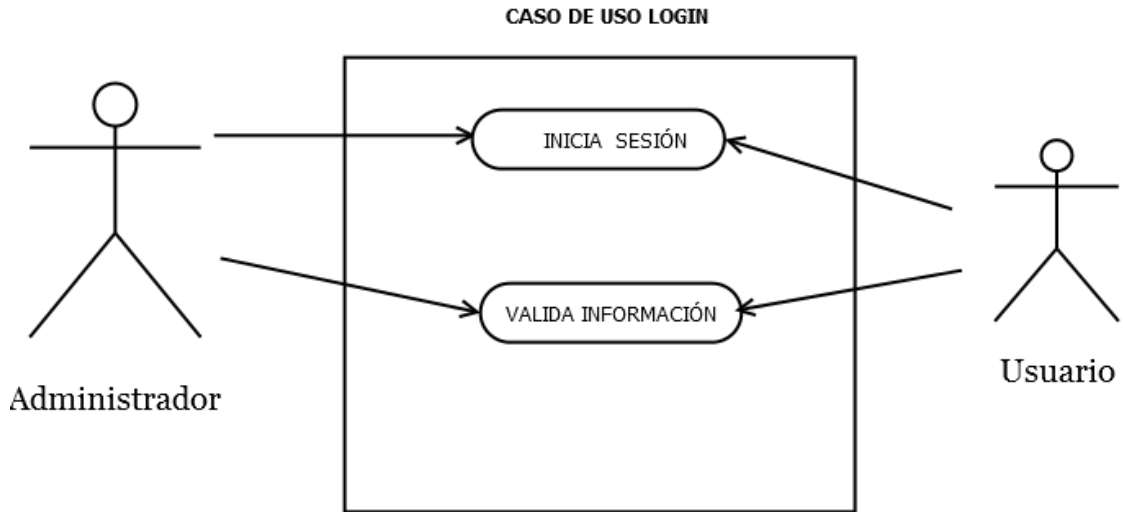
cuando se tiene un conjunto de características que son similares en más de un caso de uso y no se desea mantener copiada la descripción de la característica.

De lo anterior cabe mencionar que tiene el mismo paradigma en diseño y modelamiento de clases, en donde está la duda clásica de **usar** o **heredar**.



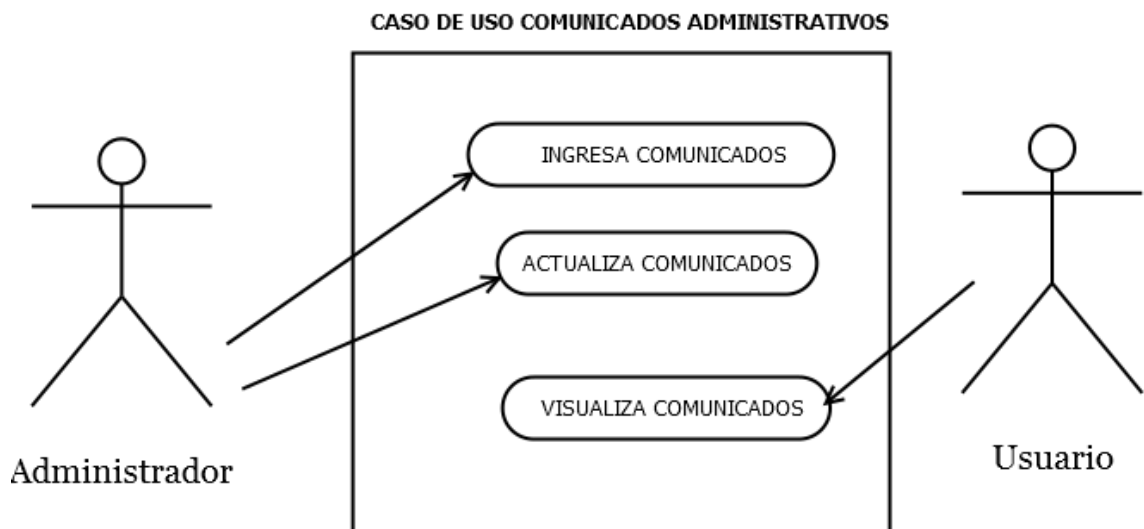
**Descripción caso de uso**

Nombre	Registro de usuarios
Actores	Administrador(Ingresa datos) Usuario(Proporciona datos, recibe usuario y password)
Función	Garantizar seguridad en el ingreso.
Descripción	El usuario ingresa sus datos y accede a la página respectiva.
Condición	El usuario debe estar registrado para acceder al sistema de información.
Poscondición	El usuario ingresa a la página después de loguearse.



**Descripción caso de uso**

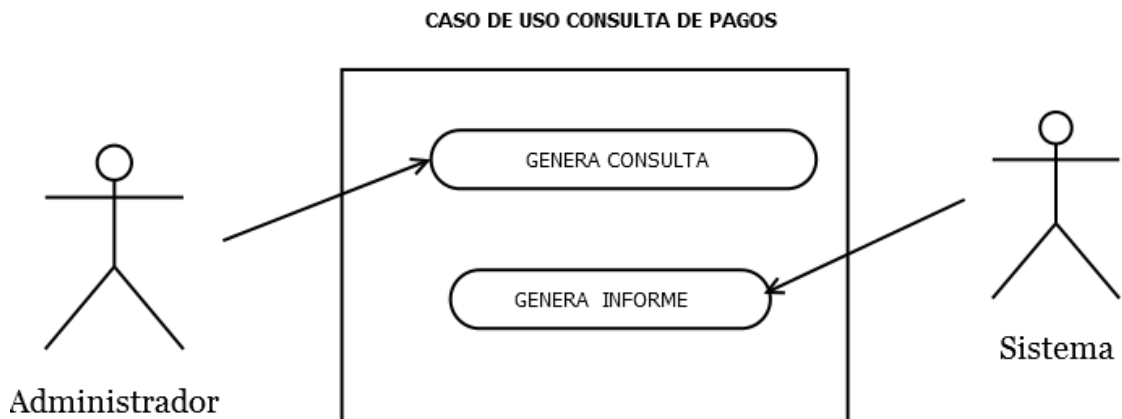
Nombre	LOGIN
Actores	Administrador(inicia sesión, valida información) Usuario (inicia sesión, valida información)
Función	Garantizar ingreso de cada usuario.
Descripción	El usuario inicia sesión y el sistema valida información.
Condición	El usuario debe estar registrado para acceder al sistema de información.
Poscondición	El usuario inicia sesión y navega por el sistema de información según los permisos.



**Descripción caso de uso**

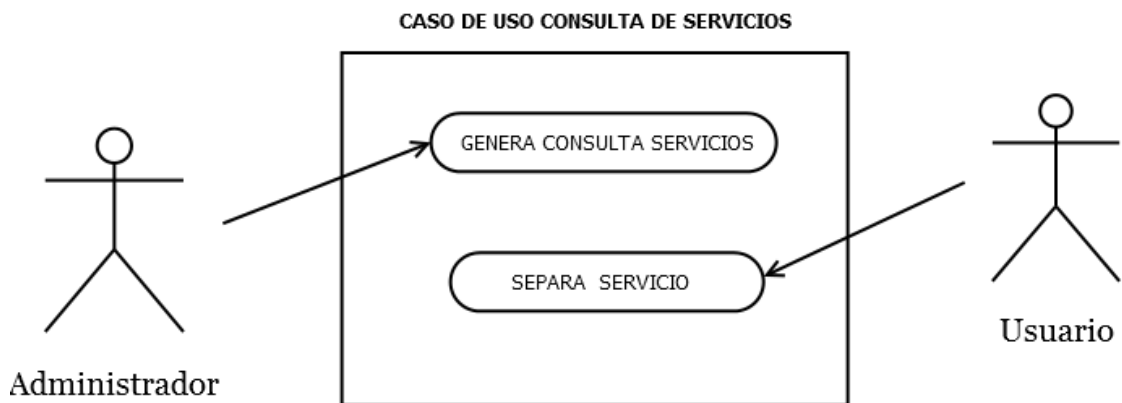


Nombre	Comunicados administrativos
Actores	Administrador(Ingresar y actualiza comunicados) Usuario (visualiza comunicados)
Función	Garantizar la comunicación asertiva.
Descripción	El administrador ingresa y actualiza comunicados y el usuario visualiza los mismos.
Condición	El administrador y el usuario deben estar debidamente registrados para acceder a los comunicados.
Poscondición	Ingresar, actualizar y visualizar comunicados.



