UNIVERSIDAD TECNOLOGICA DE CHILE
INSTITUTO PROFESIONAL
CENTRO DE FORMACION TECNICA

MANUAL

SISTEMAS DE INFORMACIÓN I

Y

SISTEMAS DE INFORMACIÓN II

Índice

1 Introducción 4

2. Fundamentos de los Sistemas de información 5

2.1 Conceptos 5

2.2 Componentes de los sistemas de Información 6

2.3 Clasificación de sistemas de información 7

2.3.1 Sistemas de Información Transaccionales (TPS, Transaction Processing Systems) 7

2.3.2 Sistemas de Información Administrativos (MIS, *Management Information Systems)* 8

2.3.3 Sistema de Apoyo a la Toma de Decisiones (DSS, *Decisión Support Systems)* 8

2.3.4 Sistemas Expertos e Inteligencia Artificial 9

2.4 Cuadro Comparativo 10

2.5 Ciclo de Vida de un Sistema de Información 11

Estudio de Factibilidad 11

Etapa de Análisis 12

Etapa de Diseño 12

Etapa de Construcción 13

Etapa de Prueba 13

Etapa de Implantación 13

Etapa de Mantención 15

2.6 Metodologías de desarrollo 16

2.6.1 El Modelo Clásico o de Cascada 16

2.6.2 El Modelo de construcción de prototipos 17

2.6.3 El Modelo Incremental 19

2.6.4 El Modelo en Espiral 20

2.6.5 Modelo DRA (Desarrollo Rápido de Aplicaciones) 21

2.7 Enfoques de Desarrollo 22

Enfoque Estructurado 22

Enfoque Orientación a Objetos 23

Enfoque Orientado a Aspectos 23

3. Enfoque Orientado a Objeto Aplicado a un Sistema de Información 23

4 Modelado de procesos de negocio 25

4.1 Conceptos y Descripciones. 25
4.2 BPMN: Business Process Management Notation 26
Definición 26
Beneficios 26
Notación 26
Proceso unificado de desarrollo (RUP) 28
Conceptos, elementos y características. 28
5 Lenguaje de Modelado Unificado UML 32
5.1 Conceptos básicos 32
5.2 Vistas de UML 33
5.3 Diagramas de UML 35
5.3.1 Diagrama de Caso de Uso 35
Descripción 35
Elementos 35
Ejercicios Propuesto 37
Descripción 40
Actividades 40
Objetos 40
Decisiones 41
Sincronización 41
Estado Inicial / Estado Final 41
Ejercicio 42
5.3.3 Diagrama de Clase 43
Descripción 43
Clase 43
Atributos 43
Métodos 43
Asociaciones 44
Multiplicidad 44
Herencias y generalización 44
Agregación 44
Composición 44

Clase Asociación 45
Ejercicio Resuelto 45
Ejemplo de creación de clases en JAVA 46
5.3.4 Diagrama de Estados 48
Descripción 48
Estados 48
Transición 48
Estado Inicial / Estado Final 48
Ejercicio Resuelto 48
5.3.5 Diagrama de Secuencias 49
Descripción 49
Objetos 49
Activación 49
Mensajes 49
Ejercicio Resuelto 50
5.3.6 Diagrama de Colaboraciones 50
Descripción 50
Objeto 51
Mensajes 51
Ejercicio propuesto 51
6 Patrones de Diseño 52
Patrones de Creación. 52
Patrones Estructurales. 53
Patrones de Comportamiento. 54
7 Ejemplos varios 55
8 Ejercicios Desarrollados 63
9 Ejercicios Propuestos 71
10 Bibliografía 75
11 Anexos 76

1 Introducción

Cómo se puede llevar a cabo lo que el usuario necesita, y más aun a través de un sistema computacional, una gran pregunta a contestar e incluso suena como una misión imposible.

Pero no es así, puesto que tenemos una serie de métodos, técnicas y herramientas que nos permitirán poder lograr dicha misión. Otra pregunta a contestar, cuál de ellas es la más adecuada?, pues bien para poder obtener una respuesta, debemos de poseer todos los antecedentes para poder seleccionar la mejor opción.

Al momento de comenzar nuestra evaluación de los antecedentes se nos bombardea con un sinfín de conceptos que nos dicen que debemos hacer. Además se nos presentan diversos enfoques a utilizar para el desarrollo de un proyecto, pero con el trascurso del tiempo nos damos cuenta que estos evolucionan y que en algún sentido los de años anteriores quedan obsoletos. Y el desarrollo de proyecto en el área informática ha evolucionado.

Los antiguos informáticos nacieron en el mundo de un enfoque estructurado, poco flexible que cuartaba en cierta medida el desarrollo de un sistema informático. En la actualidad estamos en la era de la colaboración, esto tiene referencia a ser más participe al usuario en el desarrollo de un proyecto.

Porque es importante la participación del usuario sobre todo en las primeras etapas, ya que se nos dice que el desarrollo de un sistema esta subdivido en etapas bien marcadas. Otra pregunta de gran relevancia, pues bien él es el dueño de producto que hemos de desarrollar, además el es el que tiene una necesidad a ser satisfecha.

Por lo cual contamos con metodologías que nos dice como debemos de enfrentar el desarrollo de las etapas de un proyecto. Cuando identificamos etapas estamos frente a diversas fases que deben de tener una concordancia una con otra o un hilo conductor que las acople. Es así como tres informáticos conocidos como los tres amigos (Rumbaugh, Jocaobson y Booch), crearon tres metodologías, por separado, que permitían tener una visión del desarrollo de un sistema, pero al unirlas se dieron cuenta que encajaban como un rompecabezas que permitían solventar las debilidades y crear un lenguaje que fuera universal para el desarrollo de un sistema informático, UML.

Hoy en día se habla del nuevo enfoque utilizado en el desarrollo de un sistema informático, Orientación a Objeto. Los sistemas se han vuelto más complejos, puesto que las necesidades han cambiado, pero también hay actores o piezas que interactúan. La clave está en contar con una organización del proceso de diseño en el cual los participantes comprendan y convengan con él, para el logro del sistema requerido. El enfoque orientado a objeto con UML permite esta interacción.

2. Fundamentos de los Sistemas de información

2.1 Conceptos

El uso de las tecnologías en el mundo actual es algo normal, pero hoy en día el uso de tecnologías de información ha producido cambios significativos en el manejo de información en las empresas. Todos estamos consientes de que vivimos rodeados de datos que a través de un proceso los transformamos en información.

Es así como nacen los Sistema de Información. Cuando escuchamos la palabra Sistema, la asociamos a diferentes conceptualizaciones, por ejemplo: Sistema Respiratorio, Sistema Político, Sistema Bancario, Sistema Económico, y muchas más. Son muchos los expertos que presentan una definición a este concepto. Entonces que nos queda, pues bien, establecer que entenderemos por sistema en el ámbito en el que se enmarca este documento, la informática.

Se entenderá por sistema:

"*Un conjunto de elementos interrelacionados que producen como resultado algo distinto a la simple reunión de elementos*". A través de esta definición se puede desprender que un sistema está formado por los siguientes componentes: Elementos, Relaciones y Un Objetivo.

> Que entenderemos por elementos o partes, cualquier objeto o ente del mundo real. Estos elementos pueden ser tangibles o intangibles, estáticos o dinámicos.
>
> Que entenderemos por relaciones, son las que hacen que todo sistema pueda funcionar, ya que como está formado por partes estas deben de comunicarse para el logro del objetivo trazado.
>
> Que entenderemos por objetivo, es la razón de ser del sistema. Es decir, es el que define al sistema, puesto que si no tengo bien definido lo que se desea lograr difícilmente se lograra desarrollar un sistema en forma correcta.

Se entenderá como Información:

> "U*n signo o conjunto de signos que impulsan a la acción. Se distingue de los datos, porque estos no son estímulos de la acción, si no simplemente cadenas de caracteres o patrones sin interpretar*." (Gordon B.Davis)

A continuación se dan dos definiciones de Sistema de Información:

> "*Es un conjunto integrado de personas y máquinas cuyo objetivo es entregarle a una organización la información requerida para apoyar las operaciones, la administración y la toma de decisiones. El sistema utiliza máquinas y equipos computacionales (HW), programas e instrucciones computacionales (SW), procedimientos manuales, base de datos, modelos de análisis, planificación, control y toma de decisiones*."(Davis y Olson)

"*Es aquel que tiene por objetivo proveer a una organización la información necesaria (pasado, presente y futura), en forma precisa y oportuna, para que pueda servir para la toma de decisiones en un entorno competitivo*." (Kovacevic y A. Gonzalez)

Estas definiciones engloban el concepto Sistemas de Información en el ámbito informático. Además se debe de establecer que desde el punto de vista informático tomaremos las siguientes características[1] para que un sistema sea considerado como un sistema de información:

- Existen dentro de un entorno.
- Se encuentran separados de su entorno por alguna frontera.
- Tiene entradas y salidas. Reciben entradas desde el entorno y envía salidas a su entorno.
- Transforman sus entradas de alguna forma para producir sus salidas.
- Dispone de interfaces. Las interfaces permiten la comunicación entre los sistemas.
- Un sistema puede estar formado por subsistemas.
- Los sistemas que realizan funciones disponen de un mecanismo de control
- El mecanismo de control está basado en retroalimentación. La retroalimentación maneja información sobre las operaciones del sistema o su entorno, que luego se pasara al mecanismo de control.
- Dispone de ciertas propiedades que no dependen directamente de las propiedades de sus partes.

2.2 Componentes de los sistemas de Información

Figura 1: Componentes de un SI

[1] Análisis y Diseño Orientado a Objeto de Sistemas Usando UML, Simon Bennett, Steve McRobb y Ray Farmer.

2.3 Clasificación de sistemas de información

Una empresa es un sistema, que está conformado por una serie de departamentos que poseen necesidades. De acuerdo con el modelo de Herbert Simón, las organizaciones se estructuran en *tres niveles*: el *nivel operativo*, constituido por un sistema de procesos físicos de producción y distribución; el *nivel de las decisiones programadas*, que maneja operaciones, y *el nivel de las decisiones no programadas*, que genera decisiones programadas.

Figura 2: Niveles en la Empresa

Por lo cual como tenemos diversos niveles, se pueden identificar diferentes tipos de sistema de información, tales como:

2.3.1 Sistemas de Información Transaccionales (TPS, Transaction Processing Systems)

Los sistemas de información transaccionales fueron los primeros en ser incorporados al procesamiento de un volumen considerable de datos en forma computacional. En el contexto de transaccionales se debe de establecer que una transacción es un intercambio entre el usuario que opera el equipo y el sistema de procesamiento de datos. Es decir, esto implica la captura y validación de los datos ingresados por el usuario, así como la búsqueda y actualización de archivos.

Por lo cual podemos establecer que los sistemas de información transaccional están orientados a satisfacer las necesidades del nivel operativo de la empresa. Realizando operaciones repetitivas y sencillas que conllevan a informatizar los procesos que poseen tareas rutinarias y tediosas, con el fin de minimizar los errores y disminuir la cantidad de mano de obra.

Características:

- Logran ahorro significativo de mano de obra.
- Son el tipo de SI que se implanta en la organización.
- Son intensivos en entrada y salida de información.
- Sus cálculos y procesos suelen ser simples, poco sofisticado.
- Tienen la propiedad de ser recolectores de información.
- Son fáciles de justificar ante la dirección ya que sus beneficios son visibles y palpable

En esta clasificación se pueden encontrar sistemas que prácticamente con comunes en todas las organizaciones, tales como: Contabilidad, Facturación, Inventarios, Ventas, Proveedores, Cuentas Corrientes, Cobranzas, Caja, Bancos, Sueldos, Finanzas, Compras, Planeamiento y Control de la Producción, etc.

2.3.2 Sistemas de Información Administrativos[2] (MIS, *Management Information Systems)*

Los sistemas de información Administrativos están orientados al apoyo de los niveles directivos o ejecutiva en su proceso de toma de decisiones y resolución de alguna problemática. Los directivos recurren a los datos almacenados de las operaciones transaccionales para poder evaluar las opciones que se pueden presentar para tomar una decisión, a través de una presentación de información más procesada y analizada. Por ejemplo: Sistemas de Gestión Académica, Sistema de Control de Proveedores, Sistema de Planificación de la Producción, Sistema de Evaluación de Proveedores, Sistema de Seguimiento de Ordenes de Trabajo, etc.

2.3.3 Sistema de Apoyo a la Toma de Decisiones[3] (DSS, *Decisión Support Systems)*

Los sistemas de apoyo a la Toma de Decisiones son aquellos que tienen por misión ser una herramienta para el apoyo de la función ejecutiva. No hay que dejar de lado que su base como sistema está en base a los sistemas transaccionales y administrativos, sino este tipo de sistemas no existiría.

Estos tipos de sistemas ponen un énfasis en el apoyo a la toma de decisiones en todas sus fases, aunque hay que establecer que en definitiva la responsabilidad de tomar la decisión es exclusiva del responsable. Y la finalidad de estos sistemas es aumentar la eficacia y disminuir el esfuerzo humano en el proceso de toma de decisiones.

Estos sistemas se orientan a ensamblar la información y el juicio humano para alcanzar mejores resultados decisorios.

---

[2] Análisis y Diseño de Sistemas, Kendall & Kendall

[3] Análisis y Diseño de Sistemas, Kendall & Kendall

Los objetivos[4] de un DSS son:

• Apoyar (no reemplazar) el juicio humano, de tal modo que el potencial de los procesos del hombre y de la máquina sea utilizado al máximo.
• Crear herramientas de apoyo bajo el control de los usuarios, sin automatizar la totalidad del proceso decisorio predefiniendo objetivos o imponiendo soluciones.
• Ayudar a incorporar la creatividad y el juicio de responsable de la toma de decisiones en las fases de formulación del problema, selección de los datos, y generación y evaluación de alternativas.
• Apoyar a los ejecutivos de alto nivel en la solución de problemas prácticos no totalmente estructurados y en los que, hallándose presente algún grado de estructura, el juicio sea esencial.

Características:

- Suelen introducirse después de haber implantado los sistemas transaccionales.
- Suelen ser intensivos en cálculos y escasos en entradas y salidas de información.
- La información que generan sirve de apoyo a los mandos intermedios y de alta administración en el proceso de la toma de decisiones.
- No suelen ahorrar mano de obra.
- La justificación económica para el desarrollo de estos sistemas es difícil.
- Suelen ser SI interactivos y amigables, con altos estándares de diseño gráfico y visual, ya que están dirigidos al usuario final.
- Apoyan la toma de decisiones que por su naturaleza son receptivas.
- Pueden ser desarrollados directamente por el usuario final sin la participación operativa de los analistas.

Ejemplos: Sistemas de Planificación estratégica, Sistemas Evaluación de mercados, Sistemas de Investigación, Sistemas BI, Sistema de Gestión de Clientes, Sistemas de Gestión de la Cadena de Distribución, etc.

2.3.4 Sistemas Expertos e Inteligencia Artificial[5]

La inteligencia artificial (AI, *Artificial Intelligence)* se puede considerar como el campo general para los sistemas expertos. La motivación principal de la AI ha sido desarrollar máquinas que tengan un comportamiento inteligente. Dos de las líneas de investigación de la AI son la comprensión del lenguaje natural y el análisis de la capacidad para razonar un problema hasta su conclusión lógica. Los sistemas expertos utilizan las técnicas de razonamiento de la AI para solucionar los problemas que les plantean los usuarios de negocios (y de otras áreas).

Los sistemas expertos conforman una clase muy especial de sistema de información que se ha puesto a disposición de usuarios de negocios. Un sistema experto (también conocido como sistema basado en el conocimiento) captura y utiliza el conocimiento de un experto para solucionar un problema específico en una organización. Observe que a

[4] Sistemas de Información, Raúl Horacio Saroka, Fundación OSDE
[5] Análisis y Diseño de Sistemas, Kendall & Kendall

diferencia de un DSS, que cede al responsable la toma de la decisión definitiva, un sistema experto selecciona la mejor solución para un problema o una clase específica de problemas.

Características:

- Su función no es apoyar a la automatización de procesos operativos no proporcionar información para la toma de decisiones. Sin embargo, este tipo de sistemas pueden llevar a cabo dichas funciones.
- Suelen desarrollarse “in house”
- Típicamente su forma de desarrollo es a base de incrementos y a través de su evolución permanente dentro de la organización.
- Se requiere de un experto en la estructuración y captación del conocimiento.
- Su función es lograr ventajas que los competidores no posean, tales como ventajas en costos y servicios diferenciados con clientes y proveedores.
- Apoyan el proceso de innovación dentro de la empresa. Ejemplos: Sistema de Prospección minera, Sistema de Configuración aplicaciones Distribuidas, Sistemas de Diagnóstico Médico, Sistema de Detección de Fraude Telefónico, Robots, etc.

2.4 Cuadro Comparativo

Nivel	Origen de la Información	Cantidad de Información	Tipos de Decisiones
Estratégico	Externa	Selectiva	No estructurada, no programable
Control Administrativo	Ambas	Ambas	Semi estructurada/programables
Control Operativo	Interna	Masiva	Estructuradas, programables

Tabla 1: Cuadro Comparativo por Nivel Empresa

Tipo de Sistemas	Características	Nivel
Sistemas Transaccionales	Maneja datos de procesos bien estructurados.	Operativo
Sistemas de información Administrativos	Apoyo al proceso de decisiones administrativos.	Administrativo/Estratégico
Sistemas de Apoyo a la Toma de Decisiones	Apoyo al proceso de decisiones sobre situaciones particulares.	Estratégico
Sistemas Expertos e Inteligencia Artificial	Entrega de una solución frente a una problemática.	Estratégico

Tabla 2: Cuadro Comparativo de Sistemas de Información

2.5 Ciclo de Vida de un Sistema de Información

El desarrollo completo de un sistema de información, implica la realización de una serie de etapas claramente establecidas. A estos conjuntos de etapas se le conoce como ciclo de vida de un Sistema de Información, el cual involucra desde identificar las necesidades a satisfacer hasta el desarrollo de una aplicación computacional.

Pero antes de comenzar un proyecto se debe de realizar su estudio de factibilidad, es decir, establecer si el desarrollo del sistema será realizable o no.

Estudio de Factibilidad

Al momento de identificar la realización de un sistema de información se debe de establecer si se podrá llevar a cabo o no, por lo cual se debe de evaluar los siguientes aspectos: Técnicos, Económicos, Operacionales y Legales.

Factibilidad Técnica**,** Que la alternativa sea factible en los factores más importantes que tienen que ver con los requerimientos computacionales o de equipos del sistema: Memoria, Valor del procesamiento, Impresión, Almacenamiento en disco, proceso remoto, proceso distribuido (redes), Softwares especializados (DB, MPSX). Considerar los equipos propios y las ofertas. También considerar la existencia de personal técnico propio o consultores, de nivel adecuado; Uso de Base de Datos, sistemas expertos, etc.

Factibilidad Económica**,** Considerar sólo los beneficios y costos directos realmente afectados por el funcionamiento del sistema. No valorar como beneficio la reducción de mano de obra si hay excedente y debe permanecer. No agregar como costo del sistema algún costo fijo de la Organización.

Factibilidad Operacional, Tiene que ver con el recurso humano para operar el sistema. Considerar la disciplina en la Organización y lo que requiere el Sistema, los procedimientos administrativos, la iniciativa. La idea fundamental es la evolución del estado actual de la Organización a un uso sofisticado del computador, no saltándose etapas en un corto período.

Factibilidad Legal, Tiene que ver con las normativas que el sistema debe de cumplir, es decir, si se realiza un proceso de facturación debe de manejar el trabajo con el Iva o si se utiliza un software se requiere de la licencia correspondiente.

Para mayor información revisar Anexo II.

A continuación se verán cada una de las etapas que involucra este ciclo de vida:

Etapa de Análisis

Durante la etapa de análisis se hace un examen exhaustivo de las necesidades de la organización que va a emplear el sistema, que necesita?, como lo necesita?, son algunas de las preguntas que se hacen en esta etapa. Puesto que en esta etapa es en donde se definen el ámbito y alcance del sistema a desarrollar.

Es muy importante la etapa de análisis ya que de ella depende realizar un sistema de información efectivo para los usuarios, en caso de que el análisis no se haga correctamente, es muy probable que una vez implementado se tengan que realizar adecuaciones al mismo con el fin de corregir errores o añadir requerimientos no considerados.

La fase de análisis sirve también para retomar el diseño cuando las necesidades de los usuarios rebasan las capacidades del sistema o cuando surgen novedades que tienen que ser incorporadas al sistema de información.

En esta etapa se utilizan las técnicas de recolección de datos que se ven con más detalle en el anexo I.

Etapa de Diseño

En detalle se identifica y especifica la ubicación del Sistema de Información en el contexto de la Organización. En esta etapa se establece a un nivel conceptual que es lo que el sistema desarrollara y cuáles son las operaciones que están involucradas en su funcionamiento, es decir, lo que el sistema va a hacer en la práctica.

Las actividades:

- Análisis de las características del sistema actual.
- Definición de las funciones administrativas. Su información de Entrada y de Salida.
- Evaluación de las diferentes Alternativas. Ver el Costo-beneficio y el Costo-efectividad.
- Especificar los requerimientos de información que debe satisfacer el interior
- Ver las relaciones lógicas entre todos los datos que usará el Sistema de información, (tamaño de la información).
- Especificar la agrupación física de datos (almacenamiento en archivos), que se haya decidido.
- Indicar la forma de procesar los datos, y requerimientos de software. Diagrama de procesos.
- Especificar la conversión de datos actuales y archivos al nuevo Sistema de Información.
- Documentar las indicaciones de programación para los programas que se deberá hacer.

Etapa de Construcción

Aquí el diseño se lleva a la práctica. Se procede a la construcción de la aplicación computacional que se ha diseñado en el lenguaje de programación seleccionado.

Las actividades:

- Construcción de los programas contemplados, que sean modulares y generales.
- Adquisición de programas de aplicación y de equipos.
- Comprobar la eficacia y respaldo técnico del software adquirido.
- Decisiones sobre la configuración computacional.
- Consideraciones a factores humanos, confianza, preparación y seguridad.

Etapa de Prueba

Antes de entregar el sistema a los usuarios se deben de realizar pruebas que permitan comprobar el correcto funcionamiento de las partes que componen la aplicación realizada en la etapa anterior.

Las actividades:

- Se prueban y se deben hacer compatible los distintos programas y/o sub-sistemas.
- Conversión de datos, a costo y tiempo aceptable, y que no interfiera el funcionamiento de la Organización.
- Prueba general del Sistema con datos históricos, y preparados
- Disminuir problemas por factores humanos como : resistencia del personal, temor de no entender o ser desplazado, poca confianza en los datos entregados por el computador

Etapa de Implantación

Esta etapa corresponde a la instalación del sistema en la empresa solicitante del sistema. Esta actividad se puede realizar de diferentes maneras, tales como:

Cambio Directo

Sistema Antiguo Sistema Nuevo

Con este método el cambio desde el sistema antiguo al sistema nuevo ocurre instantáneamente. Es difícil determinar errores menores en el nuevo sistema, porque los usuarios no pueden verificar las salidas del sistema nuevo ni compararlas con las salidas para los mismos datos con el sistema antiguo.

Los errores de mayor magnitud podrían causar que un proceso termine abruptamente, y no es fácil volver al sistema antiguo si el sistema llegara a fallar completamente.

Cambio en Paralelo

Sistema Antiguo

Sistema Nuevo

Con este método tanto el sistema nuevo como el antiguo se encuentran completamente operacionales durante un período de tiempo. La idea es comparar las salidas del nuevo sistema con las del antiguo para verificarlos, y, cuando todas las salidas se encuentran operando correctamente, se detiene el sistema antiguo.

Cambio Por Prototipos / Piloto

Sistema Antiguo Sistema Nuevo

Se pone en marcha el nuevo sistema en una parte de la organización. Durante la operación piloto del nuevo sistema, el sistema antiguo continúa operando en toda la organización.

En cierto modo, la operación piloto es un tipo de operación semi-paralela. Una vez que se ha probado con el éxito en unidad piloto se lleva al resto de la organización. Para decidir en qué parte la organización se llevará a cabo la implementación piloto, ha de considerarse aquellas unidades que pueden ser afectadas negativamente por la falla o error del nuevo sistema.

Cambio Por Fases

Sistema Nuevo F1

Sistema Antiguo

Sistema Nuevo F2

Este método consiste en poner en marcha el nuevo sistema por piezas o módulos. Al poner en marcha un módulo, es posible elegir cualquiera de los métodos señalados anteriormente.

A diferencia de la operación piloto, se entregan módulos a toda la organización y no el sistema completo a una parte de ella. Por consiguiente, los riesgos asociados a errores o fallas se limitan sólo al módulo que se pone en marcha. Si un módulo presenta fallas o errores a una parte de la organización es más fácil corregir el problema. No obstante, este método resulta inapropiado si el sistema no puede ser separado fácilmente en módulos.

Etapa de Mantención

Una vez que el Sistema de Información está en explotación requerirá de ajustes menores de software, en hardware, en códigos, en cantidad de datos, en tiempo respuesta, etc. En caso de cambios mayores se podría llegar al desarrollo de un nuevo Sistema de Información. (Se habrá cumplido el ciclo de vida del Sistema de Información).

Existen diversas mantenciones tale como:

Mantenimiento correctivo: Independientemente de cuán bien diseñado, desarrollado y probado está un sistema o aplicación, ocurrirán errores inevitablemente. Este tipo de mantenimiento se relaciona con la solución o la corrección de problemas del sistema. Atañe generalmente a problemas no identificados durante la fase de ejecución. Un ejemplo de mantenimiento correctivo es la falta de una característica requerida por el usuario, o su funcionamiento defectuoso.

Mantenimiento para fines específicos: Este tipo de mantenimiento se refiere a la creación de características nuevas o a la adaptación de las existentes según lo requieren los cambios en la organización o los usuarios, por ejemplo, los cambios en el código tributario o los reglamentos internos de la organización.

Mantenimiento para mejoras: Se trata de la extensión o el mejoramiento del desempeño del sistema, ya sea mediante el agregado de nuevas características, o el cambio de las existentes.

Mantenimiento preventivo. Este tipo de mantenimiento es probablemente uno de los más eficaces en función de los costos, ya que si se realiza de manera oportuna y adecuada, puede evitar serios problemas en el sistema.

2.6 Metodologías de desarrollo

2.6.1 El Modelo Clásico o de Cascada

Llamado algunas veces «ciclo de vida básico» o «modelo en cascada», el modelo lineal secuencial sugiere un enfoque sistemático, secuencial del desarrollo del software que comienza en un nivel de sistemas y progresa con el análisis, diseño, codificación, pruebas y mantenimiento.

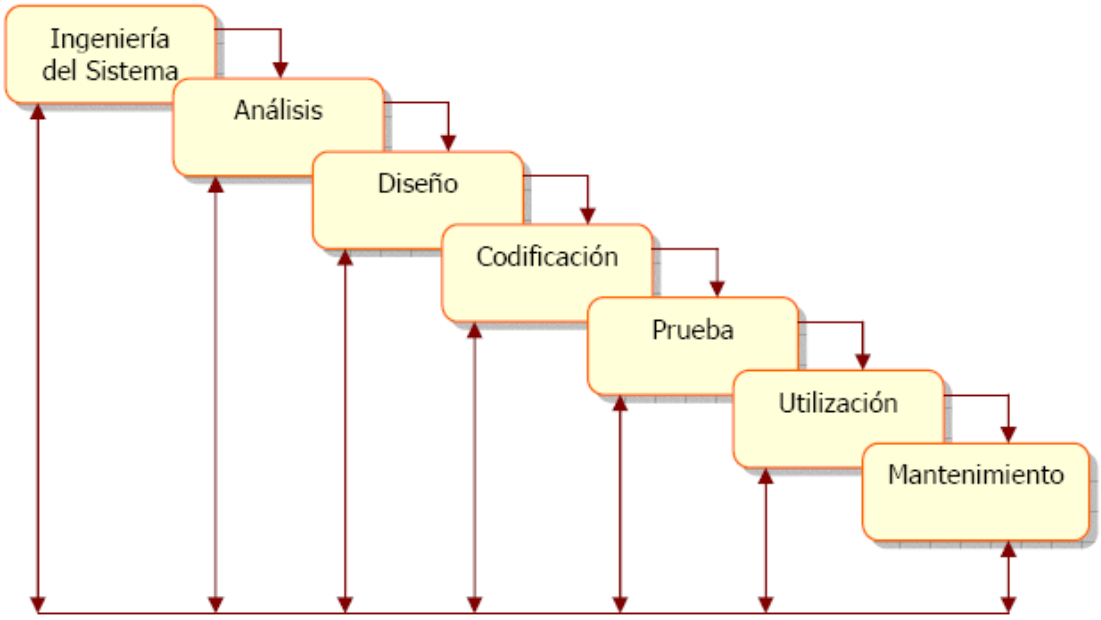


Figura 3: Modelo Clásico

Modelado según el ciclo de ingeniería convencional, el modelo lineal secuencial acompaña a las actividades siguientes:

Análisis de los requerimientos: El proceso de recopilación de los requisitos se centra e intensifica especialmente para el software. Para comprender la naturaleza de los programas que hay que construir, el ingeniero de software ("analista") debe comprender el ámbito de la información de software, así como la función, el rendimiento y las interfaces requeridas. Los requisitos, tanto del sistema como del software, se documentan y se revisan con el cliente.

Diseño: El diseño del software es realmente un proceso multipaso que se enfoca sobre cuatro atributos distintos del programa: la estructura de los datos, la arquitectura del software, el detalle procedimental y la caracterización de la interfaz. El proceso de diseño traduce los requisitos en una representación del software que pueda ser establecida de forma que obtenga la calidad requerida antes de que comience la codificación. Al igual que los requisitos, el diseño se documenta y forma parte de la configuración del software

Codificación: El diseño debe traducirse en una forma legible para la máquina. El paso de codificación realiza esta tarea. Si el diseño se realiza de una manera detallada, la codificación puede realizarse mecánicamente.

Prueba: Una vez que se ha generado el código, comienza la prueba del programa. La prueba se centra en la lógica interna del software, asegurando que todas las sentencias se han probado, y las funciones externas, realizando pruebas que aseguren que la entrada definida produce los resultados que realmente se requieren.

Mantenimiento: El software, indudablemente, sufrirá cambios después de que se entregue al cliente (una posible excepción es el software empotrado). Los cambios ocurrirán debido a que se hayan encontrado errores, a que el software deba adaptarse a cambios del entorno externo (por ejemplo, un cambio solicitado debido a que se tiene un sistema operativo o dispositivo periférico), o debido a que el cliente requiera ampliaciones funcionales o del rendimiento. El mantenimiento del software aplica cada uno de los pasos procedentes del ciclo de vida a un programa existente en vez de a uno nuevo.

