ESCUELA POLITECNICA DEL EJÉRCITO

SEDE LATACUNGA

FACULTAD DE INGENIERIA DE SISTEMAS E INFORMATICA

DESARROLLO DE UNA HERRAMIENTA PARA LA GESTION DE SEGUIMIENTOS DE PROCESOS

PROYECTO PREVIO A LA OBTENCIÓN DEL TITULO DE INGENIERO EN SISTEMAS E INFORMATICA

PAULINA DEL ROSARIO SILVA ORTEGA

MARITZA DE LAS MERCEDES VILLAMARIN RODRÍGUEZ

Latacunga, Febrero del 2006

CERTIFICACION

Se certifica que el presente trabajo fue desarrollado por Paulina Silva O. y Maritza Villamarín R, bajo nuestra supervisión.

_____

Ing. Edison Espinosa
DIRECTOR DEL PROYECTO

_____

Ing. Santiago Jácome
CODIRECTOR DE PROYECTO

I. WORKFLOW

1.1.- INTRODUCCIÓN

A partir de la llegada de las computadoras personales al ambiente empresarial se inició una nueva revolución. Uno de los cambios más drásticos fue el incremento en la velocidad de procesamiento de los datos.

Desde los años 60s se tenía la necesidad de compartir recursos de cómputo, como la memoria, las unidades de almacenamiento y principalmente el procesador; pero no se compartía la información, ya que existían diferentes barreras que lo impedían, como las distancias entre oficinas o los diferentes sistemas operativos.

En cualquier organización empresarial existen gran cantidad de procesos desarrollados por agentes ya sean estas personas, máquinas, etc. Distintas empresas poseen procesos que requieren un tratamiento muy específico y tienen el objetivo de dar un servicio de calidad y mantener un alto grado de satisfacción en sus clientes.

Con la llegada de Internet algunas barreras se rompieron, con este avance tecnológico se logró compartir información pero no se podían realizar las actividades que necesitan colaboración, a partir de esta necesidad se comenzó a trabajar en un nuevo tipo de tecnología: el software colaborativo.

Dentro de este tipo de software se encuentra: el Workflow, ya que estas tecnologías son comúnmente utilizadas dentro del ámbito empresarial.

En la actualidad la tecnología workflow y los sistemas de gestión workflow (WFMS) se derivan de varias disciplinas entre las que cabe destacar CSCW (Trabajo Cooperativo Soportado por Ordenador) y OIS (Sistemas de Información en la Oficina).

Workflow incluye un conjunto de soluciones tecnológicas que permiten automatizar los procesos de trabajo desarrollados en una empresa u organización. Su implantación es adecuada en cualquier tipo de entorno, usándose principalmente para:

- Coordinar y gestionar el trabajo.
- Encaminar automáticamente los procesos de trabajo
- Controlar, seguir y administrar a los procesos de trabajo, consiguiéndose
- Una visión global del estado del proceso completo,
- Mejorar los procesos de trabajo y conseguir un servicio más rápido y de mayor calidad,
- Acelerar los ciclos de desarrollo de productos y procesos.

Finalmente los sistemas Workflow han ido adquiriendo mayor importancia en empresas de todos los sectores ya que estas consideran que los recursos bien integrados y organizados, que a su vez permitan mayor agilidad, son las que hacen a las organizaciones ser más competitivas en el mercado.

1.2.- QUÉ ES WORKFLOW

Antes de definir lo que es Workflow debemos de tener una definición clara de qué es un Proceso de Negocio:

"Un proceso es un orden específico de actividades de trabajo, que se realizan en el tiempo, en lugares específicos y por personas o sistemas, con un principio, un fin, y entradas y salidas claramente definidas. Es decir, una estructura cohesionada y coordinada adecuadamente para la acción.**"**

Existen numerosas definiciones de Workflow, sin embargo, en todas ellas existen puntos en común: Un sistema de Workflow es un sistema informático que ofrece herramientas para el análisis de procesos administrativos y de negocio,

automatiza la ejecución de los procesos al mismo tiempo que integra las herramientas de oficina usadas.

La WfMC (Workflow Management Coalition) define a los workflows como:

"El flujo de procesos administrativos o de negocios, es el **conjunto de actividades** o tareas realizadas **en secuencia o en paralelo** por **dos o más miembros** de un equipo de trabajo para lograr **un objetivo común** siguiendo unas **reglas de negocio** preestablecidas."

Se destaca:

Conjunto de actividades: Se refiere a la amplia gama de actividades relacionadas con el negocio y su administración

En secuencia o en paralelo: Quiere decir que las actividades pueden ser realizadas una detrás de la otra o simultáneamente por individuos diferentes o una combinación de ambos

Más de dos miembros: Si una sola persona realiza la tarea, no realiza workflow. Como su nombre lo sugiere, una actividad es workflow si "fluye" de un individuo a otro.

Objetivo común: Los individuos que participan en un flujo de trabajo deben estar trabajando para lograr un objetivo común; si trabajan en proyectos independientes, no se constituye un workflow

Reglas de Negocio: Si un proceso no sigue unas reglas y ruta preestablecidas, no se trata de workflow, sino de un proceso de colaboración "ad hoc"

El Workflow es el último, de una gran línea de facilidades propuestas en respuesta de las exigencias de las empresas. Las cuales apuntan a poder reaccionar tan rápido como sea posible ante la gran demanda de la competición.

Como se mencionó anteriormente, un workflow es el control de los de procesos de negocio. Para poder identificar cada elemento dentro de cada workflow se puede utilizar los modelos de proceso.

En la **figura 1.1**, se puede observar los elementos que forman a un proceso.

Proceso de Negocio

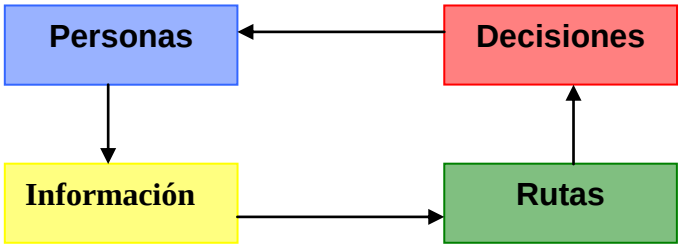


Figura 1.1 Elementos claves en un proceso de Negocio

1) Personas.- Es el componente más importante en procesos de workflow. Son los únicos que pueden crear contenido, tomar decisiones, delegar trabajos y supervisar el cumplimiento de un trabajo.

2) Decisiones.- Aquí se definen que cosas se deberán hacer, como deberá hacerse y quienes deberán hacerlo.

3) Información.- Es todos aquellos asuntos técnicos que definirán en que contexto el trabajo será hecho. Esta información permite que ciertos pasos sean implementados rápidamente

4) Rutas.- Define los caminos que siguen los objetos en una organización. En cuanto a objetos se debe entender: documentos, información, formularios, etc.

Estos cuatro elementos clave forman parte de los componentes de un proceso de negocios y por lo tanto de un workflow. Para identificar estos componentes claves dentro de un proceso, es necesario formularse las siguientes preguntas: ¿Qué rutas se siguen?, ¿Qué gente participa?, ¿Cuál es el rol que juega cada participante?, ¿Qué decisiones son tomadas?, Cómo se llevan a cabo estas decisiones?, ¿Qué información es requerida por cada participante? Estas preguntas son indispensables para poder identificar correctamente los procesos de la empresa que pueden ser mejorados e implementados a través de un workflow.

Uno de los problemas que se encuentra habitualmente en el desarrollo de aplicaciones para empresas, es que las tareas o procesos que se desarrollan en el entorno laboral de las mismas quedan inmersos en el código de la aplicación que resuelve la problemática de la empresa. Está claro que la gran mayoría de los usuarios no tienen conocimiento de estas tareas, las mismas están ocultas a sus ojos y se realizan automáticamente.

El hecho de realizar cambios en dichas áreas o procesos resulta muy costoso, es muy probable que estos cambios puedan resultar una nueva aplicación. Una buena solución al problema anterior es separar los procedimientos y asociarlos a los procesos de trabajo realizados dentro de la empresa.

Vemos entonces, que el Workflow se relaciona con la automatización de los procesos donde los documentos, la información o tareas son pasadas entre los integrantes del sistema de acuerdo a un conjunto de reglas previamente establecidas. En que el objetivo es llegar a culminar una meta común impuesta por la empresa.

Podemos ver al Workflow como un conjunto de métodos y tecnologías que nos ofrece las facilidades para **modelar y gestionar** los diversos procesos que ocurren dentro de una empresa.

Cabe mencionar que los workflows son sólo un camino para la información, para reducir tiempo, dinero y esfuerzo en la ejecución de un proceso de negocio.

En cuanto a las principales funcionalidades que Workflow provee, tenemos:

- Asignar actividades a las personas de forma automática y según cualquier criterio, o según cargas de trabajo.
- Recordar a las personas sus actividades, las cuales son parte de una cola de Workflow.
- Optimizar la colaboración entre personas que comparten actividades.
- Automatizar y controlar el flujo de procesos
- Asignarle previamente a las personas que deben ejecutar las actividades, todos los recursos necesarios Documentos, información, Aplicaciones, etc.) en cada una de ellas.
- Definir y proveer "alertas" según criterios de tiempo, condición, provocando así algún mensaje a un supervisor, un "escalado" de actividades a otras personas para que través de Email, SMS
- Proveer un alto nivel de soporte para la interacción humana

Servicios adicionales de un Workflow incluye el acceso a datos estadísticos, así como cooperación con otras herramientas (Figura 2)

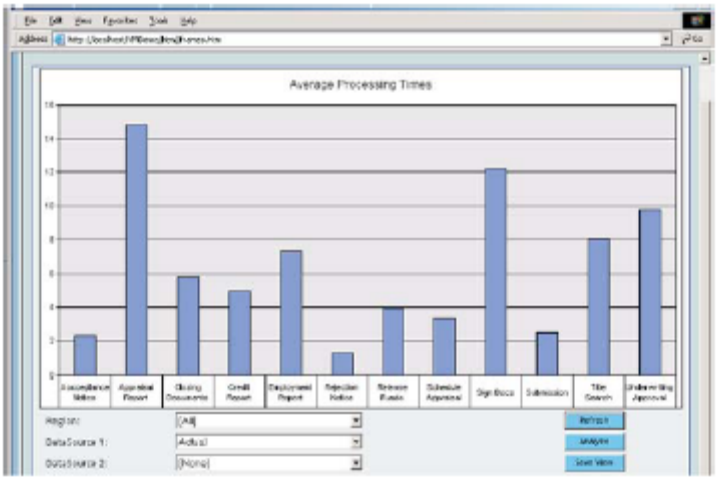


Figura 1.2 .Ejemplo Tiempo Promedio de Procesos

En general, los sistemas de Workflow ofrecen la base para la integración de todas las tecnologías usadas en el trabajo de oficina.

Los primeros esfuerzos en integración de soluciones para oficinas vinieron como una evolución de los sistemas de gestión documental. Ya en 1991 los sistemas de Workflow tenían claras ramificaciones fuera de los sistemas de gestión documental y fue cuando los primeros Workflow aparecieron en el mercado. En los últimos años muchos productos han aparecido, pero los principios esenciales son los mismos: cliente-servidor, interfaces gráficas, integración con el entorno de oficinas, editores gráficos para definición de procesos, etc.

1.3.- TIPOS DE WORKFLOW

Para muchos el Workflow es un área difícil de entender, por este motivo se han dividido los diferentes tipos de aplicaciones de Workflow en categorías.
La diversidad de **procesos de negocios** existentes explica porque el mercado de Workflow ha sido dividido en varios segmentos según el **valor del proceso a manejar** y si este proceso es **repetitivo** o no. Un proceso tiene un valor alto si representa grandes ahorros a una empresa o la relación costo-oportunidad es buena, en definitiva un proceso tiene alto valor si redunda en grandes beneficios para una empresa.

Por otra parte se estima cuan repetitivo es un proceso. Un proceso es repetitivo si cada instancia del Workflow sigue ciertas reglas, ciertos patrones que son similares para toda instancia del proceso. Por el contrario un proceso no es repetitivo si cada instancia es relativamente única. Teniendo en cuenta las consideraciones anteriores, actualmente encontramos tres tipos.

Existen tres tipos diferentes de aplicaciones de workflow:

- Workflow de Producción
- Workflow Colaborativo

- Workflow Administrativo

  En los siguientes puntos se describen dichos tipos de Workflow.

1.3.1.- WORKFLOW DE PRODUCCIÓN

Frecuentemente este tipo de Workflow es llamado Workflow de Transacciones. Esto se debe a que en este tipo, la *transacción* en una *base de datos* es considerada la clave de todo proceso.

En las aplicaciones de workflow de producción, el workflow es la tarea principal de los participantes. Dicho personal puede tener actividades adicionales en su trabajo diario, pero fundamentalmente la realización de workflow, por ejemplo: tramitar solicitudes de crédito.

El workflow de producción es similar a la producción en una línea de ensamble en una fábrica: Debe ejecutarse en el menor tiempo posible, es altamente predecible, repetitivo y de alto volumen. Los trabajadores en la línea de ensamble pasan su mayor parte del tiempo produciendo objetos; pueden participar en actividades adicionales, pero ellas son secundarias.

Debe notarse además que el workflow de producción se suele ajustar a un sólo departamento de la empresa.

En un banco por ejemplo, los individuos a cargo de la aprobación de solicitudes de crédito sólo realizan workflow para esa actividad es improbable que otros funcionarios del banco realicen esa actividad fuera del departamento.

Debido a su naturaleza de "producción", dichas aplicaciones deben cumplir con algunos de los siguientes atributos:

1. Velocidad de transferencia, o sea, la velocidad con que las tareas pasan de un paso a otro. Es muy importante en el workflow de producción, ya que es la tarea

principal de los participantes. Es improductivo que un miembro del equipo no haga nada mientras espera a que le llegue trabajo.

2. La flexibilidad de poder cambiar el proceso no suele ser importante. Una vez establecido el flujo, este permanece sin cambio por largo tiempo.

El workflow de producción suele estar circunscrito a un sólo departamento, la escalabilidad, o capacidad de "crecer" no es importante. Este tipo de soluciones están optimizadas para trasladar grandes volúmenes de información e imágenes a lo largo de rutas preestablecidas.

El workflow de producción fue el primer tipo de workflow desarrollado y mercadeado, esto, porque generalmente no se requería de una base distribuida de usuarios a lo largo de la compañía para lo que es indispensable contar con una red local (Lan).

1.3.2.- WORKFLOW COLABORATIVO

Involucra procesos estructurados y semi-estructurados que permiten a varias personas participar en un grupo de trabajo, ejemplos de ello lo constituyen el diseño arquitectónico o ingenieril, generación de informes, producción de material publicitario, revisión de documentos legales, etc.

Estos procesos involucran típicamente un "documento" que hace las veces de contenedor de la información, viajando de paso en paso y en cada uno de ellos el partícipe realiza una tarea o acción sobre el "documento.
Por tanto, las características esenciales de workflow colaborativo son las siguientes:

El "documento" y el "proceso" son claves. Es importante para la aplicación preservar la integridad tanto del documento como del proceso.

Fundamentalmente participan "knowledge workers", por tanto está restringido a ciertos grupos "creativos" dentro de la organización.

Es importante que una buena solución no sea "intrusiva" ya que el trabajo de conocimiento es un proceso mental que involucra la creatividad, la que no se desea restringir o encasillar.

El workflow colaborativo debe ser muy flexible ya que el trabajo creativo puede tomar rumbos inesperados.

Las soluciones de workflow colaborativo suelen estar centradas en el "documento". Ejemplos de ello son las soluciones avanzadas de CAD, sistemas de administración electrónica de documentos o soluciones basadas en Lotus Notes.

1.3.3.- WORKFLOW ADMINISTRATIVO

Involucra procesos administrativos tales como ordenes de compra, hojas de tiempos y movimientos, reportes de gastos, cambios de ordenes, reportes de calidad y muchas otras actividades que traspasan las barreras departamentales e inclusive de la empresa misma.

Los atributos de una buena herramienta son:

- Existen un gran número de procesos administrativos en cada organización, por ello la solución debe ser capaz de manejar muchos procesos diferentes.
- Casi cualquier persona es un participante potencial, de ahí que la escalabilidad de la solución sea de mucha importante.

El workflow administrativo es diferente para cada organización y también cambia con frecuencia; de ahí la gran importancia de poder cambiar los procesos fácilmente.

Ya que cualquiera en la empresa es un participante potencial, es necesario poder distribuir el software al mayor número de usuarios con la menor carga logística posible.

El workflow administrativo está destinado a cada escritorio y se prevé que será el segmento más grande del mercado del workflow.

Ejemplo: Aplicación de manejo de órdenes de compra.

Cualquier empleado de la empresa dentro de la empresa podría llenar una orden de compra para un cliente, esto debido a que la aplicación debe proporcionar tanto facilidades de manejo como de cálculos. Si la orden de compra es menor que cierto monto la misma pasa directamente a depósito, allí se hace un control automático de stock y si hay disponible mercadería se hace la entrega de la misma. Si la orden es mayor a un monto dado, entonces la misma es pasada al supervisor para su aprobación y luego sigue su ruta habitual si fue aprobada.

1.3.4.- ¿AD HOC WORKFLOW?

Es muy común ver el Ad hoc Workflow (Workflow caótico) como un tipo de Workflow unido con el Workflow Administrativo. Sin embargo hay quienes consideran que Ad hoc es simplemente una propiedad que puede o no tener una aplicación de Workflow.

Seguramente usted se preguntará, ¿qué significa Ad hoc?. Ad hoc puede ser interpretado como que algún usuario puede definir o tomar decisiones dentro de la aplicación en cualquier momento, por ejemplo, en una empresa los empleados pueden mandar información a clientes en forma desorganizada y no estructurada, por ejemplo Vía correo electrónico.