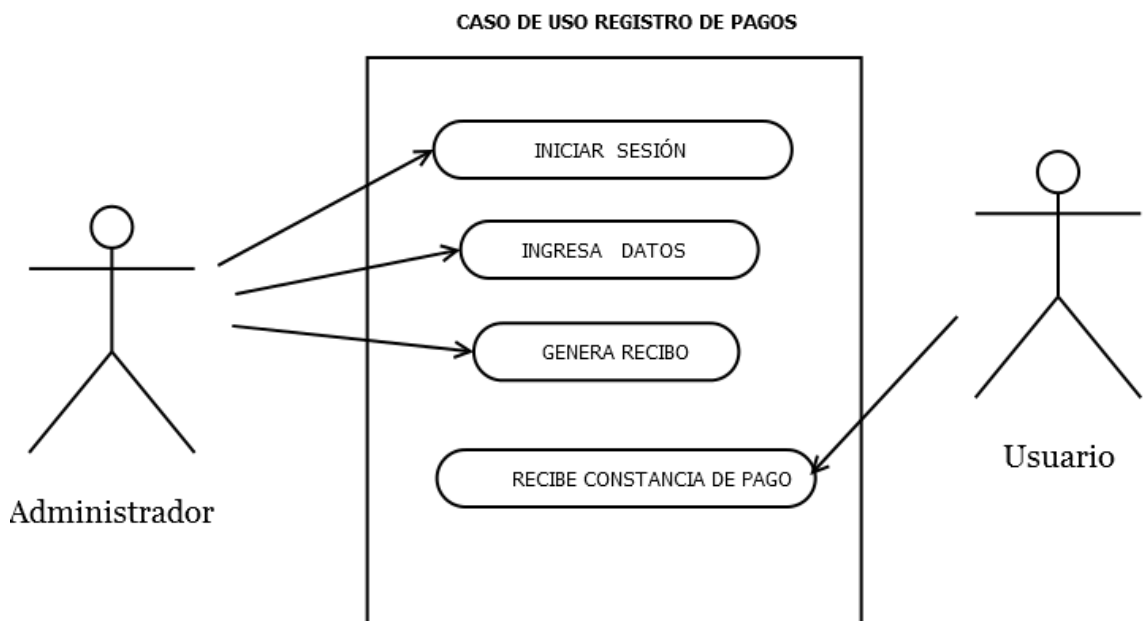
**Descripción caso de uso**

Nombre	Consulta de pagos
Actores	Administrador(genera consulta de pagos) Sistema (genera informe de pagos)
Función	Garantizar informe en tiempo real de pagos.
Descripción	El administrador genera consulta de pagos y el sistema genera informe de pagos.
Condición	El administrador debe ingresar al sistema de información.
Poscondición	Generar consulta e informe de pagos.



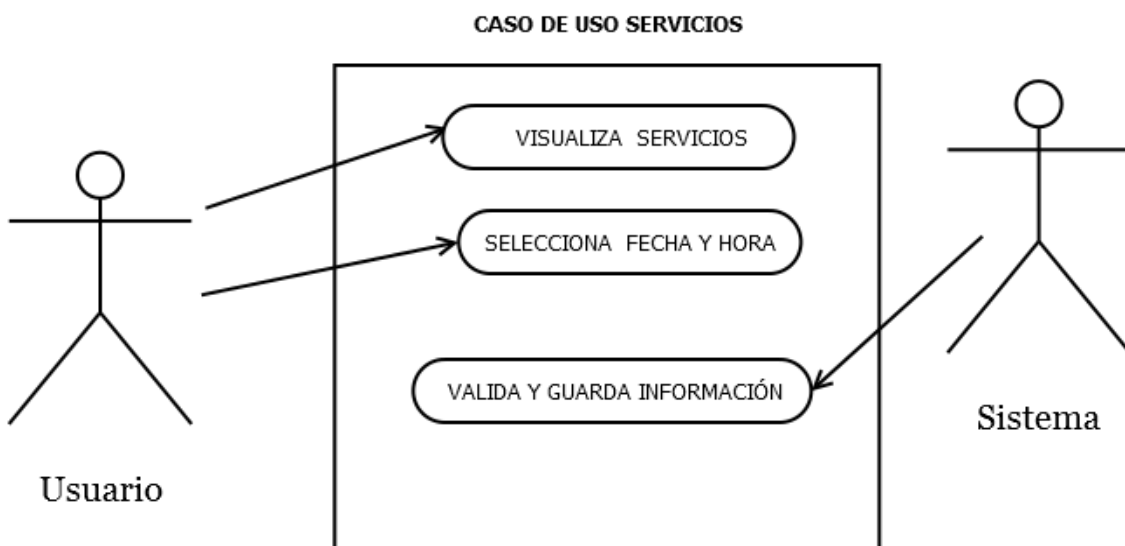
**Descripción caso de uso**

Nombre	Consulta de servicios
Actores	Administrador(genera consulta de servicios) Usuario (separa servicio)
Función	Garantizar la prestación puntual de servicios.
Descripción	El administrador genera consulta de servicios y el usuario separa el servicio requerido.
Condición	El administrador y el usuario deben estar debidamente registrados para acceder a la consulta de servicios.
Poscondición	Generar y separar servicios del conjunto residencial.



**Descripción caso de uso**

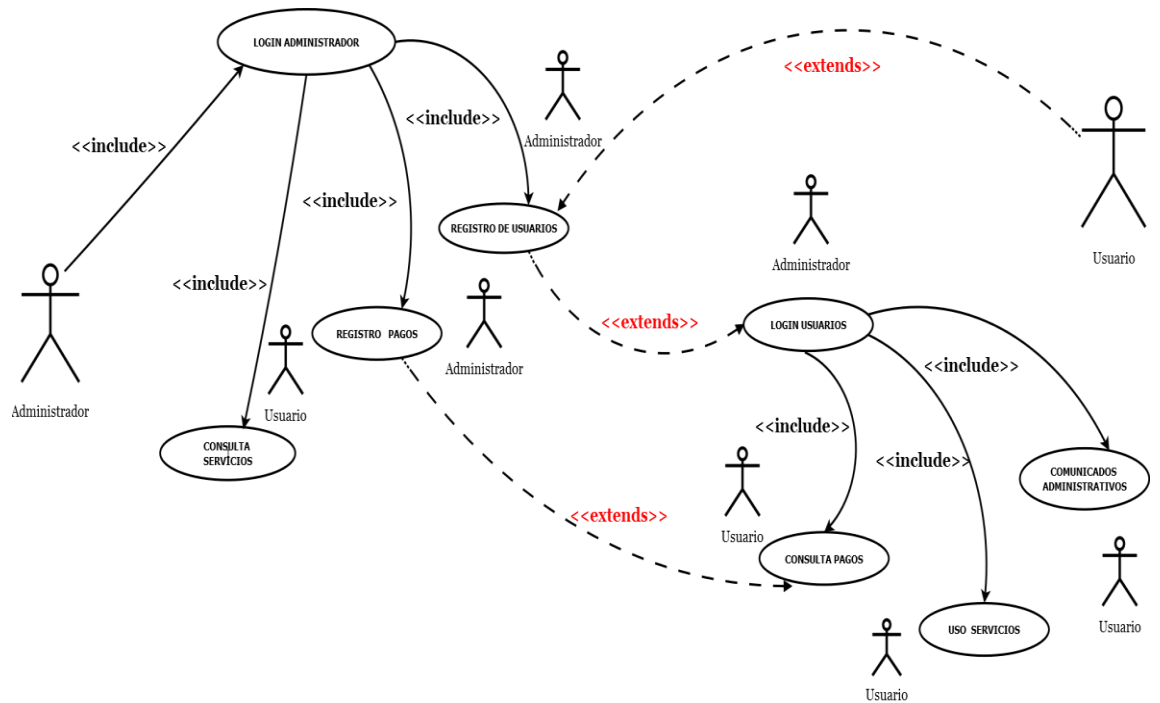
Nombre	Registro de pagos
Actores	Administrador(inicia sesión, registra pagos y genera recibo de pago) Usuario (Recibe constancia de pago)
Función	Garantizar registro de pago en tiempo real.
Descripción	El administrador inicia sesión, registra pagos y genera el recibo que entrega al usuario.
Condición	El administrador debe ingresar al sistema de información.
Poscondición	Generar recibo de pago después de registrar pago.