Características:

- Resultado de cada fase: uno o más documentos aprobados.
- Una fase comienza cuando la anterior termina.
- En la práctica, las etapas se solapan.
- Iteraciones de coste elevado y reelaboración del trabajo: tendencia a la congelación de partes del desarrollo (especificaciones).
- Se retrasa la localización y corrección de errores.
- Pueden producir sistemas poco útiles para usuarios o mal estructurados.
- Inflexibilidad del modelo: dificultad para responder a cambios en los requerimientos.

El modelo lineal secuencial es el paradigma más antiguo y más extensamente utilizado en la ingeniería del software. Sin embargo, la crítica del paradigma ha puesto en duda su eficacia.

2.6.2 El Modelo de construcción de prototipos

Un cliente a menudo define un conjunto de objetivos generales para el software, pero no identifica los requisitos detallados de entrada, procesamiento, o salida. En otros casos, el responsable del desarrollo del software puede no estar seguro de la eficacia de un algoritmo, de la capacidad de adaptación de un sistema operativo, o de la forma en que debería tomarse la interacción hombre-máquina. En éstas y en otras muchas situaciones, un *paradigma de construcción de prototipos* puede ofrecer el mejor enfoque.

El paradigma de construcción de prototipos comienza con la recolección de requisitos. El desarrollador y el cliente encuentran y definen los objetivos globales para el software, identifican los requisitos conocidos, y las áreas del esquema en donde es obligatoria más definición. Entonces aparece un diseño rápido. El diseño rápido se centra en una representación de esos aspectos del software que serán visibles para el usuario/cliente. (Por ejemplo: enfoques de entrada y formatos de salida). El diseño rápido lleva a la construcción de un prototipo. El prototipo lo evalúa el cliente/usuario y lo utiliza para refinar los requisitos del software a desarrollar. La interacción ocurre cuando el prototipo satisface las necesidades del cliente, a la vez que permite que el desarrollador comprenda mejor lo que se necesita hacer.

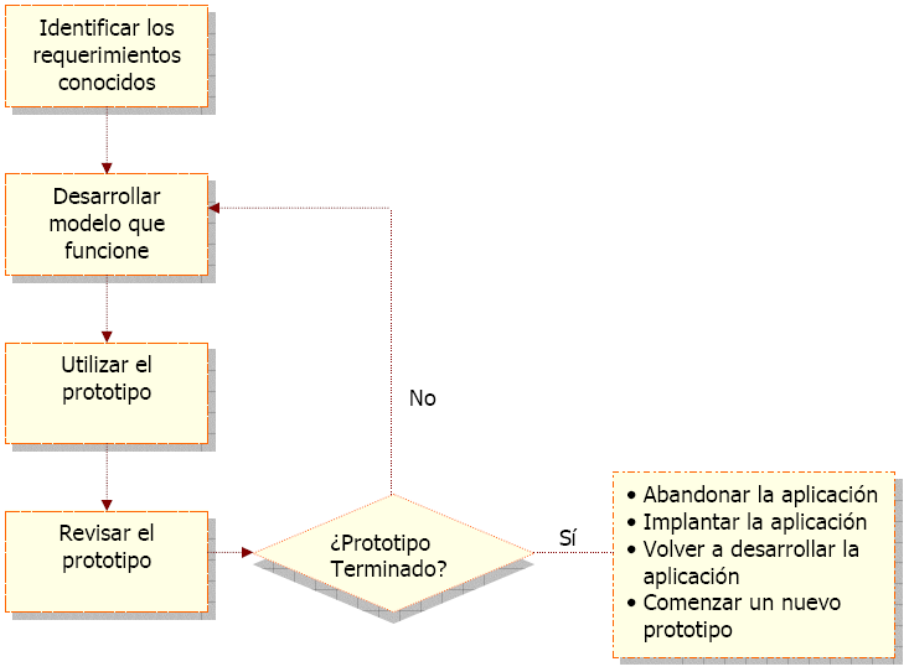


Figura 4: Modelo Prototipos

Lo ideal sería que el prototipo sirviera como un mecanismo para identificar los requisitos del software. Si se construye un prototipo de trabajo, el desarrollador intenta hacer uso de los fragmentos del programa ya existentes o aplica herramientas (p ej.: generadores de informes, gestores de ventanas, etc.) que permiten generar rápidamente programas de trabajo.

En la mayoría de los proyectos el primer sistema construido apenas se puede utilizar. Puede ser demasiado lento, demasiado grande o torpe en su uso o las tres a la vez. No hay alternativa sino comenzar de nuevo y construir una versión rediseñada en la que se resuelvan estos problemas Cuando se utiliza un concepto nuevo de sistema o una tecnología nueva, se tiene que construir un sistema que no sirva y se tenga que tirar porque incluso la mejor planificación no es omnisciente como para que este perfecta la primera vez. Por tanto, la pregunta sobre la gestión no es si construir un sistema piloto y tirarlo. Tendrá que hacerlo. La única pregunta es si planificar de antemano construir un desechable, o prometer entregárselo a los cliente.

2.6.3 El Modelo Incremental

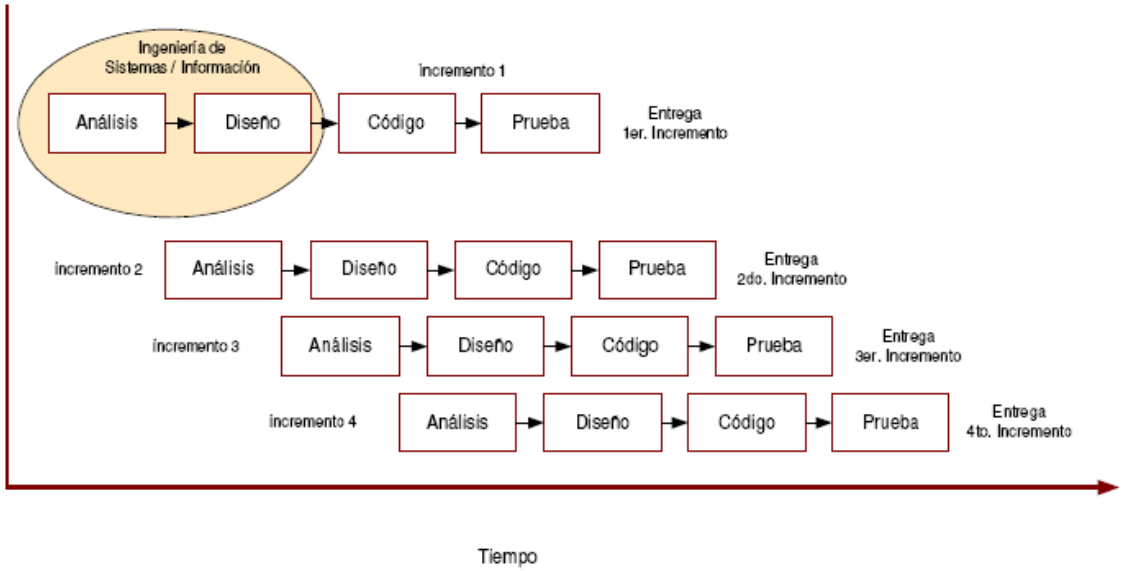


Figura 5: Modelo Incremental

El modelo incremental combina elementos del modelo lineal secuencial (aplicados repetitivamente) con la filosofía interactiva de construcción de prototipos. Como muestra la Figura el modelo incremental aplica secuencias lineales de forma sorprendente de la misma forma que progresa el tiempo en el calendario. Cada secuencia lineal produce un incremento del software. Se debería tener en cuenta que el flujo del proceso de cualquier incremento puede incorporar el paradigma de construcción de prototipos.

Cuando se utiliza un modelo incremental, el primer incremento a menudo es un *producto esencial* (núcleo). Es decir, se afrontan requisitos básicos, pero muchas funciones suplementarias (algunas conocidas, otras no) quedan sin extraer. El cliente utiliza el producto central (o sufre la revisión detallada).

El modelo de proceso incremental, como la construcción de prototipos y otros enfoques evolutivos, es interactivo por naturaleza. Pero a diferencia de la construcción de prototipos, el modelo incremental se centra en la entrega de un producto operacional con cada incremento. Los primeros incrementos son versiones «desmontadas» del producto final, pero proporcionan la capacidad que sirve al usuario y también proporciona una plataforma para la evaluación por parte del usuario.

El desarrollo incremental es particularmente útil cuando la dotación de personal no está disponible para una implementación completa en cuanto a de la fecha límite de gestión que se haya establecido para el proyecto.

2.6.4 El Modelo en Espiral

El *modelo en espiral,* propuesto originalmente por Boehm, es un modelo de proceso de software evolutivo que acompaña la naturaleza interactiva de construcción de prototipos con los aspectos controlados y sistemáticos del modelo lineal secuencial. Se proporciona el potencial para el desarrollo rápido de versiones increméntales del software. En el modelo espiral, el software se desarrolla en una serie de versiones increméntales. 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.

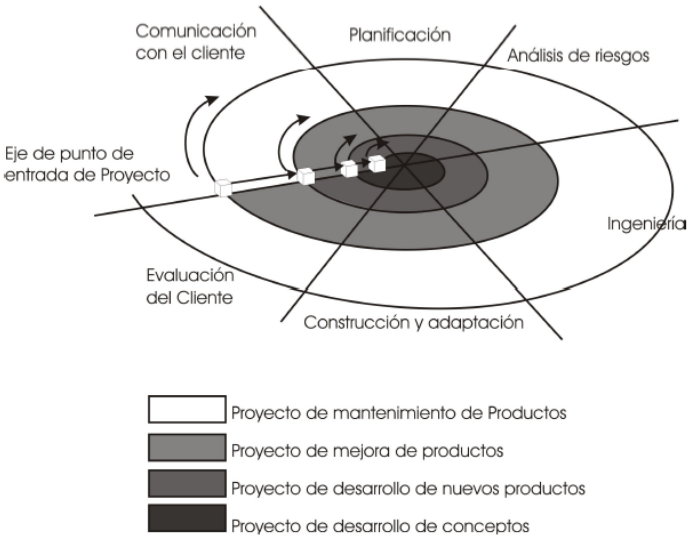


Figura 6: Modelo Espiral

El modelo en espiral se divide en un número de *actividades estructurales,* también llamadas *regiones de tareas* 6, Generalmente, existen entre tres y seis regiones de tareas. La Figura representa un modelo en espiral que contiene seis regiones de tareas:

1. Comunicación con el cliente, las tareas requeridas para establece comunicación entre el desarrollador y el cliente.

2. Planificación, las tareas requeridas para definir recursos, el tiempo y otras informaciones relacionadas con el proyecto.

3. Análisis de riesgos, las tareas requeridas para evaluar riesgos técnicos y de gestión.

4. Ingeniería, las tareas requeridas para construir una o más representaciones de la aplicación.

5. Construcción y adaptación, las tareas requeridas para construir, probar, instalar y proporcionar soporte al usuario (p.ej.: documentación y práctica).

6. 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 están pobladas 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. En todos los casos, se aplican las actividades de protección (p.ej.: gestión de configuración del software y garantía de calidad del software).

2.6.5 Modelo DRA (Desarrollo Rápido de Aplicaciones)

Es un modelo de proceso del desarrollo del software lineal secuencial que enfatiza un ciclo de desarrollo extremadamente corto. Este modelo es una adaptación a alta velocidad del modelo lineal secuencial en el que se logra el desarrollo rápido utilizando una construcción basada en componentes.

Si se comprenden bien los requisitos y se limita el ámbito del proyecto, el proceso DRA permite al equipo de desarrollo crear un «sistema completamente funcional dentro de períodos cortos de tiempo.

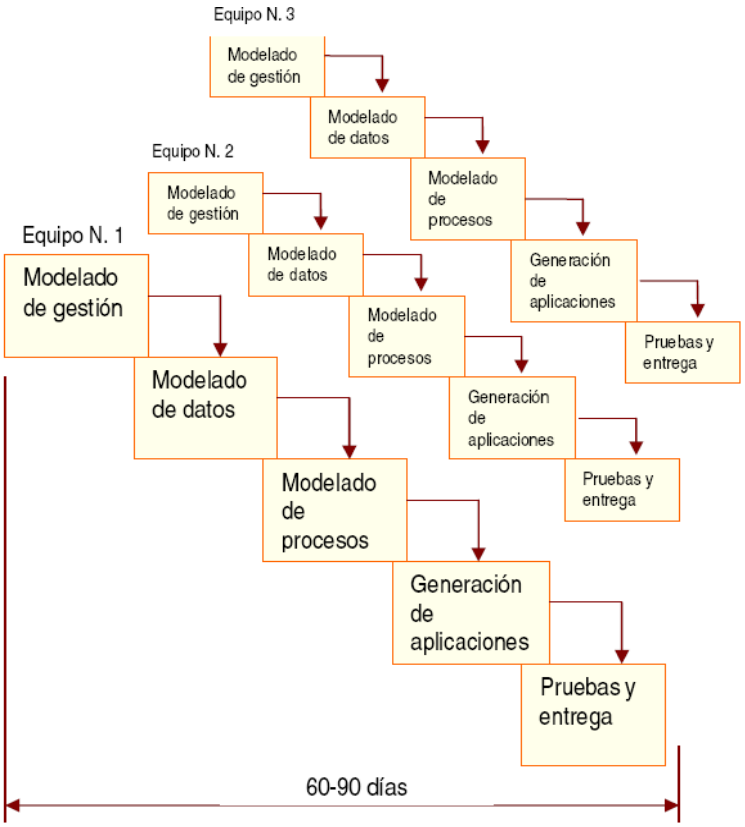


Figura 7: Modelo DRA

El enfoque DRA comprende las siguientes fases[6] :

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 procesa?

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 del 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.

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 para facilitar la construcción del software.

Pruebas y 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.

2.7 Enfoques de Desarrollo

Un enfoque de desarrollo tiene como misión el de especificar claramente todos los detalles asociados al sistema que se desea desarrollar.

Enfoque Estructurado

Este enfoque está orientado a los procesos, tomando una visión donde los datos se consideran en forma separada de los procesos que la transforman, dando una mayor importancia a la descomposición funcional del sistema.

Los elementos que este enfoque usa son:

- Diagrama de Flujo de Datos (DFDs)
- Diccionario de Datos

[6] Ingeniería de Software: Un Enfoque Práctico, Roger Pressman.

- Mini especificaciones (tablas de decisión, Arboles de decisión, Lenguaje estructurado, etc.)

Enfoque Orientación a Objetos

El enfoque orientado a objeto se centra en primer lugar en identificar los objetos del dominio de aplicación y después en establecer procedimientos que los manejen. Aunque esto pueda parecer más indirecto el software orientado a objeto se mantiene mejor ante los cambios de requerimientos porque se basa en la estructura subyacente del dominio de aplicación en vez de los requerimientos funcionales de un determinado problema.

Para producir software que cumpla su propósito hay que obtener los requisitos del sistema, esto se consigue conociendo de una forma disciplinada a los usuarios y haciéndolos participar de manera activa para que no queden “cabos sueltos”. Para conseguir un software de calidad, que sea duradero y fácil de mantener hay que idear una sólida base arquitectónica que sea flexible al cambio. Para desarrollar software rápida y eficientemente, minimizando el trabajo de recodificación y evitando crear miles de líneas de código inútil hay que disponer, además de la gente y las herramientas necesarias, de un enfoque apropiado.

Enfoque Orientado a Aspectos

Es un nuevo paradigma de desarrollo de software que se fundamenta en principios clásicos como la modularización y la separación de intereses o conceptos. Centra su aplicación en el tratamiento de intereses transversales (crosscutting concerns). Surge básicamente como una propuesta de programación que en forma rápida fue extendiéndose en las otras fases del ciclo de desarrollo del software.

3. Enfoque Orientado a Objeto Aplicado a un Sistema de Información

Vivimos en un mundo en el cual clasificamos todo, los productos, los arboles, los negocios, etc. Por lo cual no es tan sorprendente que se plantee una visión orientada a objetos para la creación de un software, es decir, una abstracción que modela el mundo de tal forma que proporciona un apoyo para entenderlo y diseñarlo.

La primera vez que se propuso un enfoque orientado a objetos para el desarrollo de software fue a finales de los años sesenta. Sin embargo, las tecnologías de objetos han necesitado casi veinte años para llegar a ser ampliamente usadas.

A medida que pasa el tiempo, las tecnologías de objetos están sustituyendo a los enfoques clásicos de desarrollo de software.

Las tecnologías de objetos llevan a reutilizar, y la reutilización de componente de software, lleva a un desarrollo de software más rápido y a programas de mejor calidad. El software orientado a objetos es más fácil de mantener debido a que su estructura es inherentemente poco acoplada. Esto lleva a menores efectos colaterales cuando se

deben hacer cambios y provoca menos frustración en los especialistas del área informática y en el cliente. Además, los sistemas orientados a objetos son más fáciles de adaptar y más fácilmente escalables (por ejemplo: pueden crearse grandes sistemas ensamblando subsistemas reutilizables).

Conceptos y Definiciones

La orientación a objetos se basa en conceptos como clase, objeto, herencia, polimorfismo, método, mensaje y encapsulamiento, y muchos más.

Objeto: Un Objeto representa una entidad del mundo real o inventada. Es un concepto, una abstracción o algo que dispone de unos límites bien definidos y tiene una significación para el sistema que se pretende modelar.

Todo lo que un objeto "conoce" está representado en sus atributos (variables y estado actual), y todo lo que puede "realizar" está definido en sus métodos (comportamiento), y en sus "interacciones" con otros objetos a través del intercambio de mensajes (dinámica del ciclo de vida).

Clase: Desde una perspectiva conceptual, una Clase representa un conjunto de Objetos que comparten: Las mismas propiedades (Atributos), el mismo comportamiento (Métodos), las mismas relaciones con otros Objetos (Mensajes) y la misma semántica dentro del sistema.

Método: Es una operación concreta de una determinada clase.

Abstracción: A partir de una abstracción del mundo real, definimos objetos que representan micromódulos de software ideales. Desde su creación, se mantienen de manera independiente unos de otros, sólo interaccionan con otros objetos a través de sus mensajes. Cada objeto configura un universo ordenado y autosuficiente.

Mensaje: Una aplicación orientada a objetos consiste en un número determinado de objetos que interactúan entre sí enviándose mensajes unos a otros para invocar sus métodos. Este intercambio de mensajes facilita su comportamiento, cambios de estado, destrucción o almacenamiento. Ya que todo lo que un objeto puede realizar está expresado en sus métodos, este simple mecanismo de mensajes soporta todas las posibles interacciones entre ellos.

Encapsulamiento: El empaquetado de métodos y atributos dentro de un objeto mediante una interface de mensajes, es lo que denominamos encapsulación. La clave está precisamente en este envoltorio del objeto. La interface que rodea por completo al objeto actúa como punto de contacto para todos los mensajes que llegan desde cualquier objeto emisor.

Polimorfismo: Diferentes objetos pueden responder a un mismo mensaje de diferentes maneras. El polimorfismo permite a los objetos interactuar entre ellos sin necesidad de conocer previamente a que tipo pertenecen.

Herencia: Es un mecanismo mediante el cual se puede crear una nueva clase partiendo de una existente, se dice entonces que la nueva clase hereda las características de la

case existentes aunque se le puede añadir más capacidades (añadiendo datos o capacidades) o modificar las que tiene.

Ventajas de la orientación a objetos

Las ventajas más importantes de la programación orientada a objetos son las siguientes:

Mantenibilidad (facilidad de mantenimiento). Los programas que se diseñan utilizando el concepto de orientación a objetos son más fáciles de leer y comprender y el control de la complejidad del programa se consigue gracias a la ocultación de la información que permite dejar visibles sólo los detalles más relevantes.

Modificabilidad (facilidad para modificar los programas). Se pueden realizar añadidos o supresiones a programas simplemente añadiendo, suprimiendo o modificando objetos.

Reusabilidad. Los objetos, si han sido correctamente diseñados, se pueden usar numerosas veces y en distintos proyectos.

Fiabilidad. Los programas orientados a objetos suelen ser más fiables ya que se basan en el uso de objetos ya definidos que están ampliamente testados.

4 Modelado de procesos de negocio

4.1 Conceptos y Descripciones.

El concepto de proceso de negocios es de naturaleza abstracta de ahí que sea difícil de entregar una definición concreta. Considerando la opinión de los expertos en el tema se pueden establecer los siguientes conceptos asociados a los procesos de negocios:

- "Un conjunto estructurado, medible de actividades diseñadas para producir un resultado especifico para un cliente particular o un mercado". (Davenport T. , 1993)
- Una colección de actividades que toman uno o más tipos de entradas (*inputs*) y crean un resultado (*output*) que es de valor para el cliente". (Hammer & Champy, 1993)
- "Serie de pasos diseñados para producir un producto o servicio". (Rummler & Brache, 1995).

De los conceptos antes mencionados se puede establecer que un proceso de negocio consta de un conjunto de tareas que poseen una relación y sentido entre ellas que tienen como norte cumplir con los resultados esperados de un negocio.

4.2 BPMN: Business Process Management Notation[7]

Definición

BPMN (Business Process Management Notation) o tambien conocida como "Notación para el modelado de negocios ", define diagramas de procesos de negocios basados en las técnicas de los diagramas de flujos adaptados para graficar modelos de operaciones de negocios

Beneficios

- Provee una notación que es entendible por todos los usuarios o participantes del negocio, entre estos se puede mencionar: analista de negocios, desarrolladores técnicos, responsables de gestión etc.
- Notación rápida y sencilla de implementar.
- Considera un único diagrama para la representación de los procesos.
- Crea un puente estandarizado entre el diseño de los procesos de negocio y la implementación de procesos.

Notación

Objetos de flujo

Eventos: Algo que ocurre durante el transcurso de un proceso de negocio.

Inicio

Intermedio

Fin

Actividades: Cualquier trabajo que realiza la compañía, pueden ser atómicas o compuestas.

Pasarelas: para controla el flujo, pueden ser tradicionales (decisiones), bifurcaciones, uniones y acoplamiento de flujos.

[7] Para mayor información revisar Anexo III.

Conectores

Flujos de secuencia: Indica el orden en el cual sera n ejecutadas las actividades del objeto de negocio.

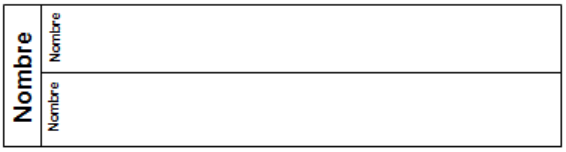
Flujo de mensaje: Se usa para el intercambio de mensaje entre entidades o roles

Asociación: Las asociaciones son usadas para mostrar las entradas y salidas de las actividades.

Calles (swimlanes)

Pool: Indica los participantes en el proceso.

Lane: Un *Lane* es una partición dentro de un *pool* y se extiende a lo largo de todo el pool, tanto vertical como horizontalmente. Los *Lanes* son usados para organizar y categorizar actividades que representan distintas áreas de responsabilidad dentro de un proceso.

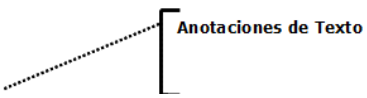


Artefactos o productos.

Datos: Muestra los datos producidos o requeridos por las actividades.

Grupos: Agrupa diferentes elementos del diagrama.

Anotaciones: Agrega información adicional.



Ejemplo de aplicación

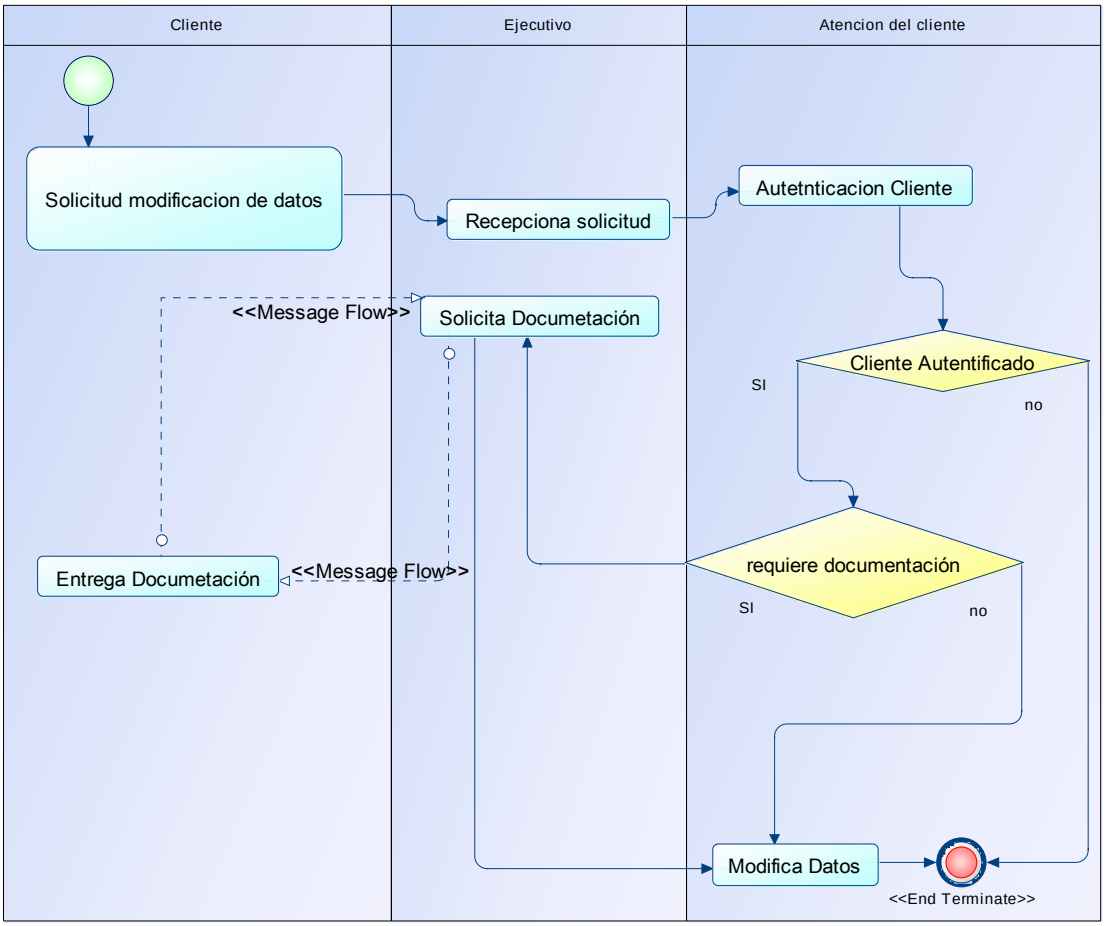


Figura 8: Business Process Management Notation

Proceso unificado de desarrollo (RUP)

Conceptos, elementos y características.

RUP (Racional Unified Process) o Proceso Unificado Racional es una metodología de la Ingeniería en Software. Entrega en forma ordenada tareas y responsabilidades, tiene como meta primordial asegurar la calidad del software.

RUP utiliza un desarrollo iterativo para enfrentar el riesgo, incorpora además el modelamiento visual a través de UML, se guía por la representación de la problemática mediante Casos de Uso.

Divide el sistema en componentes con interfaces bien definidas que posteriormente se ensamblara para formar el sistema. Así se le otorga un mayor tiempo de desarrollo y maduración a cada componente.

Etapas

Dimensión temporal

La dimensión temporal está constituida por: Ciclos, fases, iteraciones e hitos.

Los ciclos trabajan en una nueva generación del desarrollo, cada ciclo del desarrollo se divide en cuatro fases consecutivas. Cada una de las fases concluye con hito muy bien definido.

Fases del RUP

Concepción: Se hace una especificación de la visión del producto final y su caso de negocio definiendo claramente el alcance del negocio. Se identifican los principales casos de uso y los riesgos.

Elaboración: Desarrollo de planificación y recursos necesarios, complementando los casos de usos y eliminando los riesgos.

Construcción: Evolución de la visión, se desarrolla e integran el resto de los componentes. El resultado de esta fase se suele conocer como versión “Beta”.

Transición: Traspaso a los usuarios. Desarrollo de nuevas versiones, corrección de problemas. Entrenamiento y soporte hasta que los usuarios estén satisfechos.

Dimensión estática

La dimensión estática se define en bases a cuatro elementos: Roles, Actividades, productos y flujos.

Roles: Definición del comportamiento y responsabilidades de los participantes, responde a la pregunta ¿Quién?. Una persona puede representar varios roles así como el mismo rol puede ser representado por varias personas. La responsabilidad de un rol tiene asociado un conjunto de actividades y el ser dueño de un conjunto de artefactos (productos).

Actividades: Unidad de trabajo que puede ejecutar un individuo en un rol especifico. Las actividades tienen un objetivo concreto generalmente asociado a la creación o actualización de un producto.

Producto: Conocido también como artefactos son un trozo de información, que es producido usado o modificado por un proceso. Son tangibles y son el resultado de las actividades realizadas por los roles.

Flujo de trabajo: representa una forma de describir la secuencia de actividades que producen resultado y la interacción entre cargos.
Flujos de trabajo.

Los flujos de trabajo de proceso son: Modelado de Negocio, Requisito, Análisis y diseño, Implementación, Test y despliegue.

Los flujos de Trabajo de apoyo son: Administración del Proyecto, Configuración y control de cambio y Entorno.

Modelado de Negocio: Este flujo tiene como objeto llegar a un mejor entendimiento de la organización donde se va a implementar el producto.

Requisito: Con este flujo se estable explícitamente lo QUÉ es lo que exactamente hará el sistema.

Análisis y diseño: El objetivo de este flujo es traducir los requisitos a una especificación que describa como implementar el sistema.

Implementación: Se implementan clases y ficheros en archivos fuentes, binarios, ejecutables y otros. El resultado final de este flujo es un sistema ejecutable. Se deben realizar además los test de unidad.

Test: Evalúa la calidad del producto que se está desarrollando, no asegura la calidad pero si entrega una realimentación a tiempo, para que así las cuestiones de calidad puedan resolverse.

Distribución: Produce con éxito distribuciones del producto y lo distribuye a los usuarios. Se desarrolla con más intensidad en la fase de transición.

Administración del proyecto: Consigue equilibrar; la completación de los objetivos, administración del riesgo y superar restricciones para el desarrollo acorde a los requisitos.

Configuración y control de cambios: mantiene la integridad de los artefactos que se crean y mantiene la información del proceso evolutivo

Entorno: Da soporte al proyecto con las adecuadas, herramientas, procesos y métodos.

Flujo de trabajos de proceso	Fases			
	Concepción	Elaboración	Construcción	Transición
Modelado de Negocio				
Requisito				
Análisis y Diseño				
Implementación				
Test				
Despliegue				
Administración del proyecto				
Configuración y control				
De cambios				
Entorno				
	Iteración Preliminar	Iteración 1	Iteración 2	Iteración n

5 Lenguaje de Modelado Unificado UML

5.1 Conceptos básicos[8]

UML (Lenguaje Unificado de Modelamiento), a nacido creando grandes expectativas que se han cumplido y con creces. UML, ya es un estándar que no solo es utilizado en la industria de software, sino, que para cualquier industria que necesite la construcción de modelos como condición previa el diseño y posterior construcción de prototipos.