Desde nuestro punto de vista, tanto el Workflow de Producción, de colaboración, como el de Administración pueden llegar a tener características Ad

hoc. Por esto creemos que introducir un nuevo segmento simplemente trae confusión.

1.4.- ORÍGENES Y EVOLUCIÓN

Se podría decir que la tecnología de Workflow se basa en que algunas cosas son realizadas más efectivamente por las computadoras que por las personas.

Los humanos somos buenos para tomar decisiones, innovar, identificar hechos inesperados. Pero usualmente no somos eficientes en actividades tales como: buscar un documento entre cientos; tener presentes los vencimientos de las tareas que se tienen que realizar dentro de ciertos plazos; así como también el asegurarse de que el trabajo terminado pase de un lugar a otro respetando la secuencia definida.

Al igual que la evolución de la informática en general, la evolución del Workflow está ligada con el cambio en los objetivos centrales de cada época. Si resumimos la evolución de la informática en las últimas cuatro décadas, podremos ver como han cambiado los objetivos a seguir de cada época. En la década de los 60’ y 70’ el gran objetivo era resolver grandes cantidades de calculo de manera eficiente.

En los 80’ se buscaba mejorar el manejo y administración de las bases de datos y en los 90’ surge la necesidad de entender y poder manejar eficientemente el Workflow, de manera de poder sacarle el mayor provecho posible. Si miramos la actuación del Workflow dentro de estas tres etapas, podremos identificar lo que seria un Workflow Manual en la primera etapa, el Workflow Automatizado dentro de la segunda, y lo que ofrece el Workflow en la actualidad.

En el primer caso podemos ver que antes de que la informática se integrara al trabajo cotidiano, éste era realizado manualmente combinando toda la información en distintas carpetas. En este ambiente era bastante difícil determinar

el estado de una determinada carpeta, así como también el hecho de determinar el proceso a seguir. Se manejaban grandes cantidades de documentos en forma manual, con los consiguientes errores humanos que traían aparejados dichos manejos. Por esto podemos identificar un Workflow Manual inmerso en las tareas cotidianas de esta época. Surge la necesidad de remplazar las actividades manuales por actividades automáticas. Es decir, se busca tener un mayor control y coordinación sobre toda la información que se maneja para llevar a cabo las tareas de las empresas.

Cuando entramos en la década del 80’ podemos apreciar la existencia de diversos sistemas de información, donde se maneja y administra toda la información necesaria para llevar a cabo la producción de las empresas. Se ha logrado automatizar ciertas tareas, que antes se realizaban manualmente. Por esto podemos hablar de un Workflow Automatizado.

A fines de esta década se busca mejorar el flujo de la información, el desafío que se plantea es obtener la información rápida y eficientemente. Surgen las necesidades de incrementar la eficiencia, optimizar la productividad, acortar los tiempos de procesos, tener un control sobre estos, así como también de reducir los costos y mejorar la gestión. Todo esto como consecuencia del incremento de la competitividad y de la exigencia de mejores productos, dentro de un mercado que avanza a gran velocidad.

Finalmente llegamos a la actualidad, donde nos encontramos con el objetivo de resolver eficientemente el Workflow. Actualmente existe una difusión de diversos mecanismos de intercambio de información. Los mismos facilitan el manejo del flujo de la información en general. Las metas son similares a las de épocas anteriores, pero el punto de partida, las asunciones y el impacto son distintos. Dentro de la evolución actual el Workflow como tecnología identificamos la evolución y creación de ciertos productos que acompañan al Workflow. Dichos productos son:

1. **Procesamiento de imágenes:** En este caso se captura en forma de imagen electrónica (por ejemplo mediante un escáner) la información o documento que se desea, para luego ser pasada entre los diferentes participantes con distintos propósitos, durante la realización de un proceso.

2. **Administración de documentos:** Esta tecnología esta relacionada con la administración del ciclo de vida de los documentos. Esta incluye facilidades para guardar en un depósito común aquellos documentos que se comparten, así como también las facilidades para el acceso o modificación de los mismos mediante un conjunto predefinido de reglas.

3. **Correo Electrónico y Directorios:** El Correo Electrónico provee las facilidades para distribuir información entre individuos de una organización, o entre distintas organizaciones. El sistema de directorios no sólo provee una forma de identificar a los participantes dentro de un conjunto de direcciones de correo electrónico, nos ofrece además la potencialidad de registrar la información sobre los participantes, es decir, roles dentro de la empresa u otros atributos.

4. **Aplicaciones basadas en transacciones:** Las transacciones de Workflow guardan la información, reglas, roles, y otros elementos sobre un servidor de Bases de Datos Relacionales, ejecutando la aplicación de Workflow sobre una interfaz gráfica para los usuarios. Estas aplicaciones típicamente incluyen componentes gráficos para el ingreso de los datos.

5. **Procesamiento de Formularios:** El ambiente de los formularios es amigable y familiar para muchos usuarios. Éste es un excelente vehículo para el manejo de la información dentro de una aplicación de Workflow, basado en el valor de los campos de un formulario. Algunos productos para implementar aplicaciones de Workflow proveen constructores de formularios, o se integran a constructores de terceros.

1.5.- ARQUITECTURA WORKFLOW

1.5.1.- ALTERNATIVAS DE ARQUITECTURAS

En un producto de software de Workflow genérico se identifican una serie de componentes e interfaces. La implementación de esta estructura puede ser realizada de varias formas diferentes entre sí. Éste es un punto de desencuentro de los productos existentes.

En los puntos que siguen a continuación explicaremos los diferentes modelos de implementación en forma genérica. Previamente se dará una descripción de las principales componentes de un sistema de Workflow genérico.

1.5.2.- COMPONENTES

Las principales componentes de un sistema genérico de Workflow son ilustradas en la siguiente figura 1.3:

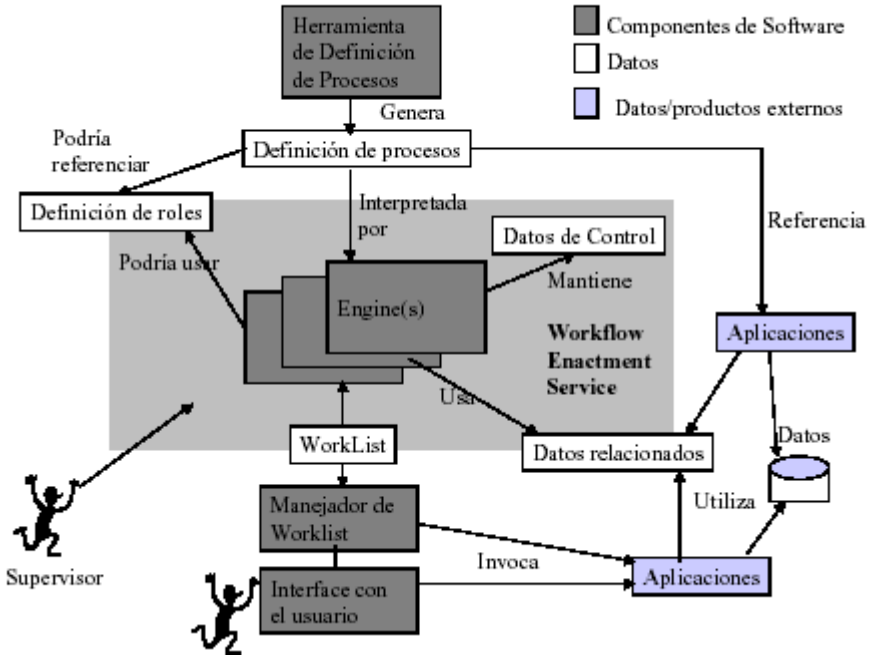


Figura 1.3 Modelo Genérico

En este modelo genérico encontramos tres tipos de componentes:

1. **De software:** Proveen soporte para gran cantidad de funciones del sistema de Workflow.

2. **Datos y Definición de procesos**: Usados por los componentes de software.

3. **Aplicaciones externas:** Trabajan en paralelo con el sistema de workflow.

A continuación describiremos los elementos más importantes mostrados en la figura:

1. **Herramienta de Definición de Procesos.** Forma parte de las componentes de software del Workflow y la podemos ver en el borde superior de la figura. Es utilizada para crear una descripción de los procesos en una forma procesable para una computadora. Esta herramienta podría estar basada en un lenguaje de definición de procesos formal, en un modelo de interacción entre objetos, o simplemente en un conjunto de reglas de ruteo para transferir información entre los participantes.

   Esta herramienta puede ser proporcionada como parte de un producto de software orientado a Workflow, o podría simplemente existir por si sola y tener integración con diferentes productos de Workflow.

2. **Definición de Procesos.** Luego de la componente anterior encontramos la Definición de Procesos, que forma parte de los datos del Workflow. Contiene, toda la información necesaria acerca de los procesos, incluye información de comienzo de actividades, condiciones, y reglas de navegación.

Podría tener referencias a la definición de roles, donde se almacena información de la estructura organizacional. Esto quiere decir que en la definición de procesos se puede mencionar que en cierto proceso participa cierto rol, el cual esta definido en la definición de roles.

3. **Workflow Enactment Service.** Esta componente interpreta la descripción de procesos y controla las diferentes instancias de los procesos, secuencia de actividades, adiciona ítems(elementos) a la lista de trabajo de los usuarios (Worklist), e invoca aplicaciones necesarias.

   Todas estas tareas son hechas por uno o más motores de Workflow (engines), los cuales manejan la ejecución de las distintas instancias de varios procesos.

4. **Worklist (lista de trabajo).** EL Worklist forma parte de los datos del Workflow y la podemos apreciar en la parte inferior de la figura. Ya que la interacción con los usuarios es necesaria en algunos casos, el motor de Workflow utiliza una worklist manejada por un manejador de worklist para controlar tal interacción. El motor deposita en la worklist ítems ha ser ejecutados para cada usuario. La worklist puede ser visible o invisible para los usuarios depende del caso, muchas veces se deja que el usuario seleccione ítems y los procese en forma individual.

5. **Manejador de Worklist.** Luego de la componente anterior encontramos el Manejador del Worklist. Es un componente de software el cual maneja la interacción entre los participantes del Workflow y el Workflow enactment service, via la worklist.

   El manejador soporta en general un amplio rango de interacción con otras aplicaciones clientes.

En la figura la interface con el usuario es mostrada como una componente separada del manejador de Worklist. En algunos sistemas estas dos componentes están agrupadas como una única entidad funcional.

1.5.3.- WORKFLOW ENACTMENT SOFTWARE

El Workflow Enactment Software consiste de uno o más Motores(engines) de Workflow, los cuales son responsables del manejo de toda, o parte, de la ejecución de las instancias de los procesos.

Este software puede ser implementado como un sistema centralizado con un único motor de Workflow, responsable del manejo de todas las ejecuciones de procesos que existen en el sistema. La otra alternativa es una implementación como un sistema distribuido, en la cual varios motores cooperan, la complejidad es mucho mayor pero en general redunda en mayores beneficios.

He aquí una representación gráfica **Figura 1.4** de lo comentado arriba:

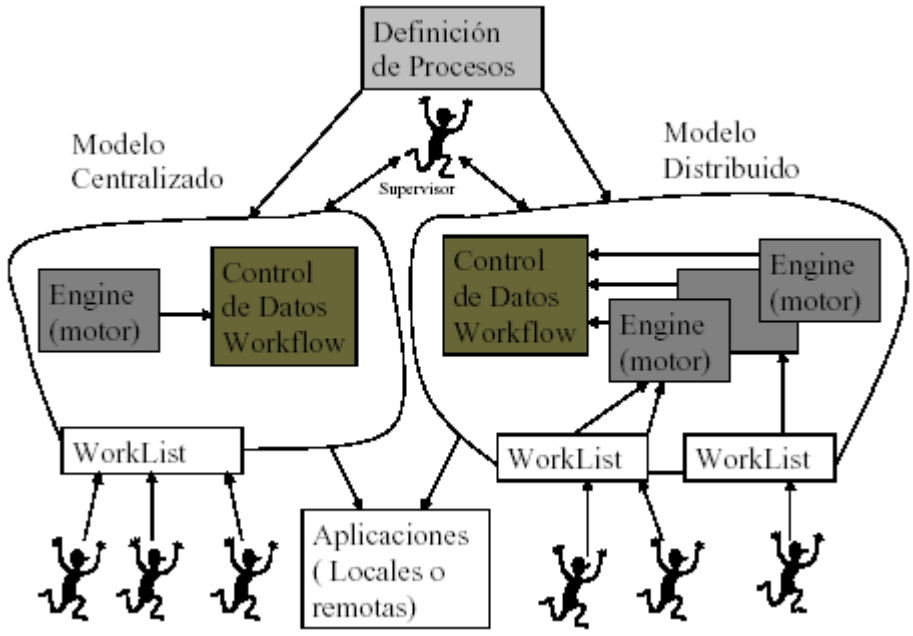


Figura 1.4. Modelo Centralizado y Modelo Distribuido

En el escenario distribuido varios motores cooperan en la ejecución de una instancia de un proceso, el control de datos asociado al proceso debe tener la capacidad de dialogar con diferentes motores. Este control de datos podría estar distribuido entre los motores o podría estar en un único motor (Motor maestro). El control de datos mantiene el estado de la información asociada a cada proceso, podría tener también checkpoints para ser usados en caso de fallas.

La definición de procesos, es usada para modelar la navegación entre los procesos, provee información acerca de entradas a procesos y criterios a tomar en cada paso de la navegación, asigna tareas a usuarios, asigna aplicaciones a cada actividad, etc. La definición de procesos también podría realizarse en forma distribuida o centralizada. La implementación de la opción distribuida implica una gran complejidad al establecer la relación entre la definición de procesos y los motores.

1.5.3.1.- Tipos de Workflow Enactment Services

Podemos encontrar Workflow Enactment Services homogéneos, los cuales están constituidos por uno o más motores de Workflow compatibles. Estos proveen un ambiente de ejecución, con un conjunto definido (especifico del producto) de atributos en la definición del proceso. La interacción entre estos motores no está estandarizada, o sea, es específica de los productos.

Se pueden encontrar también, Workflow Enactment Services heterogéneos, que están constituidos de uno o más servicios homogéneos, los cuales siguen un estándar para la interoperabilidad entre los mismos.

Se ofrecen distintos niveles de conformidad en cuanto a la estandarización. La interoperabilidad de los distintos productos depende del nivel de conformidad. Como se dijo anteriormente, tenemos distintos motores de Workflow controlando una parte del proceso e interactuando con otros motores en un dominio de trabajo distinto. Se espera que los siguientes puntos estén entre los niveles de

conformidad de los productos para poder soportar la interacción de los diversos motores:

- Se debe tener un esquema de nominación común a través de motores heterogéneos.
- Se debe soportar un proceso de definición común para los objetos y atributos, de manera que los diversos motores puedan acceder a ellos.
- Se debe soportar la transferencia de los datos relevantes del Workflow, a través de los motores.
- Se debe soportar la transferencia de procesos, sub-procesos o actividades entre los distintos motores de Workflow.
- Se debe soportar funciones de administración y monitoreo comunes, dentro de un dominio de motores de Workflow.

1.5.3.1.1.- Workflow Application Programming Interface (WAPI)

Las WAPI pueden ser vistas como un conjunto de llamadas API (Application Programming Interface) y funciones de intercambio soportadas por el Workflow Enactment Service. Las APIs son un conjunto de llamadas a funciones de software que permiten a las aplicaciones acceder a funciones de un programa. Las WAPI permiten la interacción del Workflow Enactment Service con otros recursos y aplicaciones.

- **Intercambio de Datos**

El intercambio de los datos es requerido a través de las WAPI para soportar la interacción de las tres funcionalidades siguientes en tiempo de ejecución:

- Manejador de la Worklist (interface 2)
- Aplicaciones Invocadas (interface 3)
- Intercambio entre los motores de Workflow (interface 4)

El intercambio directo de los datos de las aplicaciones es tipificado por sistemas de Workflow orientados al correo electrónico, en donde los datos son físicamente transferidos entre las actividades. En este caso no es necesario definir una relación explícita entre las actividades y la aplicación. En algunos casos es necesario proveer la conversión del formato de los datos entre las actividades. En este caso, la aplicación puede definir, como un atributo, el tipo de datos con el cual esta asociado el formato de los datos. Esto le permite a los sistemas de Workflow heterogéneos proveer la conversión de los datos sobre la base del tipo de datos de los mismos.

En el caso de un sistema de Workflow implementado por intermedio de documentos compartidos, no hay transferencia física de datos entre las actividades.

En este tipo de sistema, los datos son accedidos por la aplicación, usando la ruta de acceso apropiada. Las direcciones de los documentos deben mantenerse como variables globales en el sistema, de forma tal que los distintos servicios o motores puedan acceder a ellos. La conversión de datos puede ser modelada usando herramientas apropiadas para ello (por ejemplo un procesador de texto). Los sistemas homogéneos pueden usar convenciones privadas para el nombrado de los objetos y para el acceso a permisos; pero en sistemas heterogéneos se requiere de un esquema común.

1.5.3.2.- Definición de procesos (interface 1)

1. Herramientas de definición de procesos

Hay gran variedad de herramientas utilizadas para el análisis de procesos. Tales herramientas pueden variar desde las más informales (lápiz y papel), a las más formales y sofisticadas.

La salida de este proceso de modelización y diseño es una “definición de procesos” la cual pueda ser interpretada al instante por el o los motor(es) de Workflow.

Definición del intercambio de datos

Existe gran cantidad de herramientas de definición de procesos (independientes de los productos en muchos casos), las cuales deben comunicarse con los motores de algún producto de Workflow, lo deseable es que esta herramienta pueda comunicarse con cualquier motor, esto únicamente sería posible si se establecen ciertas normas de comunicación entre las herramientas de definición de procesos y un motor de Workflow. Por esto la WFMC propone una interface para esta comunicación.