**Descripción caso de uso**

Nombre	Servicios
Actores	Usuario(visualiza servicios, selecciona fecha y hora) Sistema (valida y guarda información)
Función	Garantizar la prestación puntual de servicios.
Descripción	El usuario visualiza servicios, selecciona fecha y hora y el sistema valida y guarda información.
Condición	El usuario debe estar debidamente registrado para acceder a la consulta de servicios.
Poscondición	Generar y separar servicios del conjunto residencial.

## CASO DE USO GENERAL



### Descripción caso de uso

Nombre	Caso de uso general
Actores	Administrador Usuario
Función	Garantizar la eficiencia del sistema de información SIPHO.
Descripción	El administrador y usuario acceden a los módulos del sistema de acuerdo al cumplimiento de los requisitos.
Condición	El administrador y usuario deben estar debidamente registrados para acceder al sistema de información.
Poscondición	Acceder efectivamente al sistema de información SIPHO.

## 14.2. DIAGRAMA DE CLASES

Los diagramas de clases son diagramas de estructura estática que muestran las clases del sistema y sus interrelaciones (incluyendo herencia, agregación, asociación, etc.). Los diagramas de clase son el pilar básico del modelado con UML, siendo utilizados tanto para mostrar lo que el sistema puede hacer (análisis), como para mostrar cómo puede ser construido (diseño). El diagrama de clases de más alto nivel, será lógicamente un dibujo de los paquetes que componen el sistema. Las clases se documentan con una descripción de lo que hacen, sus métodos y sus atributos. Las relaciones entre clases se documentan con una descripción de su propósito, sus objetos que intervienen en la relación y su opcionalidad (cuando un objeto es opcional el que intervenga en una relación).

Clase: Es la unidad básica que encapsula toda la información de un Objeto (un objeto es una instancia de una clase). A través de ella podemos modelar el entorno en estudio (una Casa, un Auto, una Cuenta Corriente, etc.).} En UML, una clase es representada por un rectángulo que posee tres divisiones: En donde: Superior: Contiene el nombre de la Clase. Intermedio: Contiene los atributos (o variables de instancia) que caracterizan a la Clase (pueden ser private, protected o public). Inferior: Contiene los métodos u operaciones, los cuales son la forma como interactúa el objeto con su entorno (dependiendo de la visibilidad: private, protected o public).

Atributos: son valores que corresponden a un objeto, como color, material, cantidad, ubicación. Generalmente se conoce como la información detallada del objeto. Ejemplo: el objeto es una puerta, sus propiedades o atributos serían: la marca, tamaño, color y peso.} Tipos de atributos: ° public (+): Indica que el atributo será visible tanto dentro como fuera de la clase, es decir, es accesible desde todos lados. ° private (-): Indica que el atributo sólo será accesible desde dentro de la clase (sólo sus métodos lo pueden utilizar). ° protected (#): Indica que el atributo no será accesible desde fuera de

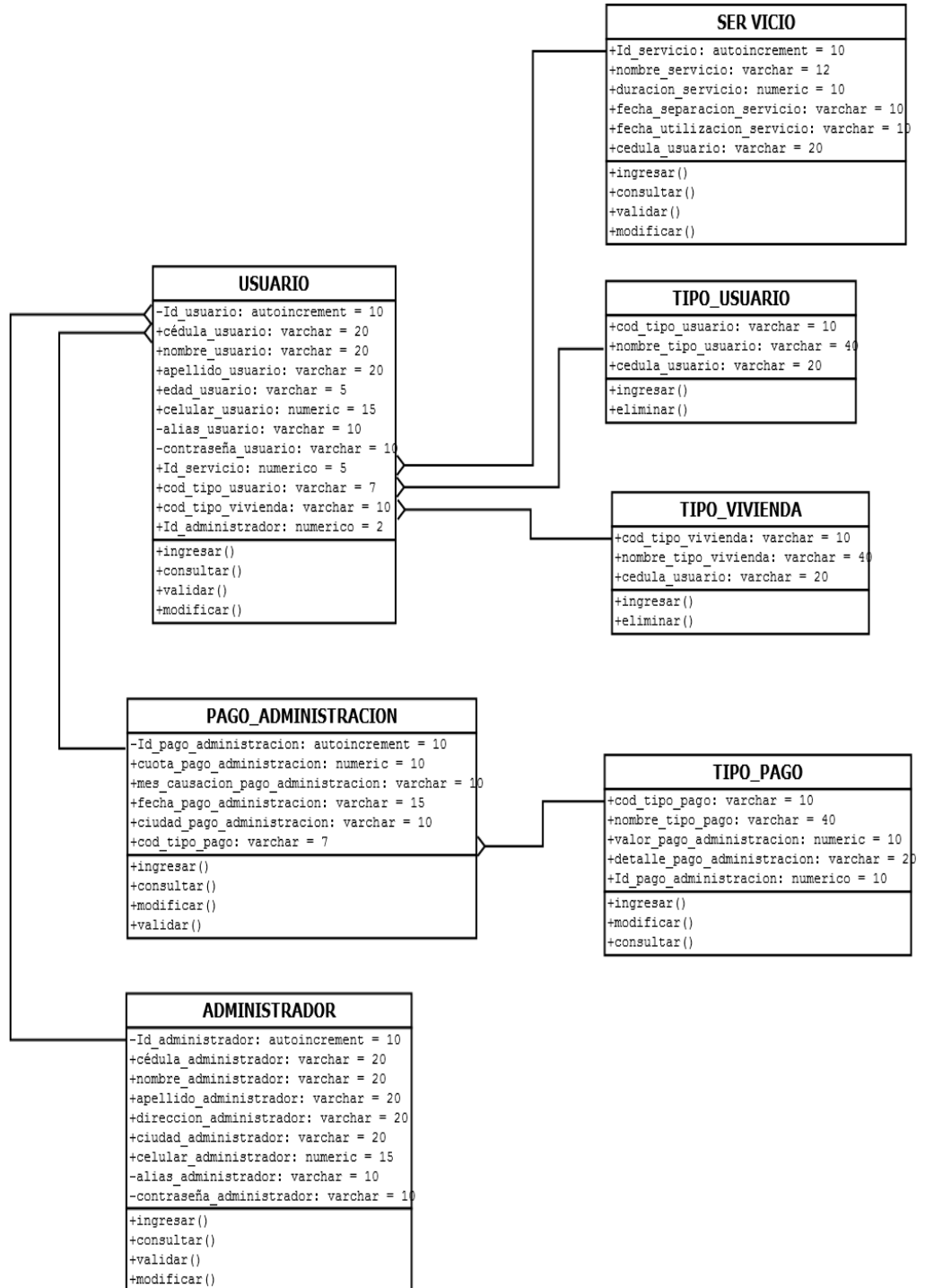
la clase, pero si podrá ser accesado por métodos de la clase además de las subclases que se deriven (ver herencia).

**Operaciones/Métodos:** son aquellas actividades o verbos que se pueden realizar con o para este objeto, como por ejemplo: abrir, cerrar, buscar, cancelar, confirmar, cargar. El nombre de una operación se escribe con minúsculas si consta de una sola palabra. Si el nombre contiene más de una palabra, cada palabra será unida a la anterior y comenzará con una letra mayúscula, a excepción de la primera palabra que comenzará en minúscula. Por ejemplo: abrirPuerta, cerrarPuerta, buscarPuerta, etc. Tipos de métodos: **public (+,):** Indica que el método será visible tanto dentro como fuera de la clase, es decir, es accesible desde todos lados.