Desde la versión conocida como Unified Method versión 0.8, originada por la unión de las metodologías Booch'94 (Grady Booch) y OMT (James Rumbaugh), la cual fue presentada a la comunidad informática en el año 1995. Hasta la unión de su tercer miembro Ivar Jocabson, con su método OOSE (concepto de Use Case) el cual vino a integrar el equipo de los tres amigos. Dado el prestigio que los tres poseían en el mundo de la ingeniería de software ellos proponen un nuevo lenguaje de modelado, conocido como UML.

La primera versión de UML 1.1, fue presentada para su estandarización al OMG (Object Management Group) en 1997 y fue aceptada.

Se bien UML a nacido como un lenguaje, es mucho más que un lenguaje de programación. Además se ha diseñado realizando combinaciones de una variedad de estándares lo que implica que su diseño es completo desde un comienzo.

UML, ayuda al usuario a entender la realidad de la tecnología y la posibilidad de que reflexione antes de invertir y gastar grandes cantidades en proyectos que no estén seguros en su desarrollo, reduciendo costos y tiempo empleado en la construcción de las piezas que constituyen el modelo.

Propiedades de UML:

a) Concurrencia, es un lenguaje distribuido y adecuado a las necesidades de conectividad actuales y futuras.
b) Ampliamente utilizado por la industria desde su adopción por OMG.
c) Reemplaza a decenas de notaciones empleadas con otros lenguajes.
d) Modelo estructuras complejas.
e) Las estructuras más importantes que soportan tienen su fundamento en las tecnologías orientadas a objetos, tales como objetos, clase, componente y nodos.
f) Emplea operaciones abstractas como guía para variaciones futuras, añadiendo variables se es necesario.
g) Comportamiento del sistema: casos de uso, diagramas de secuencias y de colaboración que sirven para evaluar el estado de las máquinas.

UML permite modelar sistemas de información y su objetivo es lograr modelos que, además de describir con cierto grado de formalismo tales sistemas, puedan ser entendidos por los clientes o usuarios de aquello que se modela. Para ellos, es muy

[8] El Lenguaje Unificado De Modelado. Manual de Referencia; James Rumbaugh, Ivar Jocaobson y Grady Booch

importante que el idioma es el que estén las palabras y textos que aparezcan en tales modelos sea el propio de estas personas.

UML, es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar elementos o componentes de un sistema de software. Además captura decisiones y conocimiento sobre los sistemas que se deben de construir. Está pensado para usarse con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios. Incluye conceptos semánticos, notación y principios generales. También se debe de establecer que este compuesto por partes estáticas, dinámicas, de entorno y organizativas.

5.2 Vistas de UML[9]

Una vista es un subconjunto de UML que modelan construcciones que representan un aspecto de un sistema. En un nivel superior, las vistas se pueden dividir en tres áreas: Clasificación estructural, comportamiento dinámico y gestión del modelo.

La clasificación estructural tiene que ver con la descripción de los elementos del sistema y sus relaciones con los otros elementos. Esta clasificación incluye la vista estática, la vista de casos de uso y la vista de implementación.

El comportamiento dinámico describe el comportamiento de un sistema en el tiempo. Esta vista incluye vista de la máquina de estados, la vista de actividades, y la vista de interacción.

La gestión del modelo describe la organización de los propios modelos en unidades jerárquicas. Esta vista cruza las otras vistas y las organiza para el trabajo de desarrollo y control de configuración.

5.2.1 Vista Estática

Esta vista modela los conceptos del dominio de la aplicación, así como los conceptos internos inventados como parte de la implementación de la aplicación. Porque recibe el nombre es estática, puesto que esta no describe el comportamiento del sistema dependiente del tiempo. Los componentes principales de esta vista son las clases y sus relaciones.

5.2.2 Vista de los Casos de uso

Esta vista modela la funcionalidad del sistema, según lo perciben los usuarios externos, conocidos como actores. El caso de uso es una unidad análoga de funcionalidad y el propósito de esta vista es enumerar a los actores, casos de usos y establecer la relación que existe entre ellos.

[9] El Lenguaje Unificado De Modelado. Manual de Referencia; James Rumbaugh, Ivar Jocaobson y Grady Booch

5.2.3 Vista de Interacción

Esta vista describe la secuencia de intercambios de mensajes entre los roles que implementa el comportamiento de un sistema. Un rol es la descripción de un objeto que desempeña un determinado papel dentro de una interacción. Esta visión permite una visión integral del comportamiento de un sistema.

En esta vista se definen dos diagramas que están enfocados a distintos aspectos de una secuencia, diagrama de secuencia y diagrama de colaboración.

5.2.4 Vista de la Maquina de Estados

Una maquina de estados modela las posibles historias de un objeto de una clase. Estos estados están conectados por transiciones. Cada estado refleja un periodo de tiempo, durante la vida del objeto, en el que se satisface ciertas condiciones; cuando ocurre un evento se puede desencadenar una transición que implica que el objeto pase a un nuevo estado. Las maquinas de estados se muestran como diagramas de estados.

5.2.5 Vista de Actividades

Un grafo de actividades es una variante de una maquina de estados, que muestra las actividades de computación implicadas en la ejecución de un cálculo. Un estado de actividad representa una actividad y estos están representados en los diagramas de actividad.

5.2.6 Vista físicas

Las vistas anteriores modelan los conceptos de la aplicación desde un punto de vista lógico. Las vistas físicas modelan la estructura de la implementación de la aplicación, su organización y su despliegue en nodos de ejecución.

Existen dos vista físicas:

Vista de implementación la cual modela los componentes de un sistema, la dependencia entre los componentes, asignación de clases y otros elementos del modelo de componentes. Esta vista es representada en los diagramas de componentes.

Vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos. Entiéndase por nodo un recurso de ejecución. Esta vista permitirá establecer las consecuencias de la distribución y de la asignación de recursos.

5.2.7 Vista de Gestión de Modelo

Vista que modela la organización del modelo. Un modelo abarca un conjunto de paquetes que contienen los elementos del modelo, tales como clases, maquinas de estados y casos de usos. Un modelo es una descripción completa de un sistema y es representado con una clase especial de paquete.

5.3 Diagramas de UML[10]

5.3.1 Diagrama de Caso de Uso

Descripción

La vista de caso de uso captura el comportamiento de un sistema, o subsistema, o de una clase, tal como se muestra a un usuario exterior. Los diagramas de caso de uso se utilizan para mostrar la funcionalidad que el sistema ofrecerá y que usuario se comunicaran con el sistema para realizar esa funcionalidad.

Elementos

Actor, es una idealización de una persona externa, de un proceso, o de una cosa que interactúa con el sistema, un subsistema o una clase. Un actor caracteriza las interacciones que los usuarios externos pueden tener con el sistema. Distintos usuarios pueden estar ligados al mismo actor.

Cada actor participa en uno o más casos de uso. Interactúa con el caso de uso intercambiando mensajes. Un actor puede ser un ser humano, otro sistema informático, o un cierto proceso ejecutable.

El actor se dibuja como una persona con trazos lineales y el nombre debajo de él. Simbología a utilizar:

Cliente

Caso de Uso, es una descripción de la funcionalidad del sistema desde la perspectiva del usuario Su propósito es definir una pieza de comportamiento coherente, sin revelar la estructura interna del sistema.

La dinámica de un caso de uso puede ser especificada por la interacción representada en los diagramas de estado, diagrama de secuencia, diagramas de colaboración o descripciones informales de texto.

A nivel de sistema, los casos de uso son una representación de cómo se muestra el sistema a los usuarios exteriores. Cada caso de uso puede participar en varias relaciones entre sí (otros casos de uso), además de asociarse a un usuario externo.

[10] El Lenguaje Unificado De Modelado. Manual de Referencia; James Rumbaugh, Ivar Jocaobson y Grady Booch

Un caso se representa como un a elipse con su nombre dentro o debajo de ella. Se conecta por líneas con trazo continuo con los actores y línea sesgada con otros casos de uso. Simbología Utilizar:

Nombre Caso de uso

Limite, corresponde a la especificación de que realiza el sistema, puesto que todo lo que está contenido en este límite es lo que el sistema realizara. En si este componente permite establecer claramente las funcionalidades incorporadas en el sistema a desarrollar, identificando claramente el alcance del sistema.

Este límite es representado por un rectángulo en cuyo interior estarán contenido todos los casos de uso que el Sistema posee, el su nivel superior estará especificado el nombre del sistema. Simbología a utilizar:

Nombre del Sistema

Inclusión/uses, la inclusión de de un caso de uso también se conoce como usar un caso de uso o también como la inserción de un comportamiento adicional al caso de uso, base, que se encuentra en otro caso de uso.

Se representa por una línea sesgada que lleva por etiqueta la palabra include o uses, según la herramienta case utilizada. Simbología a utilizar:

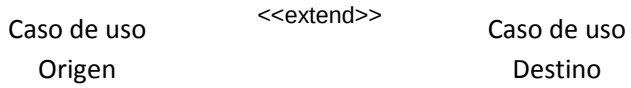
Caso de uso Origen <<Include>> Caso de uso Destino

Ejemplo:

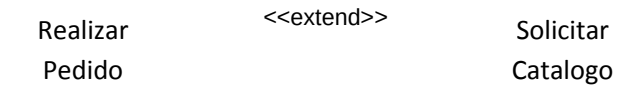
Realizar Pedido <<Include>> Pedir Producto

Extensión/extends, se establece como la inserción de un comportamiento a un caso de uso base del cual el no posee conocimiento. Es decir, es una parte de la funcionalidad del caso de uso que no siempre ocurre. Por ejemplo, en un sistema posee la funcionalidad de consultar los datos, estos serán mostrados por pantalla siempre, pero el usuario tiene como alternativa poder imprimir esta información. Pero este proceso no lo realiza siempre, sino que en algunas ocasiones, no así como el ver los datos entregados por pantalla.

Se representa por una línea sesgada que lleva por etiqueta la palabra extend utilizada. Simbología a utilizar:



Ejemplo:



Ejercicios Propuesto

Sistema de corredora de propiedades.

La corredora de propiedades “Error 404” desea la realización de un sistema informático en plataforma WEB, que permita manejar eficientemente las siguientes tareas:

- ✓ Manejo de inmuebles: En la actualidad se manejan dos tipos de inmuebles, departamentos y casas. Para esta funcionalidad se realizan las siguientes acciones;

    1. Para el ingreso de un nuevo inmueble los clientes se deben inscribir, entregando los siguientes datos: RUN, Nombre, Apellido, número de cuenta corriente, Dirección, teléfono de contacto. Los datos del inmueble a entregar son: para el departamento; Dirección, numero de departamento, población, comuna, número de habitaciones , numero de baños, metros cuadrados, piso, características varias (Logia, Vistas, tipo de cocina, calefacción, estacionamiento etc.). Para las casas; Dirección, número, población, comuna, numero de baños, número de habitaciones, metros

cuadrados construido, metros cuadrados libre, estacionamiento, numero de pisos, características varias (Casa esquina, chimenea, patente comercial asociada etc.), valor del arriendo, comisión a cobrar .

2. Valor del arriendo y comisión a cobrar para ambos tipos de inmuebles.
3. Modificación del valor de arriendos y comisiones. Modificación de datos personales del cliente, mediante solicitud enviada vía aplicación en espera de aprobación de modificación de datos.
4. Estipulación de fechas y formas de pago de arriendo.

- ✓ Manejo de arriendos:

1. El pago de arriendo se realiza hacia la corredora mediante las siguientes formas de pago:
    - o Al contado: Se debe considerar el monto en moneda nacional, realizando dicho pago en la oficina de la corredora de propiedades o vía deposito en la cuenta corriente.
    - o Cheque al día: realizado solo a través de la cuenta corriente.
    - o Transferencia Electrónica: Deposito electrónico a la cuenta corriente de la corredora con aviso mediante correo electrónico automático por parte del banco del depositante.
2. Generación de alertas Automáticas: En el caso de existir no pago de arriendo en un periodo de dos meses se generara el aviso correspondiente a la corredora, y esta se contactara con el dueño del inmueble respectivo.

- ✓ Modulo de consultas:

1. Consultas efectuadas por el dueño del inmueble:
    - o Historial de pagos efectuados, sobre todos sus inmuebles.
    - o Revisión de estado de solicitud de modificación de datos.
2. Consultas efectuadas por el administrador del sistema:
    - o Consultas estados de arriendos.
    - o Consultas por arrendatarios morosos.
    - o Consultas por historial de arriendo por inmueble.
3. Consultas arrendatario: NO realiza Consultas.

- ✓ Modulo Administrador:

1. Realización de modificación de datos.
2. aprobación de solicitud de modificación de datos.
3. Cambio de estado de clientes e inmuebles (Activo o inactivo).
4. Autorización de Deposito electrónico a los dueños de inmueble en estado de arriendo ACTIVO.

Diagrama de Caso de USO 1 (Genérico)

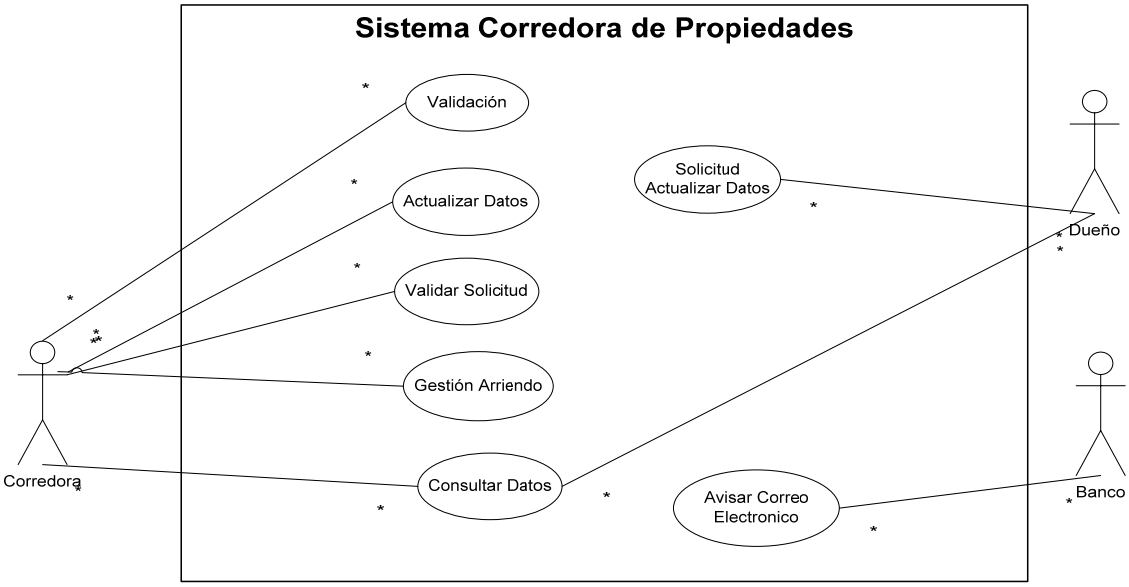
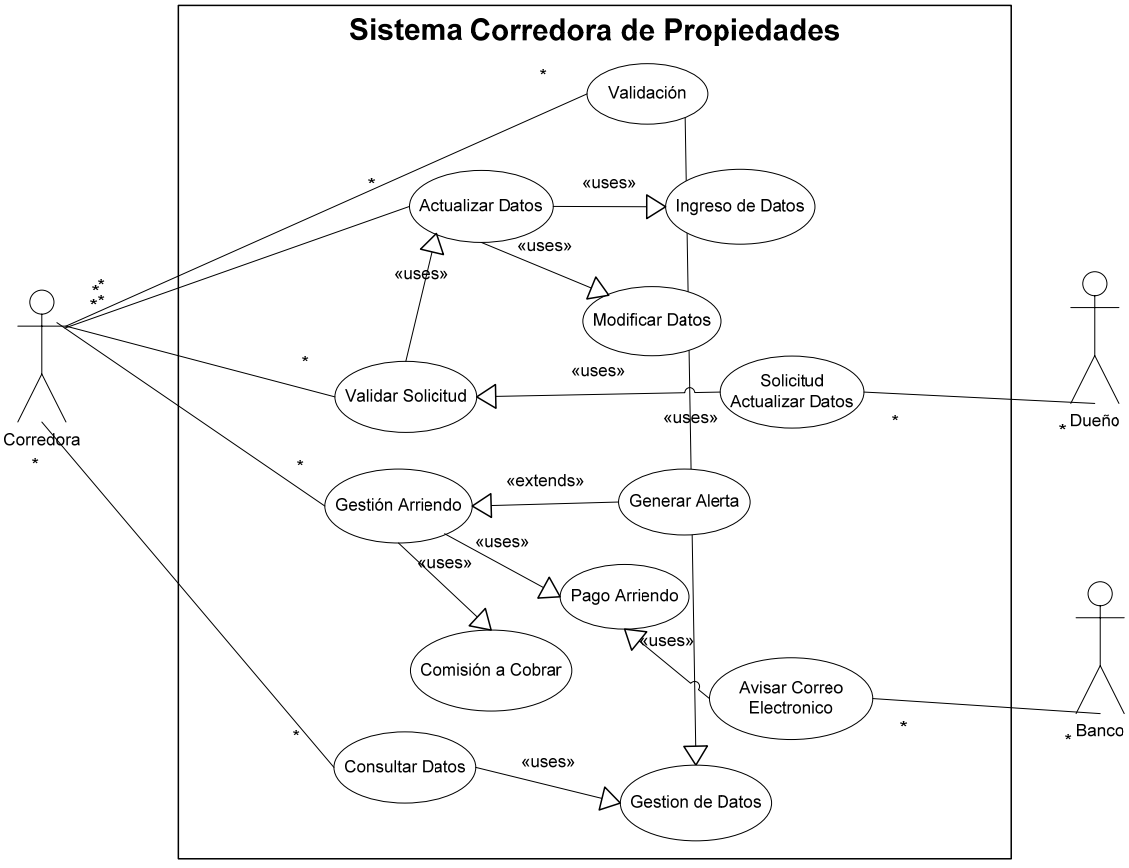


Diagrama de Caso de USO 2 (Detallado)



(Origen: Desarrollo Propio)

5.3.2 Diagrama de Actividades

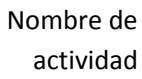
Descripción

Este diagrama es una forma especial de maquinas de estados o diagramas de estados, el cual representa los estados de ejecución de una actividad, no los estados de un objeto que participa en una actividad.

Actividades

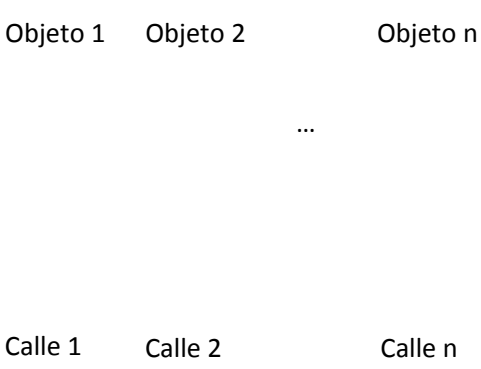
El primer elementos de estos diagramas son los estados de actividad o actividad, la cual presenta la ejecución de una secuencia en un procedimiento, o el funcionamiento de una activad en un flujo de trabajo.

La actividad es representada por una caja con los extremos redondeados. Simbología a utilizar:



Objetos

El objeto es el componente que interactúa en la actividad y que posee su calle particular de actuación, en la parte superior se establece el nombre del objeto identificado. Simbología a utilizar:



Decisiones

El un proceso siempre puede existir la opción de toma de decisión para poder continuar con el proceso. Esta condición estará representada por un rombo. Se debe de establecer que solo debe de llegar una flecha al rombo, del cual pueden salir más de dos flechas dependiendo la actividad representada. Simbología a utilizar:

Sincronización

Se representa por una línea gruesa que permite unir las fechas o hilos concurrentes. Simbología a utilizar:

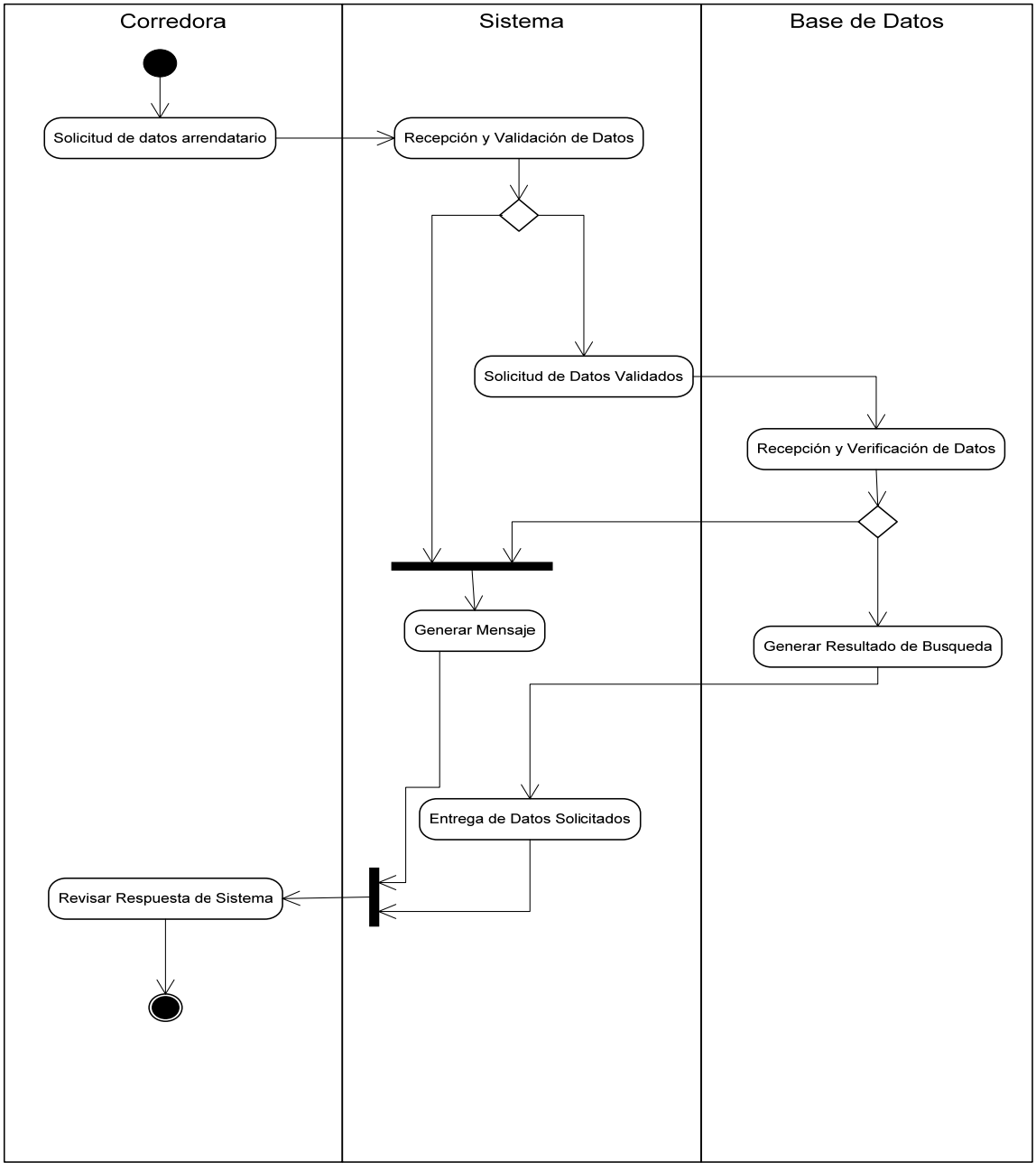
Estado Inicial / Estado Final

El Estado inicial de un diagrama de actividad se representa por medio de la siguiente simbología:

El Estado final de un diagrama de actividad se representa por medio de la siguiente simbología:

Ejercicio

Ejercicio que representa la actividad que realiza la corredora para solicitar información de un arrendatario.



(Origen: Desarrollo Propio)

5.3.3 Diagrama de Clase

Descripción

Cuando miramos a nuestro alrededor e identificamos los elementos que nos rodean, los clasificamos no los individualizamos. Por ejemplo, si pensamos en una plaza y preguntamos qué elementos identificamos solemos decir: arboles, bancos, caminos, personas, etc., no identificamos por ejemplo que hay un raulí.

Pero también debemos de establecer que estas clasificaciones que identificamos están dentro de un todo, por lo cual poseen relaciones he interacciones entre sí.

Un diagrama de clase es la representación de un conjunto de clases participan o forman parte de un sistema, junto con las relaciones que existen entre ellas.

Clase

Es un descriptor de un conjunto de objetos con una estructura, comportamiento, estado y relaciones similares. Es decir, una clase es una descripción de un conjunto de objetos similares. Una clase se representa como un rectángulo divido en tres. Simbología a utilizar:

Nombre de Clase

Atributos

Operaciones o
Métodos

Atributos

Son las características que la clase posee, es decir, estas características son aquellas comunes para todos los objetos que conforman esa clase. Por ejemplo, la clase ventana tienen como atributo: tamaño, posición o material.

Métodos

Son las operaciones que la clase puede realizar o se pueden realizar con ella. Por ejemplo: la clase ventana, se puede abrir(), cerrar() o pintar().

Asociaciones

La asociaciones es un tipo de relación que describe las conexiones con las cuales los objetos de diversas clases pueden interactuar. Simbología a utilizar, será una línea.

Multiplicidad

Establece la cantidad de objetos de una clase que se relaciona con un objeto de otra clase. Se pueden presentar las siguientes multiplicidades:

a.	Uno a uno	1……1
b.	Uno a Muchos	1…….*
c.	Uno a uno o más	1……1..*
d.	Uno a ninguno o uno	1……0,1
e.	Uno a n hasta m	1……n..m
f.	Uno a n	1……n
g.	Uno a n o m	1……n,m

Herencias y generalización

La generalización es una relación entre una descripción más general y una descripción más específica. La descripción más específica tiene todas las propiedades, miembros y relaciones de la descripción más general; y también contiene información adicional. La generalización se dibuja como una flecha con un triangulo hueco desde el hijo (descripción especifica) al padre.

Cada elemento generalizable posee un conjunto de propiedades que se puede heredar. Un hijo hereda todas las propiedades heredables de todos sus antecesores.

Agregación

En algunas ocasiones una clase está compuesta o es parte de otra clase, pero en forma completa. Por ejemplo: un automóvil esta contiene partes, pero no todas sus partes son esenciales para que sea un automóvil. Esta asociación se representa por un diamante hueco.

Composición

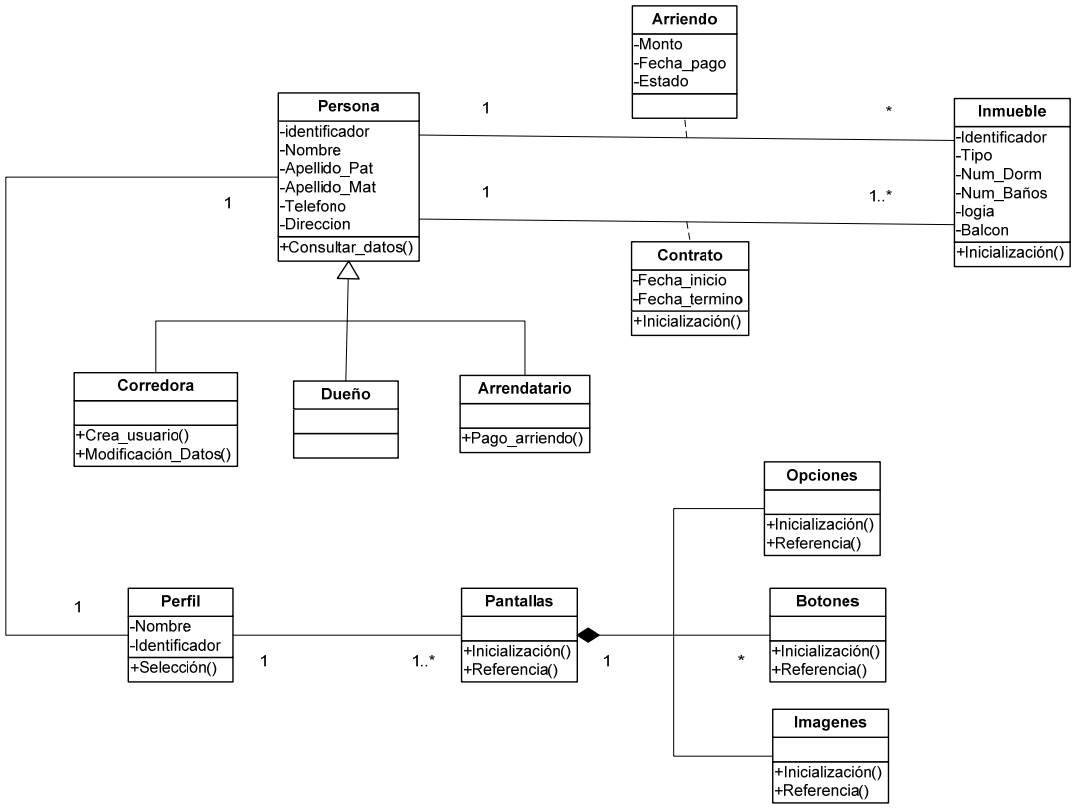
Es una forma más fuerte de asociación, es decir, que una clase es parte esencial en la construcción de una clase. Por ejemplo: un motor está compuesto de partes, si algunas de ellas falta el motor no funciona. Esta asociación se representa por un diamante relleno.

Clase Asociación

Esta clase nace cuando una asociación puede contener atributos y operaciones. Esta asociación se representa con una línea descontinuada que conecta la clase con la asociación de las clases.

Ejercicio Resuelto

Diagrama de clase para el sistema de una corredora de propiedades.



(Origen: Desarrollo Propio)

Ejemplo de creación de clases en JAVA

En el siguiente ejemplo está pensado en la creación de personaje de un juego de ROL. Existe la clase PERSONAJE de la cual heredaran CAMPESINO y HEROE. La clase que implementara ambas clases es IMPLEMENTA. Además existe una interfaz que será implementada por la clase HEROE.

```
public class Campesino extends Personaje {
    public String area_trabajo;
    public double area_sembrado;
    public void siembra(double tiempo)
    {
            area_sembrado = 10 * tiempo;
    }
}
```

```
public class Heroe extends Personaje implements Negociar{

    public String tipo_heroe;
    public String habilidad;
    public int cofre;
    public void aumenta_experiencia(double tiempo)//sobre escirtura de metodos
    {
            experiencia = 10000 + 100 * tiempo;
    }
    public void comprar() //implentación de metodos
            {
                    cofre++;
    }
    public void vender()//implementación de metodos
    {
            cofre--;
    }
}
```

```
public class Personaje {
    public String nombre;
    private double vida;//encapsulación
    public double estamina;
    public double experiencia;

    public Personaje()//constructor
    {
            this.estamina = 100;
    }
    public void aumenta_experiencia(double tiempo) {
            experiencia = 100 + 100 * tiempo;
    }
```

```
    public void aumenta_experiencia(double tiempo,double factor) //sobre carga de
metodos
    {
            experiencia = 100 + 100 * tiempo*factor;
    }
    public double getVida() {//mostra el valor de la experiencia
            return vida;
    }
    public void setVida(double valor) {//asigna el valor a experiencia
            this.vida = valor;

    }
}

interface Negociar {
    void comprar();
    void vender();
    }

import java.io.*;
public class Implementa {

public static void main(String args[]) throws Exception {

    BufferedReader unbufer = new BufferedReader(new
InputStreamReader(System.in));
    double vvida;
    Campesino uncampesino=new Campesino();//creación de objetos
    Heroe unheroe=new Heroe();

    uncampesino.nombre = "alberto";//asignación de atributos
    uncampesino.aumenta_experiencia(100);
    uncampesino.setVida(200);
    uncampesino.estamina=300;
    uncampesino.area_trabajo = "agricultura";
    uncampesino.siembra(50);
    vvida = uncampesino.getVida();
    System.out.println(vvida); //muestra vida

    unheroe.nombre = "Alex";
    unheroe.aumenta_experiencia(200);
    unheroe.setVida(400);
    unheroe.estamina=400;
    unheroe.tipo_heroe="caballero";
    unheroe.habilidad="espada";

}
}
```

(Origen: Desarrollo Propio)

5.3.4 Diagrama de Estados

Descripción

Representa el comportamiento dinámico de los objetos que son parte de un sistema, en un plazo.