El objetivo de la interface es dar un formato de intercambio y llamadas a APIs (Application Programming Interface), para soportar el intercambio de información de definición de procesos. El intercambio podría ser una completa definición de los procesos o un subconjunto de la misma. En la **Figura 1.5** se muestra la composición de la definición de procesos.

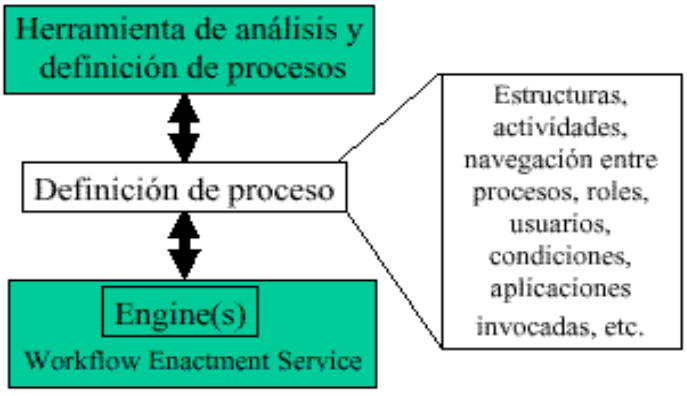


Figura 1.5 Composición de Definición de Procesos

Interface del Workflow con aplicaciones clientes (interface 2)

En el modelo planteado la interacción entre las aplicaciones clientes y el motor de Workflow esta sostenido en gran parte por el concepto de Worklist ya descrito anteriormente.

Parte de la información almacenada en la worklist es utilizada para trasmitirle al manejador de la worklist que aplicaciones hay que invocar.

La worklist podría contener ítems relacionados con diferentes instancias de un proceso o ítems de diferentes procesos. El manejador de la worklist podría estar interactuando con diferentes motores. La interface entre una aplicación cliente de Workflow y el motor de Workflow debe ser lo suficientemente flexible en los siguientes puntos:

- Identificadores de procesos y actividades.
- Estructuras de datos.
- Diferentes alternativas de comunicación.

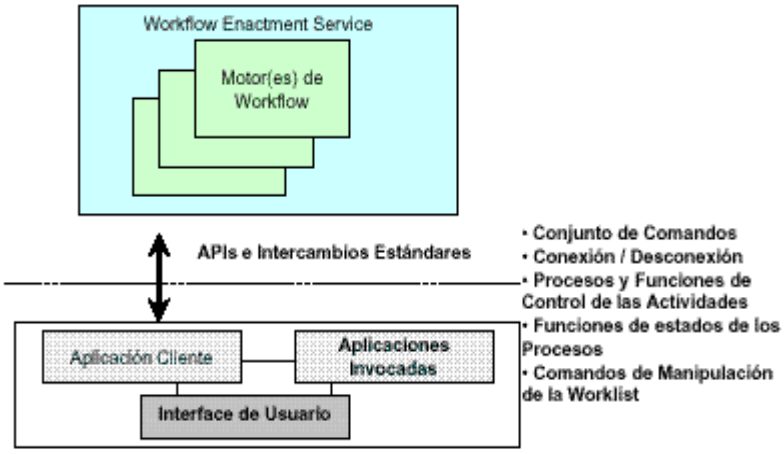


Figura 1.6 Componentes de la Interface del Workflow con Aplicaciones Clientes

Aplicaciones Invocadas (interface 3)

El siguiente diagrama muestra el alcance de esta interface, la cual esta pensada para interactuar con agentes de una aplicación, o con una aplicación entera propiamente dicha. Dichas aplicaciones deben estar orientadas con el contexto general de un sistema de Workflow, es decir, deben poder interactuar directamente con el motor de Workflow.

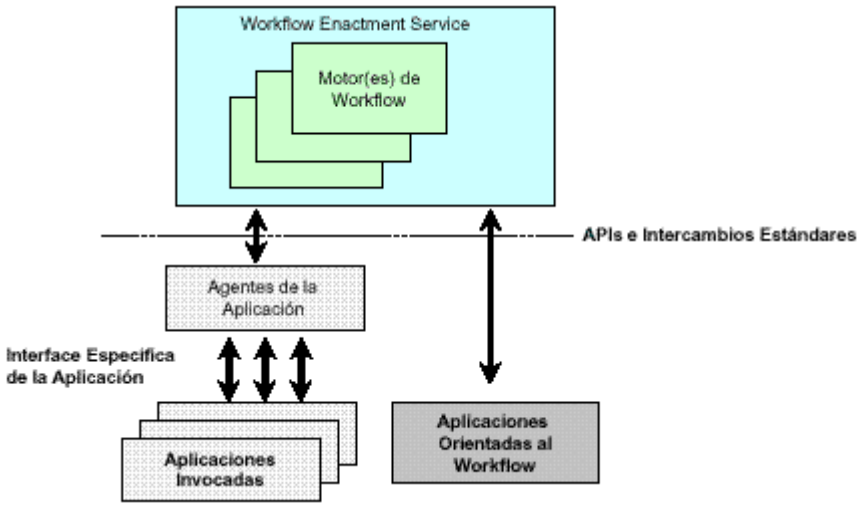


Figura 1.7 Diagrama de la Interface 3

La aplicación invocada es manejada localmente por un motor de Workflow, usando la información suministrada en la definición del proceso para identificar la naturaleza de la actividad, el tipo de aplicación a ser invocada y los requerimientos de los datos. La aplicación que se invoca puede ser local al motor de Workflow, o sea, residente en la misma plataforma, o estar en otra plataforma dentro de una red. En este caso la definición del proceso debe contener la información necesaria para poder encontrar la aplicación que se va a invocar (como ser la dirección dentro de la red).

1.5.4.- WORKFLOW MANAGEMENT COALITION (WFMC)

La WFMC es una agrupación compuesta por compañías, vendedores, organizaciones de usuarios, y consultores. El objetivo de esta agrupación es ofrecer una forma de “diálogo”común a todos. De esta forma las diferentes

herramientas que se implementen en esta área podrán tener cierto nivel de Interoperabilidad, es decir, podrán comunicarse entre ellas para poder realizar las distintas tareas involucradas en un sistema de Workflow.

1.5.4.1.- Necesidad de Estandarizar

Se estima que actualmente los distintos productos de Workflow que hay en el mercado sobrepasan los cien. Cada uno de ellos se enfoca sobre distintos aspectos funcionales, como ser, herramientas de diseño visual, en las cuales se ofrecen ciertos diagramas para representar la realidad. Otras se enfocan en la integración de los datos con las aplicaciones.

El desarrollo de estándares para la interoperabilidad de las diversas herramientas, nos permitiría la elección de los mejores productos, según el enfoque que le demos a nuestra aplicación. Por ejemplo, podríamos comprar una herramienta que nos ofrezca un entorno amigable para realizar el análisis y diseño de nuestro problema, y por otro lado comprar una componente que nos resuelva la auditoria de los datos y poder trabajar en forma integrada con las dos componentes.

Una de las estrategias que siguen actualmente las empresas, es rediseñar sus procesos, esta metodología es denominada como Reingeniería de los Procesos de Empresas (Business Process Re-engineering). Esto puede ser causa de cambios organizacionales, legislativos, cambios en los objetivos del negocio, etc. En esta situación, muchas veces es necesario relacionarse con otras organizaciones. Pero para poder hacer esto debe existir la posibilidad de que los productos de un vendedor puedan comunicarse con los de otro, pues es claro que cada empresa u organización comprará los productos que crea conveniente para su caso.

Vemos entonces la necesidad de estandarizar la forma de comunicación entre los distintos componentes de un producto de Workflow. De modo de poder tener

flexibilidad a la hora de operar con distintos productos. Esta necesidad se justifica además por las proyecciones que se tienen actualmente, sobre la penetración de la tecnología de Workflow en el mercado en los próximos años. Por esto se debe atacar el problema de potenciales incompatibilidades de antemano, y no cuando existan miles de productos en esta área, cada uno con sus particularidades.

Antes de conocer el modelo explicaremos claramente lo que es un Motor Workflow.

Modelo de Referencia de Workflow

El modelo de referencia de Workflow fue desarrollado desde estructuras genéricas de aplicaciones de Workflow, identificando las interfaces con estas estructuras, de forma de permitir a los productos comunicarse a distintos niveles. Todos los sistemas de Workflow contienen componentes genéricas que interactúan de forma definida.

Para poder tener cierto nivel de interoperabilidad entre los diversos productos de Workflow, es necesario definir un conjunto de interfaces y formatos para el intercambio de datos entre dichas componentes.

El Modelo de Workflow

En la figura siguiente se muestra las distintas interfaces y componentes que se pueden encontrar en la arquitectura del Workflow.

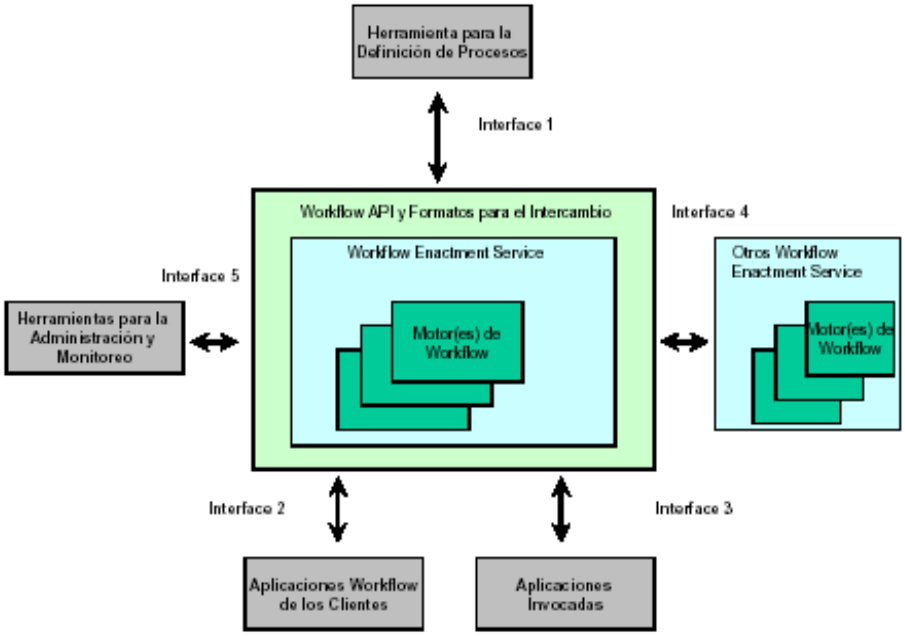


Figura 1.8 Arquitectura de Workflow

En el modelo adoptado hay una separación entre los procesos y el control de la lógica de las actividades. Esta lógica esta dentro de lo que ya definimos como el Workflow Enactment Service. Esta separación permite la integración de las diversas herramientas con una aplicación particular. La interacción del Enactment Service con los recursos externos se da por una de las dos interfaces siguientes:

- La interface de las Aplicaciones de los Clientes, a través de la cual el Motor de Workflow interactúa con el manejador de la Worklist, responsable de organizar el trabajo por intermedio de un recurso de usuario. Es responsabilidad del manejador del Worklist elegir y hacer progresar cada elemento de la lista de trabajo (Worklist).
- La interface de las Aplicaciones Invocadas, la cual le permite al motor de Workflow activar una herramienta para realizar una actividad particular. Esta interface podría ser basada en un servidor, es decir no existe la interacción con el usuario.

El Enactment Service se considera como una entidad lógica, pero físicamente podría estar centralizado o funcionalmente distribuido.

En un Enactment Service distribuido, distintos motores de Workflow controlan una parte del proceso e interactúan con un subconjunto de usuarios y herramientas relacionadas con las actividades que llevan a cabo el proceso. En este tipo de sistemas se deben usar determinados protocolos y formatos para el intercambio de información entre los distintos motores de Workflow.

Motor de Workflow (Workflow Engine)

Es el software que provee el control del ambiente de ejecución de una instancia de Workflow.

Típicamente dicho software provee facilidades para:

- Interpretación de la definición de procesos.
- Control de las instancias de los procesos: creación, activación, terminación, etc.
- Navegación entre actividades.
- Soporte de interacción con el usuario.
- Pasaje de datos al usuario o a aplicaciones.
- Invocación de aplicaciones externas.

1.6.- POR QUÉ USAR WORKFLOW

Siempre que en este mundo nuestro de las Tecnologías de la Información y las Comunicaciones brota como de la nada un nuevo término, una nueva herramienta o una nueva tecnología, sistemáticamente se habla del incremento de la productividad, de la mejora de las organizaciones, el retorno maximizado de la inversión, de la mayor integración o de la rápida adaptación a los cambios que ofrece como un remedio recién llegado. Pero la realidad parece ser más bien otra y es que en el salto de los esquemas formales, los que maneja la ingeniería informática, a los esquemas de negocio, en los que se desarrolla el día a día de las empresas, se pierda o se gane. Surgen en el negocio nuevos parámetros no contemplados y el resultado es que, si bien se da un avance efectivo y una

evolución real, con la incorporación de las nuevas técnicas no siempre se logra el aumento esperado de la productividad.

Sin pretender dar una visión desconfiada de aquello en lo que nosotros mismos trabajamos y vamos propugnando, hay que reconocer que el 'Workflow' no es la solución definitiva de las empresas, porque en realidad no hay solución definitiva para las empresas. Si analizamos los factores que han convertido a algunas compañías en las más brillantes estrellas de su firmamento, probablemente lleguemos a la conclusión de que ellas no han sido pioneras en la incorporación de novísimas herramientas.

Puede ser, eso sí, que hayan inventado la propia herramienta que los demás se esforzarán en implementar o que hayan tenido sencillamente la suerte de estar en el lugar adecuado en el momento oportuno.

Dejando a un lado esas 'afortunadas' empresas tocadas por la gracia de la oportunidad, es evidente que las organizaciones necesitan ser competitivas en un mundo vertiginosamente cambiante, necesitan que la información (oro en forma de bits) fluya cruzando las arterias de la empresa y se enriquezca creciendo desde su origen hasta su destino. Este fluir 'enrutado' en el que los elementos de la corporación, de acuerdo con sus permisos, van incluyendo sus aportaciones de un modo incremental y controlado, es el núcleo básico del 'workflow'.

Hay otro conjunto de características que no se deben olvidar en los sistemas de 'workflow'. En este sentido se nos hablará de diseñadores de flujos, del motor de 'Workflow', de las funciones de soporte gráfico, del control de colas de trabajo, de la monitorización, de las facilidades de auditoria o de la posibilidad de simulación para optimizar tiempos de resolución de problemas y expedientes.

Tal y como están actualmente diseñadas las organizaciones y dada la fiebre desatada en torno a las aplicaciones de 'workflow', cualquier usuario (de acuerdo con su 'rol' y sus permisos) puede en un momento determinado diseñar flujos de trabajo y lanzarlos a la comunidad. Habrá que preocuparse de estructurar

cuidadosamente los procesos para evitar una guerra incontrolada de flujos y el caos.

Es evidente que el contar con un sistema de Workflow proporciona grandes beneficios a las organizaciones que lo emplean. Estos beneficios no redundan únicamente en el ahorro de tiempo en el manejo de papeles, que en un principio era uno de los grandes problemas a resolver. Son varios los puntos a favor del uso de la tecnología de Workflow que se detallaran con mayor detalle en el siguiente punto.

Claramente la tecnología Workflow, combinada con una adecuada Gestión de Procesos, deben de tener características específicas para ofrecer flexibilidad y agilidad a la evolución y dinamismo de los procesos de negocio y sistemas informáticos asociados.

El primer requisito es que el proceso automatizado debe ser fácil de modificar sin ayuda de un programador, de forma que la barrera del cambio disminuya.

La tecnología Workflow ha evolucionado en esta dirección con la introducción de descripciones gráficas de los procesos, como vemos en la figura 1.9 y la posibilidad de modificar el proceso de forma inmediata, sobre la marcha.

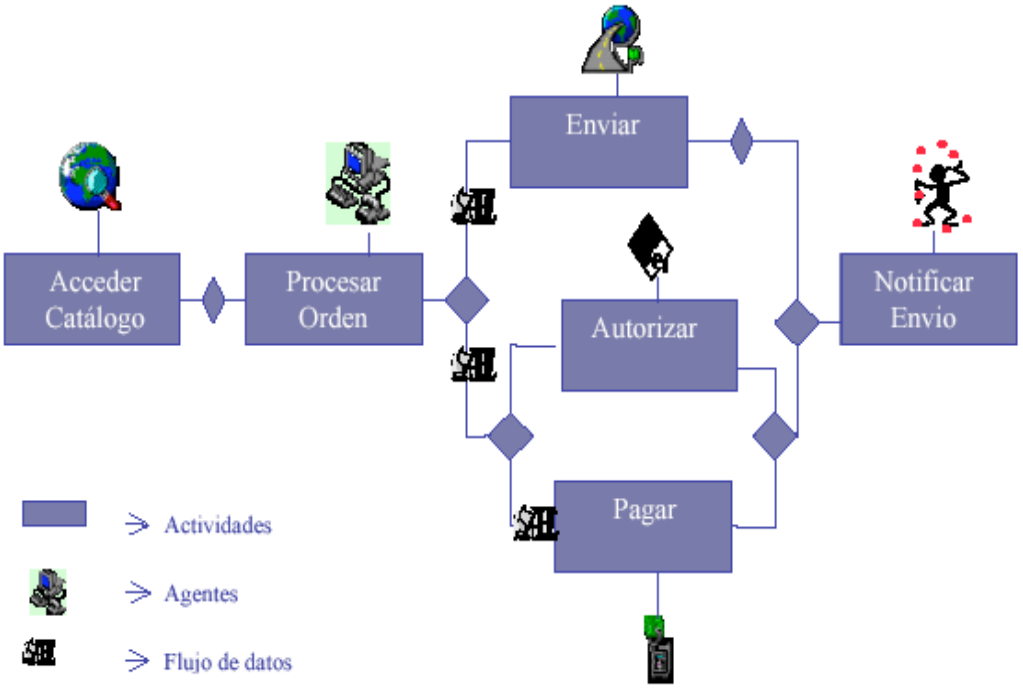


Figura 1.9 Ejemplo de un workflow que muestra en tiempo real el progreso de una instancia de un proceso. Tal cual como lo vemos, un usuario autorizado puede modificar "rápidamente" esta instancia en particular.