**private (-,):** Indica que el método sólo será accesible desde dentro de la clase (sólo otros métodos de la clase lo pueden utilizar).

**protected (#,):** Indica que el método no será accesible desde fuera de la clase, pero si podrá ser accesado por métodos de la clase además de métodos de las subclases que se deriven (ver herencia).

## DIAGRAMA DE CLASES



### 14.3. DIAGRAMA DE ESTADOS

Los diagramas de estado son una técnica conocida para describir el comportamiento de un sistema. Describen todos los estados posibles en los que puede entrar un objeto particular y la manera en que cambia el estado del objeto, como resultado de los eventos que llegan a él. En la mayor parte de las técnicas Orientadas a Objetos, los diagramas de estado se dibujan para una sola clase, mostrando el comportamiento de un solo objeto durante todo su ciclo de vida.

El estado en el que se encuentra un objeto determina su comportamiento. Cada objeto sigue el comportamiento descrito en el Diagrama de Estados asociado a su clase. Los Diagramas de Estados y escenarios son complementarios, los Diagramas de Estados son autómatas jerárquicos que permiten expresar concurrencia, sincronización y jerarquías de objetos, son grafos dirigidos y deterministas. La transición entre estados es instantánea y se debe a la ocurrencia de un evento.

Estado: Identifica un periodo de tiempo del objeto (no instantáneo) en el cual el objeto está esperando alguna operación, tiene cierto estado característico o puede recibir cierto tipo de estímulos. Se representa mediante un rectángulo con los bordes redondeados, que puede tener tres compartimientos: uno para el nombre, otro para el valor característico de los atributos del objeto en ese estado y otro para las acciones que se realizan al entrar, salir o estar en un estado (entry, exit o do, respectivamente).

Evento: Es una ocurrencia que puede causar la transición de un estado a otro de un objeto. Esta ocurrencia puede ser una de varias cosas: Condición que toma el valor de verdadero o falso, recepción de una señal de otro objeto en el modelo, recepción de un mensaje, paso de cierto período de tiempo, después de entrar al estado o de cierta hora y fecha particular. El nombre de un evento tiene alcance dentro del paquete en el cual está definido, no es local a la clase que lo nombre.

Envío de mensajes: Además de mostrar la transición de estados por medio de eventos, puede representarse el momento en el cual se envían mensajes a otros objetos.



Esto se realiza mediante una línea punteada dirigida al diagrama de estados del objeto receptor del mensaje.

Transición simple: es una relación entre dos estados que indica que un objeto en el primer estado puede entrar al segundo estado y ejecutar ciertas operaciones, cuando un evento ocurre y si ciertas condiciones son satisfechas.

Transición interna: es una transición que permanece en el mismo estado, en vez de involucrar dos estados distintos. Representa un evento que no causa cambio de estado. Se denota como una cadena adicional en el compartimiento de acciones del estado.

Los diagramas de estados son buenos para describir el comportamiento de un objeto a través de varios casos de uso.

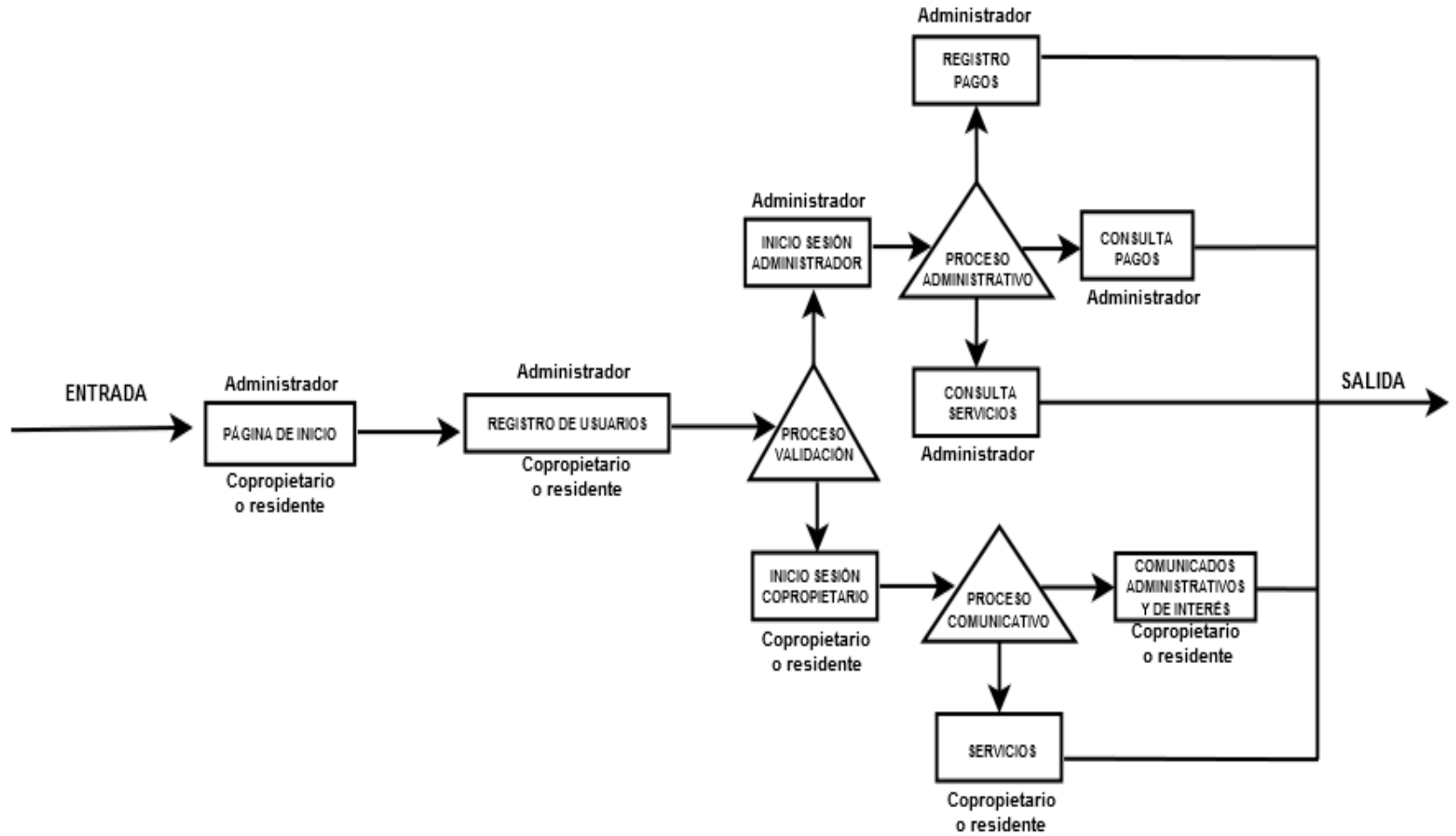
#### 14.4. FLUJOGRAMA DE PROCESOS

El flujograma también es conocido como diagrama de flujo y en este sentido, representa de manera gráfica de un proceso que puede responder a diferentes ámbitos: programación informática, procesos dentro de una industria, psicología de la cognición o el conocimiento, economía, entre otros.