Estados

Describe un periodo de tiempo durante la vida de un objeto de una clase. Un estado se muestra como un rectángulo con las esquinas redondeadas.

Transición

La unión de dos o más estado se conoce como transición. En general, una transición tiene un evento que lo activa (evento disparador), una condición de guarda (expresión booleana), una acción y un estado destino.

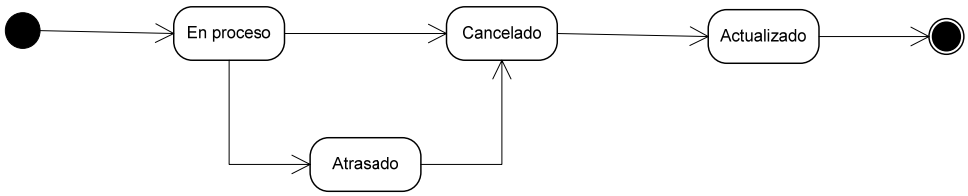
Estado Inicial / Estado Final

El Estado inicial de un diagrama de actividad se representa por medio de la siguiente simbología:

El Estado final de un diagrama de actividad se representa por medio de la siguiente simbología:

Ejercicio Resuelto

Se tomara el objeto arriendo para poder establecer sus estados dentro del sistema.



(Origen: Desarrollo Propio)

5.3.5 Diagrama de Secuencias

Descripción

El diagrama de estado está enfocado a los diferentes estados de un objeto, pero establece solo una pequeña parte del todo. Es así como este diagrama permite poder visualizar la interacción de los objetos entre sí en un periodo de tiempo.

El diagrama de secuencia presenta dos dimensiones: la vertical es el eje de tiempo, y la horizontal muestra a los distintos objetos que interactúan. El tiempo avanza desde arriba hacia abajo.

Horizontal

Vertical

Objetos

Es una entidad discreta con identidad, estado y un comportamiento invocable. Los objetos son piezas individuales.

Activación

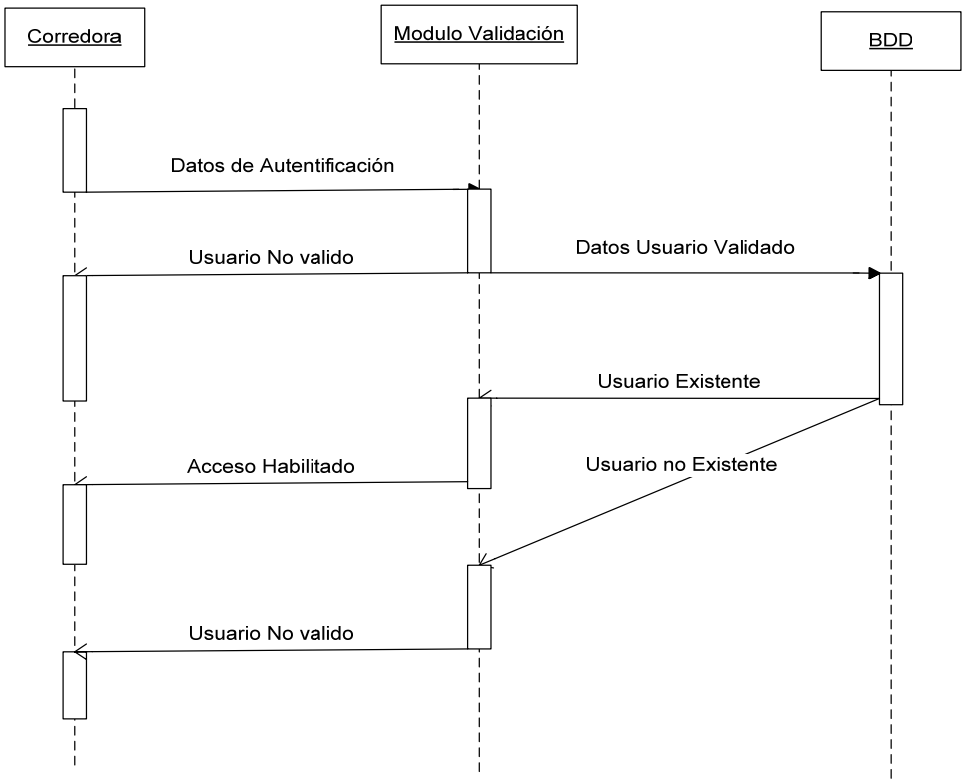
Es la ejecución de un procedimiento, incluyendo en el tiempo que espera a los procedimientos anidados para ejecutarse. Este se define como un rectángulo en la línea de tiempo que posee el diagrama.

Mensajes

Presenta la interacción que los objetos tienen en un diagrama de secuencia. Existe tres tipos de mensajes: El mensaje síncrono, es el que representa la comunicación en donde un objeto envía el mensaje y este espera una respuesta ( ); mensaje asíncrono, solo representa un flujo de control que no se espera de una respuesta ( ); mensaje simple, se utilizan mayormente como un retorno a un mensaje síncrono ( ).

Ejercicio Resuelto

Diagrama que presenta la secuencia del proceso de validación de un usuario.



(Origen: Desarrollo Propio)

5.3.6 Diagrama de Colaboraciones

Descripción

Establece como las instancias específicas de las clases trabajan juntas para lograr un objetivo en común. En cierta forma, un diagrama de colaboración se detalla las asociaciones que se muestran en un diagrama de clases, describiendo el intercambio de mensajes entre objetos y las relaciones entre los objetos.

Objeto

Un objeto se representa por un rectángulo que contiene el nombre del objeto.

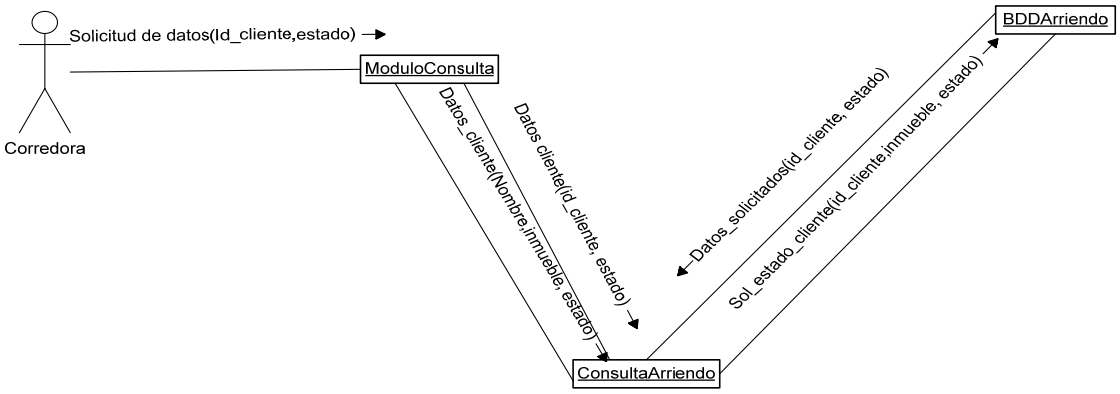
: Nombre objeto

Mensajes

Los mensajes se muestran como una flecha etiquetados. Cada mensaje tiene un número de secuencia, una lista opcional de mensajes precedentes, una condición opcional de guarda, un nombre, lista de argumentos y un nombre de valor de retorno opcional.

Ejercicio propuesto

Se establece el diagrama de colaboración que permite responder a la consulta de datos de un arrendatario por parte de la corredora.



(Origen: Desarrollo Propio)

6 Patrones de Diseño

Según **Christopher Alexander** Master en matemáticas y licenciado en arquitectura al hablar de un patrón decía: “Cada patrón define un problema que ocurre una y otra vez en nuestro entorno, así como la solución de esos problemas, de modo que se puede aplicar un solución un millón de veces, sin hacer lo mismo dos veces ”. Aunque el hacía referencia a patrones de ciudades y edificios esta afirmación es igual de válida para los patrones de diseño orientados a objetos.

Los patrones de diseño poseen cuatro elementos esenciales:

- El nombre del patrón: permite describir en una o más palabras un problema de diseño junto con su solución y consecuencias
- El problema: describe cuando aplicar el patrón y describe el problema en su contexto.
- La solución: describe los elementos que constituyen el diseño, sus responsabilidades, sus relaciones y colaboraciones. El patrón proporciona una solución abstracta del problema y como lo resuelve una disposición general de elementos.
- Las consecuencias: son las ventajas e inconvenientes de aplicar el patrón.

Una definición más técnica de patones de diseño: “Descripciones de clases y objetos relacionados que están particularizados para resolver un problema de diseño general en un determinado contexto”.

Un patrón de diseño nomina, abstrae e identifica los aspectos claves de una estructura de diseño común, lo que los hace útiles para crear un diseño orientado a objeto. Cada patrón de diseño se centra en un problema especifico además proporciona código de ejemplo para ayudar a la implementación del diseño.

Patrones de Creación.

Los patrones de creación cumplen con el objetivo de crear una abstracción en el proceso de creación de los objetos.

Un patrón de creación de clases se basa en el paradigma de la herencia para cambiar la clase de la instancia a crear, en cambio, un patrón de creación de objeto deja esta responsabilidad a otro objeto la creación de esta instancia. Esto permite que un sistema sea independiente de “como” se crean, componen y se representan sus objetos.

La importancia de estos patrones se ve resaltada a medida que los sistemas van evolucionando, haciendo que los sistemas dependan más de la composición de los objetos que la herencia clases.

Características de estos patrones son que encapsula el conocimiento de las clases concretas del sistema y además ocultan como se crean y asocian las instancias de dichas clases. Esto se logra haciendo que el sistema solo conozca sus interfaces tal

como las defines sus propias clases abstractas, por lo tanto se crea un alto grado de flexibilidad en que los que se crea, quien lo crea y no menos importante cuando. Esta configuración puede ser estática (en tiempo de configuración) o dinámica (en tiempo de ejecución).

Para estos patrones se han definido los siguientes:

Abstract Factory. Proporciona una interfaz para crear familias de objetos o que dependen entre sí, sin especificar sus clases concretas.
Builder. Separa la construcción de un objeto complejo de su representación, de forma que el mismo proceso de construcción pueda crear diferentes representaciones.
Factory Method. Define una interfaz para crear un objeto, pero deja que sean las subclases quienes decidan qué clase instanciar. Permite que una clase delegue en sus subclases la creación de objetos.
Prototype. Especifica los tipos de objetos a crear por medio de una instancia prototípica, y crear nuevos objetos copiando este prototipo.
Singleton. Garantiza que una clase sólo tenga una instancia, y proporciona un punto de acceso global a ella.

Patrones Estructurales.

Los patrones estructurales se ocupan de cómo se combinan las clases y los objetos para formas estructuras más grandes. Los patrones estructurales de clase hacen uso de la herencia para componer interfaces o implementaciones en cambio los patrones estructurales de objetos definen formas de componer objetos para obtener nueva funcionalidad. La flexibilidad añadida a la composición de objetos viene dada por la capacidad de cambiar la composición de objetos en tiempo de ejecución, lo que es imposible con la composición estática.

Para estos patrones se han definido los siguientes:

Adapter. Convierte la interfaz de una clase en otra distinta que es la que esperan los clientes. Permiten que cooperen clases que de otra manera no podrían por tener interfaces incompatibles.
Bridge. Desvincula una abstracción de su implementación, de manera que ambas puedan variar de forma independiente.
Composite. Combina objetos en estructuras de árbol para representar jerarquías de parte-todo. Permite que los clientes traten de manera uniforme a los objetos individuales y a los compuestos.
Decorator. Añade dinámicamente nuevas responsabilidades a un objeto, proporcionando una alternativa flexible a la herencia para extender la funcionalidad.
Facade. Proporciona una interfaz unificada para un conjunto de interfaces de un subsistema. Define una interfaz de alto nivel que hace que el subsistema sean más fácil de usar.
Flyweight. Usa el compartimiento para permitir un gran número de objetos de grano fino de forma eficiente.
Proxy. Proporciona un sustituto o representante de otro objeto para controlar el acceso a éste.

Patrones de Comportamiento.

Los patrones de comportamiento tienen que ver con la asignación de responsabilidades a los objetos y los algoritmos. Pero estos patrones no solo se limitan a describir patrones de clases y objetos sino también patrones de comunicación entre ellos.

Este patrón permite concentrarse en el modo que se interconectan los objetos. Los patrones de comportamiento basado en clases usan la herencia. Ejemplo de esto es un método plantilla que es una definición abstracta de un algoritmo, que define al algoritmo paso a paso y cada paso invoca a una operación abstracta generalmente. Una subclase completa el algoritmo definiendo las operaciones abstractas.

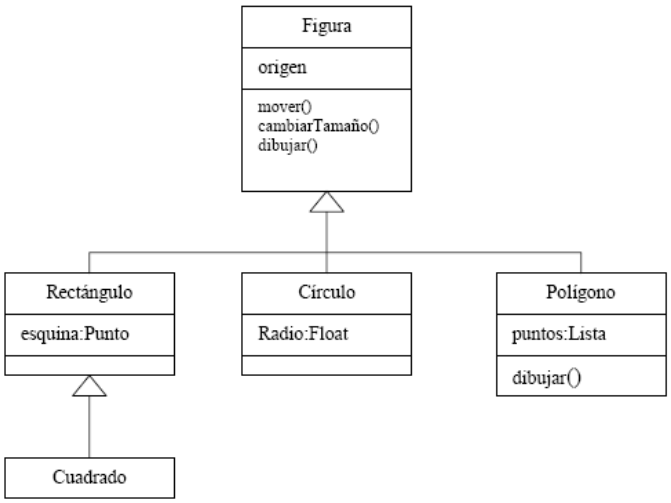
Los patrones de comportamiento basados en objetos usan la composición de los objetos en vez de la herencia. Alguno de ellos describen como cooperan un grupo de objetos realizando una tarea que por sí solos no podrían realizar. Para esto existe un patrón mediador que introduce un objeto mediador el cual proporciona un bajo acoplamiento entre los objetos.

Para estos patrones se han definido los siguientes:

Chain of Responsibility. Evita acoplar el emisor de una petición a su receptor, al dar a más de un objeto la posibilidad de responder a la petición. Crea una cadena con los objetos receptores y pasa la petición a través de la cadena hasta que esta sea tratada por algún objeto.
Command. Encapsula una petición en un objeto, permitiendo así parametrizar a los clientes con distintas peticiones, encolar o llevar un registro de las peticiones y poder deshacer la operaciones.
Interpreter. Dado un lenguaje, define una representación de su gramática junto con un intérprete que usa dicha representación para interpretar las sentencias del lenguaje.
Iterator. Proporciona un modo de acceder secuencialmente a los elementos de un objeto agregado sin exponer su representación interna.
Mediator. Define un objeto que encapsula cómo interactúan un conjunto de objetos. Promueve un bajo acoplamiento al evitar que los objetos se refieran unos a otros explícitamente, y permite variar la interacción entre ellos de forma independiente.
Memento. Representa y externaliza el estado interno de un objeto sin violar la encapsulación, de forma que éste puede volver a dicho estado más tarde.
Observer. Define una dependencia de uno-a-muchos entre objetos, de forma que cuando un objeto cambia de estado se notifica y actualizan automáticamente todos los objetos.
State. Permite que un objeto modifique su comportamiento cada vez que cambia su estado interno. Parecerá que cambia la clase del objeto.
Strategy. Define una familia de algoritmos, encapsula uno de ellos y los hace intercambiables. Permite que un algoritmo varíe independientemente de los clientes que lo usan.
Template Method. Define en una operación el esqueleto de un algoritmo, delegando en las subclases algunos de sus pasos. Permite que las subclases redefinan ciertos pasos del algoritmo sin cambiar su estructura.
Visitor. Representa una operación sobre los elementos de una estructura de objetos. Permite definir una nueva operación sin cambiar las clases de los elementos sobre los que opera.

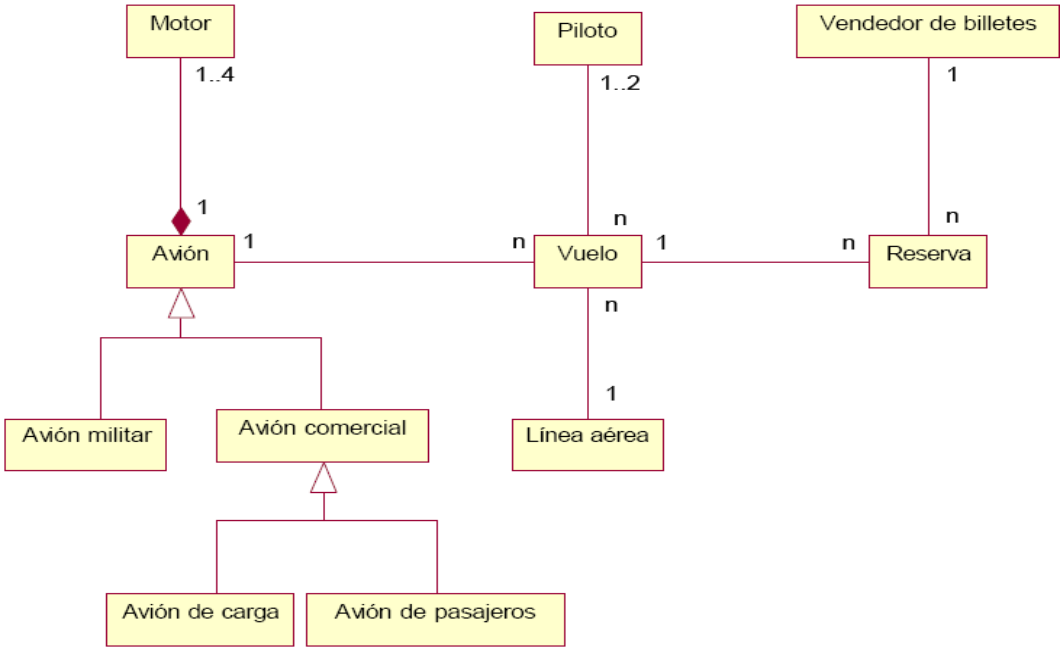
7 Ejemplos varios

1. Ejemplo de Herencia en un diagrama de clases.



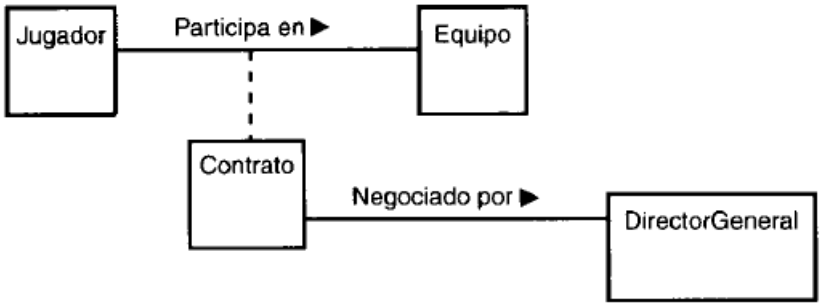
(Origen: Extraído de documento de internet)

2. Ejemplo diagrama de clase de una Aerolínea



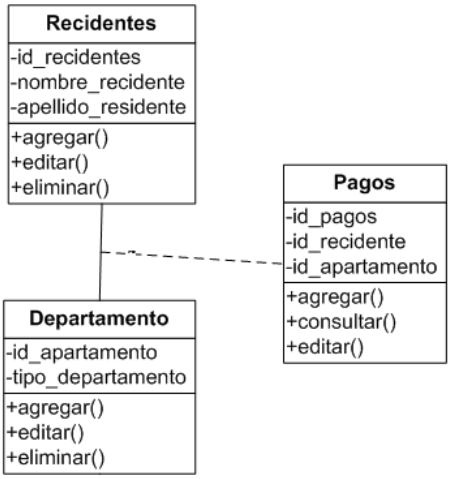
(Origen: Extraído de documento de internet)

3. Ejemplo Clase Asociación



Clase Asociación

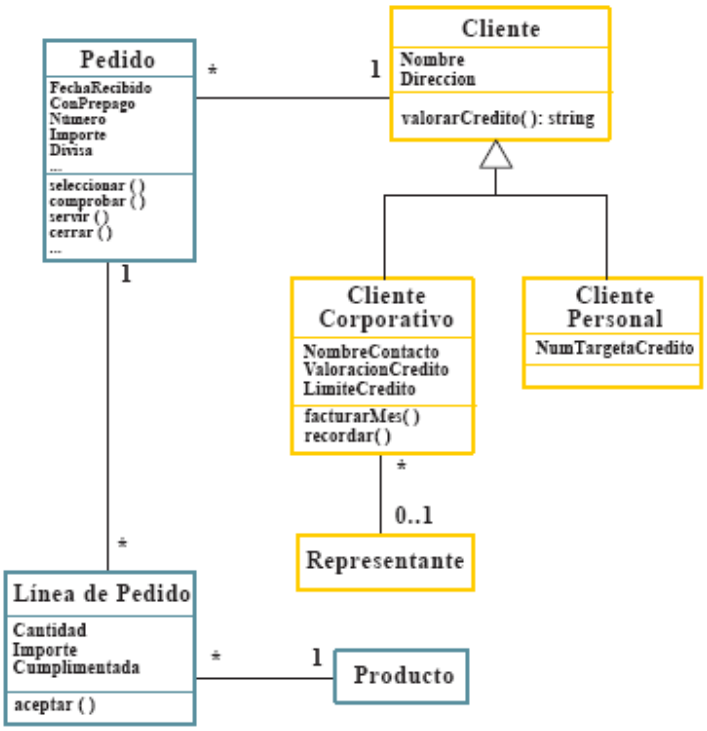
(Origen: Extraído de Libro Aprendiendo UML en 24 Horas)



Clase Asociación

(Origen: Propio)

4. Diagrama de Clases



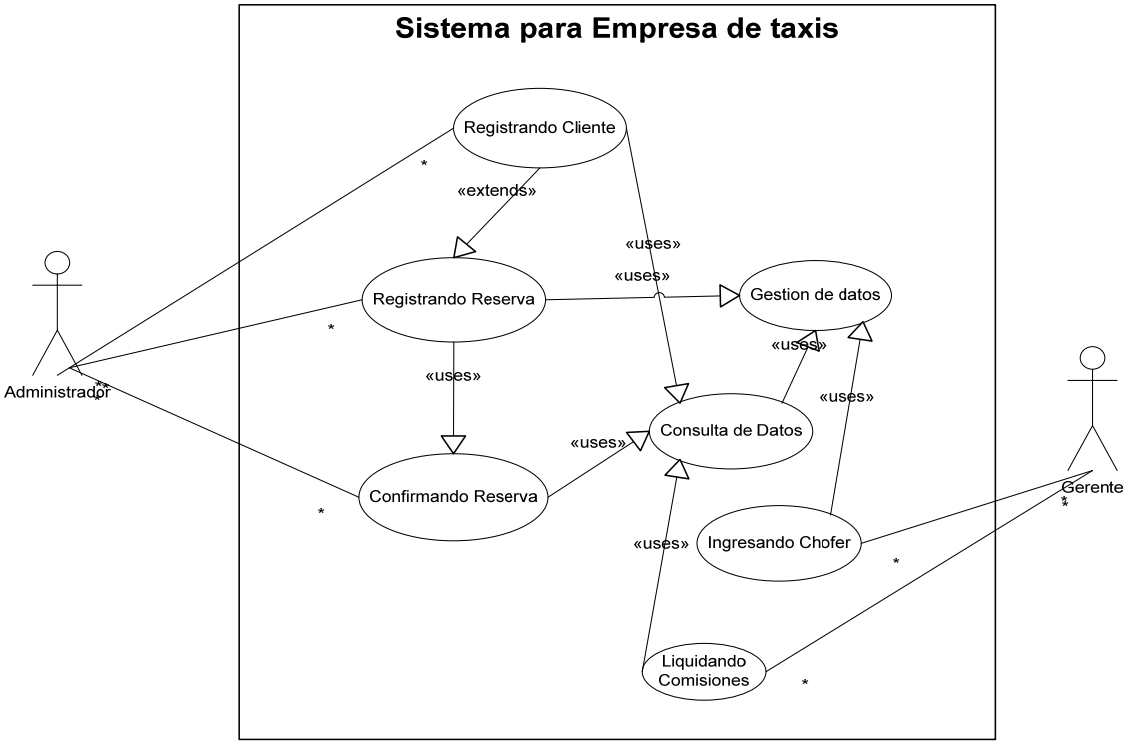
(Origen: Extraído de documento de internet)

5. Diagrama de caso de uso.

Enunciado:

La empresa de taxis nos ha solicitado la confección de un sistema para la administración de la misma. Se puede establecer que solo hay dos tipos de usuarios: Administrativos y el Gerente.

Los administrativos poseen la misión de: Ingresar nuevos clientes, ingresar reservas de viajes indicando el cliente, el chofer solicitado, la dirección de origen, de destino y la hora de salida. La empresa ha solicitado que si al ingresar una reserva, el cliente en cuestión no existe en el sistema se pueda ingresarlo directamente. También ha solicitado que el sistema brinde la opción de confirmar inmediatamente la reserva que se está ingresando. El gerente podrá realizar todas las operaciones que pueden realizar los Administrativos. Además podrán Ingresar nuevos choferes al sistema y Liquidar las comisiones de los choferes mensualmente.



(Origen: Desarrollo Propio)

6. Ejemplo de diccionario de datos

6.1.- Si tomamos el Diagrama de Caso de Uso Anterior se puede tener la siguiente descripción:

Caso de uso: Registrando Reserva	
Actor: Administrativo	
Usa: Gestión de datos, confirmación de reserva.	
Extiende: Registrando cliente	
Curso Normal	Alternativas
1.- El administrativo ingresa el numero de cliente	1.1.- Si el cliente no existe, el administrativo registrara al cliente en la base de datos.
2.- El sistema muestra la información del cliente.	
3.- El administrativo ingresa la fecha, origen, destino y hora de la reserva. Verificando los datos ingresados	
4.- El administrativo confirma la reserva sí.	4.1 si él los datos ingresados no poseen

	disponibilidad se vuelve al paso 5.
5.- Si el administrador quiere ingresar una nueva reserva debe de volver al paso 3, sino al paso 6.	
6.- Terminar la operación realizada.	

(Origen: Formato extraído apuntes de internet)

6.2.- Otra forma de Documentar.

Actualizar el catálogo de productos

1. Descripción
El actor que inicia este use case es el «técnico de mercadeo». Su objetivo en este use case es actualizar el catálogo de Videos y CDs -añadir, eliminar, modificar- que contiene la información de descripción, características, proveedor, precio, etc.

2. Flujo de Eventos
2.1 Flujo Básico

a) El sistema presenta las opciones: Crear, Modificar o Desactivar registro del catálogo de productos, también presenta la opción de Salir.
b) Si el usuario selecciona la opción Salir, el sistema termina la transacción.
c) Si el usuario selecciona la acción de Crear ejecuta la acción correspondiente.
d) Si el usuario selecciona la acción de Modificar o Desactivar:
d.1 - El sistema presenta la lista de los productos registrados en la base de datos.
d.2 - Si el usuario selecciona un registro de la lista que el sistema le presenta.
d.3 - El usuario puede modificar los datos del registro seleccionado y elegir la acción de Modificar o simplemente puede seleccionar la acción de Desactivar.
d.4 - Si el usuario selecciona la acción de Modificar, el sistema ejecuta la acción correspondiente.
d.5 - Si el usuario selecciona la acción de Desactivar, el sistema ejecuta la acción correspondiente.
d.6 - Regresa al paso 4.1 para seleccionar otro ítem.

2.2 Flujos Alternativos

a) De acuerdo con la selección del usuario, se ejecutarán las acciones de SALIR, CREAR, MODIFICAR o DESACTIVAR.

3. Precondiciones

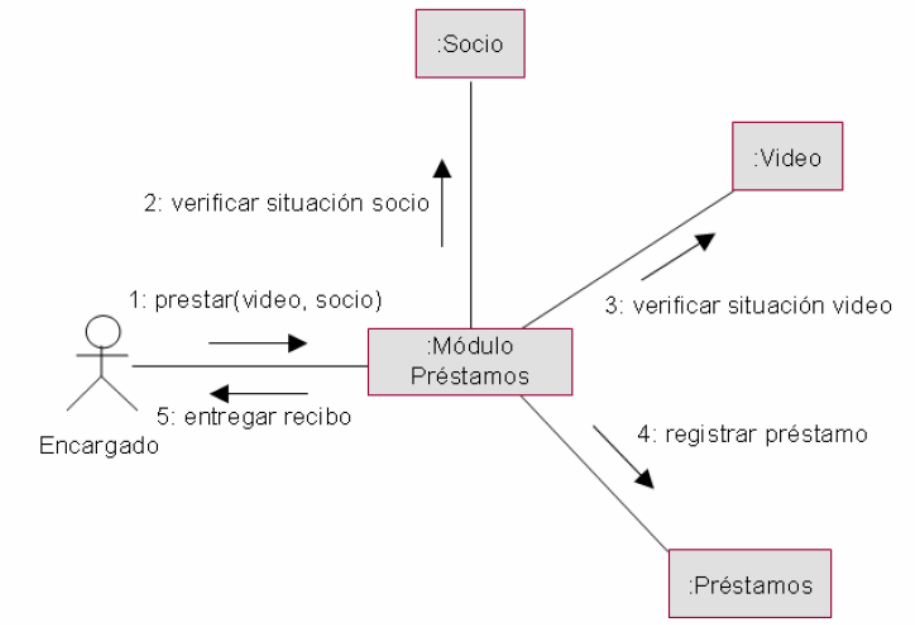
3.1 En el caso de un nuevo producto la carátula correspondiente habrá sido escaneada previamente y se almacenará en el directorio de carátulas con el mismo código asignado al producto.