1.7.- BENEFICIOS DE WORKFLOW

La automatización de los procesos de negocio de una empresa trae grandes beneficios como la reducción del tiempo de búsqueda de papeles o el menor gasto en papelería, estos problemas son los primeros que se atacaron con la tecnología de workflows.

A continuación conoceremos algunos de los beneficios al adoptar una solución de workflow.

Eficiencia en los procesos y estandarización de los mismos. Esto conduce a:
Una reducción de costos dentro de una empresa.
La estandarización de los procesos lleva a tener un mayor conocimiento de los mismos, lo que a su vez conduce a obtener una mejor calidad de éstos.

Administración de los Procesos. Utilizando la tecnología de Workflow es posible monitorear el estado actual de las tareas así como también observar cómo evolucionan los planes de trabajo realizados. Permite ver cuales son los embotellamientos dentro del sistema, es decir aquellas tareas o decisiones que están requiriendo de tiempo no planificado y se tornan en tareas o decisiones críticas.

Asignación de tareas a la gente. La asignación de tareas se realiza mediante la definición de roles dentro de la empresa, eliminando la tediosa tarea de asignar los trabajos caso por caso.

Recursos disponibles. Se asegura que los recursos de información (aplicaciones y datos) van a estar disponibles para los trabajadores cuando ellos los requieran.

Diseño de procesos. Se fomenta a pensar los procesos de una manera distinta a la tradicional forma jerárquica que se utiliza para diseñarlos en la actualidad.

Hay además muchos aspectos operacionales por los cuales es deseable contar con una tecnología de Workflow ya que aspectos como la secuencia de tareas, quiénes realizan dicha secuencia, los mecanismos de control y monitoreo, son implementadas en el software de Workflow.

Parece ser obvio que son grandes los beneficios. Organizaciones que no hayan evaluado esta tecnología podrían encontrarse con desventajas en un futuro, ya que con la Tecnología Workflow:

El trabajo no queda atascado o extraviado.

Los jefes pueden enfocarse más en los problemas del negocio y del personal, tal como el rendimiento y capacitación individual, mejoras de procedimientos, y casos especiales, más que en la rutina de asignación de tareas.

Los usuarios no gastan tiempo escogiendo sobre cual caso trabajar, aplazando quizás aquellos casos más importantes pero de mayor dificultad.

Se logra el procesamiento paralelo, donde 2 o más actividades no dependientes, dependientes pueden ser realizadas concurrentemente, generando

así beneficios en cuanto a reducción de tiempo de los procesos, mejor servicio al cliente y reducción de costes.

Convertimos el entorno de trabajo de "Reactivo" a un entorno "ProActivo", con todas las ventajas y beneficios que esto conlleva.

Definir y controlar "alertas" según criterios de tiempo, de evento o de condición, provocando así algún mensaje a un supervisor, un "escalado" de actividades a otras personas para que las resuelvan, y/o una resignación automática.

Modificar los procesos y gestionar excepciones "en vivo", o "al vuelo", y desde cualquier lugar, es decir, permitir modificar cualquier instancia de proceso ya iniciada, sin necesidad de volver a iniciarla y sin necesidad de meter mano informáticamente.

Proveer una vista "on-line" para supervisores del estado e histórico de cada instancia de proceso, de cada actividad, y del desempeño de las personas.

Hacerles llegar a cada persona sus actividades y alertas, independientemente de su ubicación geográfica, SMS, o cualquier otro dispositivo móvil.

II. MODELAMIENTO DE PROCESOS

2.1.- INTRODUCCIÓN

Apesar de la gran variedad de productos de Workflow que se encuentran en el mercado, se puede ver que los conceptos y terminologías utilizadas no varían en gran forma. Esto permite que se tienda a realizar un modelo de implementación general.

Actualmente se busca identificar los principales componentes de un sistema de Workflow, de modo que puedan ser volcados dentro de un mismo modelo abstracto. Es necesaria la representación de un modelo que permita la realización de sistemas sobre diversos escenarios, con el propósito de tener la posibilidad de que distintos sistemas de Workflow puedan interactuar entre sí.

Uno de los pilares básicos de un sistema de gestión workflow es su capacidad para modelar formalmente la realidad. Para poder implementar un WFMS es necesario el uso de una metodología de modelado, la cual facilitará las siguientes funciones:

- Identificación, representación y comprensión de los procesos,
- Análisis y simulación de procesos y mecanismos de demostración, para detectar los posibles cuellos de botella, evaluar costes, etc.
- Comunicación entre procesos,
- Documentación para asegurar la calidad.

Para comprender mejor la importancia de la tarea de modelado de procesos dentro de la construcción de un sistema de workflow, se enumeran las siguientes características que cumplen la mayoría de los procesos en este tipo de sistemas:

Conocimiento distribuido.- Los sistemas de workflow generalmente automatizan procesos en los que intervienen muchas personas. Pues comúnmente el conocimiento global de las reglas de negocio que rigen estos procesos, está distribuido entre todas las personas que participan y no en una sola.

Sustitución de procesos manuales documentales.- Los sistemas de workflow no son una sustitución o mejora de una tecnología ya existente. Es decir, la mayoría de los procesos que se automatizan en los sistemas de workflow son procesos que antes se llevaban a cabo a mano a través de formularios y documentos con información poco estructurada y con escaso o sin registro en sistemas informáticos.

Alta variabilidad.- A diferencia de lo que puede ser un proceso rígido con pocas posibilidades de variar como la línea de montaje de automóviles en una fábrica, los procesos modelados por sistemas de workflow son propensos a sufrir cambios. Esta característica se debe principalmente a la alta participación de personas en tales procesos las cuales pueden ser reubicadas, ascendidas o suprimidas en el organigrama de una organización implicando así la variación temporal o definitiva de un determinado proceso. También la búsqueda constante de mejora en los tiempos de procesos por parte de la organización hace que los mismos sean modificados para eliminar “cuellos de botella” detectados luego de modelarlos.

De las anteriores características surge que la metodología a utilizar para el modelado o representación de los procesos debe cumplir con las siguientes propiedades:

- Fácil de usar y entendible para el común de los usuarios del sistema.
- Debe tener en cuenta y permitir una clara representación de la información con la que se trabaja en los procesos desde los puntos de vista de contenido y forma es decir documentos, formularios, etc.

- Ofrecer la mayor cantidad posible de elementos que permitan tanto la representación de procesos sencillos así como también de procesos altamente complejos y soportar la variabilidad de los mismos.

2.2.- CATEGORÍAS DE ANÁLISIS DE UN SISTEMA DE WORKFLOW

2.2.1.- CONCEPTOS MANEJADOS PARA MODELAR EL WORKFLOW

En toda organización, cuando se modela un sistema de Workflow generalmente se identifican y utilizan definiciones de los distintos elementos que se pueden encontrar dentro de dicho sistema. A continuación se lista estos elementos, para luego dar una descripción o definición de cada uno de ellos:

- Tareas.
- Personas (Usuarios).
- Roles.
- Rutas.
- Reglas de Transición.
- Datos.
- Eventos.
- Plazos (Deadlines).
- Procesos.
- Políticas.

- **Tareas.** Cada tarea es un conjunto de acciones o actividades manejadas como una sola unidad. Generalmente son desempeñadas por una única persona dentro de los roles que pueden realizar dicha tarea. Las tareas surgen del análisis del flujo del trabajo, donde se define por quienes deben ser ejecutadas.

- **Personas (Usuarios).** Las tareas son realizadas en un orden definido por determinadas personas (o agentes automatizados tomando el rol de las personas) basados sobre las condiciones o reglas del negocio.

- **Roles.** Cada rol se define independientemente de las personas físicas a las cuales se les van a asignar dichos roles. Una persona puede tener más de un rol.

- **Rutas.** Una ruta especifica la secuencia de pasos que será seguida por los documentos (o información) dentro de un sistema de Workflow. La capacidad de rutear las tareas a usuarios remotos u ocasionales es trascendental en una aplicación de Workflow. Para asegurar el éxito del flujo de la información y decisiones, todos los miembros del equipo deben ser capaces de tomar parte en este proceso.

Se distinguen varios tipos de rutas:

- *Rutas Fijas:* En este caso los documentos siguen siempre el mismo camino. Se define de antemano cual es la próxima etapa a seguir.

- *Rutas Condicionales:* El camino a seguir depende de la evaluación de condiciones. Estas decisiones se toman en el mismo momento que se pasa por el punto donde hay que evaluar las condiciones.

- *Rutas Ad Hoc: E*n este caso el usuario elige explícitamente cual es la siguiente etapa a seguir.

- **Reglas de Transición.** Son reglas lógicas que determinan la navegación del documento dentro del sistema. Indican que acción se va a tomar dependiendo del valor de expresiones lógicas. La definición de las reglas puede ser muy complicada, con múltiples opciones, variaciones, y excepciones. Un ejemplo sencillo podría ser el siguiente: Un cliente solicita

un préstamo por US$ 1000, entonces la siguiente regla expresa el camino a seguir sobre la base de la solicitud: “SI la cantidad solicitada es mayor que el tope del cliente ENTONCES enviar la solicitud al supervisor del área, CASO CONTRARIO, entregar el dinero”. La regla anterior muestra, de manera sencilla, el tipo de reglas que comúnmente se expresan.

- **Datos.** Los datos son los documentos, archivos, registros de la Base de Datos, y otros utilizados como información para llevar a cabo el trabajo.

Entre los datos manejados por el Workflow encontramos:

- *Datos de Control:* Son los datos internos manejados por la lógica del sistema de Workflow.

- *Datos Relevantes:* Son aquellos datos utilizados para determinar el ruteo de las distintas tareas del sistema.

- *Datos de la Aplicación:* Estos datos son específicos de la aplicación, no son accedidos por la lógica del Workflow.

La noción de documento como recipiente de información que se transmite de una tarea a otra, es muy utilizada. Por esto, cuando nos refiramos a datos manejados por el sistema, los nombraremos por documentos.

Existen ciertas propiedades que se le pueden asociar a un documento, como ser: la definición de los derechos de acceso a los mismos; las vistas definidas sobre ellos; también se pueden definir formas de relacionar datos provenientes de fuentes externas al documento, como ser, datos de la aplicación o de la Base de Datos.

- **Eventos.** Un evento es una interrupción que contiene información, el mismo tiene un origen y uno o más destinatarios. La información contenida

en el mensaje que se produjo por el evento puede ser implícita o dada por el usuario. Los eventos pueden ser disparados voluntariamente por el usuario; o en forma implícita durante un proceso según el estado de los datos o de decisiones tomadas por el usuario; o en forma automática. Por ejemplo, cuando un gerente de un banco hace una consulta sobre ciertos datos para hacer una auditoria, se dispara un evento que le devuelve la información de dicha consulta.

- **Plazos (Deadlines).** Podemos ver a los plazos como los tiempos que se le asignan a ciertos elementos. Ejemplos de plazos pueden ser: el tiempo máximo que se le asigna a una tarea para que sea terminada; el tiempo máximo para recorrer una ruta; terminar una tarea antes de cierta fecha; terminar el recorrido de una ruta antes de cierta fecha; etc. A los plazos podemos asignarles eventos, de forma tal de que cuando venza determinado plazo se disparen ciertos eventos asignados por el usuario, o programados para que se disparen automáticamente.

- **Procesos.** Comúnmente los procesos no son “diseñados”, sino que son identificados en la realidad, por el uso diario que se les da. “Nosotros siempre lo hemos hecho así” es una expresión común que se identifica al momento de evaluar estos procesos. Es común que se piense en poner todos los procesos dentro de una aplicación, pero suele ocurrir que sólo algunos de ellos compongan la aplicación final.

- **Políticas.** Las políticas son una manera formal de expresar sentencias de cómo serán manejados ciertos procesos. Por ejemplo, todas las empresas tienen políticas de licencias vacacionales y beneficios para sus empleados, y podrían definir además como se manejarán los distintos procesos de que la componen.

A continuación se muestra un ejemplo donde se identifican algunos de los elementos explicados en los puntos anteriores:

Ejemplo

En la figura 2.1 se muestra un diagrama sencillo de un proceso de solicitud de compra de algún producto en una empresa. Esta solicitud ingresa al sistema vía teléfono. En él se pueden identificar las tareas que comprenden el proceso, ruta por la cual fluye la solicitud, reglas de transición entre las tareas, así como también usuarios, roles y eventos.
Las tareas son las que están representadas por rectángulos con una descripción asociada dentro de los mismos.

El diagrama está dividido en tres partes, cada una de las partes identifica las tareas realizadas por el rol que corresponde. Los roles que identificamos son: el empleado de atención al público, la sección de ventas y el empacador. Observar que el rol de empacador puede asociarse a una persona en particular y que el rol de sección de ventas no necesariamente identifica a una persona.

Vemos como el evento asociado al ingreso de una solicitud de compra por teléfono hace que se dispare todo un proceso donde la solicitud de compra va recorriendo ciertas rutas según las condiciones que se van dando. Las reglas de transición se identifican por rombos con una condición asociada. Según el valor de estas condiciones la solicitud de compra toma uno u otro camino. Una de las rutas más sencillas que se pueden identificar es cuando existe producto en stock para el producto de la solicitud de compra, luego de lo cual se pasa a facturar y finalmente se empaca y se envía.

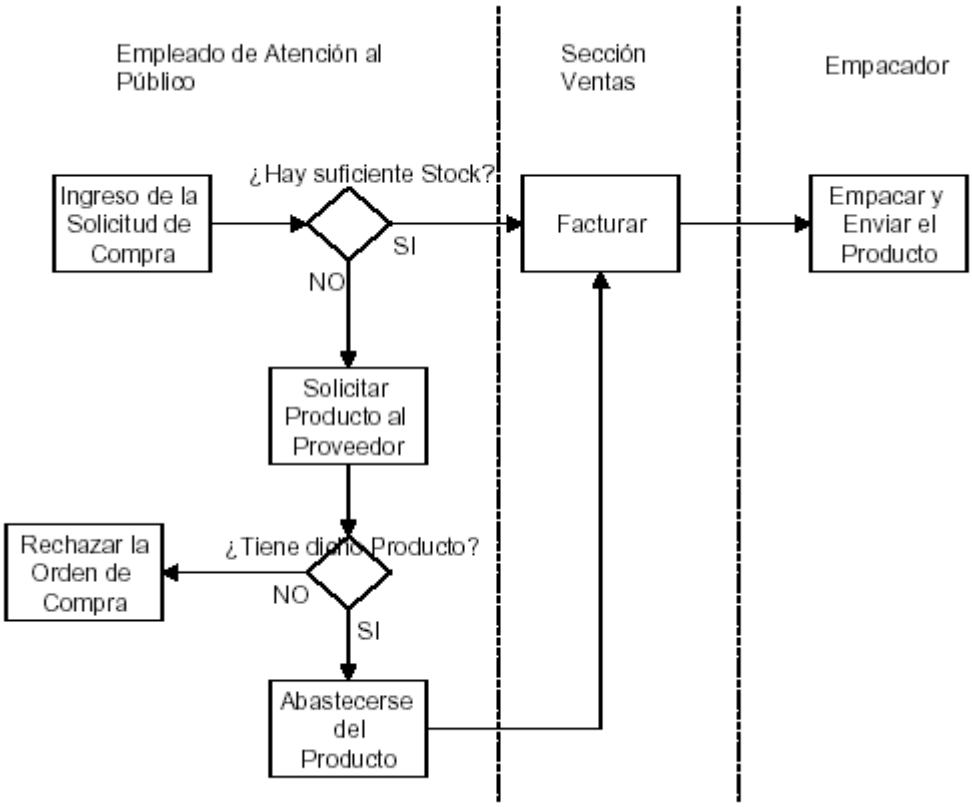


Figura 2.1.- Ejemplo de Diagrama de un Proceso de Solicitud de Compra de un Producto en una Empresa

2.2.2.- PRINCIPALES CATEGORÍAS DE ANÁLISIS

A efectos de poner en marcha un Sistema de Administración Workflow, y con el propósito de guiar las tareas de relevamiento, diseño e implementación, se utilizarán las siguientes categorías de análisis:

- **Quienes.** Definición de participantes (usuarios), roles, ubicaciones (unidades u oficinas) y las distintas relaciones que puedan existir entre ellos tales como dependencias jerárquicas o de reemplazo. Los participantes son los individuos que intervienen en un proceso con su trabajo.
- **Sobre qué información.** Definición de la información (Documentos, Formularios) sobre la cual los participantes trabajan. Esta definición no solo

involucra a los documentos en sí, sino también a los *estados* por los cuales éstos pasarán durante su ciclo de vida en un proceso.

- **Qué hacen.** Definición de las acciones que los usuarios pueden realizar sobre la información.

  Esta área se refiere a las *actividades* que los participantes de un proceso pueden realizar.

- **Lógica de Procesos (Reglas del Negocio).** Definición de las reglas que rigen el flujo de la información entre los participantes del proceso.

La división del problema en cuatro categorías de análisis (o sub-problemas) ayuda al entendimiento del sistema a modelar y facilita su construcción en las etapas de relevamiento, análisis, diseño e implementación.

Una buena metodología de modelado de procesos para sistemas de workflow debe tener en cuenta las categorías de análisis arriba presentadas.