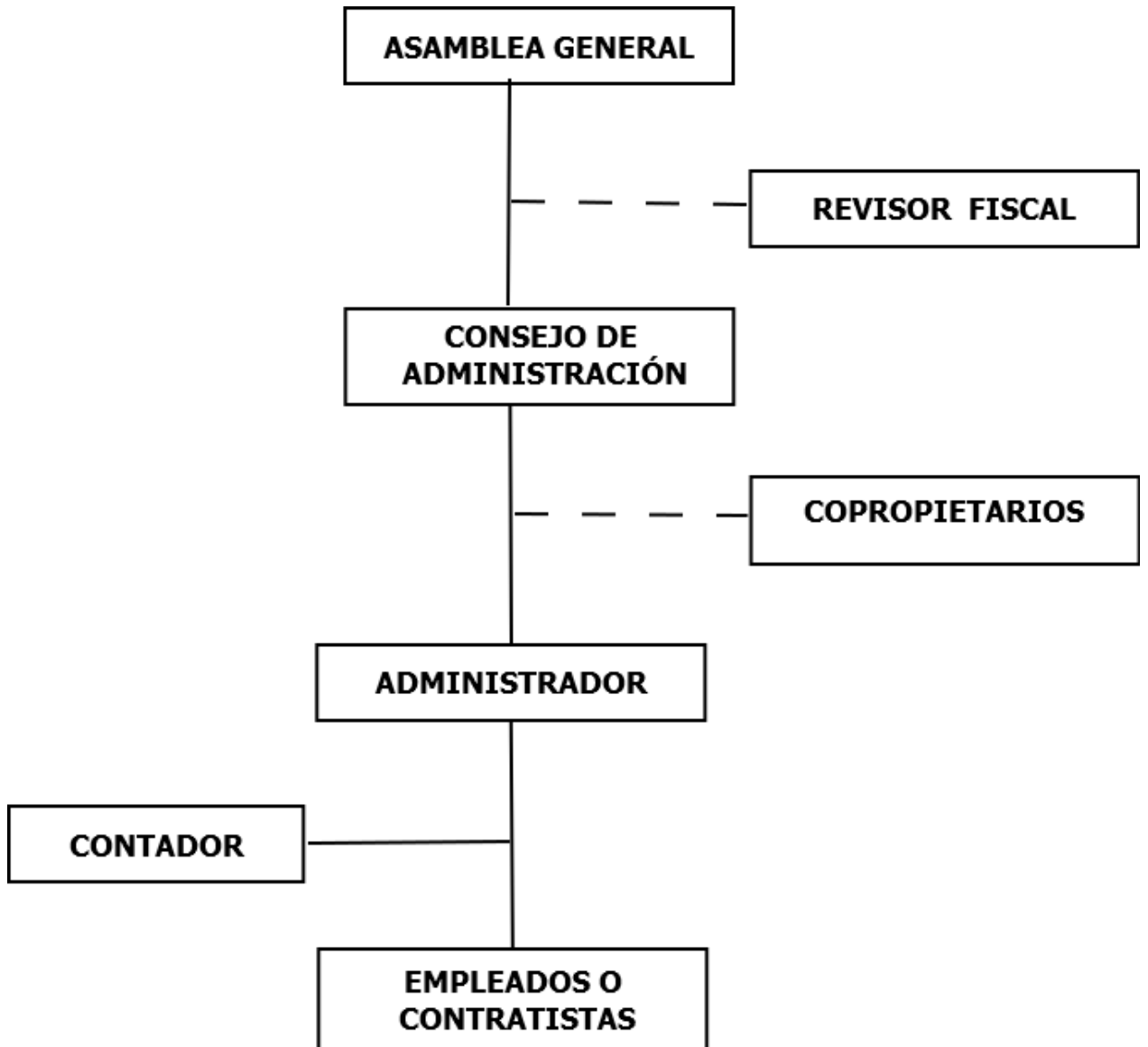
Por ejemplo, el diagrama de flujo puede ser utilizado para describir paso a paso las operaciones que se realizarán dentro del proceso de fabricación de un producto, o asimismo la perspectiva comercial de una empresa o negocio.

Los flujogramas utilizan una variedad de símbolos definidos donde cada uno representa un paso del proceso, y la ejecución de dicho proceso es representado mediante flechas que van conectando entre ellas los pasos que se encuentran entre el punto de inicio (comienzo) y punto de fin del proceso (final). Una característica importante de los diagramas de flujo es que sólo pueden poseer un único punto de inicio o comienzo, y un solo punto final o de fin del proceso.

## FLUJOGRAMA DE PROCESOS



#### 14.5. ORGANIGRAMA DE LA EMPRESA



## 14.6. DIAGRAMA DE FLUJO DE DATOS

Es una representación gráfica del "flujo" de datos a través de un sistema de información. Un diagrama de flujo de datos también se puede utilizar para la visualización de procesamiento de datos (diseño estructurado). Es una práctica común para un diseñador dibujar un contexto a nivel de DFD que primero muestra la interacción entre el sistema y las entidades externas. Este contexto a nivel de DFD se "explotó" para mostrar más detalles del sistema que se está modelando.

Los diagramas de flujo de datos son útiles a lo largo del proceso de análisis y diseño. Existen compromisos para decidir que tanto deben ser explotados de los flujos de datos. Se desperdiciara tiempo y se sacrificara comprensibilidad si los diagramas de flujo de datos son exclusivamente complejos. Por otro lado, si los diagramas de flujo de datos están muy poco explotados, pueden ocurrir errores u omisiones que pueden eventualmente afectar el sistema que está en desarrollo. Por último, recuerde que los diagramas del sistema de flujo pueden ser usados para documentar niveles altos o bajos del análisis y para ayudar a sustentar la lógica subyacente en los flujos de datos de la organización.

Los diagramas de flujo de datos desarrollan:

- Muestran que debe hacer el sistema sin referencias.
- Son diagramas explícitos y comprensibles.
- Dan la posibilidad de representar el sistema a diferentes niveles de complejidad, desde lo más global a lo más detallado solo requieren de 4 símbolos.
- Son fácil de mantenimiento, pues los cambios afectan solo algunos de sus elementos y no al todo.

Reglas para la creación de los diagramas:

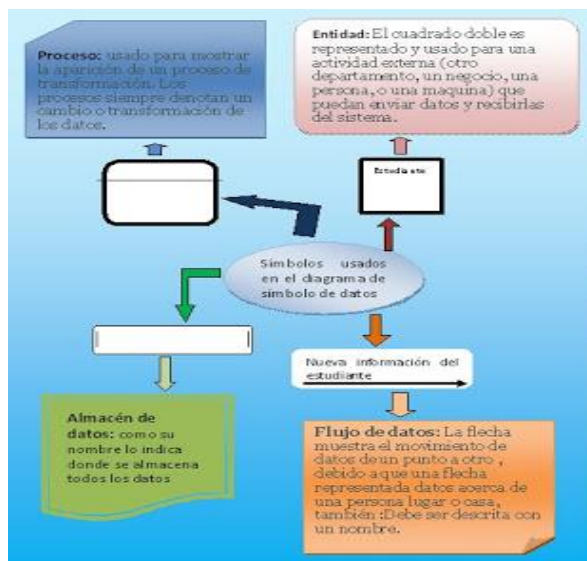
- Los diagramas de flujo deben escribirse de arriba hacia abajo y/o de Izquierda a derecha.
- Los símbolos se unen con líneas, las cuales tienen en la punta una flecha que indica su dirección que fluye la información procesos, se deben utilizar solamente líneas de flujo horizontal o vertical (nunca diagonales).

- Se debe evitar el cruce de líneas, para lo cual se quisiera separar el flujo del diagrama a un sitio distinto, se pudiera realizar utilizando los conectores, se debe tener en cuenta que solo se van a utilizar conectores cuando sean estrictamente necesario.