4. Poscondiciones

4.1 El catálogo de productos ha quedado actualizado.

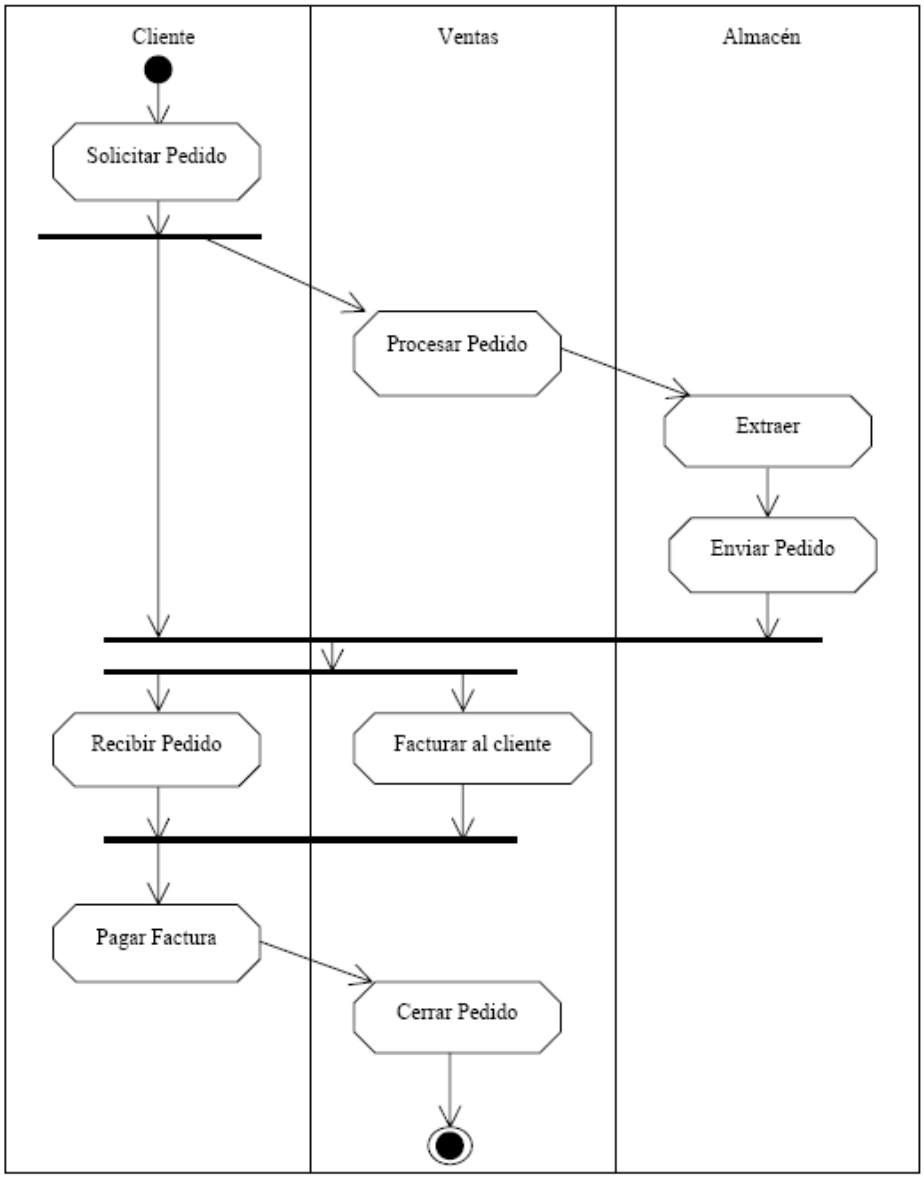
(Origen: Extraído apuntes de internet)

7. Ejemplo de Diagrama de Colaboración



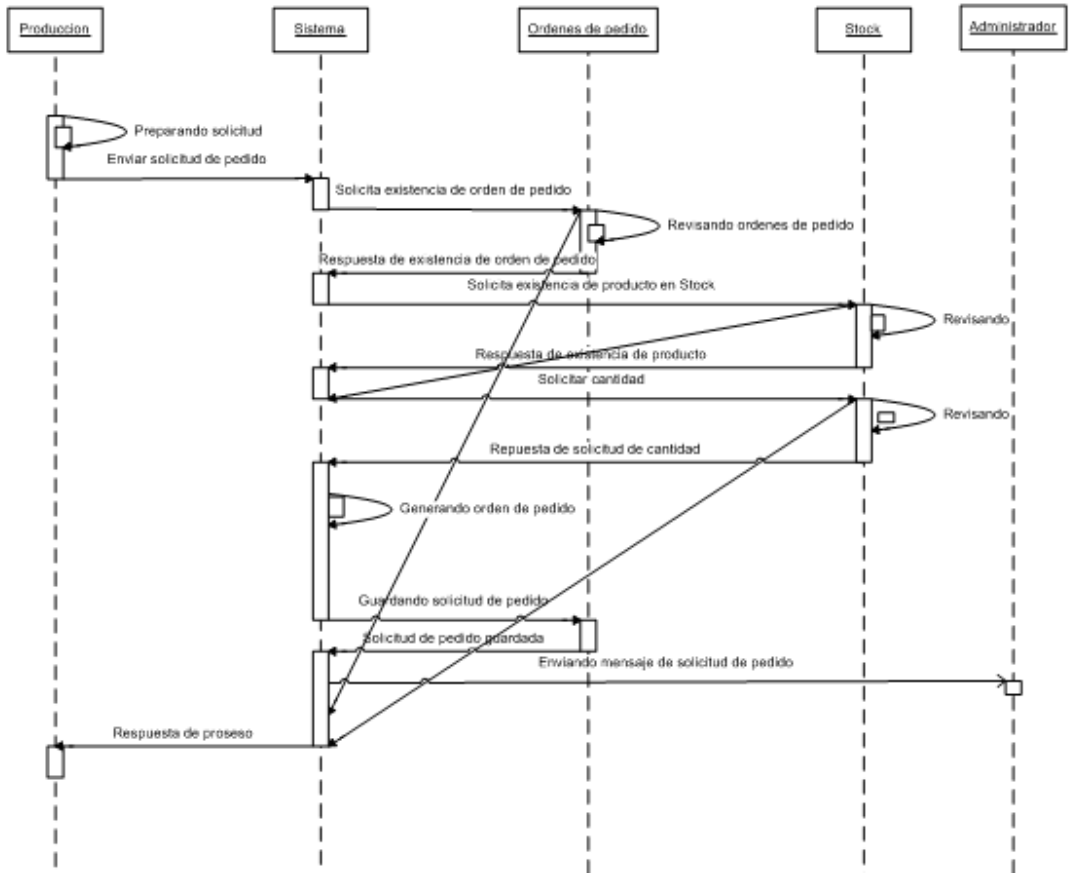
(Origen: Extraído apuntes de internet)

8. Ejemplo de Diagrama de Actividad



(Origen: Extraído apuntes de internet)

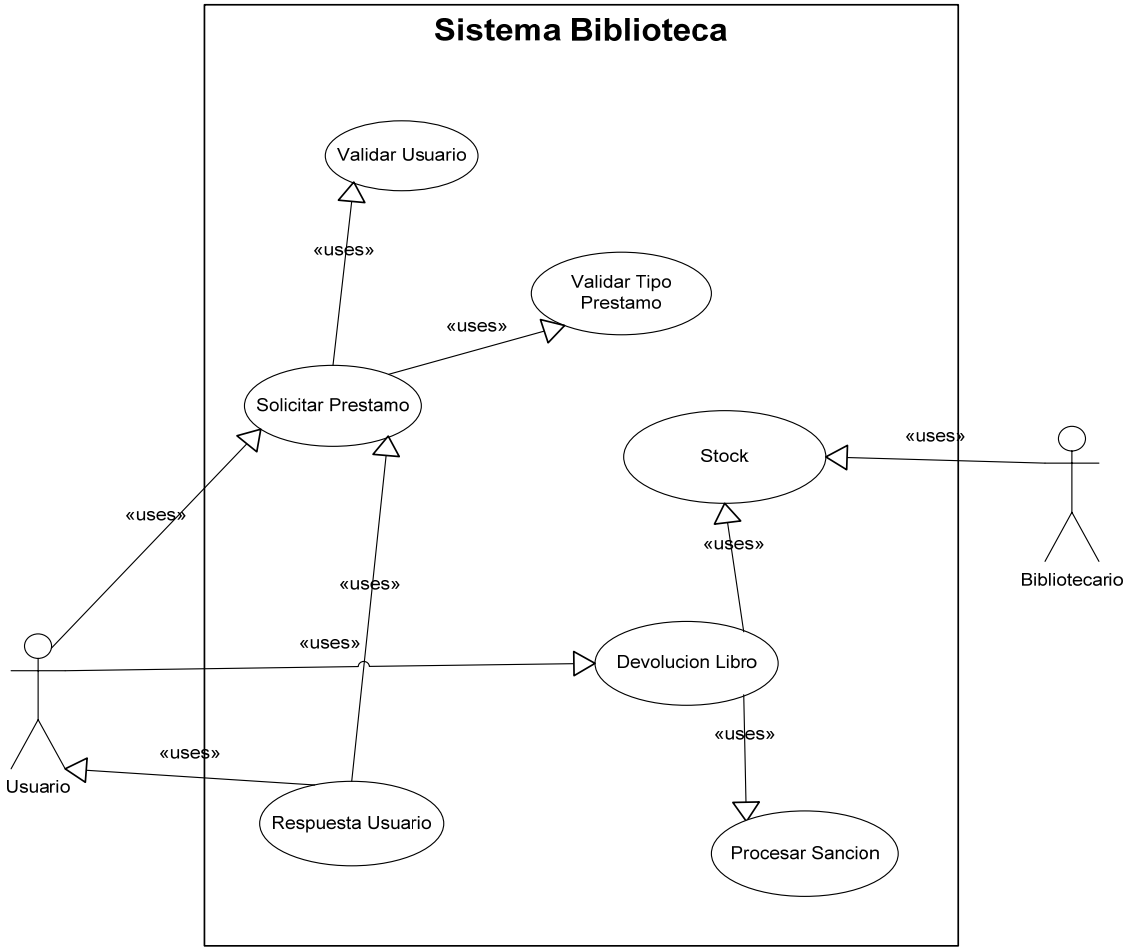
9. Ejemplo de diagrama de Secuencia



(Origen: Extraído Trabajo Alumno)

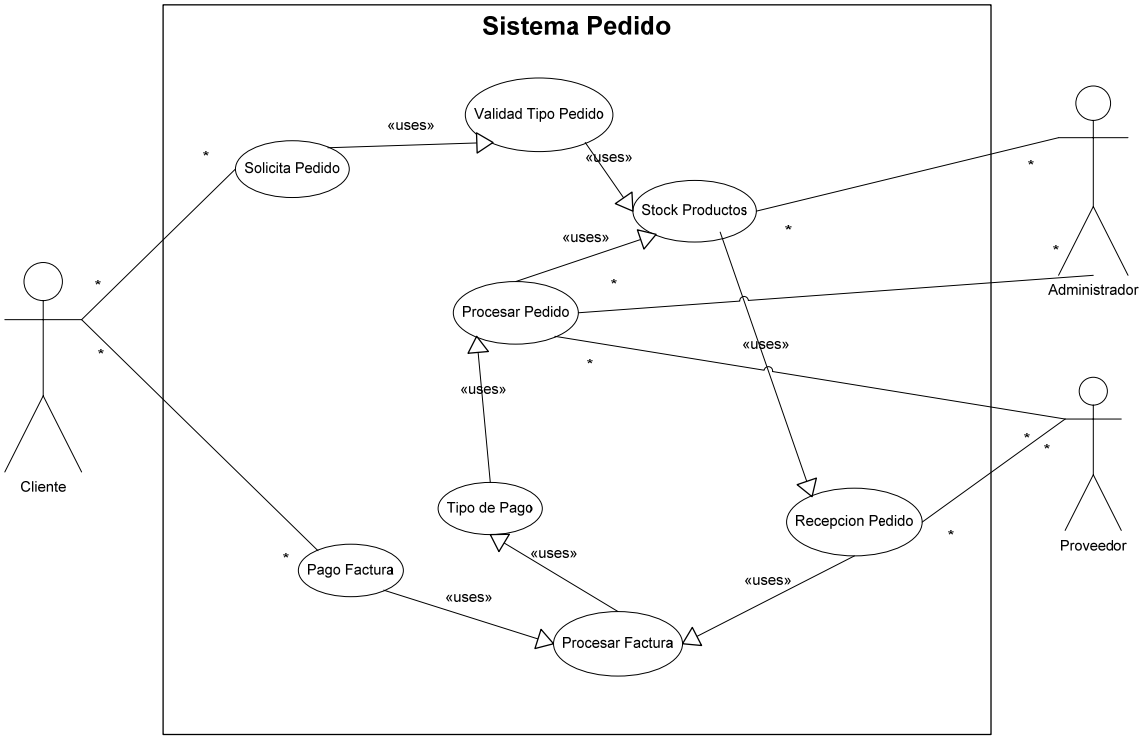
8 Ejercicios Desarrollados

1. Modulo de préstamo y devolución de un sistema de Biblioteca.



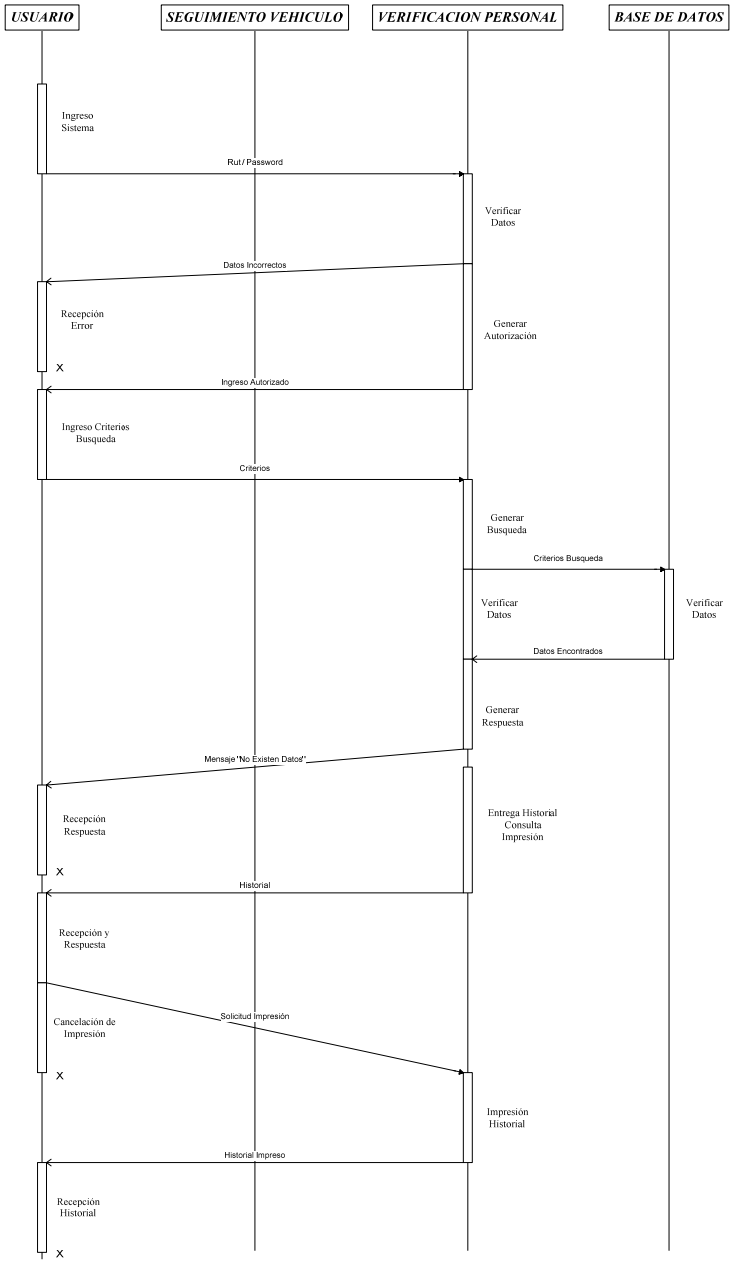
(Origen: Desarrollo Propio)

2. Sistema informático que procesa un pedido de una empresa. En este sistema el cliente puede realizar un pedido a través de la aplicación, al igual que el proveedor cuando se le notifica una solicitud de pedido.



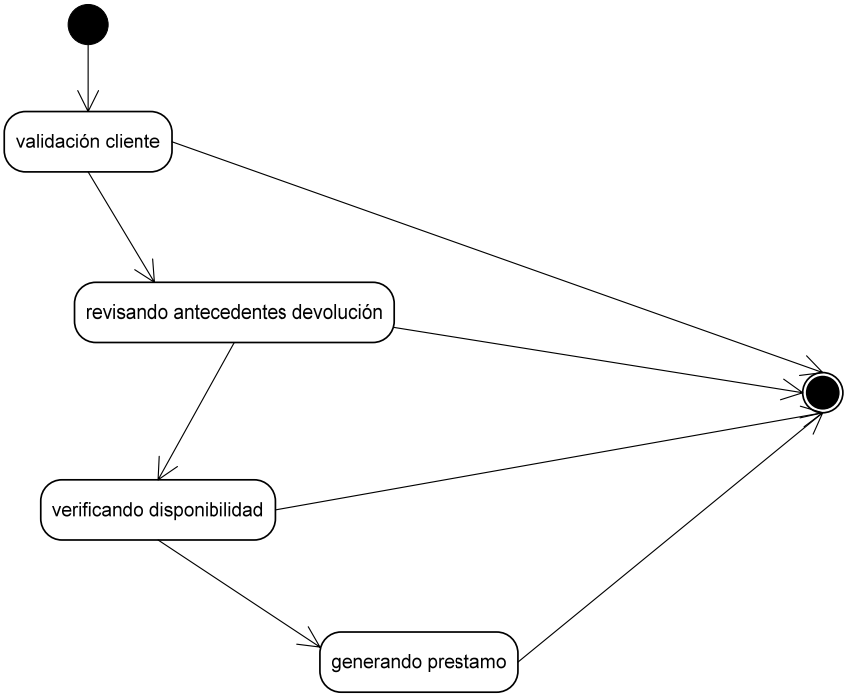
(Origen: Desarrollo Propio)

3. Diagrama de Actividad que representa el proceso de consultar información de un sistema para un taller mecánico desarrollado por alumnos de la asignatura SIA II.



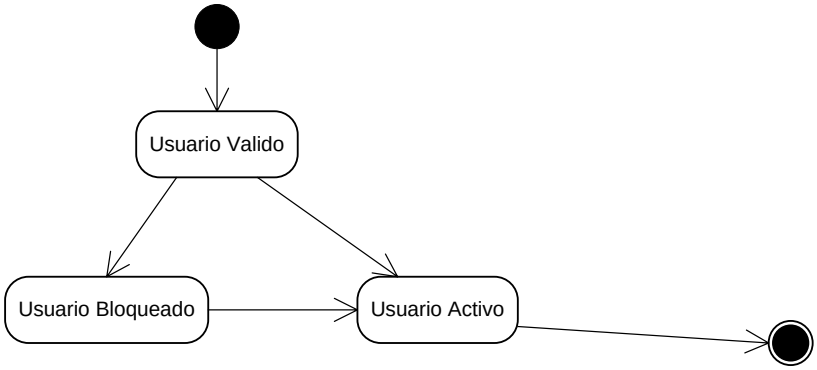
(Origen: Extraído Trabajo Alumno)

4. Diagrama de Estado que permite presentar los estados de una solicitud de préstamo.



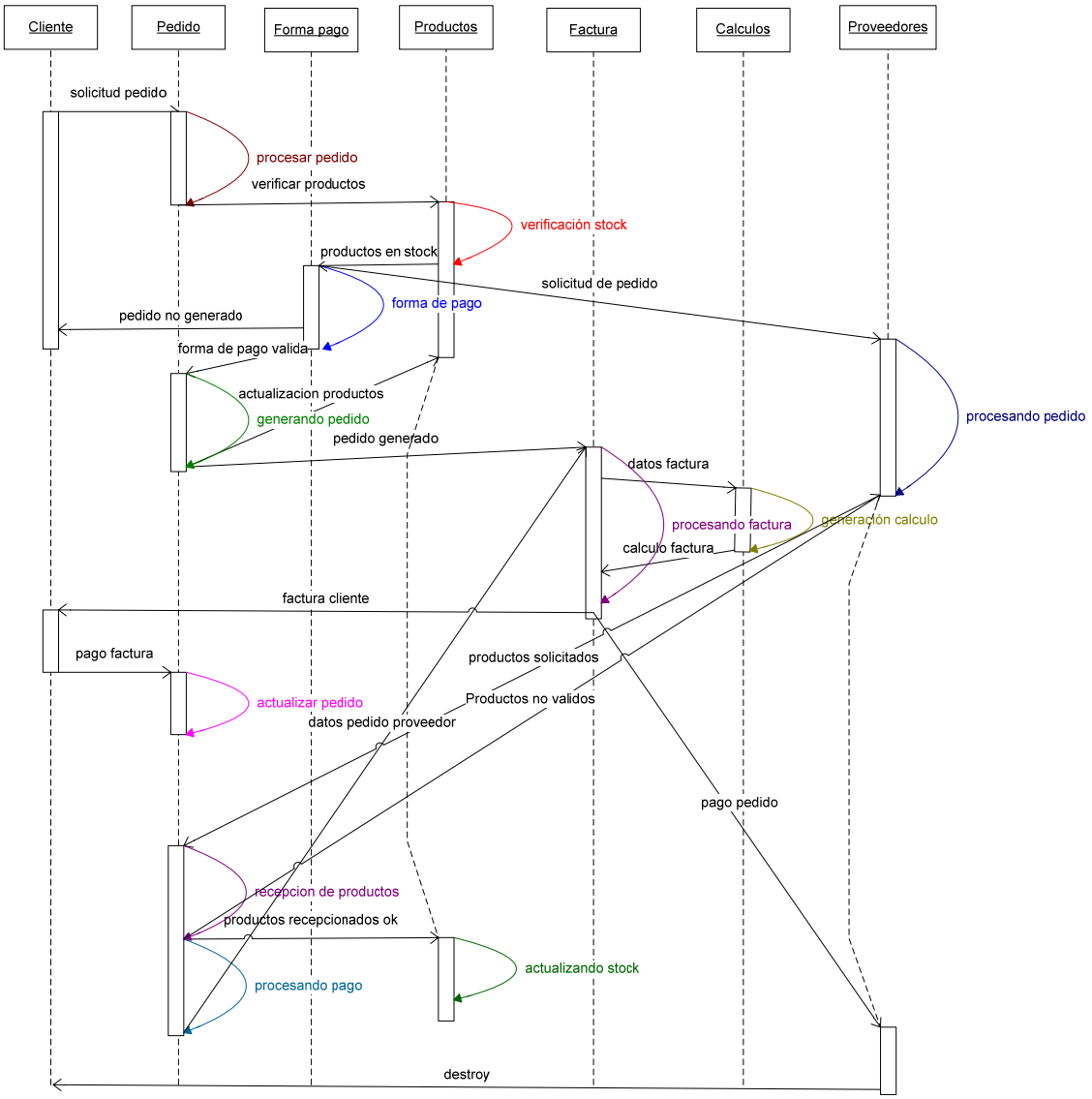
(Origen: Desarrollo Propio)

5. Diagrama de estado del objeto usuario



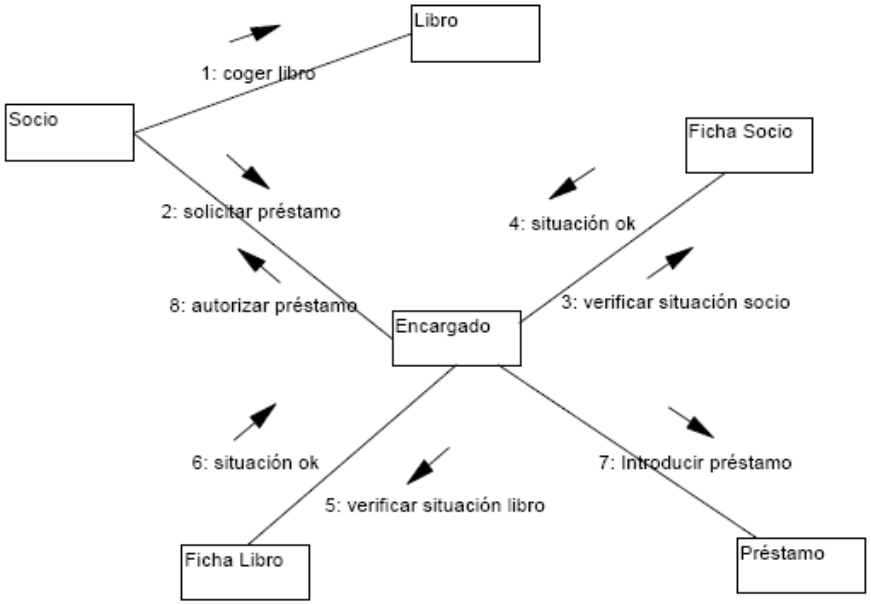
(Origen: Desarrollo Propio)

6. Diagrama de Secuencia que presenta la secuencia de la solicitud de un pedido.



(Origen: Desarrollo Propio)

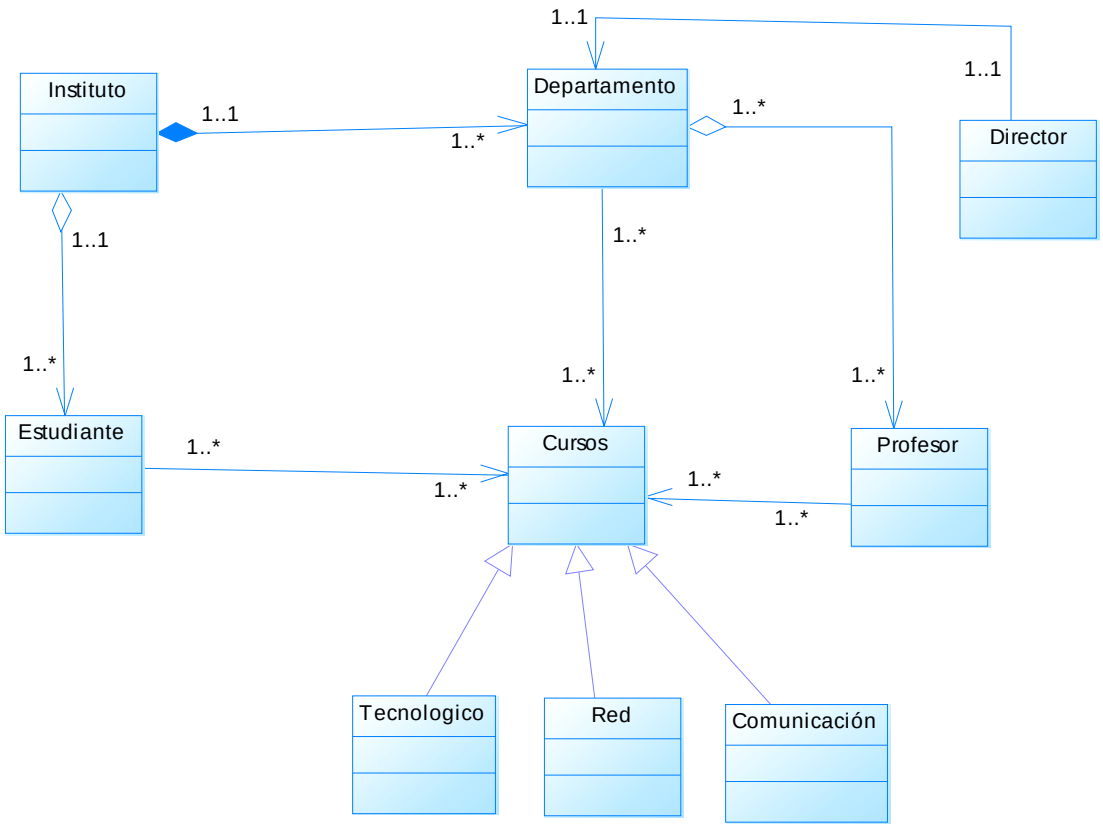
7. Diagrama de colaboración del proceso de préstamo de un libro.



(Origen: Extraído apuntes de internet)

8. Se desea desarrollar un sistema para los institutos tecnológicos de la región. Cada instituto deberá registrar a los respectivos alumnos asociados. Los diversos institutos deben tener al menos un departamento el cual poseerá un director asociado.

   Existen variados cursos, los cuales estarán asociados a cada departamento. Estos cursos son básicamente de tres tipos: Redes, Tecnológicos y Comunicación. Los profesores podrán dictar clases de diferentes cursos y los alumnos podrán asistir a más de un curso. Los profesores serán partes de un o más departamentos.

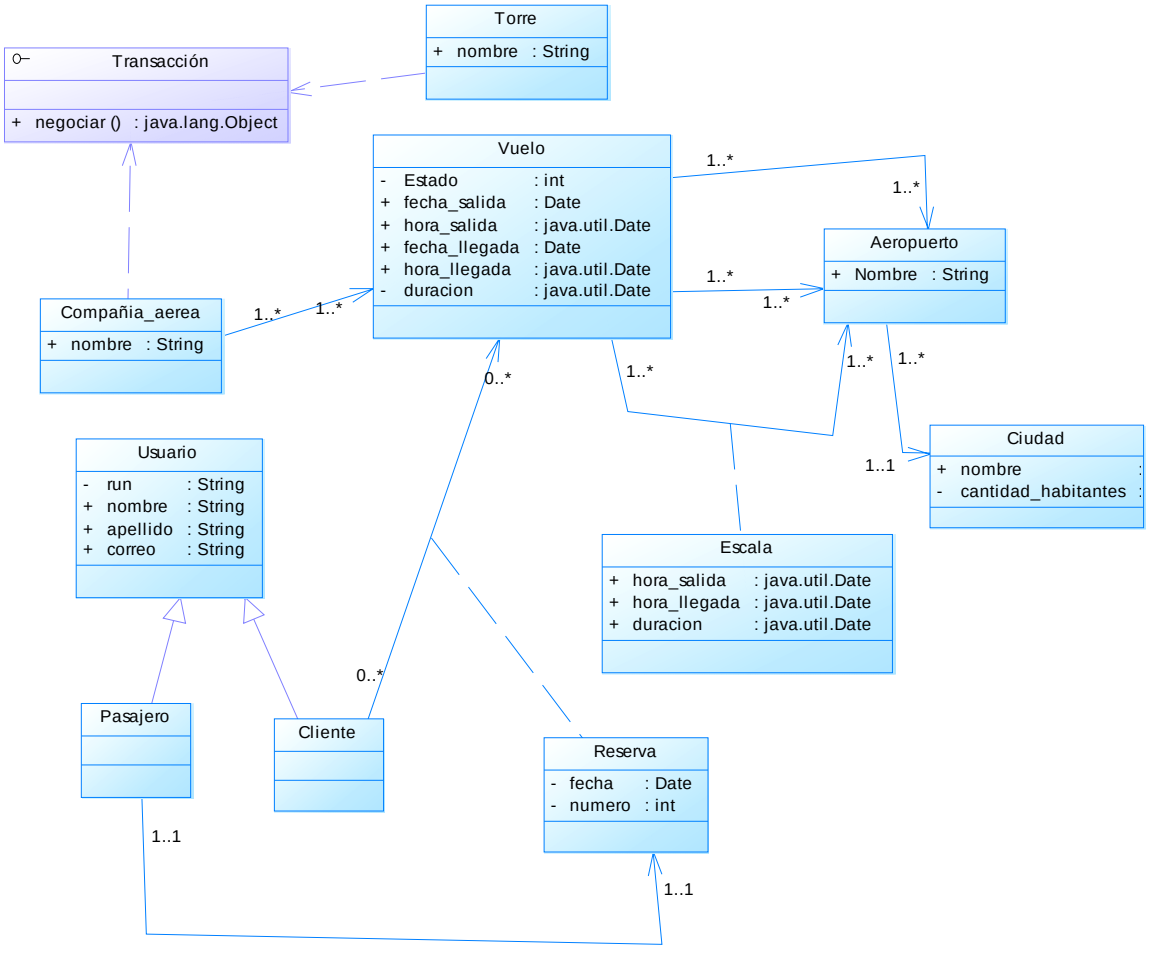


(Origen: Desarrollo Propio)

9. Se desea desarrollar un sistema para una agencia de vuelos. Los vuelos poseen como características: Estado, fecha de salida, hora de salida, fecha de llegada, hora de llegada y duración. Cada compañía aérea ofrece una serie de vuelos y negocia las transacciones de creación o cancelación de vuelos. Cada vuelo arriba y despega de diversos aeropuertos. En una ciudad puede existir más de un aeropuerto .Interesa conocer de la ciudad el nombre y la cantidad de habitantes. El vuelo realiza diversas escalas, la cual están asociadas al aeropuerto respectivo. Los datos de la escala son: hora de salida, hora de llegada y duración.