La primera categoría **Quienes** no es exclusividad de los sistemas de workflow, de hecho se encuentra presente en casi todo sistema informático desde sistemas operativos hasta aplicativos de toda clase. Corresponde a la clásica “definición de usuarios”. Lo que sí es importante destacar es qué en los sistemas de administración workflow ésta categoría tiene mayor importancia, y por lo tanto complejidad, que en otro tipo de sistemas.

La segunda categoría **Sobre qué información** hace referencia a la parte de datos del sistema, la cual en los sistemas de workflow tiene un fuerte componente documental. En esta categoría es tan importante el *estado* de los datos como los datos en sí mismos. Tomando por *documento* al conjunto de datos concretos y particulares de una instancia de un proceso; y a *estado* como el valor que

identifica la situación de un documento en un momento dado. El estado de un documento provee una idea global del mismo desde tres puntos de vista:

- Qué datos contiene en ese momento
- En qué etapa del proceso se encuentra
- Qué es lo próximo que hay que hacer sobre el documento en el proceso

La tercera categoría **Qué hacen** representa las *actividades* que los participantes del proceso deben llevar a cabo para completar el mismo.

La cuarta y última categoría de análisis, **Lógica de Procesos** es característica propia de los sistemas de workflow y claramente distingue a éstos de otros tipos de sistemas informáticos. Se refiere a las reglas que rigen el flujo de datos a lo largo del o los procesos modelados.

2.3.- METODOLOGÍAS DE MODELADO EXISTENTES

Un proceso existente dentro de una organización, puede tener varias representaciones así como diversas metodologías para modelarlo. Incluso utilizando la misma metodología, dos analistas podrían modelar el mismo proceso de diferente forma. Esto se debe a que el modelo es una abstracción de los aspectos más importantes de dicho proceso y no el proceso como tal. Es decir, basta con que los dos analistas que modelan determinado proceso tengan distintas visiones del mismo o consideren como importantes diferentes aspectos de éste (de acuerdo a intereses particulares de cada uno) para que el modelo sea diferente. Algo similar ocurre con las metodologías. Cada metodología modela o representa de un proceso lo que considera más importante o más útil para trabajar con el mismo.

Consecuentemente, la clave del éxito de una metodología está en identificar cuales son esos aspectos más importantes o fundamentales a representar de un proceso cuando se trata de un sistema de workflow.

Por lo tanto, si para modelar un proceso de un sistema de workflow, se utiliza una metodología existente que no representa los aspectos fundamentales de este tipo de sistemas, evidentemente se estará frente a una tarea más compleja que si se utiliza una metodología que sí los represente.

Se puede proponer como aspectos fundamentales a representar de un proceso de un sistema de workflow, las cuatro categorías de análisis presentadas anteriormente.

En base a esto, se analizan brevemente algunas de las metodologías existentes y luego se presenta la metodología DEA (Diagrama de Estados-Actividades) como propuesta de representación de las categorías de análisis.

2.3.1.- DIAGRAMA DE COATS & MELLON CMOS

Coats y Mellon, ingenieros de Motorola, proponen un método simple de representación del comportamiento de los procesos desde el punto de vista del usuario. Tal como explican sus autores, este método fue diseñado para ser usado con lápiz y papel con el objetivo de capturar en forma rápida y efectiva la información operacional más importante de un proceso sin necesidad de entrar en el uso de herramientas CASE. Claramente este método está orientado al uso en conjunto con usuarios finales y por eso es muy útil en el modelado de procesos para sistemas de workflow.

A modo de ejemplo, la Figura 2.2, muestra la especificación operacional obtenida utilizando el método CMOS al analizar el proceso de operación de un cajero automático.

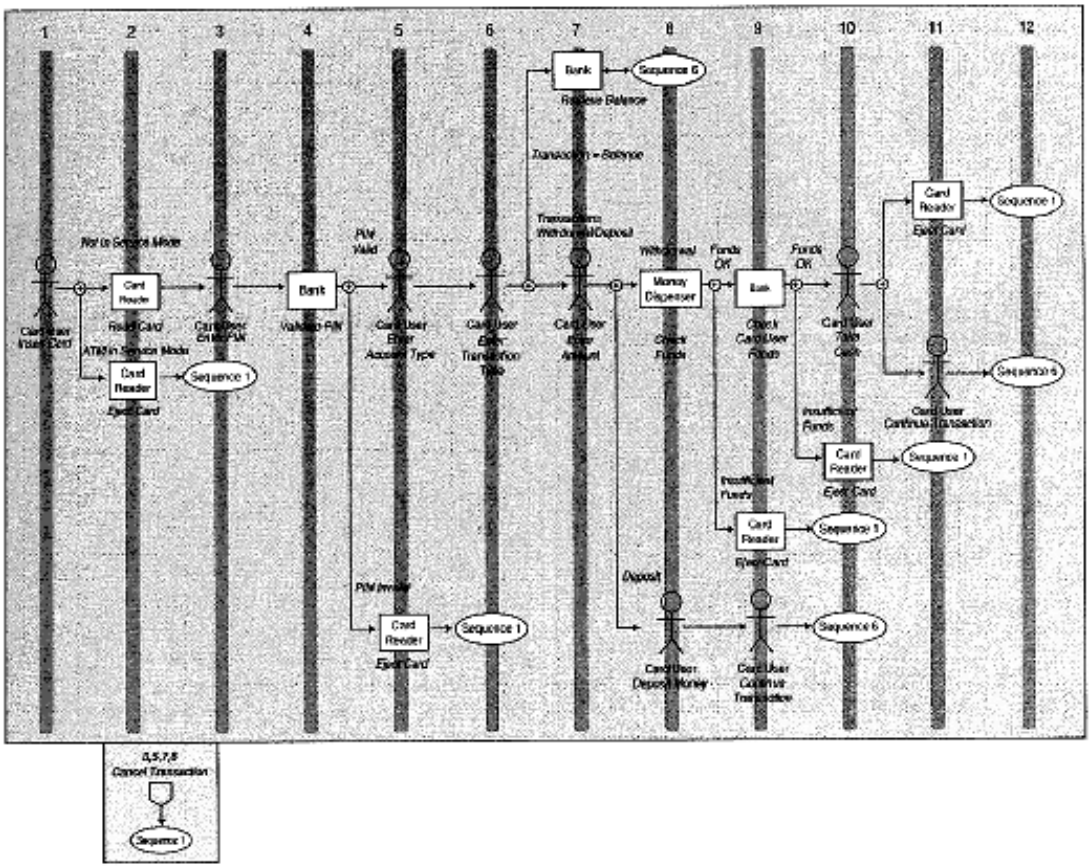


Figura 2.2.-: Diagrama CMOS para proceso de operación de cajero automático

Tal como se puede apreciar, el diagrama es conceptualmente similar a un diagrama de actividades solo que más sencillo. En el mismo se identifica qué participante (usuario o máquina) hace determinada tarea y en qué momento. Claramente muestra también el orden o la secuencia en que suceden las actividades o eventos.

2.3.2.- DIAGRAMA DE ACTIVIDAD (UML)

UML (Unified Modeling Language) [UML] es un estándar ampliamente difundido y muy utilizado en el análisis y diseño de sistemas informáticos. En lo que a sistemas de workflow se refiere UML aporta los diagramas de actividad y los diagramas de estado.

Como ejemplo, la Figura 2.3, muestran la especificación operacional obtenida utilizando el método UML al analizar el proceso de entrega de citación de un crédito.

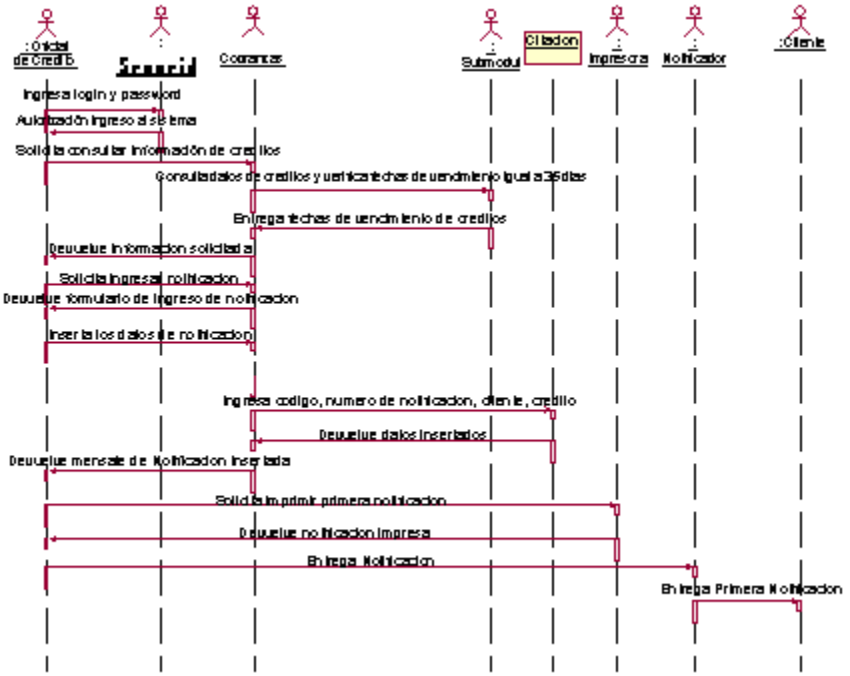


Figura 2.3.- Diagrama UML para proceso de entrega de citación de créditos

El diagrama de actividad UML, al igual que CMOS representa qué participante realiza cuál actividad y en qué momento. Una ventaja del diagrama de actividades frente al CMOS es que el primero es un estándar cuya nomenclatura es ampliamente conocida y utilizada. Por otro lado, CMOS es más fácil de entender y de utilizar además de ofrecer una representación más clara de la secuencialidad de las actividades, lo cual para procesos complejos es difícil de visualizar en los diagramas de actividades.

2.3.3.- DIAGRAMA DE ESTADOS

Tanto el diagrama CMOS como el DA-UML dejan fuera de su representación los datos o la información sobre la cual se trabaja en un proceso. El diagrama de estados aborda directamente este problema representado un proceso como la

evolución de la información (documentos) del mismo, a través de distintos estados.

La representación más natural para estos diagramas, es la máquina de estados finita. En la misma que se muestra para cada documento (conjunto de información) cuales son los estados por los que transita dicho documento mientras los usuarios trabajan sobre él dentro de un proceso.

Los diagramas de estado son claramente más complejos que los anteriores y por lo tanto es más difícil su utilización con usuarios finales. Su utilidad cobra mayor importancia en las etapas de diseño e implementación y no tanto en el relevamiento del sistema. Una de sus principales ventajas es el nivel de detalle con que se describe el comportamiento o el flujo de los datos sobre todo en sistemas documentales donde la información se agrupa por formularios.

2.4.- DIAGRAMA DE ESTADOS – ACTIVIDADES (DEA)

El diagrama de estados – actividades tiene por objetivo unificar, en una metodología de representación, los aspectos fundamentales (categorías de análisis) de los procesos en los sistemas de workflow. Como corolario se obtiene “lo mejor de dos mundos” integrando los diagramas enfocados a estados con los diagramas enfocados a actividades.

2.4.1.- ELEMENTOS BÁSICOS

2.4.1.1.- Actividad (Activity)

Este elemento representa el trabajo a ser realizado por un participante dentro de un proceso, es una porción del trabajo total necesario para cumplir con el proceso. Para el mismo se aplica la definición de la Workflow Management Coalition (WfMC).

La representación gráfica de este elemento se muestra en la figura 2.4:

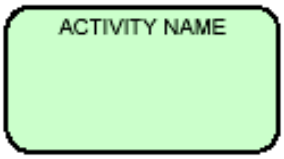


Figura 2.4.- Representación Gráfica de la Actividad

Para ilustrar el uso de éste y los demás elementos básicos del DEA se pone a consideración el ejemplo de proceso en un sistema de workflow en la figura 2.5.

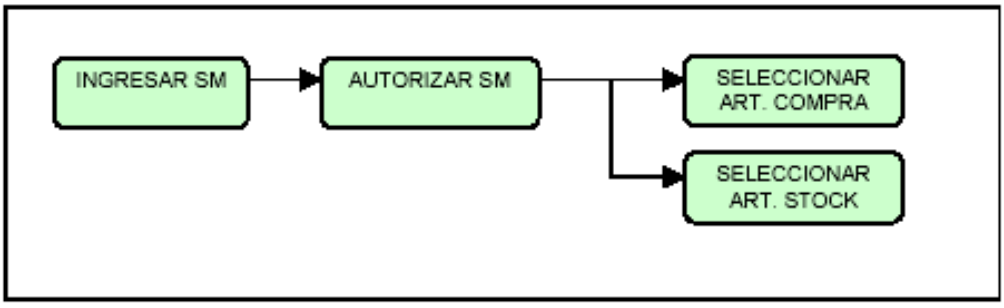


Figura 2.5.- Ejemplo de proceso

En este ejemplo se representa una parte del proceso de compra de insumos para una organización estatal proveedora de servicios. En el mismo se tienen cuatro actividades.

Ingresar SM representa el ingreso de una Solicitud de Materiales por parte de un usuario solicitante dentro de la organización. Luego *Autorizar SM* representa la actividad de aprobación de la solicitud por parte del superior inmediato (jefe) de quien la ingresó. Y finalmente se representan las actividades de *selección de artículos* para *comprar* y para sacar de *stock* llevadas a cabo por la oficina de planificación de la organización.

Claramente este elemento del diagrama hace referencia a la categoría de análisis **Qué hacen**.

2.4.1.2.- Documento – Estado (Document-State)

Este elemento representa los datos del sistema conjuntamente con el estado de los mismos. Es importante comprender que cuando se habla de Documento se hace referencia a cualquier tipo de datos de una instancia de proceso, y no solamente a aquellos almacenados en forma documental. En efecto, más allá del tipo de repositorio de datos que el producto utilice (RDBMS, DDBMS, ASCII, HTML, XML, etc.) se define como documento a un conjunto de datos lógicamente agrupados (por su función dentro del proceso) independientemente de su estructura de almacenamiento o la forma de acceso a ellos.

La segunda parte del nombre de este elemento (state) hace referencia al estado de los datos representados por el documento y que, e identifican la situación del documento en una momento dado en una instancia de un proceso.

Representación gráfica en el la figura 2.6:

Figura 2.6.- Representación Gráfica de Documento - Estado

Nuevamente considerando el ejemplo de la figura 2.5, en el mismo se puede ver que el documento Solicitud de Materiales pasa por los estados de la figura 2.7.

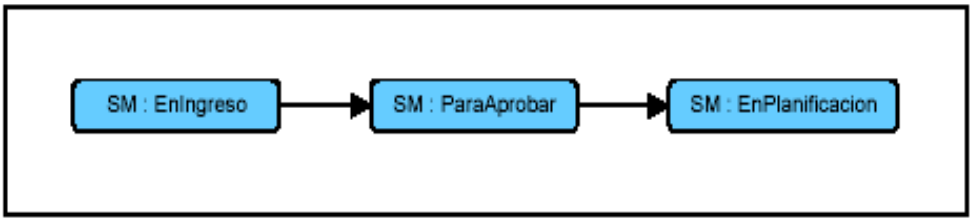


Figura 2.7.- Estados del documento SM

Con este ejemplo, se observa claramente que no son lo mismo las Actividades que los Estados del Documento. Tampoco es posible, encontrar una correspondencia uno-a-uno entre los Estados del Documento y las Actividades. Como se puede ver, a los dos primeros estados de la SM se les puede asociar las dos primeras actividades respectivamente, pero al tercer estado se le deben asociar las últimas dos actividades en forma conjunta. Este es un caso en el cual un solo Estado del Documento tiene asociadas más de una Actividad. También puede darse, como se verá más adelante, que una sola Actividad tenga asociados varios Estados del Documento.

El Estado del Documento representa en el DEA a la categoría de análisis **Sobre qué información**.

2.4.1.3.- Participantes (Participant)

Representa a los participantes de un proceso de workflow. En la mayoría de los casos los participantes son los usuarios del sistema, es decir las personas que trabajan con la aplicación workflow, sin embargo, en ocasiones el participante no es una persona sino un sistema externo a la aplicación de workflow.

Los participantes del proceso son de gran importancia en un sistema de workflow y deben ser representados claramente en el modelo. Son los recursos que llevan a cabo determinadas actividades dentro de un proceso. Se diferencian dos tipos de participantes:

- **PARTICIPANTE USUARIO.** Recurso humano (persona) definido con acceso al sistema, es decir un usuario.

- **PARTICIPANTE SISTEMA.** Recurso no humano que participa en un proceso, puede ser una aplicación externa a la Aplicación de workflow o el motor workflow. Es importante aclarar que para que un recurso no humano sea considerado un participante en un proceso, éste debe realizar alguna actividad en el mismo, es decir, debe tener un papel activo.

  Dentro de los *participantes sistema* distinguimos dos tipos:

- Las aplicaciones externas a la Aplicación de workflow.Cuando se habla de las aplicaciones externas se refiere a las aplicaciones que participan en el proceso y no a aquellas que son invocadas por el mismo. Por ejemplo, un servidor de correo que recibe información para iniciar un trámite y la manda automáticamente al sistema de workflow para que éste la procese, es un participante sistema de tipo aplicación externa. Por el contrario, un procedimiento almacenado (stored procedure) que es ejecutado desde el sistema de workflow para llevar a cabo determinada tarea en el proceso es una aplicación invocada y no un participante sistema.

- El o los motores de workflow componentes del sistema de workflow es decir el enlace entre el diagramador y el sistema de administración.

  Representación gráfica en la figura 2.8:

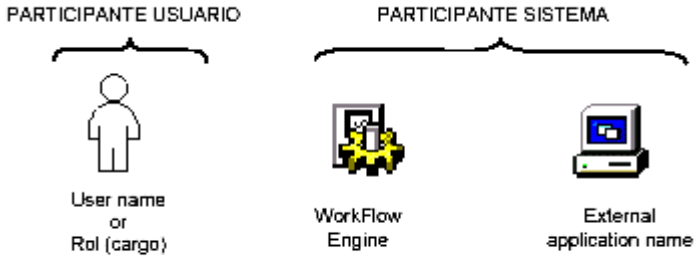


Figura 2.8.- Representación gráfica de Participante Usuario y Participante Sistema

En el proceso de la figura 2.5 los participantes son todos usuarios y están representados en la figura 2.9.