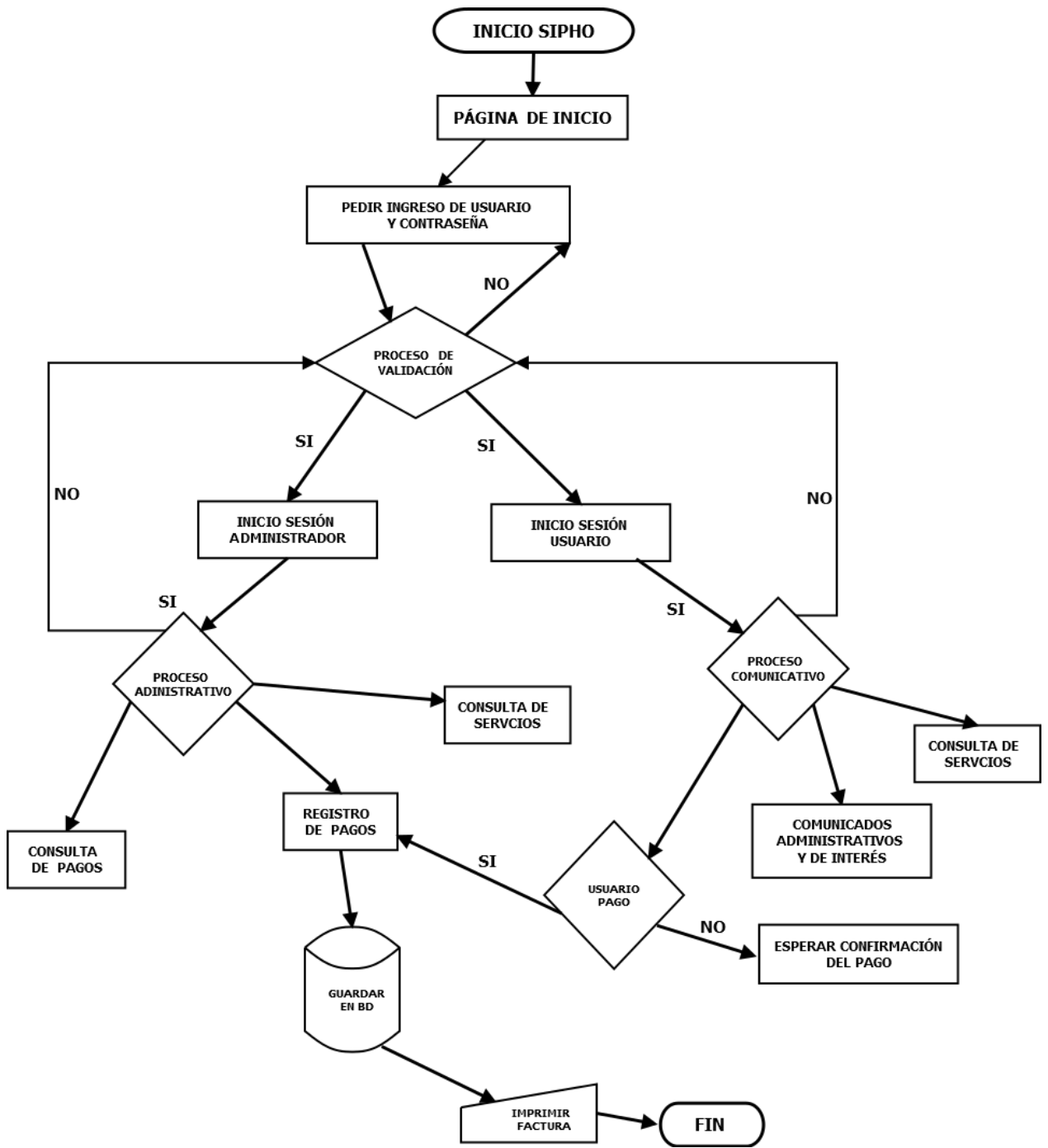
- No deben quedar líneas de flujo sin conectar.
- Todo texto escrito dentro de un símbolo debe ser legible, preciso, evitando el uso de muchas palabras.

- Todos los símbolos pueden tener más de una línea de entrada, a excepto del símbolo final.
- Solo los símbolos de decisión pueden y deben tener más de una línea de flujo de salida.

Ejemplo de Diagrama de Flujo: Diagrama de flujo que encuentra la suma de los primeros 50 números naturales.