   Los clientes pueden realizar reservas de los cuales se conoce al menos la fecha de la reserva y la cantidad reservada. Un cliente puede reservar no necesariamente para el mismo, es decir el pasajero del vuelo no es necesariamente quien reserva. De los clientes se necesitan los datos tales como: RUN, nombre, apellido, teléfono y correo electrónico. De los pasajeros interesa saber. RUN, Nombre y Apellido.

   Una torre de control negocia para cada vuelo las transacciones de indicaciones asociadas tanto al despegue como el aterrizaje.



(Origen: Desarrollo Propio)

9 Ejercicios Propuestos

A continuación se presentan una serie de enunciados con los cuales pueden ser desarrollados los diferentes diagramas tratados en este manual.

1. Sistema gestión hotel.

   Dada la siguiente especificación de requisitos, identifica el diagrama de caso de uso.

   El proceso de reserva de una habitación en un hotel es iniciado por una petición de un cliente, quién establece sus necesidades. El sistema comprobará la disponibilidad y si hay disponible una habitación que se ajusta a la petición se crea una reserva y se envía una confirmación al cliente por e-mail. A partir de ese momento pueden suceder cuatro cosas: el cliente llega al hotel y ocupa la habitación; el cliente la cancela con antelación; el cliente cambia los detalles de la petición y se crea una nueva reserva; el cliente no aparece el día previsto y se le carga la cantidad prevista.

2. Se desea estudiar el sistema de pedidos de libros, realizados por los clientes, en una librería y su posterior envío y facturación. Se supone que la librería no mantiene stock de libros y por tanto debe pedir los libros solicitados a las editoriales correspondientes, con las cuales tiene concertado un sistema de descuentos en función de la cantidad de libros solicitados.

   Cada cliente tiene asociado un crédito permitido que debe ser controlado por el sistema para no aceptar pedidos si éste ha sido superado. Una vez validados los pedidos son agrupados por editorial para realizar un pedido de reaprovisionamiento asociado a los pedidos de los clientes. Estos pedidos se realizan dos días por semana.

   Cada editorial tiene establecido un tiempo estándar de respuesta. Una vez transcurrido este tiempo más una semana el pedido reaprovisionamiento puede ser anulado. Tras recibir y validar que lo enviado por la editorial se corresponde con lo solicitado, se deben asociar los pedidos de reaprovisionamiento y los de los clientes.

   Cuando el pedido del cliente está completo debe añadirse la dirección de envío y generar una prefectura, la cual irá acompañando a los libros solicitados por el cliente. Una vez recibido el paquete con los libros y la prefectura, el cliente deberá realizar el pago asociado a dicha prefectura.

   Al ser recibido un pago del cliente, deberá asociarse a una prefectura pendiente y enviar una factura definitiva al cliente. Si el pago no se efectúa en un período de 30 días desde el envío de la prefectura, el pedido llevará un recargo adicional.

   La dirección de ventas desea obtener mensualmente una estadística de compras por cliente, para de este modo poder clasificar a sus clientes en función a su volumen de pedidos. Junto a este informe, la misma dirección desea enviar un catálogo general anualmente y otro de novedades con carácter mensual sobre aquellos temas de más interés para cada cliente, para lo cual desea disponer de una estadística que indique los temas más frecuentemente solicitados.

   Una petición normal de los clientes, una vez solicitado un pedido, es saber en qué situación se encuentra.

3. Se desea desarrollar un cajero para el pago automático de los estacionamientos de vehículos de los parkings subterráneos de un Mall. El sistema estará inicialmente a la espera de que el usuario opere con el cajero automático. Se tienen las siguientes necesidades:

    a. Al momento de ingresar al Mall, el cliente retirara un ticket para que la barrera se levante y lo deje pasar.
    b. A la vez el cajero le avisara en donde se puede estacionar, para agilizar el ingreso de los vehículos al recinto.
    c. Luego cuando el usuario se retira, introducirá el ticket del aparcamiento, de tal forma que el cajero pueda contabilizar el tiempo de estacionamiento y solicitar el pago al usuario.
    d. Una vez calculado el pago a cobrar, el sistema lo mostrará al usuario, y se pondrá a la espera hasta que se introduzca el dinero. Si el sistema no tuviese cambio, lo indicaría al mismo tiempo que indica el importe a introducir.
    e. El usuario podrá introducir una cantidad de dinero igual o mayor que el importe indicado (siempre que hubiese cambio en el sistema), y en ese caso el sistema devolverá el ticket y el dinero sobrante si es que lo hubiese. Si pasados 15 segundos el usuario no hubiese recogido el ticket, el sistema le avisará mediante un pitido, que terminará cuando el usuario retire el ticket de la ranura. Si el usuario no recogiese el ticket de la ranura, el sistema tras otros 15 segundos de espera mientras se produce el pitido, tragaría el ticket sin oportunidad alguna de recuperarlo. En este caso debería mostrar un mensaje en pantalla indicando “ticket requisado” y la barrera no se levantara. Después volverá a la situación inicial.

4. La famosa cadena de videoclubes nos ha contratado con el fin de desarrollar un sistema para informatizar sus locales.

    Hasta hoy en día se han mantenido una serie de reuniones con el cliente con el fin de determinar los requerimientos del sistema. De tales reuniones, se ha determinado lo siguiente:

    El sistema deberá permitir que los clientes consulten el catálogo de películas. A partir del mismo, una vez seleccionada una película, se deberá poder acceder a la información de la misma como ser su clasificación, su género y un breve resumen de la misma. Asimismo, opcionalmente, se deberá poder consultar la disponibilidad del video y hacer la solicitud de arriendo correspondiente.

    Los empleados del videoclub deberán poder, a través del sistema, registrar arriendos (el es que procesa el arriendo cuando el cliente lo solicita) y devoluciones por parte de los clientes, y consultar, dado un cliente, los videos que éste posea alquilados. Si registrando un arriendo, resulta que el cliente no se encuentra registrado, el sistema deberá permitir que se efectúe su ingreso. Además del sistema deberá ser capaz de avisar si el cliente puede o no realizar arriendos.

    Nuestro cliente también pidió que el sistema, todas las mañanas genere de forma automática un informe que muestre todos los clientes que se encuentran atrasados con sus devoluciones. Cuando se le preguntó a que se refería con “todas las mañanas” aclaró: “Que todos los días a las 9:00 a.m. imprima o muestre por pantalla el listado de los atrasados.

5. Se pretende modelar el funcionamiento de un servicio de atención médica. El proyecto en cuestión es el desarrollo de un Módulo Informatizado de agenda medica, con él se pretende que el centro médico cuente con una herramienta que facilite la asignación de horas a los pacientes. Para poder llevar a cabo sus funciones el sistema deberá poder consultar información sobre horas de atención y la disponibilidad de los médicos de acuerdo a sus especialidades. Para la obtención de las posibles horas de atención el sistema cuenta con un módulo subordinado (al que emite solicitudes) denominado validación que es el encargado de definir y procesar las horas de la agenda.

   En el caso de que un paciente desee una hora y no está disponible, se le asignara la siguiente que esté disponible y se almacenara cuando el paciente confirme la hora asignada. En el caso que al paciente se le asigne un tratamiento, el sistema debe de ser capaz de asignar el horario completo del tratamiento.

   También se debe de tratar el caso en que el doctor no asista a sus horarios de atención, por lo cual el sistema deberá de ser capaz de reasignar las horas del médico. Además de cancelar alguna hora en la cual el paciente no pueda asistir.

6. Sistema Inmobiliario.

   Una agencia inmobiliaria quiere desarrollar un sistema web que facilite a compradores y vendedores la compra-venta de inmuebles, poniéndolos en contacto de una forma simple y eficiente. Es posible hacer y responder a ofertas, listar y buscar inmuebles, seleccionar agentes, negociar términos de la venta, gestionar préstamos bancarios, cerrar el contrato, proporcionar información sobre el inmueble. La aplicación será un punto de encuentro entre compradores y vendedores.

   Un vendedor añade los datos de un inmueble y selecciona un agente. Un comprador selecciona un agente y realiza una búsqueda de un inmueble introduciendo unos criterios de búsqueda. Si el comprador realiza una oferta por un inmueble, el sistema notifica al vendedor y a su agente. El vendedor responde con una contraoferta y el sistema notifica al comprador y al agente. Si comprador y vendedor llegan a un acuerdo, el sistema lo registra y el comprador puede solicitar un préstamo, en este caso, el sistema selecciona una entidad bancaria e informa al comprador que puede contactar con ella para llegar a un acuerdo.

7. Un profesor desea el desarrollo de un sistema computacional que le permita poder mantener un control más expedito de los alumnos que forman parte de sus cursos. Para lo cual ha tenido la idea de la realización de un sistema en plataforma WEB, en la cual puedan interactuar él y sus alumnos.

   Lo que el alumno puede realizar con esta aplicación es lo siguiente: Ingresar sus datos personales, realizar consultas (ya sea por correo como por foro), ver el estado de revisión de sus trabajos, notas asociadas, observaciones del docente y su promedio. En el caso del docente, puede desarrollar operaciones tales como: crear usuarios, crear actividades, modificar, ingresar datos, generar reportes, actualizar estados, e ingreso de notas, además de mantener actualizado su centro de documentación.

   Los dos usuarios para poder hacer uso de la aplicación deben de estar previamente validados.

8. Un taller mecánico desea la realización de un sistema informático que le permita poder manejar de mejor forma el seguimiento de los trabajos desarrollador en sus dependencias. Esto quiere decir, que la administración desea saber en un instante cualquiera el estado de un trabajo desarrollado.

   Se debe de conceptualizar que los datos de los clientes y sus respectivos vehículos los ingresa la secretaria del taller, el encargado de taller es el responsable de asignar el mecánico y estimar el tiempo de trabajo. El mecánico es el responsable de ingresar el estado del trabajo desarrollado.

   Loa mecánicos pueden poseer más de una especialidad por lo cual pueden desarrollar distintos trabajos.

   A su vez la planta administrativa desea manejar información específica y general del funcionamiento del taller en lo referente a los servicios entregados. Además de contar con la información necesaria que le permita poder aplicar políticas que permitan retener al cliente.

9. Una sociedad que se dedica a la confección de velas aromáticas desea comenzar la promoción de sus productos a través de un medio innovador como lo es Internet. Esta idea ha tenido una buena acogida en las empresas que forman parte de la sociedad.

   Hasta el momento existen 8 empresas que conforma esta sociedad, en ellas se pueden encontrar empresas con una gran infraestructura y otras en las cuales la empresa la forma un grupo familiar reducido.

   Dentro de los productos que estas empresas poseen se pueden encontrar: Velones, Velas con Figuras, Velas flotantes, Velas de Parafinas, Velas de Gel, Velas con incrustaciones, Velas con diseño, Velas de Miel, Porta Velas

   Cada empresa que forma parte de la sociedad esta trabando con los mismos proveedores, pero en algunas oportunidades ellos no son capaces de entregar todas las materias primas por lo cual se recurren a otros proveedores que quedan registrados como alternativos.

   Por lo cual lo que se desea realizar es un sistema que tenga plenamente identificados los productores de velas y sus productos, control de los proveedores para evitar retrasos en las entregas de pedidos, además de establecer un medio de solicitud de pedidos de los clientes en forma más expedita.

10. Una empresa de transporte escolar desea la realización de un sistema informático que le permita poder manejar los datos de sus clientes en forma más expedita, puesto que el nivel de crecimiento de la empresa ha ido en aumento año a año.

    La dueña desea poder contar con una aplicación que le permita saber exactamente en qué furgón esta un estudiante a consultar, además de que el sistema pueda generar en forma automática la ruta que debe de seguir el furgón previo ingreso de los antecedentes por la dueña o administradora. A su vez quiere conocer el estado de las cuotas del servicio y si el cliente no ha pagado que generar una alerta indicando al alumno y el monto a cancelar.

    Se debe de establecer que un alumno no puede ser asignado a un furgón sino está ingresado en la BDD, así como cualquier otro dato que forme parte del proceso que implica el transporte escolar.

10 Bibliografía

- ✓ Análisis y Diseño Orientado a Objeto de Sistemas Usando UML; Simon Bennett, Steve McRobb y Ray Farmer.
- ✓ Análisis y Diseño de Sistemas; Kendall & Kendall.
- ✓ Aprendiendo UML en 24 horas; Joseph Schmuller
- ✓ El Lenguaje Unificado De Modelado. Manual de Referencia; James Rumbaugh, Ivar Jocaobson y Grady Booch.
- ✓ Ingeniería de Software: Un Enfoque Práctico; Roger Pressman.
- ✓ Patrones de Diseño, Elementos de Software Orientado a Objetos Reutilizable; Erich Gamma, Reichard Helm, Ralph Johnson, John Vlissides.
- ✓ Sistemas de Información, Raúl Horacio Saroka, Fundación OSDE.
- ✓ UML y Patrones, Introducción al Análisis y Diseño Orientado a Objeto; Graig Larman
- ✓ UML Gota a Gota; Martin Fowler, Kendall Scott.
- ✓ http://www.bpmn.org

11 Anexos

ANEXO I

Técnicas de Recolección de Datos

Las necesidades del sistema a estudio se obtendrán principalmente de los usuarios del mismo. Esta información se podrá conseguir haciendo uso de diversas herramientas como son las entrevistas, los cuestionarios, la observación directa, los archivos del sistema y los muestreos.

La entrevista es la principal herramienta para la obtención de información de un sistema. El objetivo de la entrevista es obtener información de un sistema que se desea analizar e informatizar.

El éxito de una entrevista se basa en la experiencia del entrevistador. Se requiere especial habilidad cuando los entrevistados no se sienten cómodos, hay que saber escuchar, dejar hablar, no adelantarse a las respuestas, realizar las preguntas desde diferentes puntos de vista, etc.

Muchos entrevistadores creen que las entrevistas son similares a las conversaciones normales y prefieren no estructurar las preguntas. En una entrevista completamente estructurada todo está planeado, existirán preguntas generales del sistema, pero las preguntas específicas serán el centro principal de la misma.

Las entrevistas no estructuradas, es decir, sin orden establecido en las preguntas, pueden hacer que el sistema sea más difícil de entender. Esto no implica una falta de preparación. También, el tiempo de contacto requerido es mayor en una entrevista no estructurada que en una estructurada.

La única forma de decidir si las preguntas son las más apropiadas en su orden, es anticipar las posibles respuestas (lo que la otra persona podría decir) y ver como se suceden.

Existen diversos factores que afectan a la persona que realiza las entrevistas: la educación, la inteligencia y las emociones. Es necesario elaborar la entrevista antes de hacerla, analizar por qué se quiere hacer, qué se preguntará, a quién se preguntará y qué cuestiones harán de la misma conseguir los objetivos deseados.

Fuentes de información.

Entre las fuentes de información más usuales se pueden establecer las siguientes:

- **USUARIOS**: Forman la primera fuente de información a investigar por el analista. De estos se obtienen las actividades del sistema existente y se determinan sus objetivos y necesidades. Los métodos utilizados son la

entrevista, el cuestionario y la observación directa de las tareas que realizan.

- **PROGRAMAS:** Si existe un sistema informático, se estudiarán las características del mismo, así como las mejoras, los problemas existentes, etc.

- **CATÁLOGOS DE USUARIOS:** Indica las funciones de cada uno de los miembros de la organización.

- **DOCUMENTACIÓN DEL SISTEMA:** Indica todas las posibles salidas que se realizan en el sistema. Esta fuente debe acompañarse de una explicación por parte del usuario de sus objetivos y si desea cambios. También se puede acceder a normativas o políticas organizacionales.

Técnicas de Recolección de Datos

Entrevistas

Existen cinco pasos principales en la preparación de una entrevista, lo cuales se presentan a continuación.

a) *Recoger el material histórico*: obtener y comprender tanta información histórica sobre los entrevistados y su organización, como sea posible. Normalmente, se puede obtener a través de la persona de contacto de la organización, el informe anual actual y por alguna publicación hecha para explicar su funcionamiento al público.

   Leyendo este material, será posible entender el lenguaje que usan de los miembros de la organización para describirse a ellos mismos y a dicha organización. Lo que se intenta hacer es construir un vocabulario que nos permita construir las preguntas de la entrevista de tal manera que el entrevistado pueda comprenderlas.

   Otra ventaja de investigar la organización es que se maximiza el tiempo invertido en la entrevista dado que no es necesario preguntar por cuestiones generales.

b) *Establecer los objetivos de la entrevista*: usando la información histórica así como la propia experiencia, se deben establecer los objetivos de la entrevista. Qué aspectos hay sobre procesamiento de información y sobre toma de decisiones. Estos puntos incluyen las fuentes de la información, los formatos de la información, la frecuencia en la toma de decisiones, la calidad de la información y en el estilo de toma de decisiones.

c) *Decidir a quién entrevistar*: cuando se decide a quién entrevistar se incluye a las personas clave en todos los niveles afectadas en el sistema de alguna forma.

   Hay que estructurar correctamente las necesidades del usuario. La organización propondrá las personas de su entorno que serán entrevistadas. Tendrá también idea de quién debería ser entrevistado. Pero el analista puede cambiar a los posibles usuarios.

d) *Preparar la entrevista*: Se debe preparar a los entrevistados, avisándoles con suficiente tiempo. Hay que organizar el tiempo de los encuentros. Las entrevistas no deben durar más de 45 ó 60 minutos.

e) *Decidir el tipo de preguntas*: Las preguntas escritas deben cubrir las áreas clave de toma de decisiones descubiertas cuando se establecieron los objetivos.

Existen distintos tipos básicos de preguntas que a continuación de presentan:

a) Preguntas generales, no limitadas:

Las opiniones son abiertas, sin límites, dado que su respuesta puede ser dos palabras o dos párrafos.

Por ejemplo, algunas preguntas generales podrían ser:

¿Cuál es su opinión sobre el sistema informático actual en su organización?
¿Cuáles son las ambiciones en este departamento?
¿Cómo relataría el trabajo que hace?
¿Cuáles son los errores comunes que se cometen en los datos de entrada en este departamento?

Las ventajas de usar preguntas generales incluyen:

1. Hacer que el entrevistado se sienta cómodo.
2. Permitir al entrevistador captar el vocabulario del entrevistado que refleja su educación, valores, aptitudes y creencias.
3. Proveer de numerosos detalles.
4. Revelar la forma de hacer futuras preguntas que podrían estar mal adaptadas.
5. Darle más importancia al entrevistado.
6. Permitir mayor espontaneidad.
7. Que el entrevistador pueda expresarse más fácilmente.
8. Usarlas en un apuro si el entrevistador se coge de improviso.

Como se puede ver, existen bastantes ventajas, pero también están las desventajas:

1. Estas preguntas podrían llevar a demasiados detalles irrelevantes.
2. Existe la posibilidad de perder el control de la entrevista.
3. Se permiten respuestas que llevan mucho tiempo en relación a la información útil generada.
4. Potencialmente puede parecer que el entrevistador no está preparado.

b) Preguntas específicas:

Son aquellas cuya respuesta es determinada, por ejemplo "¿Cuántos subordinados tiene?" donde las posibles respuestas pueden ser sólo un número finito como "Ninguno", "Uno" o "Cinco".

Otros ejemplos de este tipo de preguntas son :

¿Cuántos informes genera al mes?
¿Qué pasaría si el informe no se genera o se atrasa?
¿Cuántas veces y cuando se genera la nómina?
¿Cuánto tiempo toma en desarrollar este proceso?
¿Cuáles son las dos prioridades básicas del departamento de marketing?

Una pregunta específica o determinada puede limitar la respuesta al entrevistado si ofrece opciones para responder una de ellas. Se deber estar preparado a las contestaciones de este tipo de preguntas, ya que son más específicas.

Una clase especial de este tipo de preguntas, es la pregunta binaria, es decir, aquélla que posee sólo dos respuestas posibles una la negativa y otra la positiva, por ejemplo si o no, verdadero o falso, de acuerdo o en desacuerdo.

Las ventajas de usar estas preguntas, de cualquier clase, son:

1. Ahorrar tiempo.
2. Facilidad para comparar entrevistas.
3. Mantener el control de la entrevista.
4. Cubrir rápidamente el terreno.
5. Obtener los datos relevantes.

Los inconvenientes de usar preguntas específicas o determinadas son:

1. Pueden aburrir al entrevistado.
2. Dejar de obtener detalles interesantes (debido a que el entrevistador proporciona y limita el marco de referencia al entrevistado).
3. Perder ideas principales por la razón anterior.
4. Dejar de estar de acuerdo entrevistador y entrevistado.

c) Sondeos:

Un tercer tipo de preguntas constituyen el sondeo o entrevista complementaria.

El sondeo más simple se basa en la pregunta "¿Por qué?". Otras son: ¿Puede darme algún ejemplo ?, o ¿Podría darme más explicaciones a cerca de eso?.

El propósito del sondeo es ir más allá de la respuesta inicial para obtener más información, clarificar y expandir el punto de vista del entrevistado. Los sondeos pueden ser preguntas limitadas y no limitadas.

Es recomendable modificar algunas preguntas pobres o inoportunas que puedan falsear los datos. Estas preguntas se llaman preguntas inductoras y preguntas dobles.

- Evitar preguntas inductoras: Este tipo de preguntas tienden a inducir al entrevistado a responder lo que el entrevistador quiera. La respuesta está influenciada por el entrevistador.

Un ejemplo de este tipo de preguntas es " Está de acuerdo con otros directivos en que el control de inventario debería ser informatizado, ¿verdad?" De esta forma el entrevistador hace bastante incómoda la postura de estar en desacuerdo. Como alternativa es preferible la forma " ¿Qué opina sobre la informatización del control de inventario? " y así los datos obtenido serán más reales, válidos, fáciles de entender y más útiles.

- Evitar preguntas dobles: Son aquellas cuestiones que usan sólo una pregunta que normalmente se separaría en dos, por ejemplo "¿Qué decisiones se toman durante un día normal y cómo las lleva a cabo?". Si el entrevistado responde a este tipo de preguntas puede que los datos sufran variaciones no deseadas.

Una pregunta doble no es una buena elección ya que los entrevistados podrían responder sólo a una de las cuestiones, o el entrevistador podría confundir la pregunta a la que responden y llegar a conclusiones equivocadas.

Formas de registrar

Registrar la entrevista es el aspecto más importante. Dependerá de a quién se esté entrevistando y de qué se hará con la información una vez terminada la entrevista, la forma en que se registrara.

a) Grabar la entrevista:

Se debe informar al entrevistado que se desea grabar la entrevista, además de lo que se hará posteriormente con la cinta como que será escuchada por el entrevistador y por otros miembros de su equipo y luego será destruida, o bien será transcrita y usada como información para el desarrollo del sistema. Puede que el entrevistado se niegue prefiera que se tomen notas.

Las ventajas de esta forma son:

1. Proporciona una grabación exacta de lo que cada persona dijo.
2. Libera al entrevistador de escuchar y responder rápidamente.
3. Permite un mayor y mejor contacto visual entre el entrevistador y el entrevistado.
4. Permite responder a la entrevista a otros miembros del equipo.

Las desventajas son:

1. Posibilidad de poner al entrevistado nervioso y menos apto para responder libremente.
2. Posibilidad de que el entrevistador esté menos atento a las respuestas dado que todo quedará grabado.
3. Dificultad de localizar partes importantes en una cinta muy larga.
4. Incrementa los costos ya que es necesario transcribir las cintas.

La decisión de grabar la entrevista se hará sobre el conocimiento que tenga el profesional de la misma y el proyecto en particular.

b) Tomar notas

Este método sería la única alternativa ante una negativa de grabación de la entrevista.

Sus ventajas son:

1. Mantiene al entrevistador alerta.
2. Ayuda a insistir en cuestiones importantes.
3. También ayuda a dirigir la entrevista por la dirección correcta.
4. Muestra el interés del entrevistador.
5. Demuestra la capacidad y preparación del entrevistador.

Las desventajas incluyen:

1. Se pierde el contacto visual entre entrevistado y entrevistador.
2. Se pierde el curso de la conversación.
3. Puede hacer que el entrevistado esté indeciso a hablar cuando se toman mal las notas.
4. Excesiva atención hacia los hechos y poca para los sentimientos y opiniones.

Luego, para preparar la entrevista debidamente, el entrevistador debe analizarse a sí mismo y al entrevistado, investigar la organización, analizar las áreas no conocidas, contactar con los que serán entrevistados, escribir las cuestiones y formular un plan para llevarla a cabo.

Dirección

Al iniciar la entrevista se debe informar al que será entrevistado lo que se hará con los datos tomados y asegurarle la total confidencia de los mismos. El entrevistador debe preparar la grabadora o el bloc de notas según el caso.

Es conveniente empezar siempre por preguntas generales que ayude a relajar el ambiente, así como comprender la actitud, cultura y vocabulario empleado por el entrevistado.

Cuando sea necesario, el entrevistador debe informar al entrevistado el nivel de detalle que requiere en la contestación de alguna pregunta: pedirle algún ejemplo en cuestiones que necesita profundizar, o bien informar que un "sí" o un "no" son suficientes.

Controlar el tiempo de la entrevista e intentar que en ningún caso supere el tiempo planificado por pregunta.

Para finalizar la entrevista, podemos seguir otros procedimientos como resumir y nombrar las impresiones globales, o informar al entrevistado sobre la secuencia de pasos tomada y lo que el entrevistador hará con esos datos.

Documentación

Aunque complete con éxito la entrevista, el trabajo del entrevistador no hecho más que empezar. Se debe resumir los contenidos de la misma en un informe escrito, lo más pronto posible, para asegurar así la calidad de los datos.

Después de escribir ese resumen inicial, el entrevistador debe entrar en más detalle, identificando los puntos principales de la entrevista y su propia opinión.

Revisar el informe de la entrevista ayuda a clarificar el significado de la misma y a tener en mente lo que el entrevistado sabe, notando así este último que el entrevistador está interesado suficientemente en tomarse tiempo para entender su punto de vista y percepciones.

Ejemplo de una entrevista, que ayudará al entrevistador a planear las próximas entrevistas:

Entrevistado: Pedro Gómez Fecha: 28 de noviembre de 2008
Entrevistador: Juan Gómez Tema: Uso del ordenador

Objetivos:

- Encontrar una opinión sobre el uso de los ordenadores
- Conocer la estimación del usuario sobre su uso
- Conocer la opinión del nuevo sistema propuesto

Objetivos del sondeo:

- Descubrir qué le parece el sistema propuesto en su departamento.
- Conseguir opiniones sobre a quién entrevistar después.

Principales puntos de la entrevista

Las funciones del entrevistado están claras.
Usa el ordenador para todas sus funciones.

Opiniones del entrevistador

Persona adecuada para obtener la información requerida.
Uso de los ordenadores es fundamental.
Existe una mala estructura de la información.

Cuestionarios

Los cuestionarios son una técnica de recolección de información que permite a los analistas de sistemas estudiar actitudes, comportamientos, y características de las personas clave en la organización que pueden ser afectadas por los sistemas actuales y los propuestos.

Mediante el uso de cuestionarios, el analista puede buscar la cuantificación de lo que se obtuvo en las entrevistas. Los cuestionarios pueden utilizarse para determinar lo general o limitada que es una opinión expresada en una entrevista.

Los cuestionarios pueden usarse para recoger información de una gran muestra de usuarios del sistema a fin de detectar los problemas o destacar cuestiones importantes antes de planificar las entrevistas.

Hay muchas similitudes entre las entrevistas y los cuestionarios. Lo ideal sería utilizar ambas conjuntamente, o complementar respuestas imprecisas de un cuestionario con una entrevista o diseñando el cuestionario con base a lo que se descubrió en la entrevista. Sin embargo, cada técnica tiene sus funciones específicas propias y no es siempre necesario o deseable el uso de ambas.

Planificación

A primera vista, los cuestionarios pueden parecer una manera rápida de reunir grandes cantidades de datos sobre cómo los usuarios evalúan el sistema actual, qué problemas experimentan con su trabajo, y qué espera la gente de un sistema nuevo o uno modificado.

Para desarrollar un cuestionario útil se necesita mucho tiempo de planificación. Se debe decidir primero qué es lo que intenta obtener mediante el uso de un cuestionario. Por ejemplo, si usted quiere saber qué porcentaje de usuarios prefiere un centro de formación como medio para aprender acerca de nuevos paquetes de software, entonces un cuestionario podría ser la técnica correcta. Si queremos hacer un análisis en profundidad de un proceso de toma de decisiones de la dirección, entonces una entrevista es una elección mejor.

Los cuestionarios se utilizan cuando:

1. La gente a la que necesitamos preguntar está geográficamente dispersa.
2. Un gran número de personas está involucrado en el proyecto, y es significativo conocer qué proporción de un grupo determinado aprueba o desaprueba un aspecto particular del sistema propuesto.
3. Se está realizando un estudio exploratorio y queremos medir la opinión total antes de que el proyecto tome alguna dirección específica.
4. Deseamos estar seguros de que cualquier problema con el sistema actual está identificado y ha sido tratado en las entrevistas.
5. Una vez que determinamos la necesidad de usar un cuestionario y hemos descubierto los objetivos que han de ser cumplidos mediante su uso, podemos crear las preguntas.

Tipos de preguntas

La mayor diferencia entre las preguntas utilizadas en las entrevistas y las utilizadas en cuestionarios es que la entrevista permite interacción con respecto a las preguntas y sus significados.