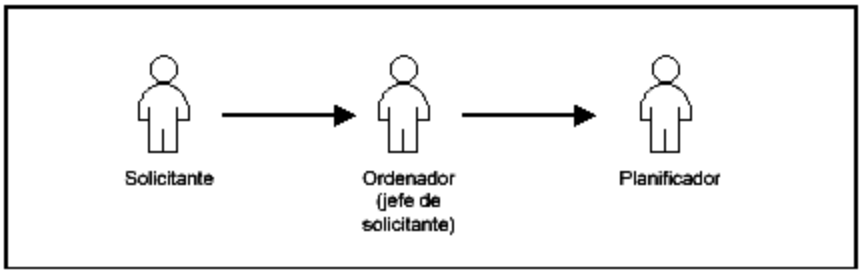


Figura 2.9.- Participantes del proceso

Los participantes representan en el DEA la categoría de análisis **Quienes**.

2.4.1.4.- Conectores y Etiquetas de Conectores (Conectors & Conectors labels)

Los conectores representan las transiciones de un proceso. En los diagramas enfocados a actividades las transiciones representan el paso de una actividad finalizada (actividad de salida) hacia la próxima a efectuarse (actividad de llegada), es decir hay un cambio de actividad. En los diagramas enfocados a estados las transiciones representan el paso de parte o toda la información de la instancia del proceso de un estado a otro, es decir hay un cambio de estado. En los diagramas DEA, como se verá más adelante, las transiciones representan ambas cosas simultáneamente, es decir, cambio de actividades y/o estados. Todas las transiciones en un diagrama DEA tienen asociada una condición lógica.

En el caso más sencillo la condición es **True**, es decir siempre verdadera. Para que una transición se lleve a cabo debe devolver True como verdadero, la cual es previamente evaluada por el motor de workflow. La condición lógica correspondiente a una transición, puede ser tan compleja como el proceso lo

requiera y puede tener en cuenta para su evaluación datos correspondientes a la instancia del proceso ejecutándose, datos de otras instancias o incluso datos obtenidos de fuentes externas.

Las condiciones lógicas son representadas en el diagrama por las etiquetas de los conectores. Cuando la condición es true (siempre verdadera) la etiqueta puede omitirse para mayor claridad del diagrama.

La representación gráfica de las etiquetas se muestra en la figura 2.10:

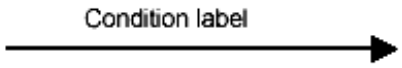


Figura 2.10.- Representación de Etiqueta de Conector

En la figura 2.9 se observan dos transiciones con condición siempre verdadera, es decir, dos conectores sin etiqueta.

Desde el punto de vista formal la definición anterior de *Conectores* es suficiente para modelar cualquier tipo de transición con cualquier condición lógica asociada en cualquier proceso. Sin embargo, por razones de práctica y claridad se definirá a continuación el concepto de Conector Automático (*Automated Conector)*, el cual está incluido en los conectores normales pero es mejor representarlo de forma diferenciada en los diagramas DEA.

2.4.1.5.- Conectores Automáticos (Automated Conectors)

Los conectores automáticos también representan transiciones dentro un proceso. Son aquellos en los cuales la condición lógica es exclusivamente una condición temporal, por eso también se les puede llamar *Conectores de Tiempo*, es decir una condición que evalúe variables de tiempo.

Para comprender la utilidad de estos conectores tomemos como ejemplo que se quiere modelar un proceso en donde existe la actividad de aprobación de un documento y que se desea que luego de 24 hs. de no haber sido aprobado dicho documento, el mismo quede denegado automáticamente. Un caso como este se modela con una transición automática entre la actividad de aprobación y la de archivado como denegado con un conector temporal de condición: **tiempo transcurrido > 24 hs**.

La representación gráfica de conector automático se muestra en la figura 2.11:

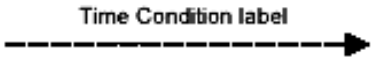


Figura 2.11.- Representación de Conector Automático

De esta manera se tiene una forma clara de diferenciar en el diagrama cuando una transición puede darse por el transcurrir de determinado tiempo exclusivamente, lo cual resulta muy útil para usuarios finales y administradores de sistemas.

Un diagrama de definición de procesos que diferencie estos tipos de transiciones puede resultar muy útil para el motor de workflow que debe interpretarlo.

2.4.1.6.- Flow Hubs

Los ejes de flujo representan las bifurcaciones (split) y las uniones (join) del flujo del proceso. Los mismos que son usados en combinación con los conectores.

Se entiende por bifurcación del flujo del proceso la división de un hilo de ejecución en más de uno. Análogamente la unión del flujo del proceso es la unificación de varios hilos de ejecución en uno solo.

Para comprender la utilidad de los ejes de flujo, es necesario primero hacer algunas consideraciones con respecto a la dinámica de ejecución de los procesos.

2.4.1.6.1.- Instancias de Procesos e Hilos de Ejecución

Una ***instancia de un proceso*** es “la representación de la ocurrencia de un proceso incluyendo sus datos asociados”. Es decir, se puede ver la relación entre la definición de un proceso y una instancia del mismo, como la relación que existe entre una clase y un objeto de dicha clase.

Por ejemplo un proceso de solicitud de tarjetas de crédito podría tener dos instancias: la solicitud de Juan Méndez y la solicitud de Ana Pérez. Por lo tanto una instancia de un proceso se obtiene cuando se ejecuta el mismo con un conjunto de datos reales (llamados “relevant data” por la WfMC)

Un ***hilo de ejecución*** representa la parte de una instancia de proceso que se encuentra en cierto estado o actividad o que recorre cierta transición (evolución) en un momento dado.

Una instancia de proceso puede tener uno o varios hilos de ejecución.

En el diagrama DEA se definen seis tipos de ejes de flujo que permiten seis diferentes usos.

- **AND Paralelo (AND -Split)**

El proceso sigue en paralelo por todas las transiciones que sigan al AND, lo que significa que a partir de un lugar fuente, los documentos son distribuidos hacia varios destinos simultáneamente. Todas las transiciones que siguen al AND tienen como condición True, no admitiendo otra condición.

Representación gráfica:

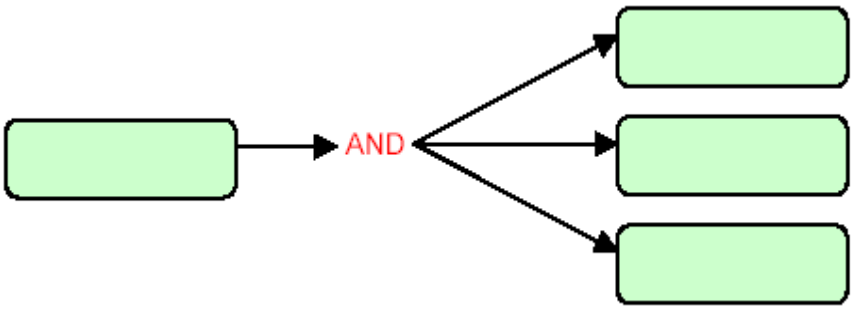


Figura 2.12.- Representación de un Eje de Flujo AND-Split

- **AND Unión (AND Join)**

El proceso sigue luego del AND si y solo si todas las transiciones anteriores a dicho AND son ejecutadas, por lo tanto a partir de varios lugares fuentes, los documentos convergen, sincrónicamente, hacia un único destino.

Representación gráfica:

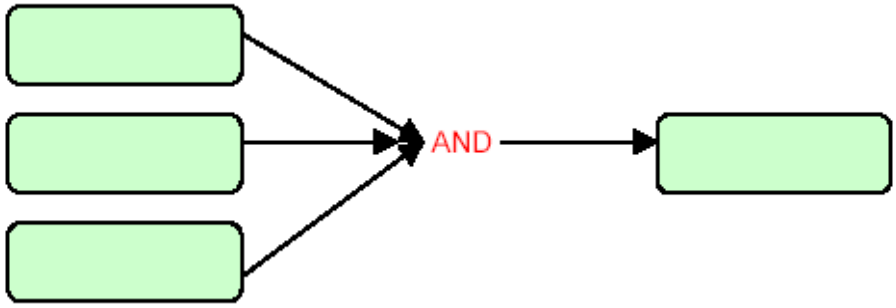


Figura 2.13.- Representación de un Eje de Flujo AND-Join

- **OR Bifurcación (condicional) (OR-Split)**

El proceso sigue por una y solo una de las transiciones que siguen al Or; es decir, a partir de un lugar origen, los documentos toman un destino entre varios posibles.

En las bifurcaciones no existe un orden o prioridad para la evaluación de las condiciones de las transiciones de salida.

Representación gráfica:

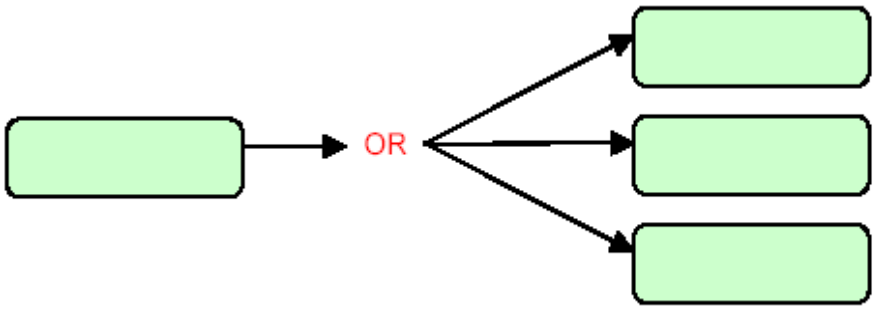


Figura 2.14.- Representación de un Eje de Flujo OR-Split

- **OR Unión (OR-Join)**

El proceso sigue luego del OR si cualquiera de las transiciones anteriores a dicho OR es ejecutada, es decir, a partir de uno o más lugares de origen, dentro de varios posibles, convergen hacia un único destino (no se requiere sincronización).

Representación gráfica:

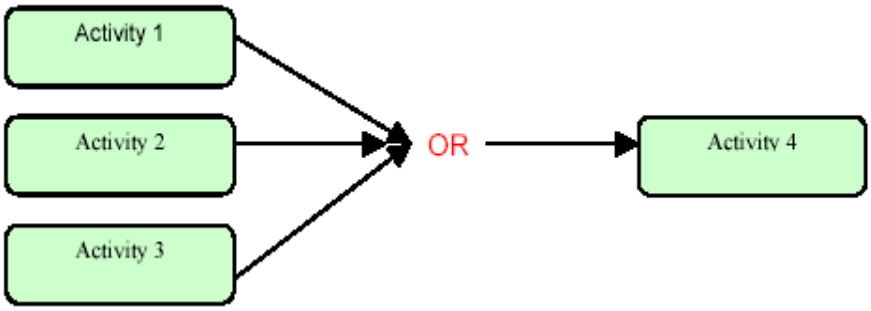


Figura 2.15.- Representación de un Eje de Flujo OR-Join

- **AND Exclusivo**

El proceso, en principio, sigue en paralelo por todas las transiciones salientes del AND/E, pero inmediatamente después de que alguna de las mismas se lleva a cabo las otras actividades paralelas quedan anuladas (se termina su hilo de ejecución). Todas las transiciones salientes del AND/E tienen como condición True, no admitiendo otra condición.

Representación gráfica:

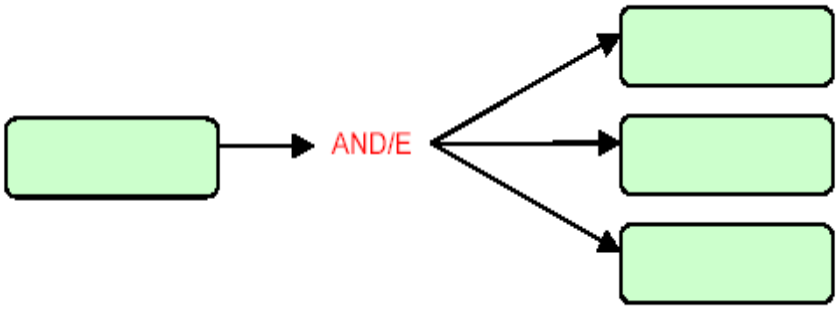


Figura 2.16.- Representación de un Eje de Flujo AND Exclusivo

- **OR Exclusivo (condicional)**

El proceso sigue luego del OR/E si cualquiera de las transiciones anteriores a dicho OR/E es ejecutada y se les anula el hilo de ejecución a todas las restantes transiciones que llegan al OR/E una vez ejecutada la transición.
Representación gráfica:

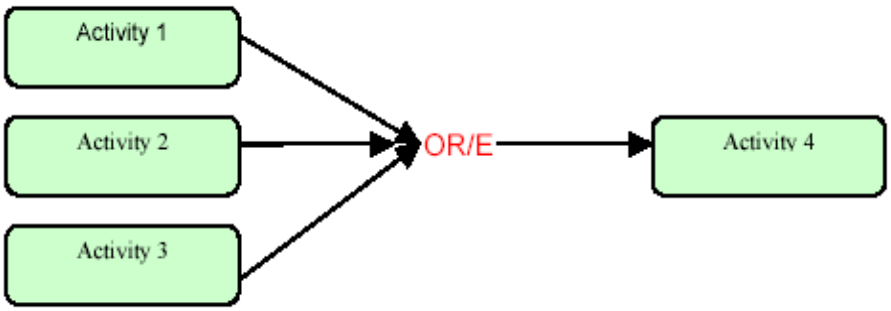


Figura 2.17.- Representación de un Eje de Flujo OR Exclusivo

2.4.1.7.- Fin de Hilo (End Thread)

Este elemento representa la finalización de un hilo de ejecución dentro de la instancia de un proceso. La finalización de un hilo de ejecución no significa la finalización de la instancia del proceso ya que la misma puede tener varios hilos de ejecución como resultado de la utilización de AND's Paralelos. Algunos procesos nunca terminan totalmente y lo que tienen es estados de "archivo" para

los datos de las instancias los cuales pueden ser reactivados en cualquier momento.

Representación gráfica:

Figura 2.18.- Representación de un Hilo de Ejecución

Los conectores, etiquetas, ejes de flujo y fin de hilo representan en DEA la categoría de análisis **Lógica de Procesos (Reglas del Negocio)**.

2.4.2.- INTEGRACIÓN DE DIAGRAMAS

Como se mencionó anteriormente, el diagrama de estados – actividades tiene por objetivo unificar, en una metodología de representación, los aspectos fundamentales (categorías de análisis) de los procesos de workflow integrando los diagramas enfocados a estados con los diagramas enfocados a actividades.

Para ilustrar esta integración se utilizará el ejemplo representado en las figuras 2.5, 2.7 y 2.9. Los diagramas 2.5 y 2.9 representan las actividades y los participantes respectivamente, como habitualmente en los diagramas enfocados a actividades éstas aparecen conjuntamente con los participantes (como en el diagrama de actividades de UML) la integración de las figuras 2.5 y 2.9 será la primera que haremos y el resultado es la figura 2.19.

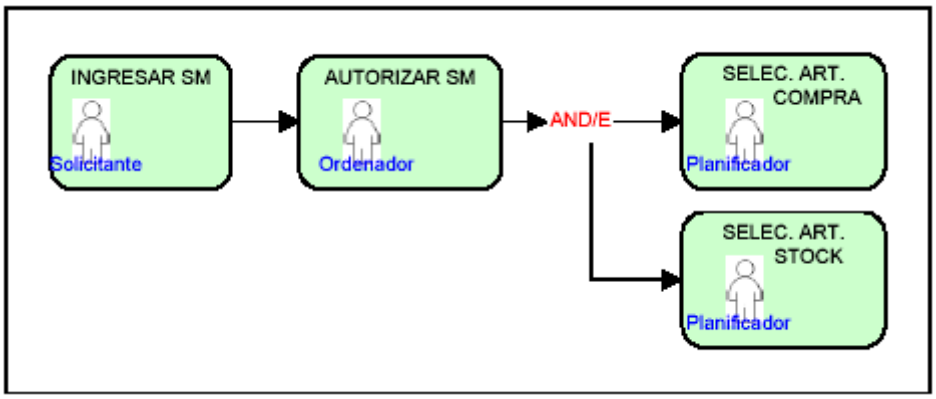


Figura 2.19: Integración de actividades y participantes

Como se puede observar, al hacer la integración de actividades con participantes se debe duplicar la representación del participante *Planificador* ya que el mismo lleva a cabo las dos últimas actividades.

A continuación se hace la integración del diagrama anterior con el diagrama de documentos-estados de la figura 2.7, obteniéndose el diagrama DEA de la figura 2.20.

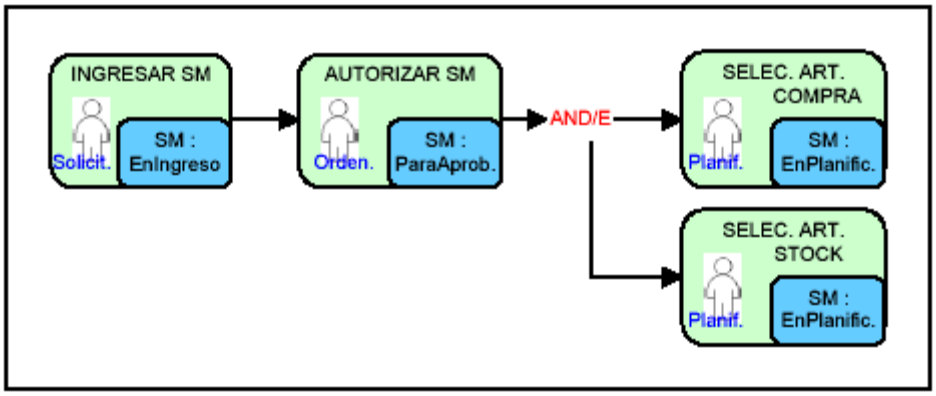


Figura 2.20: Integración de actividades, participantes y documento-estados

El anterior es un ejemplo sencillo de diagrama DEA en el cual cada actividad tiene asociado un solo participante (o rol) y un solo documento-estado. Sin embargo, es posible que las actividades de un proceso tengan asociados más de un participante (usuarios responsables de la actividad) y más de un documento-estado (información sobre la cual se realiza la actividad).