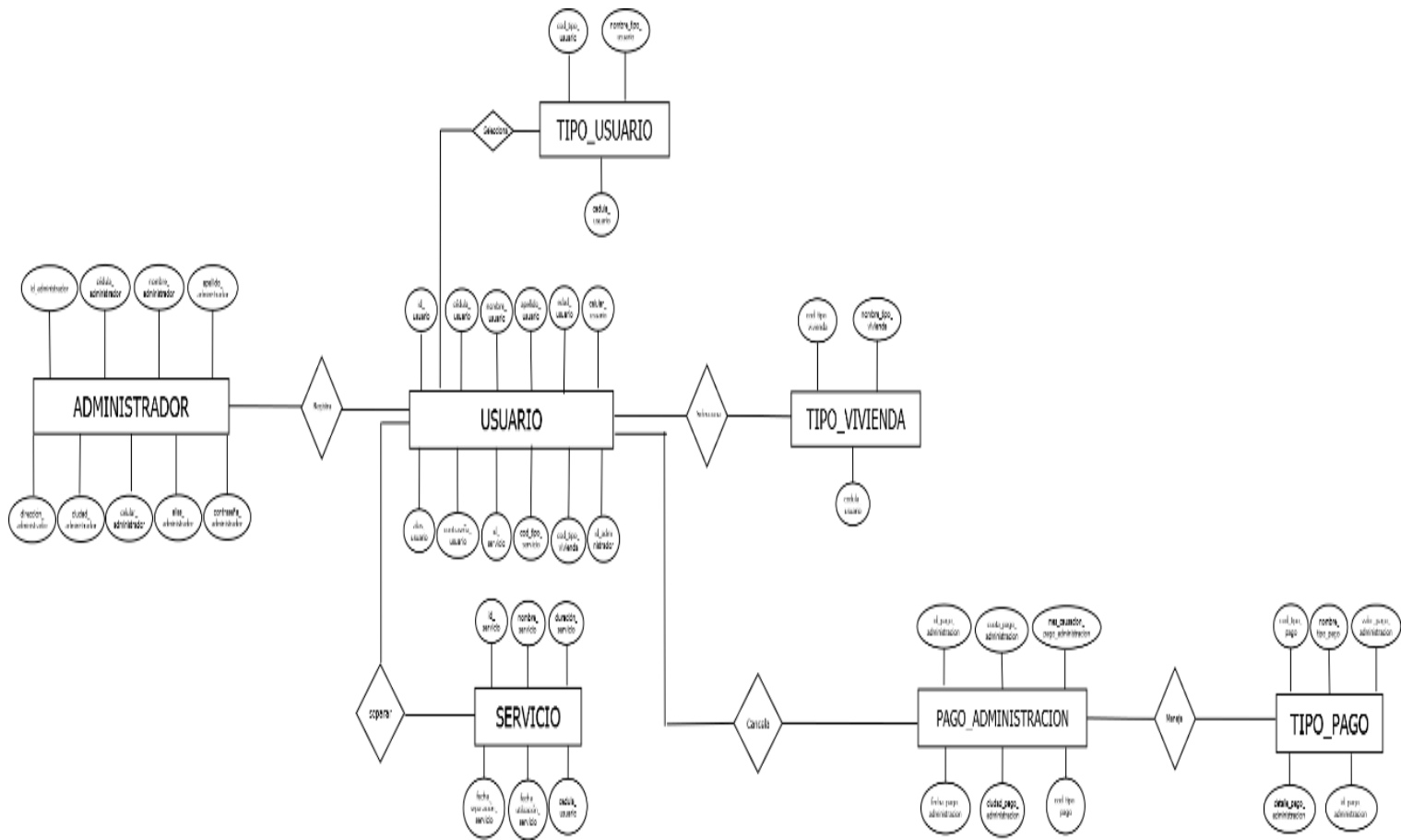


# DIAGRAMA DE FLUJO DE DATOS



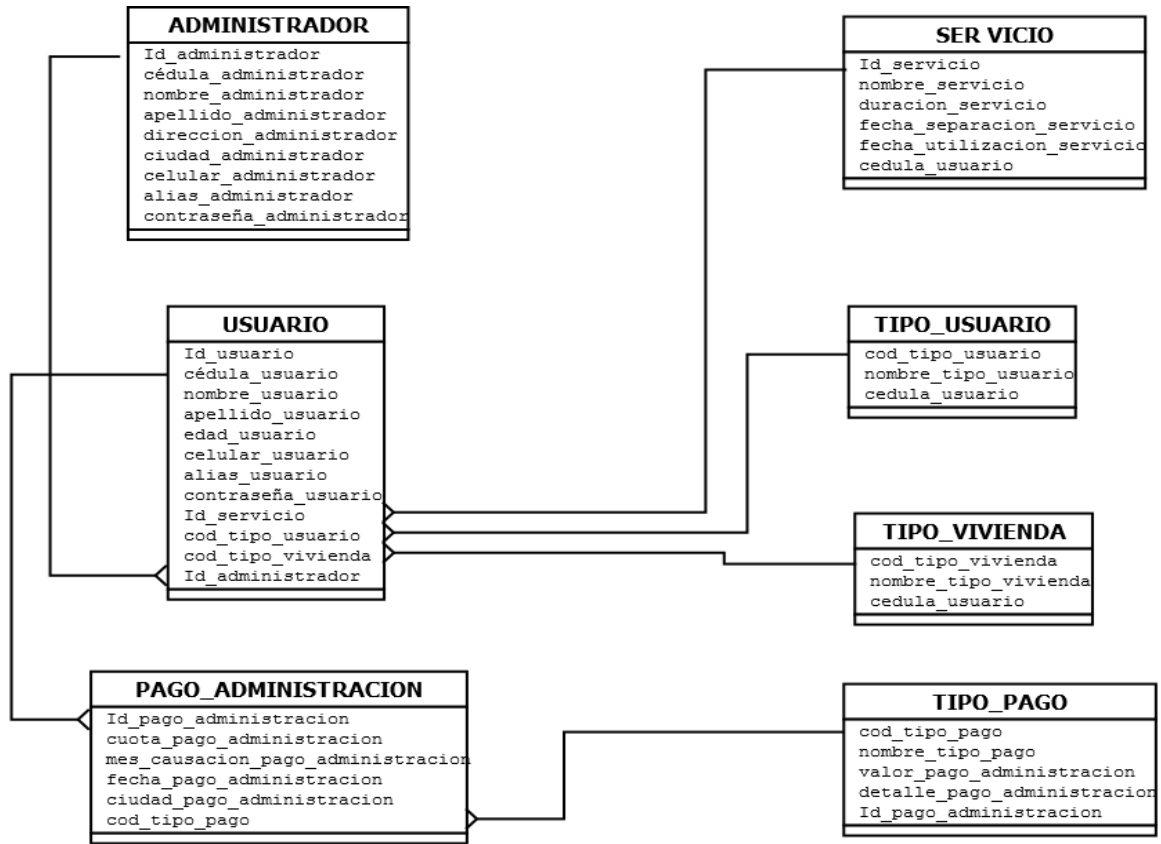
## 15. MODELOS DE DATOS

### 15.1. MODELO E – R

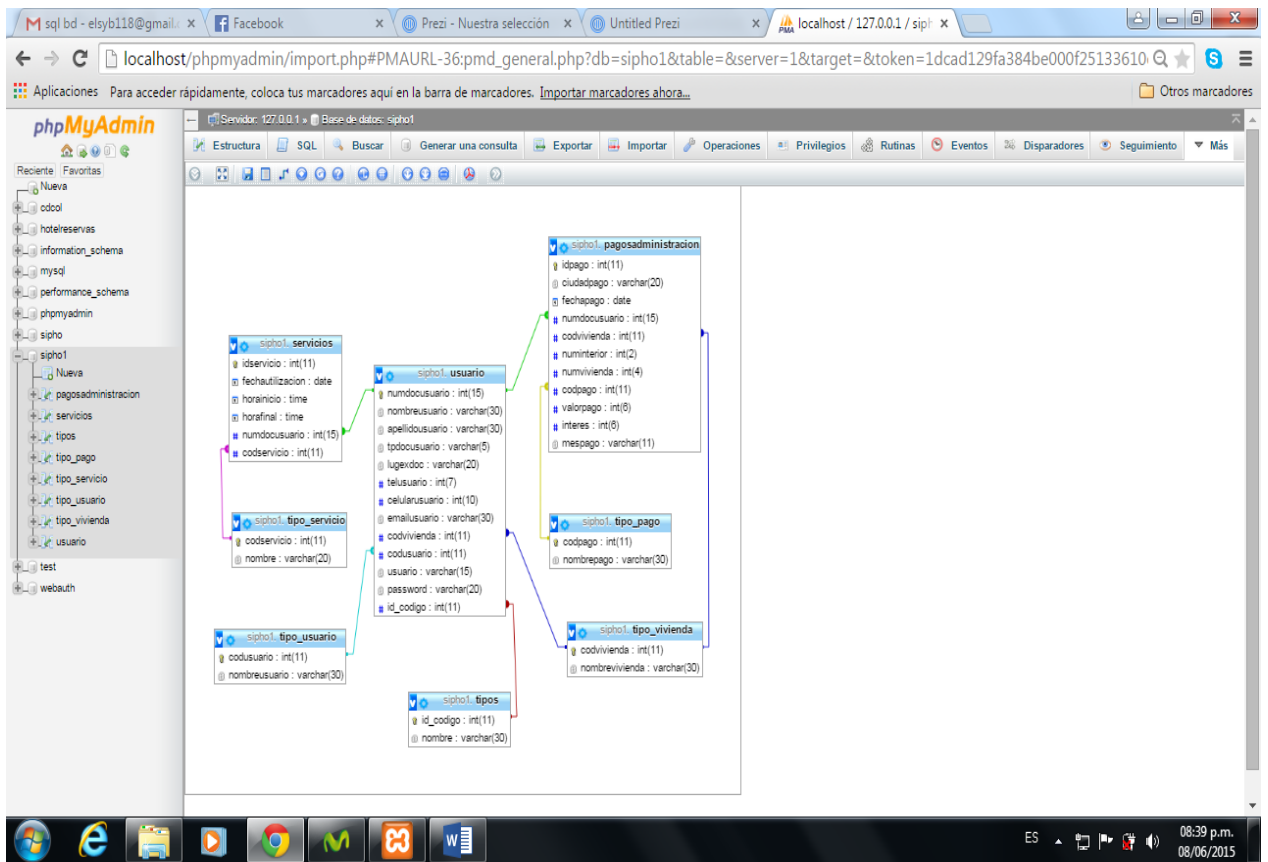




## 15.2. MODELO RELACIONAL



## 15.3. MODELO TABULAR



## 16. VIABILIDAD O FACTIBILIDAD

Se conoce como análisis de viabilidad al estudio que intenta predecir el eventual éxito o fracaso de un proyecto. Para lograr esto parte de datos empíricos (que pueden ser contrastados) a los que accede a través de diversos tipos de investigaciones (encuestas, estadísticas, etc.).

Según Varela, “se entiende por Factibilidad las posibilidades que tiene de lograrse un determinado proyecto”. El estudio de factibilidad es el análisis que realiza una empresa para determinar si el negocio que se propone será bueno o malo, y cuáles serán las estrategias que se deben desarrollar para que sea exitoso. Según el Diccionario de la Real Academia Española, la Factibilidad es la “cualidad o condición de factible”. Factible: “que se puede hacer”.

### 16.1. TÉCNICA

El sistema de información requiere un software y un hardware con los requerimiento mínimos: un equipo de cómputo de escritorio o portátil, sistema Operativo Microsoft Windows XP Professional Servipak 2 o superior, Procesador Intel Pentium II, o AMD cómo mínimo de 2600 + 1.60 GHz y 1024 de Ram, espacio de disco duro 100 MB libres o superior, unidad de CDROM o puerto de conexión USB, monitor revolución mínima de 800x600 píxeles, teclado, Mouse, impresora , no necesitan hacer actualizaciones, pero en caso de que lo hagan los componentes se encuentran en el mercado actualmente a precios muy bajos, acceso permanente a internet.

## **16.2. HUMANA**

Para el manejo adecuado del sistema de información se capacitará a dos personas del conjunto residencial, en este caso al administrador y a la secretaria del consejo.

## **16.3. LEGAL**

Por ser software libre no requiere comprar licencias para su utilización legal.

En Colombia se cuenta con un marco legal que ha determinado el manejo y la aplicación de las leyes en las empresas que administran propiedad horizontal: Ley 675 del año 2001.

## **16.4. FINANCIERA**

El conjunto residencial Quintas del portal debe hacer una inversión de tres millones de pesos \$3.000.000 para el equipo de cómputo y la infraestructura adecuada para su funcionamiento; igualmente tener conexión a internet.

Guiados por el estudio hecho se concluye que la implementación del sistema de información SIPHO no representa grandes costos y es posible que otras propiedades horizontales quieran adquirir este sistema de información.

## **17. COSTOS**

### **COSTOS DEL PROYECTO**

Los valores relacionados en las siguientes tablas se deriva de la sumas de los tres participantes durante un término en tiempo de dos meses de desarrollo.