En una entrevista, el analista tiene una oportunidad para refinar una pregunta, definir un término confuso, cambiar el curso de las preguntas, responder a una mirada confusa, y generalmente controlar el contexto.

a) Preguntas abiertas

Las preguntas abiertas son aquellas que dejan todas las opciones posibles de respuesta abiertas al entrevistado. Por ejemplo, las preguntas abiertas en un cuestionario podrían decir, "Describa algún problema que usted tenga respecto a la introducción de datos en el sistema" o "¿Son claros los manuales que existen acerca del uso de los programas?".
Debemos ser lo suficientemente concreto para orientar a los entrevistados a responder de una manera específica. Las preguntas abiertas son particularmente favorables en las situaciones en que se quiere conseguir opiniones de los miembros de la organización sobre algún aspecto del sistema, ya sea el producto o el proceso.

Se necesitará utilizar preguntas abiertas, ya que es imposible enumerar de forma efectiva todas las respuestas posibles a la pregunta.

Las preguntas abiertas son útiles en situaciones exploratorias. Estas ocurren cuando el analista de sistemas no es capaz (a causa de la diversidad de opiniones) de determinar con precisión qué problemas existen con el sistema actual.

¿Cuáles son los problemas más frecuentes que usted experimenta con la salida de los informes por la impresora?.

A. _____
_____
B. _____
_____
C. _____
_____

De los problemas que ha enumerado anteriormente, ¿cuál es el más molesto?

_____
¿Por qué? _____
_____
_____

Seguidamente se formulan preguntas sobre usted mismo. Por favor, rellene en los espacios vacíos de la mejor manera que pueda.

¿Cuánto tiempo ha trabajado usted para esta compañía?
_____ años y _____ meses.

¿Cuánto tiempo ha trabajado usted en la misma ocupación?
_____ años y _____ meses.

¿Cuál es su edad?
_____ años.

b) Preguntas cerradas

Son aquellas que limitan las opciones de respuesta disponibles al entrevistado. Las preguntas cerradas en los cuestionarios pueden ayudar a asegurar las respuestas.

Un ejemplo sería: de los programas siguientes, marque con una x, el que más utiliza.

Las preguntas cerradas deberían usarse cuando el analista es capaz de enumerar con efectividad todas las respuestas posibles a la pregunta y cuando todas las respuestas enumeradas son mutuamente excluyentes, para que la elección de una excluya la elección de cualquiera de las otras.

Las respuestas a preguntas abiertas pueden ayudar a los analistas a obtener conocimientos de exploración amplios, así como también anchura y profundidad sobre un tema. Aunque las preguntas abiertas pueden escribirse fácilmente, las respuestas a ellas son difíciles y consumen tiempo para ser analizadas.

A continuación se presenta un ejemplo:

Responda las siguientes preguntas marcando en la casilla apropiada.

Seguidamente se muestran los seis paquetes de software disponibles actualmente en el Centro de información. Por favor, marque el paquete que usted use personalmente con mayor frecuencia.

[ ] Excel	[X] Word para Windows
[ ] Lotus	[ ] Works
[ ] PowerPoint	[X] Excelerator

"Los informes sobre el stock normalmente son incorrectos"

[ ] De acuerdo [X] En desacuerdo

Responda a las preguntas

"Cuando las gráficas de ventas se preparan mediante los servicios de datos de la computadora se retrasan"

Nunca	Raramente	A veces	A menudo	Siempre
1	2	3	4	5

Responda a la siguiente pregunta:

El Departamento al que pertenezco se llama

Inversiones
Gestión de Cobros
Contabilidad

ANEXO II

Ejemplo de Factibilidad del Proyecto

Intranet con Servidor Propio.

Descripción:

Esta alternativa contempla, la compra de un servidor para alojar la Base de Datos del liceo, que estará desarrollada en un motor de SQL Server; o bien, se podría acondicionar un computador con características de servidor, en este deberá estar instalado el sistema operativo Windows 2003 Server y funcionar como servidor de aplicación, además de estar configurado como servidor DHCP para las demás maquinas y Web Services, incluyendo las herramientas de aplicaciones en el servicio IIS provisto por el sistema operativo, ya que sólo existirá una intranet; donde los usuarios podrán acceder a los datos desde dentro del liceo sin tener la oportunidad de acceso remoto desde cualquier punto geográfico. Los usuarios podrán acceder al sistema por medio de una clave que se le asignará al momento de poner el funcionamiento este nuevo sistema.

Factibilidad Operacional:

La factibilidad operacional existe, ya que la mayoría de las personas hoy en día están familiarizadas con el ambiente y aplicaciones web, en este caso será una ventaja, ya que el sistema está diseñado en dicho ambiente y por ende es fácil de utilizar; y en caso de surgir algún inconveniente, en el proyecto está contemplada una capacitación para los distintos grupos de usuarios de este sistema, además ante la idea de un nuevo sistema de información en la empresa las personas están más optimistas y dispuestas a aprender a utilizar nuevas tecnologías.

Factibilidad Técnica:

Gracias a los adelantos tecnológicos es posible el desarrollo de este sistema, ya que muchas de las herramientas, tanto de software como de hardware necesarias para la realización de éste se encuentran disponibles en el mercado. La empresa en cuestión cuenta con la mayoría del hardware necesario, existiendo 15 computadores, 4 impresoras, conexiones de red en puntos estratégicos para la comunicación de los distintos equipos y el cableado ya existe, debido a que en el liceo está implementado el proyecto Enlaces, además la empresa cuenta con un switch, el cual puede ser reutilizado en beneficio del sistema; y en cuanto al software los computadores trabajan con el sistema operativo Windows 98, el cual permanecería sin alteraciones. En cuanto al servidor, como esta opción sólo incluye una Intranet, no se necesita un gran poder de procesamiento por lo cual el servidor podrá ser de mediana capacidad, es decir, se podría acondicionar un computador con características de servidor, en este deberá estar instalado el sistema operativo Windows 2003 Server y funcionar como servidor de aplicación, además de estar configurado como servidor DHCP para las demás maquinas y Web Services, incluyendo las herramientas de aplicaciones en el servicio IIS provisto por el sistema operativo. La ubicación física del servidor, deberá estar localizada en una pieza a propósito acondicionada y protegida, de tal forma que se tenga un rápido acceso a ella en caso de emergencia. Por último el servidor estará protegido contra cortes de energía por una UPS ubicada físicamente en el mismo lugar, la cual brindará un lapso de tiempo

de 15 minutos para bajar los servicios y apagar correctamente el servidor. En cuanto a los conocimientos que deben tener las personas que utilizarán el sistema, se necesita:

- Un ingeniero informático que tenga conocimientos de gestión, planificación y contabilidad, y modelamiento de bases de datos, además de tener vastos conocimientos en su área.
- Un analista de sistemas que posea los conocimientos necesarios para desarrollar aplicaciones en flash, ASP y base de datos.
- Dos personas que tengan conocimientos básicos de computación, las cuales estarán debidamente capacitadas para reaccionar ante cualquier emergencia que ocurra en el servidor, específicamente apagar el servidor cuando se produzca una falla de energía y encenderlo cuando ésta llegue.

Los componentes necesarios para el desarrollo de esta opción en los cuales se incurrirá en gastos son los siguientes:(excluyendo, los de hardware y software ya existentes en el liceo, los cuales se especifican en la factibilidad económica.)

Hardware:

- Un servidor, con procesador Pentium 4 HT de 3500 mhz, 1 GB memoria RAM, 200 GB. Disco Duro SCSI, tarjeta de red ethernet 1000 BaseT 10/100/1000.
- 100 metros de cable UTP.
- Switch de 6 bocas
- 1 UPS 1200 Watts.
- Conectores RJ45, Capuchones RJ45
- Impresoras.

Software:

- Microsoft SQL Server 7.0
- Windows 2003 Server
- Macromedia Dreamweaver MX y Macromedia Flash MX
- Microsoft ASP

Recursos Humanos:

- Ingeniero Informático.
- Analista Programador.
- Encargado del Servidor (persona encargada de cuidar el servidor en caso de cortes de energía eléctrica)

Otros recursos:

- Una pieza especial para instalar el servidor, la cual estará construida de material no inflamable, de preferencia albañilería de ladrillo con estuco de cemento y piso del mismo material, las medidas pueden ser de 2 x 3 metros, dicha pieza deberá tener una puerta con llave y un sistema efectivo de ventilación.

Factibilidad Económica.

HERRAMIENTAS DE DESARROLLO Y SUS VALORES			
DESCRIPCIÓN DE LA HERRAMIENTA	**CANTIDAD**	**VALOR UNITARIO**	**TOTAL**
1) Hardware ya existente			
Computadores	**15**	**-**	**-**
Impresoras	**4**	**-**	**-**
Switch	**1**	**-**	**-**
Cableado Instalado y puntos de red	**-**	**-**	**-**
2) Hardware a adquirir			
Servidor, con procesador Pentium 4 HT de 3500 mhz,1 GB memoria RAM, 200 GB. Disco Duro SCSI, Tarjeta de red ethernet 1000 BaseT 10/100/1000.	**1**	**850.000**	**850.000**
UPS de 1200 watts	**1**	**220.000**	**220.000**
Switch de 6 bocas	**1**	**35.000**	**35.000**
Cable de red UTP 10 baseT	**100 mts**	**230**	**23.000**
Conectores RJ45	**50**	**85**	**4.250**
Capuchones RJ45	**50**	**65**	**3.250**
3) Software a Adquirir			
Microsoft SQL Server 7.0	**1**	**720.000**	**720.000**
Windows 2003 Server	**1**	**517.000**	**517.000**
Macromedia Dreamweaver MX	**1**	**400.000**	**400.000**
Macromedia Flash MX	**1**	**690.000**	**690.000**
Microsoft ASP	**1**	**330.000**	**330.000**
4) Licencias a Adquirir			
Licencia Microsoft SQL Server 7.0 incluida con el software	**1**	**-**	**-**
Licencia Windows 2003 Server incluidas en el software	**5**	**-**	**-**
Licencia Windows 2003 Server adicionales	**10**	**55.000**	**550.00**
Licencia Macromedia Dreamweaver MX	**1**	**-**	**-**
Licencia Macromedia Flash MX incluida en el software	**1**	**-**	**-**
Licencia Microsoft ASP incluida en el SW	**1**	**-**	**-**
5) Recursos Humanos			
Ingeniero Informático (sueldo x 8 meses)	**1**	**850.000**	**6.800.000**
Analista Programador (sueldo x 7 meses)	**1**	**400.000**	**2.800.000**
Encargado mantención del Servidor por parte del usuario	**2**	**-**	**-**
6) 0tros Recursos			
Pieza del servidor de 2 x 3 metros por parte del usuario	**1**	**-**	**-**
VALOR TOTAL			**13.942.500**

Factibilidad Legal:

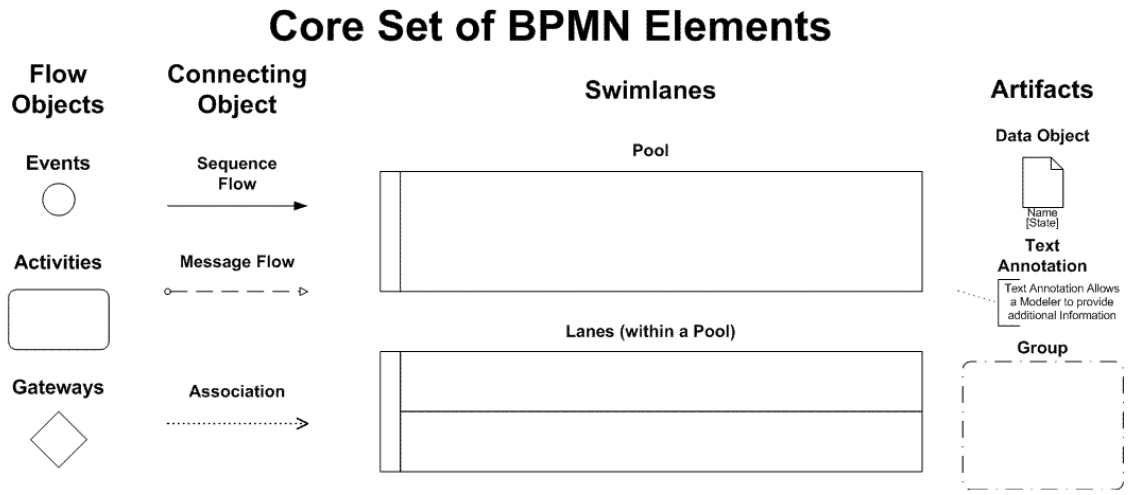
Al igual que en la factibilidad anterior, la empresa actualmente no cuenta con ningún tipo de licencias o permisos de software para la utilización de algún programa o aplicación particular, sólo las licencias que están incluidas en el proyecto Enlaces, las que subvenciona el gobierno, las cuales son todas las licencias del sistema operativo actual (Windows 98), Office 2000, y una casilla de correo.

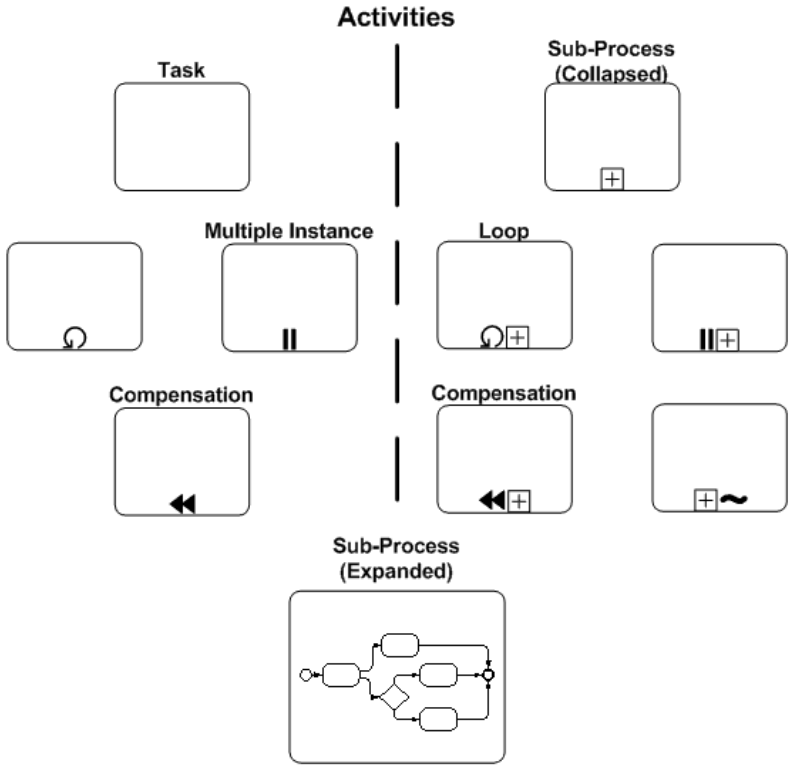
Para implantar el sistema se necesita de las siguientes licencias de software:

- Por la compra del software Windows 2003 Server se incluye 5 licencias para 5 equipos.
- Para 10 equipos restantes se necesita ampliar la licencia del servidor en 10 equipos más.
- Para el motor de base de datos, se necesita una licencia Microsoft SQL Server para 15 usuarios.
- Para diseñar el ambiente Web, se requiere de una licencia Macromedia Dreamweaver MX y Macromedia Flash MX, además de la licencia para utilizar Microsoft ASP.

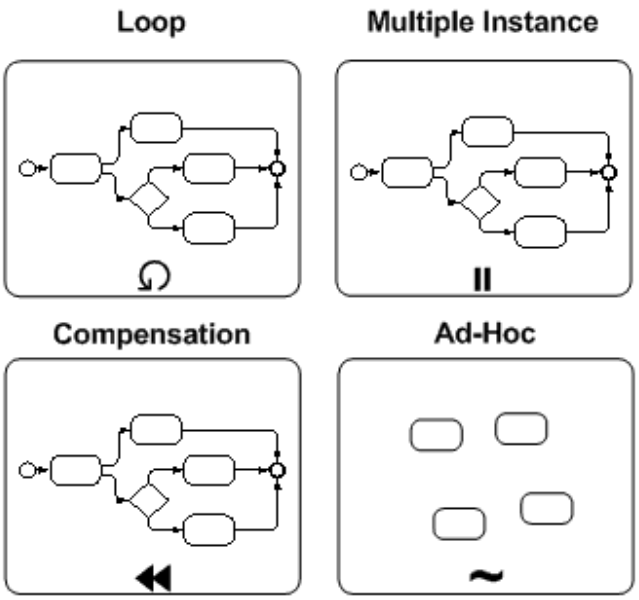
ANEXO III

BPMN Business Process Modeling Notation[11]



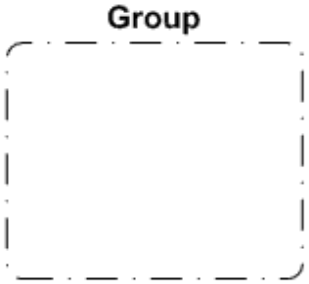


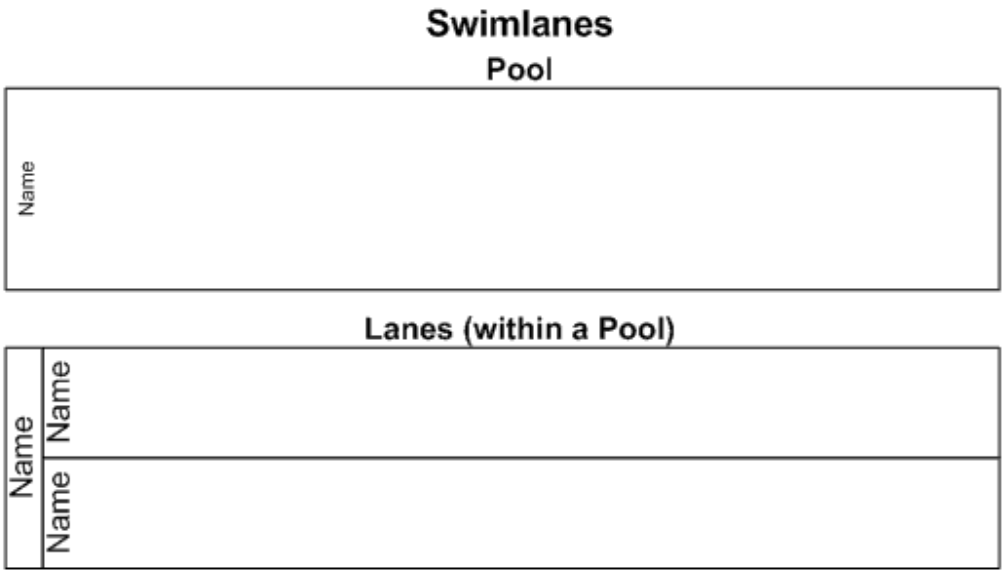
[11] http:// www.bmpn.org – http://bpmn.itposter.net



Atributos especiales de Tarea/Subproceso		
Looping	↺	La tarea o sub-proceso es repetida.
Ad Hoc	~	Las tareas en el sub-proceso no pueden ser conectadas con la secuencia en tiempo de diseño.
Instancias múltiples	‖	Múltiples instancias de una tarea o sub-proceso pueden ser creadas.
Compensación	◀◀	El símbolo representa la compensación de la tarea o subproceso.

Secuencia de flujo Normal	Name, Condition, Code, or Message	Una secuencia de flujo es usada para mostrar el orden en el que las actividades del proceso son realizadas.
Secuencia de flujo condicional	Name, Condition, or Code	Una secuencia de flujo puede tener expresiones de condición que son evaluadas en tiempo de ejecución para determinar si o no el flujo será usado
Secuencia de flujo predeterminado	Name or Default	Para decisiones exclusivas basadas en datos o decisiones inclusivas, un tipo de flujo es el de condición predeterminada de flujo. Este flujo solo puede ser ocupado si todas las otras salidas de flujo condicionales no son verdaderas en tiempo de ejecución.
Flujo de mensajes	Name or Message	Un flujo de mensaje es usado para mostrar el flujo entre dos participantes que son preparados para enviar y recibir los mensajes. En BPMN, Dos pool separados en un diagrama pueden representar dos participantes.
Asociación		Una asociación (Directa, indirecta) es usada para asociar información con flujos de objetos. Textos y gráficos que no son objetos de flujo no pueden ser asociados con objetos de flujo

Artefactos (Artifacts)		
Objeto de datos		Objetos de datos dan información sobre qué actividades se requiere que sean gatilladas y/o que es lo que ellas producen. Ellas son consideradas, ya que no tienen ningún efecto directo en la secuencia del flujo o flujo de mensajes del proceso. El estado del objeto de datos debe ser también establecido
Grupo		El agrupamiento puede ser usado para la documentación o por propósitos de análisis. Los grupos pueden también ser usados para identificar actividades de una transacción distribuida que es mostrada a través de Pools. El agrupamiento no afecta la secuencia o flujo de mensajes.
Comentario		Comentarios son un mecanismo del modelador para dar mayor información a los lectores de diagrama BPMN.



Eventos(Events): Un evento es algo que »pasa o sucede« durante el proceso. Este evento afecta el flujo del proceso y usualmente tiene una causa (algo que lo gatilla) y un impacto (resultado).
Ejemplos: 'Email recibido', '3:00 en punto', 'Deposito Vacio', 'Error Critico',...

	Inicio	Intermedio	Fin	
General				El evento inicial indica donde un proceso en particular empezará. Los eventos intermedios ocurren entre un evento inicial y un evento final. El afectará el flujo del proceso, pero no empezará o (directamente) terminará el proceso. El evento final indica donde el proceso termina.
Mensaje				Un mensaje llega de un participante y activa el evento. Esto causa que el proceso {comience, continúe, Termine} donde está esperando por mensajes o cambie el flujo si la excepción se realiza. Eventos de mensaje de fin indican que un mensaje es enviado al termino del proceso
Tiempo				Un tiempo especifico o ciclo puede establecer que se gatillará el inicio de un proceso o continuar con el proceso. Eventos intermedios de tiempo pueden ser usados para modelar lo retrasos en tiempo que se tengan.
Error				Este tipo de fin indica que el nombre del error debiera ser generado. Este error será capturado por un evento intermedio dentro del contexto del evento.
Cancelados				Este tipo de evento es usado dentro de un sub-proceso de transacción. Este tipo de evento DEBE estar adjunto a la frontera del sub-proceso. El deberá ser activado si un evento final de cancelación es alcanzado dentro de la transacción del sub-proceso.
Comprensión				Estos son usados para manipular la compensación—Ambos establecen y realizan compensación. La compensación es llamada si el evento es pate del flujo normal. Esta reacciona al llamado del nombre de la compensación cuando está adjuntada a la frontera de la actividad. Muy útil para revertir las acciones del modelado que están dentro de la transacción.
Reglas				Este tipo de evento es activado, cuando las condiciones de una regla se convierta en verdadera. Las Reglas pueden ser muy útiles para interrumpir el ciclo del proceso, por ejemplo: 'El número de repeticiones= N'. La regla intermedia es usada solo para manipular la excepción
Enlaces				Un enlace, es un mecanismo para conectar el fin (resultado) de un proceso a el inicio de otro. Típicamente, estos son 2 subprocesos dentro de proceso padre, puede ser usado, por ejemplo, cuando el área de trabajo (pagina) es pequeña – va a otra página.
Termino				Este tipo de fin indica que todas las actividades en el proceso deben estar inmediatamente terminadas. Esto incluye todas las instancias o multiinstancias. El proceso es terminado sin compensación o evento de Manipulación.
Múltiple				Este tipo de evento indica que hay múltiples formas de gatillar el proceso. Solo una de ellas será requerida para {empezar, continuar, finalizar} el proceso.

Compuertas (Gateways)	
Exclusive Decision/Merge (XOR) Data-Based or Name	Decisión exclusiva basada en datos o fusionada. Ambos símbolos tienen igual significado. Ver también flujo condicional
Event-Based	Evento basado solamente en una decisión exclusiva.
Inclusive Decision/Merge (OR)	Decisión inclusiva basada en datos o fusionada.
Complex Decision/Merge	Condición compleja (Una combinación de condiciones básicas)
Parallel Fork/Join (AND)	Ramificación paralela y unión (sincronización).