2.5.- VENTAJAS DEL DIAGRAMA DEA

La principal ventaja del diagrama DEA está en su simplicidad y completitud. Los elementos que lo componen son sencillos y de fácil entendimiento incluso para usuarios finales. Al mismo tiempo agrupa los principales conceptos a considerar en la construcción de una Aplicación de workflow mediante un sistema de workflow.

Otra ventaja es la flexibilidad. Al ser este un diagrama resultado de la integración de otros dos se puede “filtrar” fácilmente para obtener cualquiera de los diagramas que lo componen por separado. En efecto, si dado un caso particular dentro de una organización se considera más importante o útil la representación del proceso exclusivamente con un diagrama de actividades, basta con “filtrar” el diagrama DEA sacando los documento-estados. Si por el contrario lo que se desea es ver solamente el diagrama de estados se puede “filtrar” y “simplificar” el diagrama DEA para obtener el diagrama de estados.

La operación de simplificación consiste en unificar los documento-estados que quedan repetidos en el gráfico una vez que se quitaron las actividades.

La utilización del diagrama DEA en las etapas de relevamiento, análisis y diseño de una Aplicación de workflow hacen que éstas etapas sean independientes de la etapa de implementación. Una vez obtenido el diagrama, el mismo puede ser filtrado para alimentar un producto comercial enfocado a estados o uno enfocado a actividades manteniendo una única definición del proceso.

III. LENGUAJE DE MODELAMIENTO UNIFICADO UML

3.1.- INTRODUCCIÓN

El modelado es una técnica utilizada en todas las ingenierías. Como tal la Ingeniería de Software debe basarse en el modelado como una parte central de todas las actividades que conducen a la producción de software de calidad.

Podemos definir el proceso *software* como “El conjunto de actividades del desarrollo *software* y las relaciones entre ellas”. Estas actividades varían según la organización y el tipo de sistema, pero en general la gestión del proceso *software* exige el disponer de un modelo. Construimos modelos para comunicar la arquitectura y el comportamiento deseado en nuestro sistema. Construimos modelos para visualizar y controlar la arquitectura del sistema y para comprender mejor el sistema que estamos construyendo, muchas veces descubriendo posibilidades de simplificación y reutilización. En general, se puede decir que construimos modelos para controlar el riesgo:

- Los modelos nos ayudan a visualizar cómo es o qué queremos que sea un sistema.
- Los modelos nos permiten especificar la estructura o el comportamiento de un sistema.
- Los modelos nos proporcionan plantillas que nos guían en la construcción de un sistema.
- Lo modelos documentan las decisiones que hemos adoptado.

Un modelo es básicamente una abstracción que incluyen lo esencial de un problema complejo o estructura, filtrando los detalles no esenciales, de forma que el problema se hace más comprensible. No se trata, por tanto, de una representación alternativa del sistema. Idealmente, una “representación” de un sistema debería mantener toda la información de la entidad que representa; por

contra, una abstracción deliberadamente simplifica, quedándose con las características más sobresalientes.

Siendo un modelo una simplificación de la realidad, dicha simplificación no es única. Muy al contrario, todo sistema debería ser descrito desde diferentes perspectivas utilizando diferentes modelos y cada uno de estos modelos será una abstracción semánticamente cerrada del sistema. Así podemos trabajar en modelos estructurales, destacando la organización del sistema, o en modelos de comportamiento, resaltando su dinámica.

Un único modelo no es suficiente. Cualquier sistema no trivial se aborda mejor a través de un pequeño conjunto de modelos “casi” independientes. Aunque se pueden crear modelos de distinta naturaleza su elección no debe ser arbitraria, sino que tiene una gran influencia posterior a la hora de comenzar la resolución del problema.

Aunque todos los sistemas pueden beneficiarse del uso de modelado, también es cierto que cuanto más grande y complejo sea un sistema mayor es la importancia de la utilización de buenas técnicas de modelado: existe un límite en la capacidad humana de comprender la complejidad. Por tanto, en un sistema *software* de magnitud industrial el desarrollar un modelo antes de su construcción o renovación es esencial para la comunicación entre equipos de trabajo y para asegurar la solidez de la arquitectura.

Para el desarrollo de sistemas *software*, debemos abstraer diferentes vistas del sistema, construir modelos utilizando notaciones no ambiguas, verificar que los modelos satisfacen los requisitos del sistema, y añadir de forma gradual detalles que transforman el modelo en una implementación. En general los modelos nos ayudan a organizar, visualizar, comprender y crear sistemas complejos.

Una vez aceptada la necesidad de modelar el sistema, en muchas ocasiones dicho modelado se realiza informalmente. La importancia de la realización de un modelado formal es que las bases del modelado son comunes lo que conlleva una comunicación no ambigua entre los individuos que forman parte del proyecto e incluso con los usuarios.

Sin embargo, en el momento de enfrentarse a una realidad, los profesionales de informática tienden a realizar diversos diagramas para describir la realidad que se quiere modelar. El tipo de diagrama utilizado depende mucho de la formación que hayan adquirido estos profesionales.

En general los conceptos que se modelan en los diversos diagramas utilizados son los mismos. A causa de esto, a un mismo usuario se le muestran los mismos conceptos con distinta representación, causando así una mayor confusión, lo cual dificulta la comunicación entre el técnico y el usuario. A su vez se dificulta la comunicación entre los consultores, debido a que se utilizan diferentes nombres para los mismos conceptos.

En resumen la falta de estandarización en la manera de representar gráficamente un modelo impide que los diseños gráficos realizados se pudieran compartir fácilmente entre distintos diseñadores.

Se necesitaba por tanto un lenguaje no sólo para comunicar las ideas a otros desarrolladores sino también para servir de apoyo en los procesos de análisis de un problema. Con este objetivo se creo el Lenguaje Unificado de Modelamiento (UML: *Unified Modeling Language*). UML se ha convertido en ese estándar tan ansiado para representar y modelar la información con la que se trabaja en las fases de análisis y, especialmente, de diseño.

UML es una especificación de notación orientada a objetos. Se ha convertido en el estándar de facto de la industria, debido a que ha sido impulsado por los

autores de los tres métodos más usados de orientación a objetos: BOOCH, RUMBAUGH y JACOBSON.

UML divide cada proyecto en un número de diagramas que representan las diferentes vistas del proyecto. Estos diagramas juntos son los que representa la arquitectura del proyecto.

3.2.- UML COMO LENGUAJE

UML es ante todo un lenguaje. Un lenguaje proporciona un vocabulario y unas reglas para permitir una comunicación. En este caso, este lenguaje se centra en la representación gráfica de un sistema. Este lenguaje nos indica cómo crear y leer los modelos, pero no dice cómo crearlos. Esto último es el objetivo de las metodologías de desarrollo.

UML es un lenguaje de modelamiento, no un método. La mayoría de los métodos consisten, en principio, de dos cosas: un lenguaje de modelamiento y un proceso. El lenguaje de modelamiento es la notación que los métodos usan para expresar el diseño. El proceso son los pasos a seguir al hacer el diseño.

El lenguaje UML tiene una notación gráfica muy expresiva que permite representar en mayor o menor medida todas las fases de un proyecto informático: desde el análisis con los casos de uso, el diseño con los diagramas de clases, objetos, etc., hasta la implementación y configuración con los diagramas de despliegue.

Con UML nos debemos olvidar del protagonismo excesivo que se le da al diagrama de clases, este representa una parte importante del sistema, pero solo representa una vista estática, es decir muestra al sistema parado. Sabemos su estructura pero no sabemos que le sucede a sus diferentes partes cuando el sistema empieza a funcionar. UML introduce nuevos diagramas que representa una visión dinámica del sistema. Es decir, gracias al diseño de la parte dinámica

del sistema podemos darnos cuenta en la fase de diseño de problemas de la estructura al propagar errores o de las partes que necesitan ser sincronizadas, así como del estado de cada una de las instancias en cada momento. El diagrama de clases continua siendo muy importante, pero se debe tener en cuenta que su representación es limitada, y que ayuda a diseñar un sistema robusto con partes reutilizables, pero no a solucionar problemas de propagación de mensajes ni de sincronización o recuperación ante estados de error. En resumen, un sistema debe estar bien diseñado, pero también debe funcionar bien.

UML también intenta solucionar el problema de propiedad de código que se da con los desarrolladores, al implementar un lenguaje de modelado común para todos los desarrollos se crea una documentación también común, que cualquier desarrollador con conocimientos de UML será capaz de entender, independientemente del lenguaje utilizado para el desarrollo.

UML permite la modificación de todos sus miembros mediante estereotipos y restricciones. Un estereotipo nos permite indicar especificaciones del lenguaje al que se refiere el diagrama de UML. Una restricción identifica un comportamiento forzado de una clase o relación, es decir mediante la restricción estamos forzando el comportamiento que debe tener el objeto al que se le aplica.

UML ha puesto fin a las llamadas “guerras de métodos” que se han mantenido a lo largo de los 90, en las que los principales métodos sacaban nuevas versiones que incorporaban las técnicas de los demás. Con UML se fusiona la notación de estas técnicas para formar una herramienta compartida entre todos los ingenieros software que trabajan en el desarrollo orientado a objetos.

UML no tiene propietario y está basado en el común acuerdo de gran parte de la comunidad informática. No pretende ser un método de desarrollo completo, pues no incluye un proceso de desarrollo paso a paso; pero si incluye todos los conceptos que se consideran necesarios para utilizar un

proceso moderno iterativo, basado en construir una sólida arquitectura para resolver requisitos dirigidos por casos de uso.

UML necesita ser lo suficientemente expresivo para manejar todos los conceptos que se originan en un sistema moderno, así como también los mecanismos de la ingeniería de software.

Los objetivos de la unificación fueron: el mantenerlo simple, el quitar elementos de los lenguajes de Booch, OMT y OOSE que no funcionaran en la práctica, el añadir elementos de otros métodos que fueran mas efectivos y el inventar nuevas construcciones solamente cuando la solución existente no estuviera disponible.

Dentro de los objetivos principales de la creación de UML están:

- Posibilitar el intercambio de modelos entre las distintas herramientas CASE orientadas a objetos del mercado. Para ello era necesario definir una notación y semántica común. UML es ahora un Standard, no existe otra especificación de diseño orientado a objetos, ya que es el resultado de las tres opciones existentes en el mercado. Su utilización es independiente del lenguaje de programación y de las características de los proyectos, ya que UML ha sido diseñado para modelar cualquier tipo de proyectos, tanto informáticos como de arquitectura, o de cualquier otro ramo.
- Busca combinar y "unificar" las notaciones provenientes de los distintos mecanismos para:

1) Modelado Orientado a Objetos
2) Modelado de Datos
3) Modelado de Componentes
4) Modelado de Flujos de Trabajo (Workflows)

Los objetivos de las funciones de UML son muchos, pero se pueden sintetizar en:

- ***Visualizar:*** UML permite expresar de una forma gráfica un sistema de forma que otro lo puede entender.
- ***Especificar:*** UML permite especificar cuáles son las características de un sistema antes de su construcción.
- ***Construir:*** A partir de los modelos especificados se pueden construir los sistemas diseñados.
- ***Documentar:*** Los propios elementos gráficos sirven como documentación del sistema desarrollado que pueden servir para su futura revisión.

UML es además un método formal de modelado. Esto aporta las siguientes ventajas:

- Mayor rigor en la especificación.
- Permite realizar una verificación y validación del modelo realizado.
- Se pueden automatizar determinados procesos y permite generar código a partir de los modelos y a la inversa (a partir del código fuente generar los modelos). Esto permite que el modelo y el código estén actualizados, con lo que siempre se puede mantener la visión en el diseño, de más alto nivel, de la estructura de un proyecto.

UML está pensado para modelar sistemas complejos con gran cantidad de software, el lenguaje es los suficientemente expresivo como para modelar sistemas que no son informáticos, como flujos de trabajo (*workflow*) en una empresa, diseño de la estructura de una organización y por supuesto, en el diseño de hardware.

Aunque el UML define un leguaje preciso, no es una barrera para el desarrollo futuro en los conceptos de modelaje. Se han incorporado muchas técnicas líderes, pero se espera que técnicas adicionales influyan las versiones futuras del

UML. Muchas técnicas avanzadas pueden ser definidas usando el UML como base. El UML puede ser extendido sin redefinir su núcleo.

3.3.- HISTORIA DE UML

Con el advenimiento del mundo de la orientación a objetos (OO), mucha gente se preguntó como iban a encajar los métodos de diseño en ese mundo. Los métodos de diseño se hicieron muy populares en la década de los 70 y 80.

Muchos sintieron la necesidad de elaborar técnicas para ayudar a la gente a hacer un buen análisis y diseño. Todas estas técnicas eran bastante parecidas pero conservaban diferencias que hacían que un mismo concepto se llamara de dos formas distintas, lo cual causaba confusión en los clientes.

La necesidad de realizar un estándar demoró en prosperar debido a grandes oposiciones de gente que ya tenía sus propios métodos de diseño.

El lenguaje UML comenzó a gestarse en octubre de 1994, cuando Rumbaugh se unió a la compañía *Rational* fundada por Booch (dos acreditados investigadores en el área de metodología del software). El objetivo de ambos era unificar dos métodos que habían desarrollado: el método Booch y el OMT (*Object Modelling Tool*). El primer borrador apareció en octubre de 1995. En esa misma época otro conocido investigador, Jacobson, se unió a *Rational* y se incluyeron ideas suyas.

Estas tres personas son conocidas como los “tres amigos”. Además, este lenguaje se abrió a la colaboración de otras empresas para que aportaran sus ideas.

Todas estas colaboraciones condujeron a la definición de la primera versión de UML.

Esta primera versión se ofreció a un grupo de trabajo para convertirlo en 1997 en un estándar del OMG (*Object Management Group*). Este grupo, que gestiona estándares relacionados con la tecnología orientada a objetos (metodologías, bases de datos orientadas a objetos, etc.), propuso una serie de modificaciones y una nueva versión de UML (la 1.1), que fue adoptada por el OMG como estándar en noviembre de 1997.

Desde aquella versión ha habido varias revisiones que gestiona la *OMG Revision Task Force*. La última versión aprobada es la 1.4. En estos momentos se está desarrollando una nueva versión en la que se incluirán cambios importantes que conducirán a la versión 2.0.

En el proceso de creación de UML han participado, no obstante, otras empresas de gran peso en la industria como Microsoft, Hewlett-Packard, Oracle o IBM, así como grupos de analistas y desarrolladores.

Esta notación ha sido ampliamente aceptada debido al prestigio de sus creadores y debido a que incorpora las principales ventajas de cada uno de los métodos particulares en los que se basa.

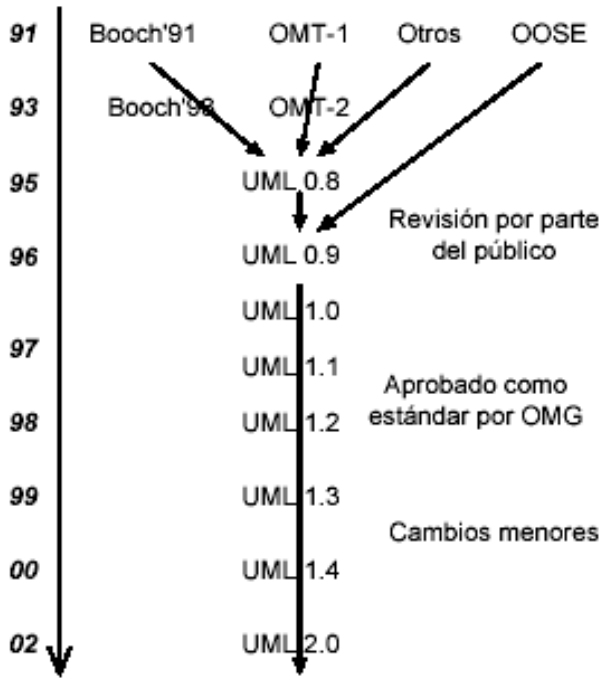


Figura 3.1.- Evolución de UML

3.4.- QUE ES UML

Es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Se usa para entender, diseñar, configurar, mantener y controlar la información sobre los sistemas a construir.

UML capta la información sobre la estructura estática y el comportamiento dinámico de un sistema. Un sistema se modela como una colección de objetos discretos que interactúan para realizar un trabajo que finalmente beneficia a un usuario externo.

El lenguaje de modelado pretende unificar la experiencia pasada sobre técnicas de modelado e incorporar las mejores prácticas actuales en un acercamiento estándar.

Las herramientas pueden ofrecer generadores de código de UML para una gran variedad de lenguaje de programación, así como construir modelos por ingeniería inversa a partir de programas existentes.

Es un lenguaje de propósito general para el modelado orientado a objetos. UML es también un lenguaje de modelamiento visual que permite una abstracción del sistema y sus componentes.

Existían diversos métodos y técnicas Orientadas a Objetos, con muchos aspectos en común pero utilizando distintas notaciones, se presentaban inconvenientes para el aprendizaje, aplicación, construcción y uso de herramientas, etc.