## TABLA DE COSTOS

### 17.1. FIJOS

<b>COSTOS FIJOS</b>		
<b>REF</b>	<b>DETALLE</b>	<b>VALOR</b>
1	SERVICIO DE LUZ X3	140000
2	SERVICIO DE INTERNET X3	270000
3	SERVICIO DE AGUA X3	300000
4	SALARIOS MINIMO X 3	4080000
	<b>CFT</b>	4790000

### 17.2. VARIABLE

<b>COSTOS VARIABLES</b>		
<b>REF</b>	<b>DETALLE</b>	<b>VALOR</b>
1	ALIMENTACION X3	1200000
2	TRANSPORTE X3	600000
3	SUMINISTROS	300000
	<b>CVT</b>	2100000

17.3. UTILIDAD DE 20% = 6.890.000 \* 20%= 1.378.000

17.4. VALOR TOTAL DEL PROYECTO = 8.268.000

## 18. MATRIZ FODA:

<p><b>FORTALEZAS:</b></p> <p>SIPHO es un sistema de información que pueden utilizar las viviendas de propiedad horizontal, en su versión beta.</p> <p>Ha sido diseñado siguiendo las normas legales vigentes.</p> <p>Es un sistema de información de fácil manejo y con una interfaz agradable.</p> <p>Dispone de las herramientas necesarias para su funcionamiento.</p>	<p><b>OPORTUNIDADES</b></p> <p>Amplía oportunidad para la organización de la información de pequeñas empresas.</p> <p>Acceder a un producto con calidad.</p> <p>Ampliación en oferta a otros conjuntos residenciales de la versión beta.</p> <p>Poner en práctica los contenidos aprendidos a lo largo de la tecnología en informática.</p>
<p><b>DEBILIDADES:</b></p> <p>Falta optimización del tiempo de elaboración.</p> <p>Falta de tiempo para mejorar las pruebas.</p> <p>Falla del administrador en seguir procesos.</p>	<p><b>AMENAZAS:</b></p> <p>Fortaleza comercial de empresas competidoras.</p> <p>Cambio en la reglamentación legal de propiedad horizontal.</p> <p>Los usuarios no utilicen el sistema de información de manera asertiva.</p>

## 19. DICCIONARIO DE DATOS

SIPHO: sistema de información para propiedad horizontal.

Aplicación de Escritorio: Aplicación que almacena sus datos en un archivo o en una base de datos central muchas de estas aplicaciones funcionan en un sistema operativo como Microsoft Windows o Linux.

Base de Datos o Banco de Datos: Es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso.

Modelo de Base de Datos: Estructura o formato de la base de datos descrita en un lenguaje formal soportada por un sistema.

Modelo Entidad Relación: cuando se necesita gestionar información se construye una base de datos pero antes de construirla se diseña definiendo las entidades y sus relaciones

Entidad: Objeto real del proyecto donde queremos almacenar información estas se componen por atributos que son los datos que definen a la entidad:

Atributos: Define las propiedades de una entidad.

Relación: Asociación entre las entidades. Pueden ser de tres tipos: Relaciones 1- 1 relaciones 1-n y relaciones n-n.

Clave Primaria: Un campo o a una combinación de campos que identifica de forma única a cada fila de una tabla.

Clave Foránea: Vinculo que existe entre las tablas.

Cardinalidad: Elementos básico del modelo de entidad relación, define el número máximo de relaciones de objetos que pueden participar en una relación.

## 20. GLOSARIO

**Framework:** infraestructura digital. En el desarrollo de software, es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos de *software* concretos, que puede servir de base para la organización y desarrollo de software. Típicamente, puede incluir soporte de programas, bibliotecas y en un lenguaje interpretado, entre otras herramientas, para así ayudar a desarrollar y unir los diferentes componentes de un proyecto. Representa una arquitectura de software que modela las relaciones generales de las entidades del dominio, y provee una estructura y una especial metodología de trabajo, la cual extiende o utiliza las aplicaciones del dominio.

**Bootstrapp:** Es un framework o conjunto de herramientas de software libre para diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como, extensiones de JavaScript opcionales adicionales.

**PHP:** Es un lenguaje de programación código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML en lugar de llamar a un archivo externo que procese los datos. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera la página Web resultante. PHP ha evolucionado por lo que ahora incluye también una interfaz de línea de comandos que puede ser usada en aplicaciones gráficas independientes. Puede ser usado en la mayoría de los servidores web al igual que en casi todos los sistemas operativos y plataformas sin ningún costo.

## 21. CIBERGRAFIA



- 22.1. <http://www.guiadesolucionestic.com/soluciones-verticales/sector-propiedad-horizantal/gestion-de-propiedad-horizantal>
- 22.2. <http://www.softgafin.com/index.php/propiedad-horizantal>
- 22.3. <http://helisa.com/helisaniif/>
- 22.4. [http://metodologiadesoftware.blogspot.com/2012/11/fases-del-modelo-rup\\_27.html](http://metodologiadesoftware.blogspot.com/2012/11/fases-del-modelo-rup_27.html)
- 22.5. <http://www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=4162>
- 22.6. [http://www.upf.edu/hipertextnet/numero-1/sistem\\_infor.html](http://www.upf.edu/hipertextnet/numero-1/sistem_infor.html)
- 22.7. <http://www.monografias.com/trabajos7/sisinf/sisinf.shtml>
- 22.8. <http://www.monografias.com/trabajos29/ciclo-sistema/ciclo-sistema.shtml>
- 22.9. [http://es.wikipedia.org/wiki/Desarrollo\\_en\\_espiral](http://es.wikipedia.org/wiki/Desarrollo_en_espiral)
- 22.10. [http://www.sites.upiicsa.ipn.mx/polilibros/portal/Polilibros/P\\_externos/Administracion\\_informatica\\_de\\_las\\_organizaciones\\_Ramon\\_E\\_Enriquez\\_Gonzalez/AIO\\_2\\_Mod\\_ESPIRAL.html](http://www.sites.upiicsa.ipn.mx/polilibros/portal/Polilibros/P_externos/Administracion_informatica_de_las_organizaciones_Ramon_E_Enriquez_Gonzalez/AIO_2_Mod_ESPIRAL.html)
- 22.11. [http://es.wikipedia.org/wiki/Desarrollo\\_en\\_cascada](http://es.wikipedia.org/wiki/Desarrollo_en_cascada)
- 22.12. <http://eii.ucv.cl/pers/gbustos/PDF/Evalua.PDF>
- 22.13. <http://www.monografias.com/trabajos60/metodologias-desarrollo-software/metodologias-desarrollo-software.shtml>
- 22.14. <http://metodologiarad.weebly.com/>
- 22.15. [http://es.wikipedia.org/wiki/Programaci%C3%B3n\\_extrema](http://es.wikipedia.org/wiki/Programaci%C3%B3n_extrema)
- 22.16. <http://procesosdesoftware.wikispaces.com/METODOLOGIA+XP>
- 22.17. <http://profesores.fi-b.unam.mx/carlos/aydoo/uml.html>
- 22.18. <http://users.dcc.uchile.cl/~psalinas/uml/casosuso.html>
- 22.19. <http://es.slideshare.net/nedowwhaw/diagrama-de-clases-16208245>
- 22.20. <https://www.google.com.co/url?sa=t&rct=j&q=&esrc=s&source=web&cd=15&ved=0CFEQFjAO&url=http%3A%2F%2Fvirtual.usalesiana.edu.bo%2Fweb%2Fpractica%2Farchiv%2Festados.doc&ei=pYFGVcjjIsPCggSmhIDwDg&usg=A>

FQjCNFOSRCAGO0oenAqwgOEnK65zfcKWQ&bvm=bv.92291466.d.eXY&c  
ad=rjt

**22.21.** <http://ilandinezsanchez.blogspot.com/2009/08/diagramas-de-flujo.html>

**22.22.** <http://definicion.mx/flujogramas/>

**22.23.** <http://definicion.de/viabilidad/>

**22.24.** <http://estudiodefactibilidadyproyectos.blogspot.com/2010/09/factibilidad-y-viabilidad.html>