Ejemplos de notación utilizada

Imagen 1

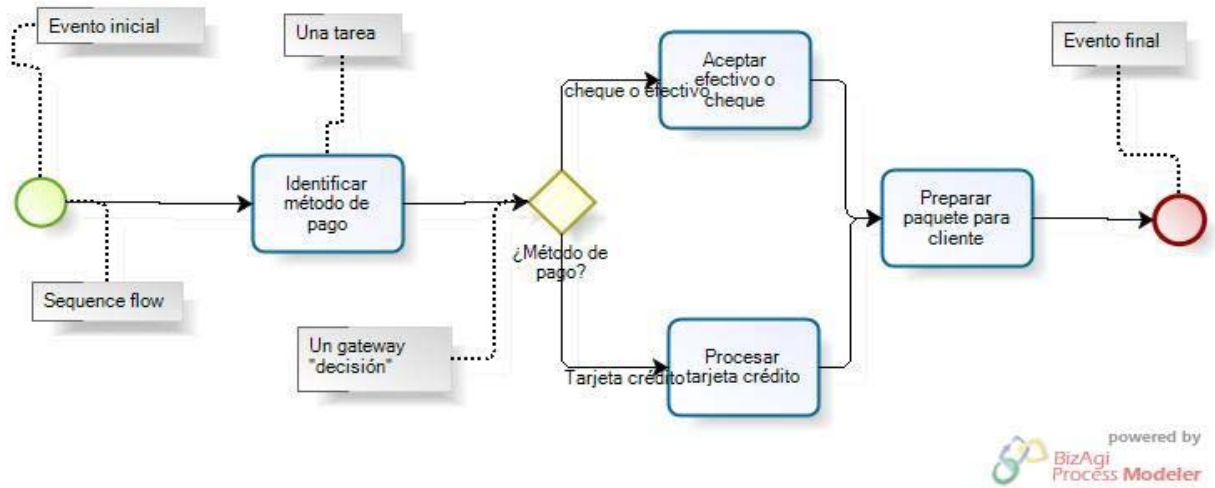


Imagen 2

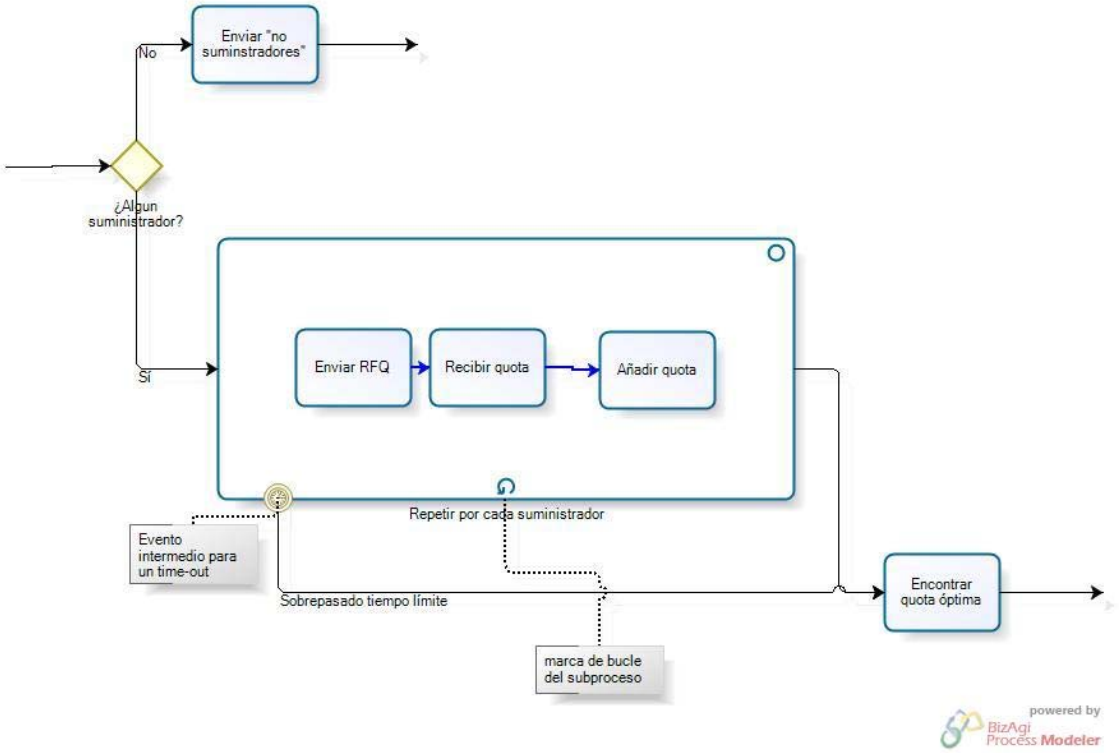


Imagen 3

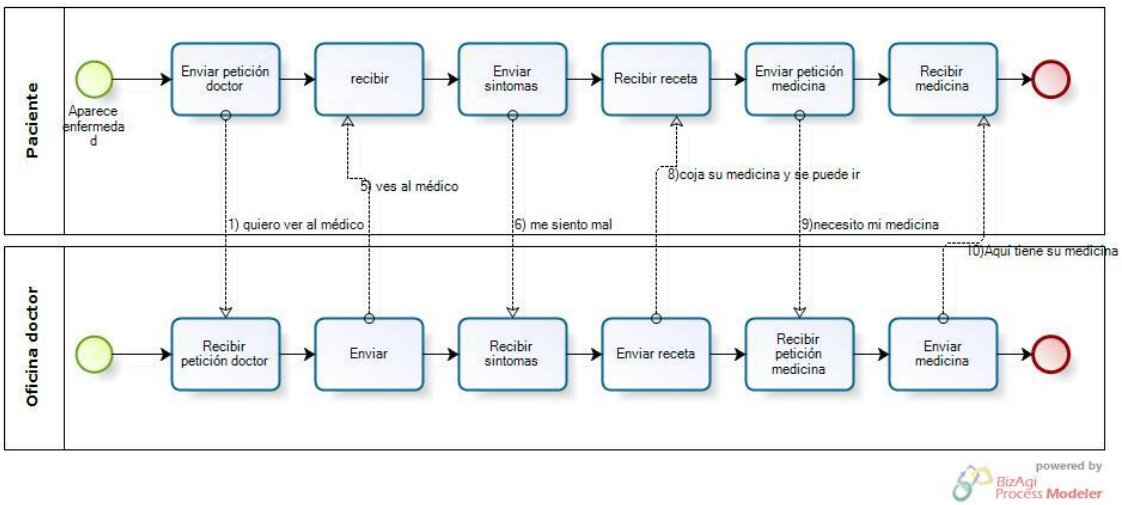


Imagen 4

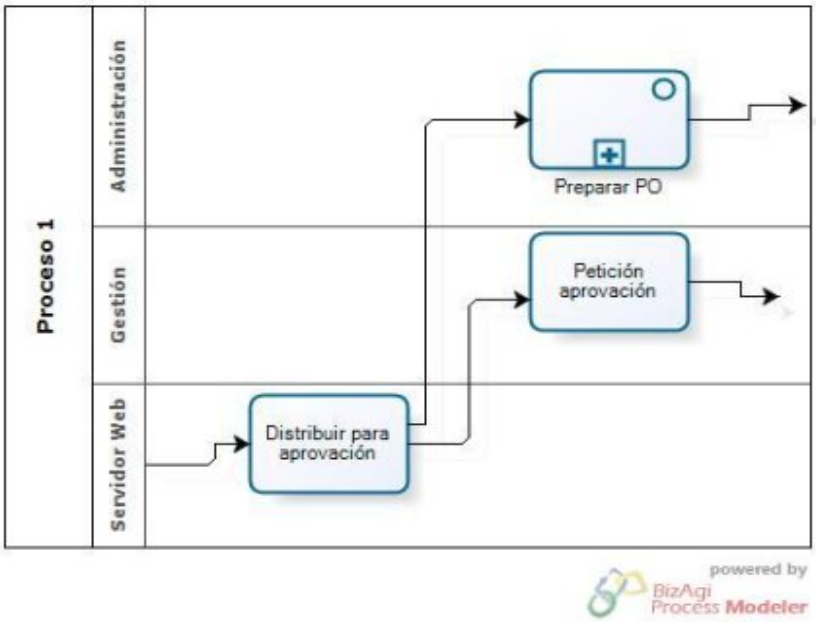


Imagen 5

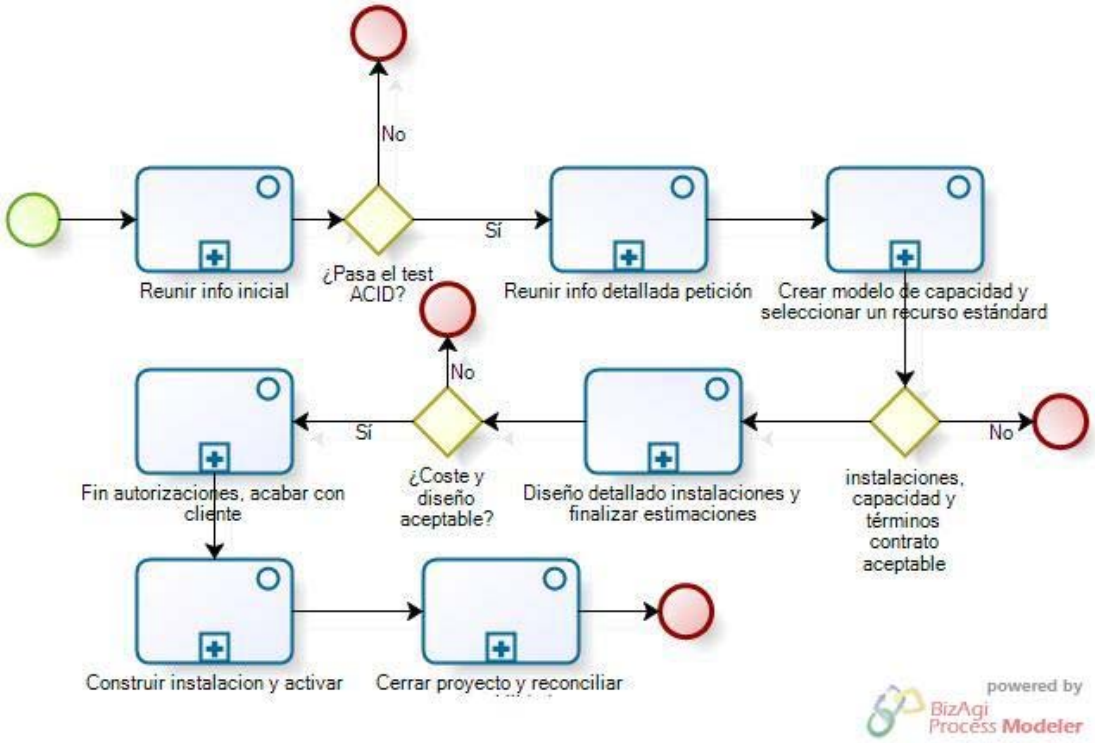


Imagen 6

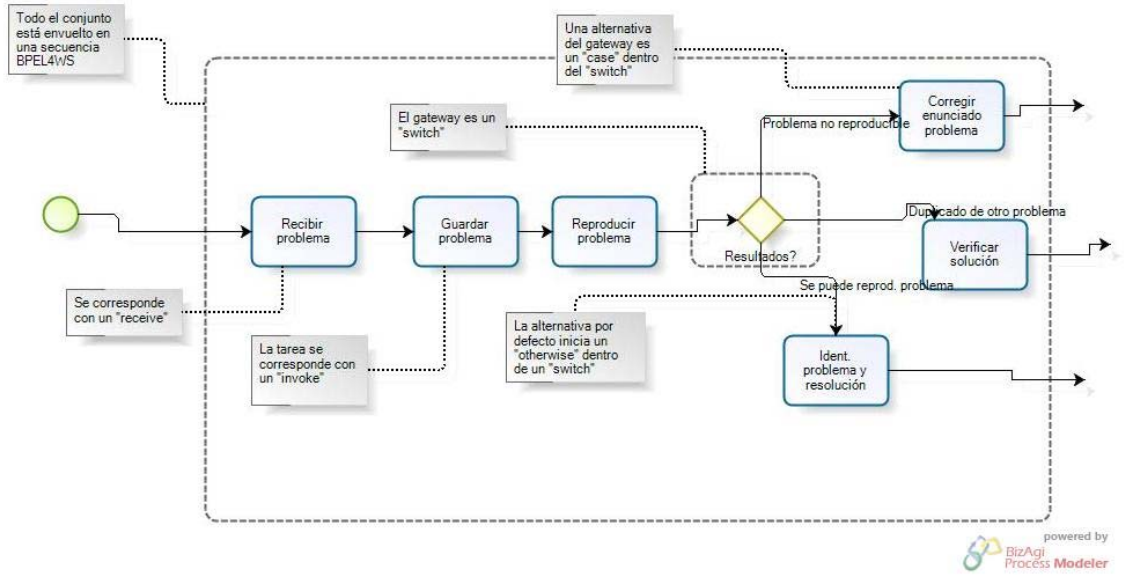


Imagen 7

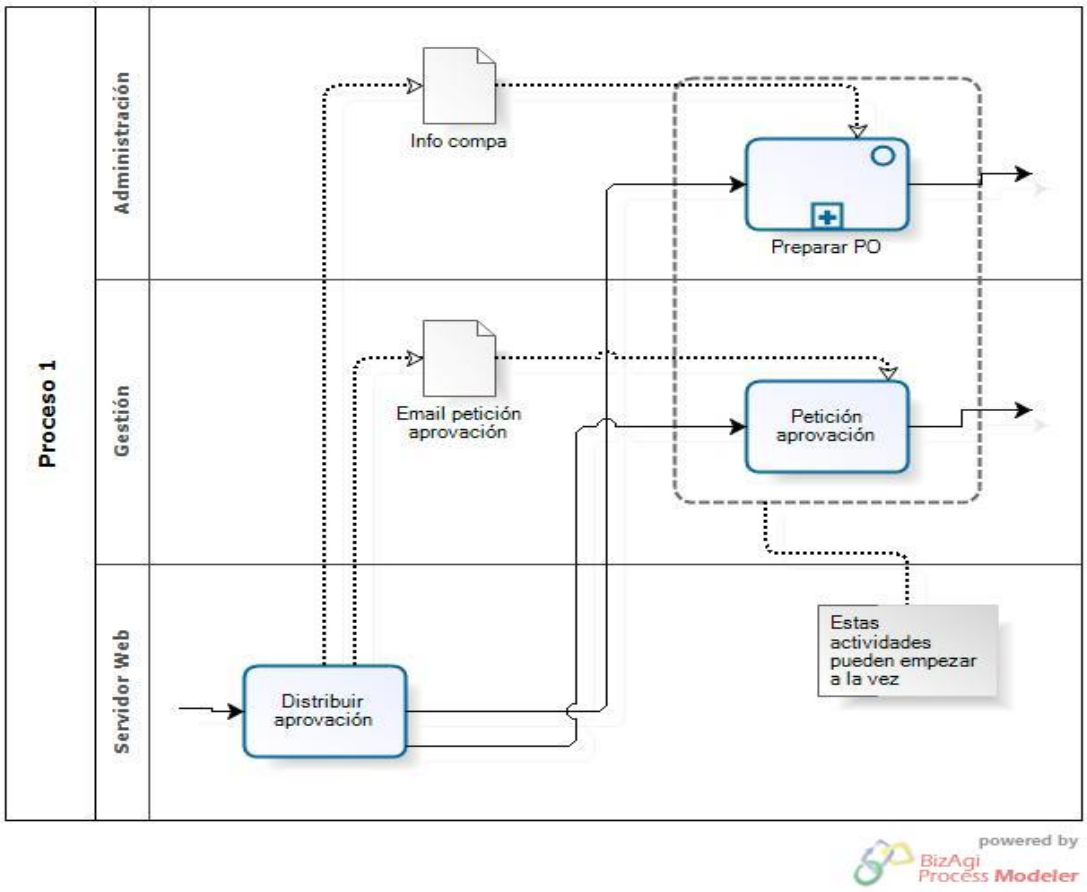
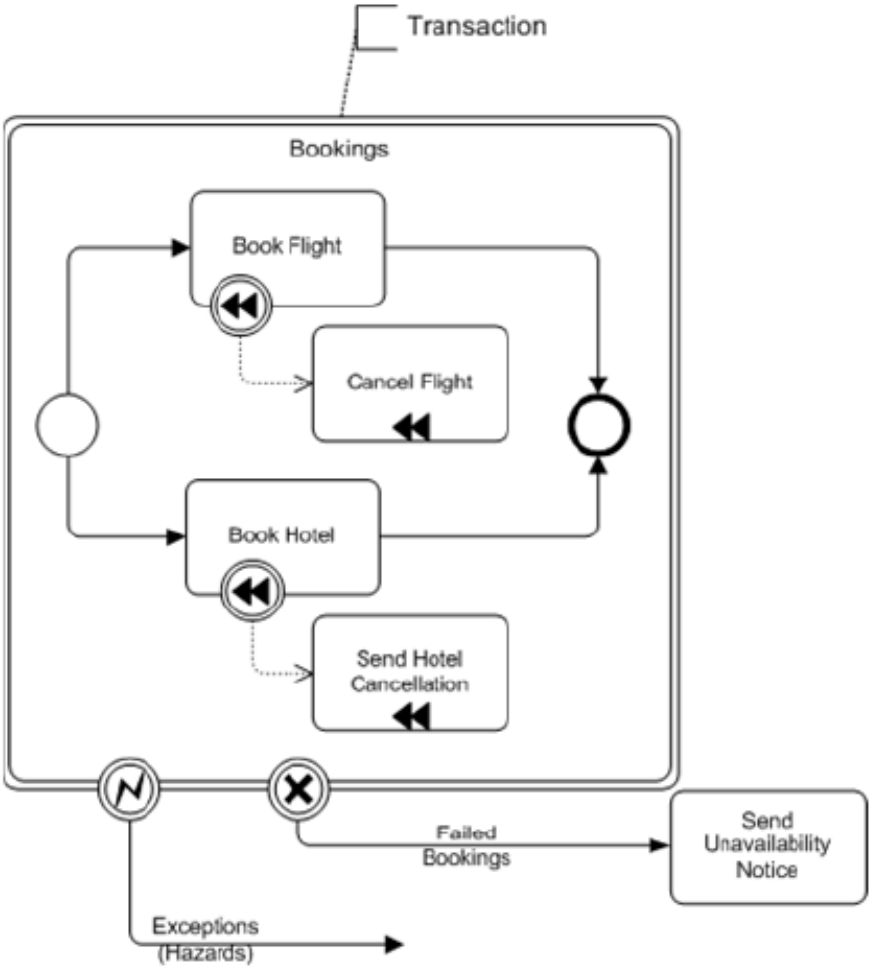


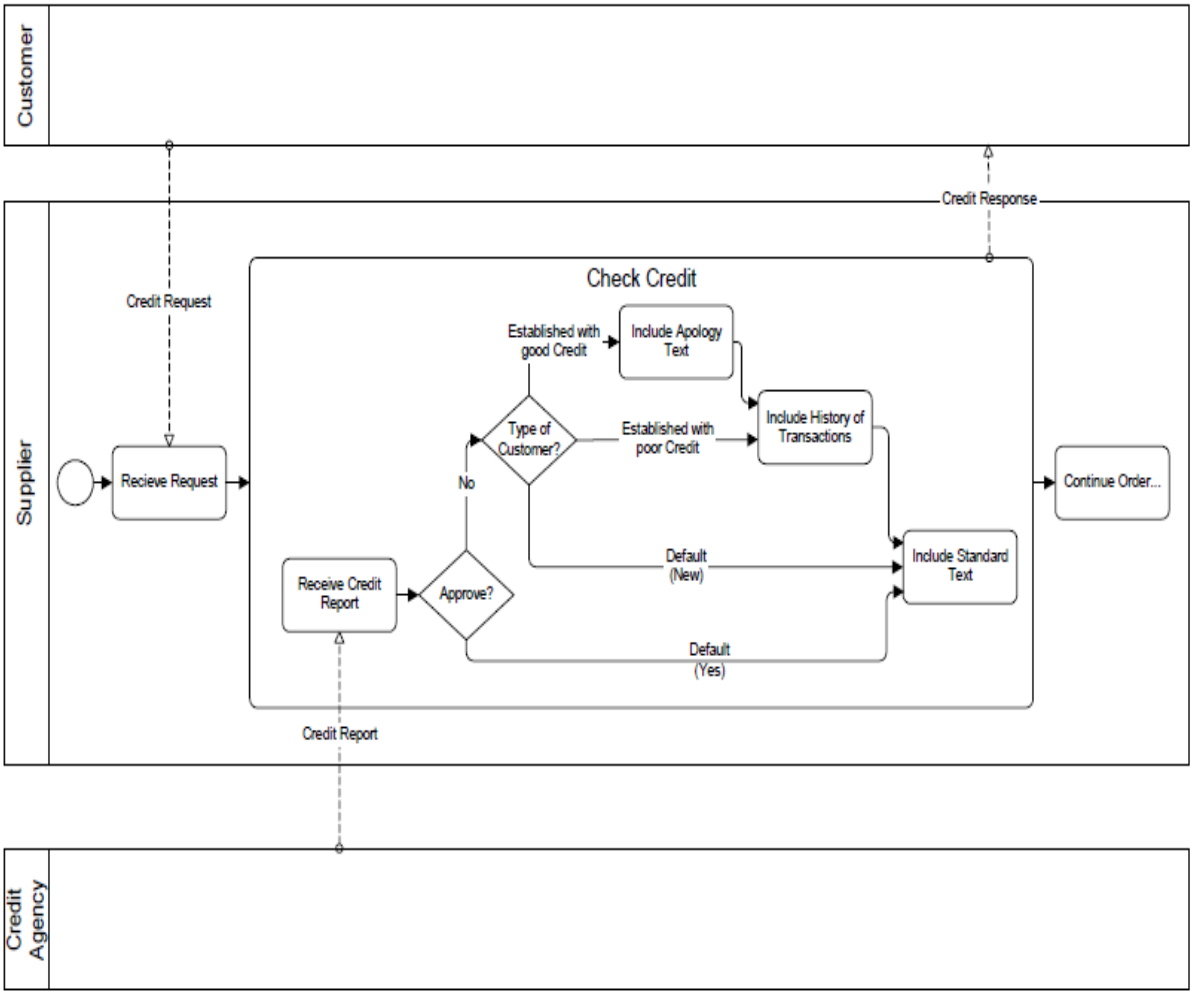
Imagen 8[12]



[12] Business Process Modeling Notation (MPMN) Version 1.0

Imagen 9[13]

Ejemplo de cómo se conecta el límite del subproceso y los objetos internos.



[13] Business Process Modeling Notation (MPMN) Version 1.0

Imagen 10

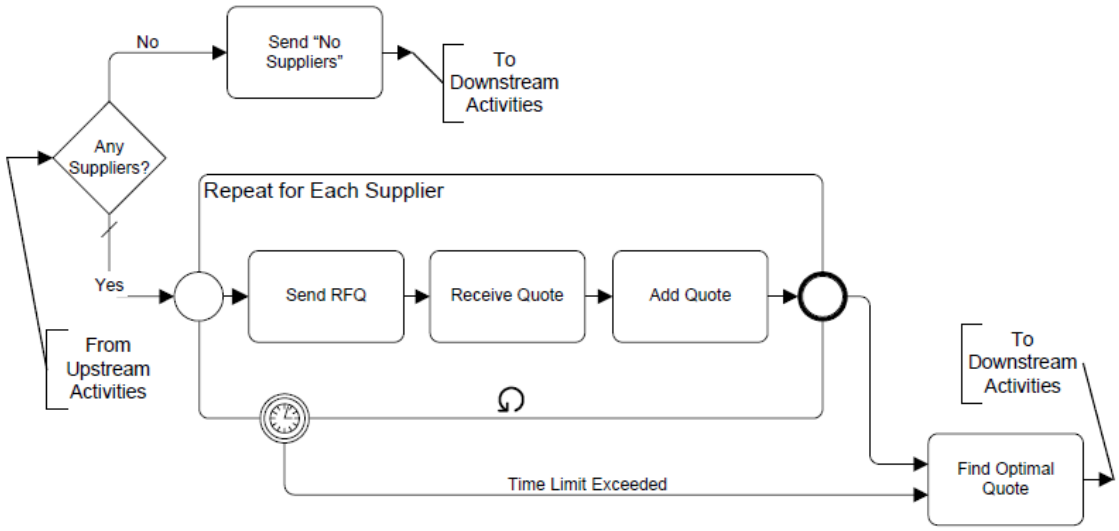
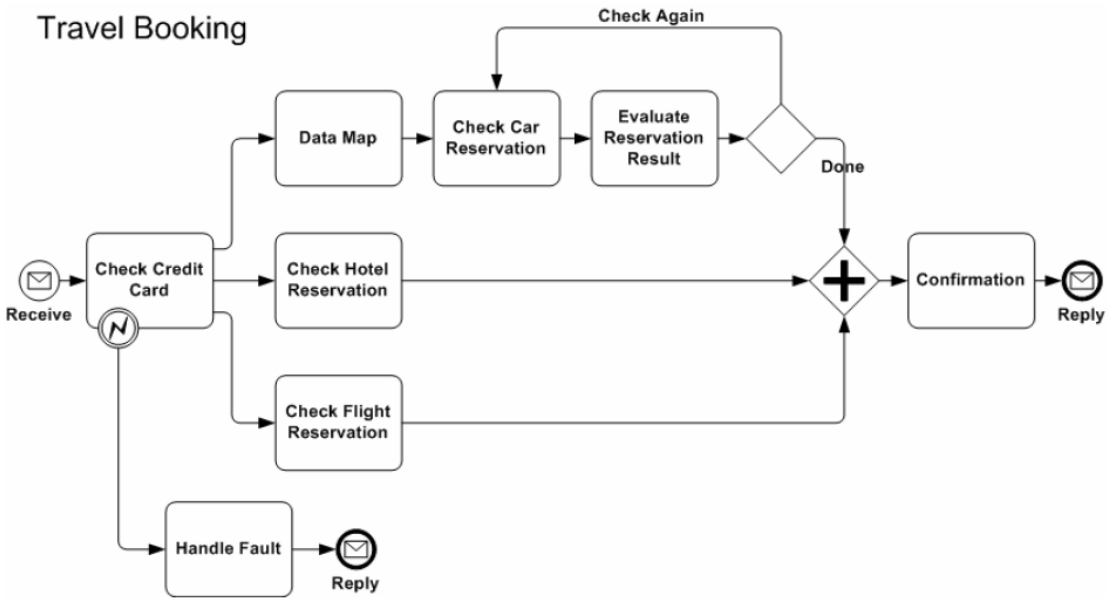
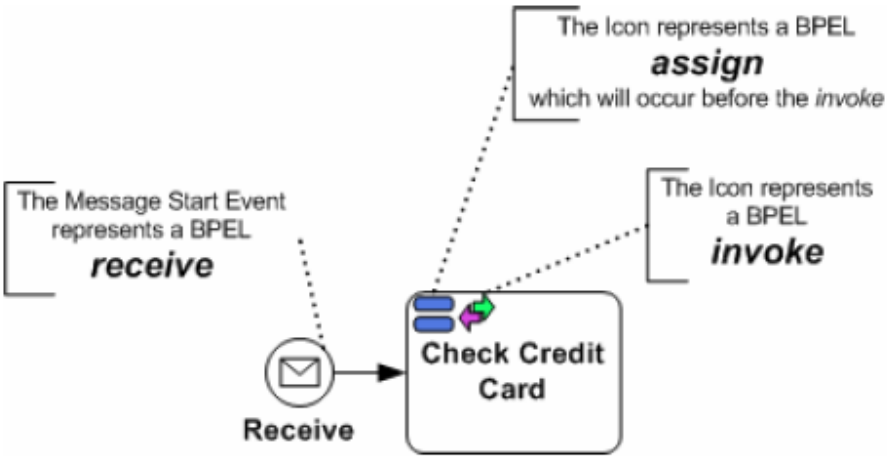


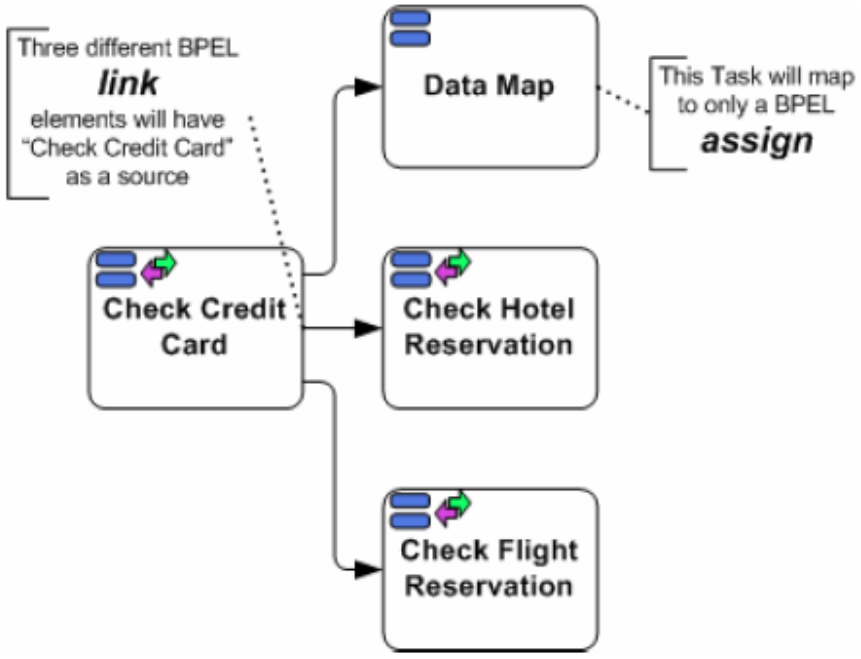
Imagen 11[14]



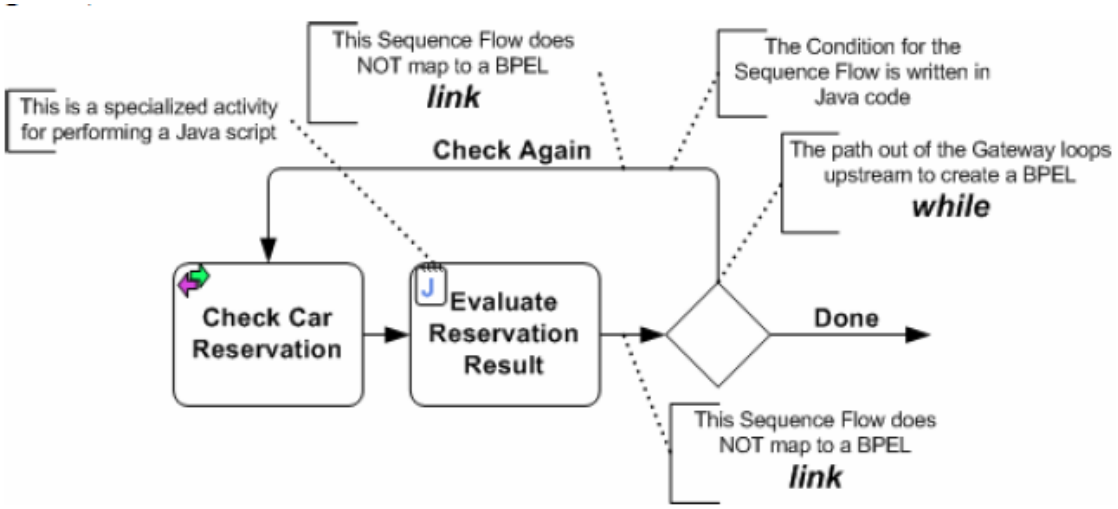
[14] Using BPMN to Model a BPEL Process Stephen A. White, IBM Corp., United States



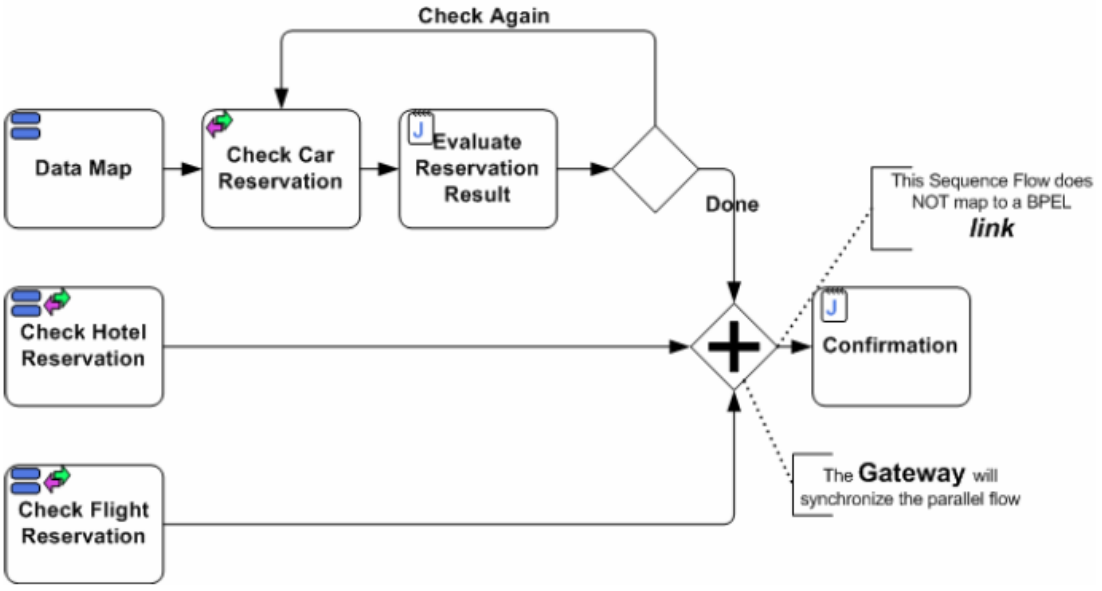
The Beginning of the Travel Booking Process



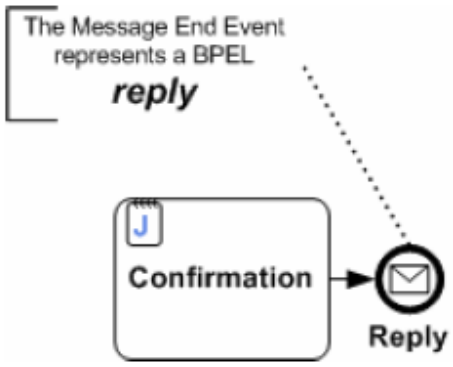
Parallel Flow within the Process



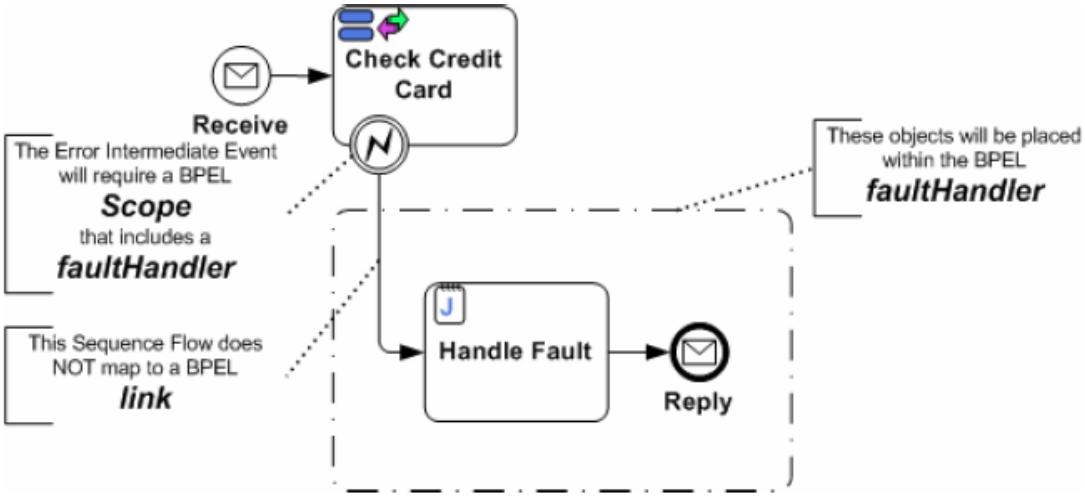
A Loop within the Process



Flow Synchronization within the Process



The Conclusion of the Process

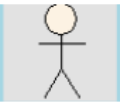
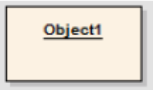
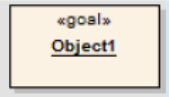


Fault Handling for the Process

ANEXO IV

Modelado de Proceso Según Ericksson & Penker [15]

Este modelo se encuentra dentro de la vista de procesos, y tiene como objetivo modelar un proceso identificando que evento o eventos lo provocan, que información lo apoya, que recursos necesita, a que meta da soporte y que resultados se obtienen.

Objeto	Imagen	Descripción
Actor		Rol (Interno o Externo) o Organización que genera el evento inicial que da comienzo al proceso.
Evento		Evento que genera el comienzo del proceso.
Proceso		Representa el proceso de negocio.
Información, Recursos y Salidas		**Información:** representa información o datos que serán utilizados en el proceso, pero no participan en la transformación realizada en él. La asociación con el proceso se representa con una línea bidireccional con stereotype <<supply>>. **Recurso:** representa cosas que serán utilizados en el proceso, y participan en la transformación realizada en el proceso. La asociación con el proceso se representa con una línea bidireccional con stereotype <<input>>. **Salidas:** representan los resultados obtenidos con el proceso. La asociación con el proceso se representa con una línea unidireccional con stereotype <<outcome>>.
Meta u Objetivo		Representa las metas que son soportadas por el proceso negocio.

[15] Modelo de Procesos de Negocio con UML: Evaluación de un caso real, Sergio Sánchez

Ejemplos uso Nomenclatura