La necesidad de realizar un estándar demoró en prosperar debido a grandes oposiciones de gente que ya tenía sus propios métodos de diseño. Luego de varias idas y venidas es en enero de 1997, que varias organizaciones proponen un estándar para facilitar el intercambio entre los modelos. Este propósito se enfoco principalmente en un meta-modelo y opcionalmente una notación. Nace así UML.

3.5.- COMPRENDIENDO UML

La definición de UML se basará en los diagramas, en lugar de en vistas , ya que son estos la esencia de UML.

3.5.1.- DIAGRAMAS

Cada diagrama usa la notación pertinente y la suma de estos diagramas crean las diferentes vistas. Las vistas existentes en UML son:

- Vista casos de uso: Se forma con los diagramas de casos de uso, colaboración, estados y actividades.

- Vista de diseño: Se forma con los diagramas de clases, objetos, colaboración, estados y actividades.
- Vista de procesos: Se forma con los diagramas de la vista de diseño. Recalcando las clases y objetos referentes a procesos.
- Vista de implementación: Se forma con los diagramas de componentes, colaboración, estados y actividades.
- Vista de despliegue: Se forma con los diagramas de despliegue, interacción, estados y actividades.

Se dispone de dos tipos diferentes de diagramas los que dan una vista estática del sistema y los que dan una visión dinámica. Los diagramas estáticos son:

- Diagrama de clases: muestra las clases, interfaces, colaboraciones y sus relaciones. Son los más comunes y dan una vista estática del proyecto.
- Diagrama de objetos: Es un diagrama de instancias de las clases mostradas en el diagrama de clases. Muestra las instancias y como se relacionan entre ellas. Se da una visión de casos reales.
- Diagrama de componentes: Muestran la organización de los componentes del sistema. Un componente se corresponde con una o varias clases, interfaces o colaboraciones.
- Diagrama de despliegue.: Muestra los nodos y sus relaciones. Un nodo es un conjunto de componentes. Se utiliza para reducir la complejidad de los diagramas de clases y componentes de un gran sistema. Sirve como resumen e índice.
- Diagrama de casos de uso: Muestran los casos de uso, actores y sus relaciones. Muestra quien puede hacer que y relaciones existen entre acciones (casos de uso). Son muy importantes para modelar y organizar el comportamiento del sistema.

Lo diagramas dinámicos son:

- Diagrama de secuencia, Diagrama de colaboración: Muestran a los diferentes objetos y las relaciones que pueden tener entre ellos, los

mensajes que se envían entre ellos. Son dos diagramas diferentes, que se puede pasar de uno a otro sin perdida de información, pero que nos dan puntos de vista diferentes del sistema. En resumen, cualquiera de los dos es un Diagrama de Interacción.

- Diagrama de estados: muestra los estados, eventos, transiciones y actividades de los diferentes objetos. Son útiles en sistemas que reaccionen a eventos.
- Diagrama de actividades: Es un caso especial del diagrama de estados. Muestra el flujo entre los objetos. Se utilizan para modelar el funcionamiento del sistema y el flujo de control entre objetos.

Como podemos ver el número de diagramas es muy alto, en la mayoría de los casos excesivos, y UML permite definir solo los necesarios, ya que no todos son necesarios en todos los proyectos. En el documento se dará una breve explicación de todos, ampliándose para los más necesarios.

3.5.1.1.- Elementos Comunes a Todos los Diagramas

- **Notas.** Una nota sirve para añadir cualquier tipo de comentario a un diagrama o a un elemento de un diagrama. Es un modo de indicar información en un formato libre, cuando la notación del diagrama en cuestión no nos permite expresar dicha información de manera adecuada. Una nota se representa como un rectángulo con una esquina doblada con texto en su interior. Puede aparecer en un diagrama tanto solo como unida a un elemento por medio de una línea discontinua.

Puede contener restricciones, comentarios, el cuerpo de un procedimiento, etc.

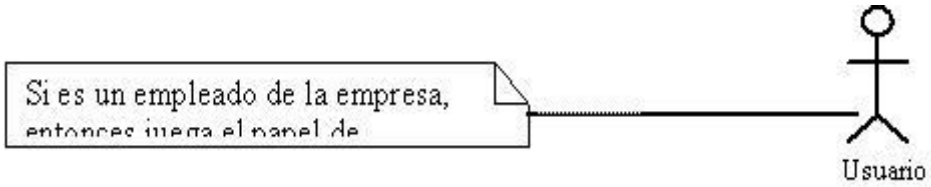


Figura 3.2 Nota

- **Dependencias.** La relación de dependencia entre dos elementos de un diagrama significa que un cambio en el elemento destino puede implicar un cambio en el elemento origen (por tanto, si cambia el elemento destino habría que revisar el elemento origen). Una dependencia se representa por medio de una línea de trazo discontinuo entre los dos elementos con una flecha en su extremo. El elemento dependiente es el origen de la flecha y el elemento del que depende es el destino (junto a él está la flecha).

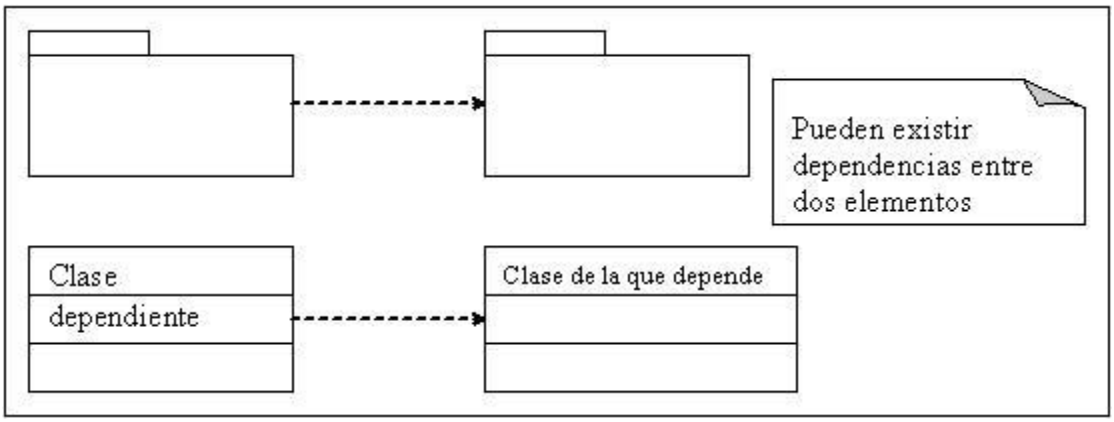


Figura 3.3 Dependencia

3.5.2.- DIAGRAMA DE CLASES

Forma parte de la vista estática del sistema. En el diagrama de clases como ya hemos comentado será donde definiremos las características de cada una de las clases, interfaces, colaboraciones y relaciones de dependencia y generalización. Es decir, es donde daremos rienda suelta a nuestros conocimientos de diseño orientado a objetos, definiendo las clases e implementando las ya típicas relaciones de herencia y agregación.

En el diagrama de clases debemos definir a estas y a sus relaciones.

3.5.2.1.- La Clase

Una clase esta representada por un rectángulo que dispone de tres apartados, el primero para indicar el nombre, el segundo para los atributos y el tercero para los métodos.

Cada clase debe tener un nombre único, que las diferencie de las otras.

Un atributo representa alguna propiedad de la clase que se encuentra en todas las instancias de la clase. Los atributos pueden representarse solo mostrando su nombre, mostrando su nombre y su tipo, e incluso su valor por defecto.

Un método es la implementación de un servicio de la clase, que muestra un comportamiento común a todos los objetos. En resumen es una función que le indica a las instancias de la clase que hagan algo.

Para separar las grandes listas de atributos y de métodos se pueden utilizar estereotipos.

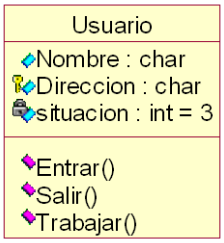


Figura 3.4 Lista de atributos clase usuario

La clase usuario contiene tres atributos. Nombre que es public, dirección que es protected y situación que es private. Situación empieza con el valor 3. También dispone de tres métodos Entrar, Salir y Trabajar.

3.5.2.2.- Relaciones entre clases

Existen tres relaciones diferentes entre clases, Dependencias, Generalización y Asociación. En las relaciones se habla de una clase destino y de una clase origen. El origen es desde la que se realiza la acción de relacionar. Es decir desde la que parte la flecha, el destino es el que recibe la flecha. Las relaciones se pueden modificar con estereotipos o con restricciones.

- **Dependencias.** Es una relación de uso, es decir una clase usa a otra, que la necesita para su cometido. Se representa con una flecha discontinua va desde la clase utilizadora a la clase utilizada. Con la dependencia mostramos que un cambio en la clase utilizada puede afectar al funcionamiento de la clase utilizadora, pero no al contrario.

  Aunque las dependencias se pueden crear tal cual, es decir sin ningún estereotipo (palabreja que aparece al lado de la línea que representa la dependencia) UML permite dar mas significado a las dependencias, es decir concretar mas, mediante el uso de estereotipos de relación Clase – objeto como:

  Bind: La clase utilizada es una plantilla, y necesita de parámetros para ser utilizada, con Bind se indica que la clase se instancia con los parámetros pasándole datos reales para sus parámetros.

  Derive: Se utiliza al indicar relaciones entre dos atributos, indica que el valor de un atributo depende directamente del valor de otro. Es decir el atributo edad depende directamente del atributo Fecha nacimiento.

  Friend: Especifica una visibilidad especial sobre la clase relacionada. Es decir podrá ver las interioridades de la clase destino.

  InstanceOF: Indica que el objeto origen es una instancia del destino.

Instantiate: indica que el origen crea instancias del destino.

Powertype: indica que el destino es un contenedor de objetos del origen, o de sus hijos.

Refine: se utiliza para indicar que una clase es la misma que otra, pero mas refinada, es decir dos vistas de la misma clase, la destino con mayor detalle.

- **Generalización.** Pues es la herencia, donde tenemos una o varias clases padre o superclase o madre, y una clase hija o subclase. UML soporta tanto herencia simple como herencia múltiple. Aunque la representación común es suficiente en el 99.73% de los casos UML nos permite modificar la relación de Generalización con un estereotipo y dos restricciones.

Estereotipos de generalización:

Implementación: El hijo hereda la implementación del padre, sin publicar ni soportar sus interfaces.

Restricciones de generalización.

Complete: La generalización ya no permite más hijos.

Incomplete: Podemos incorporar mas hijos a la generalización.

Disjoint: solo puede tener un tipo en tiempo de ejecución, una instancia del padre solo podrá ser de un tipo de hijo.

Overlapping: puede cambiar de tipo durante su vida, una instancia del padre puede ir cambiando de tipo entre los de sus hijos.

- **Asociación.** Especifica que los objetos de una clase están relacionados con los elementos de otra clase. Se representa mediante una línea continua, que

une las dos clases. Podemos indicar el nombre, multiplicidad en los extremos, su rol, y agregación.

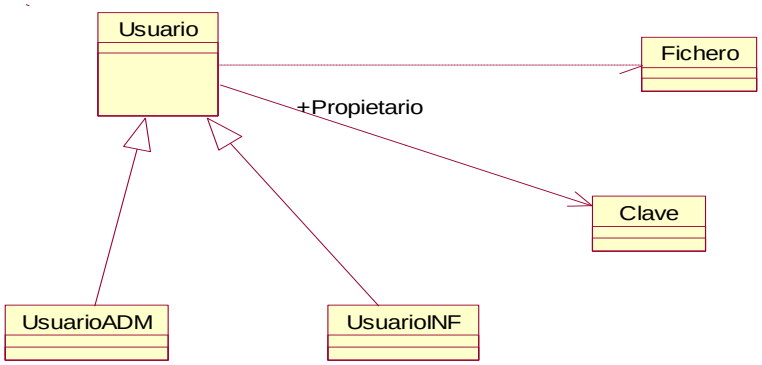


Figura 3. 5 Asociación

Ejemplo.

En este diagrama se han creado cuatro clases. La clase principal es Usuario, que tiene dos clases hijas UsuarioADM y UsuarioINF. El usuario mantiene una relación de asociación con la clase Clave, se indica que es propietario de una clave, o de un número indeterminado de ellas. Se le crea también una relación de dependencia con la clase Perfil, es decir las instancias de usuario contendrán como miembro una instancia de Perfil.

3.5.3.- DIAGRAMA DE OBJETOS

Forma parte de la vista estática del sistema. En este diagrama se modelan las instancias de las clases del diagrama de clases. Muestra a los objetos y sus relaciones, pero en un momento concreto del sistema. Estos diagramas contienen objetos y enlaces. En los diagramas de objetos también se pueden incorporar clases, para mostrar la clase de la que es un objeto representado.

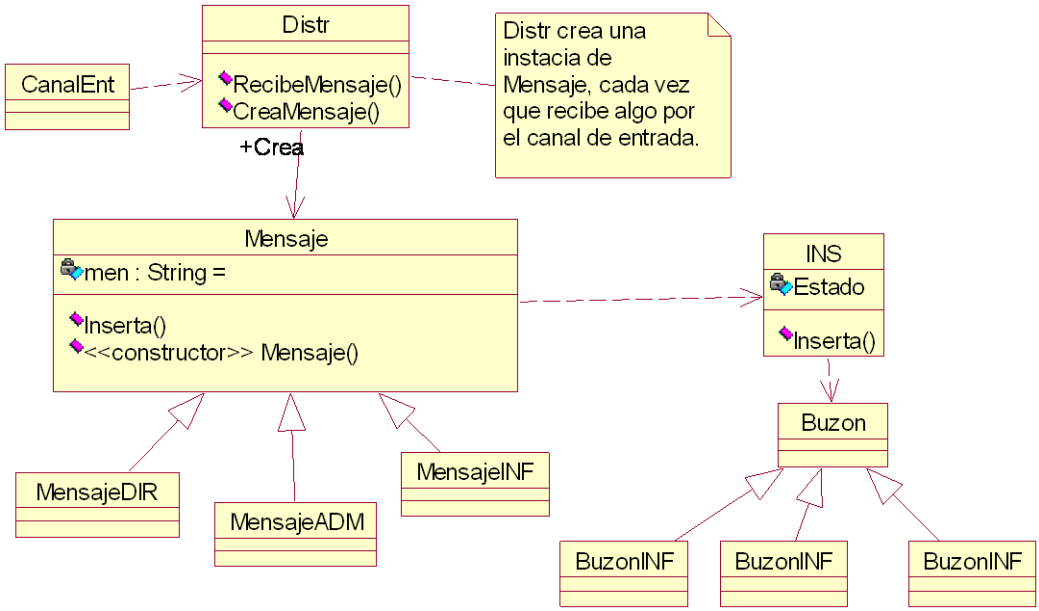


Figura 3.6 se muestra un estado del diagrama de eventos

En este diagrama figura 3.6 se muestra un estado del diagrama de eventos. Para realizar el diagrama de objetos primero se debe decidir que situación queremos representar del sistema. Es decir si disponemos de un sistema de mensajería, deberemos decidir que representaremos el sistema con dos mensajes entrantes, los dos para diferentes departamentos, dejando un departamento inactivo. Para el siguiente diagrama de clases:

Tendríamos un diagrama de objetos con dos instancias de Mensaje, mas concretamente con una instancia de MensajeDIR y otra de MensajeADM, con todos sus atributos valorados. También tendríamos una instancia de cada una de las otras clases que deban tener instancia. Como CanalEnt, INS, Distr, y el Buzon correspondiente a la instancia de mensaje que se este instanciando.

En la instancia de la clase INS se deberá mostrar en su miembro Estado, que esta ocupado realizando una inserción.

En un diseño no podemos encontrar con multitud de diagramas de objetos, cada uno de ellos representando diferentes estados del sistema.

3.5.4.- DIAGRAMA DE COMPONENTES

Se utilizan para modelar la vista estática de un sistema. Muestra la organización y las dependencias entre un conjunto de componentes. No es necesario que un diagrama incluya todos los componentes del sistema, normalmente se realizan por partes. Cada diagrama describe un apartado del sistema.

En el situaremos librerías, tablas archivos, ejecutables y documentos que formen parte del sistema.

Uno de los usos principales es que puede servir para ver que componentes pueden compartirse entre sistemas o entre diferentes partes de un sistema.

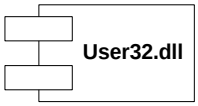


Figura 3.7 Componente del sistema de Windows

Aquí tenemos figura 3.7 un componente del sistema de Windows. En el diagrama de componentes de Windows debe salir este componente, ya que sin el sistema no funcionaría.

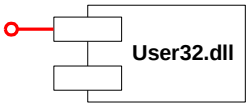


Figura 3.8 Componente del sistema de Windows

En esta otra figura tenemos el mismo componente, pero indicamos que dispone de un interface. Al ser una Dll el interface nos da acceso a su contenido. Esto nos hace pensar que la representación anterior es incorrecta, pero no es así solo corresponde a un nivel diferente de detalle.

Como ya hemos indicado antes todo objeto UML puede ser modificado mediante estereotipos, los standards que define UML son:

- Executable
- Library
- Table
- File
- Document.

Aunque por suerte no estamos limitados a estas especificaciones. Que pasa si queremos modelar un proyecto de Internet donde nuestros componentes son ASP, HTML, y Scripts, y queremos marcarlo en el modelo. Pues utilizamos un estereotipo. Existen ya unos definidos WAE (Web Applications Extensión).

Podemos modelar diferentes partes de nuestro sistema, y modelar diferentes entidades que no tiene nada que ver entre ellas.

- Ejecutables y bibliotecas.
- Tablas.
- API
- Código fuente.
- Hojas HTML.

- **Ejecutables.** Nos facilita la distribución de ejecutables a los clientes. Documenta sus necesidades y dependencias. Si disponemos de un ejecutable que solo se necesita a él mismo para funcionar no necesitaremos el diagrama de componentes.

  Los pasos a seguir para modelar, a priori no a posteriori, son:
  - Identificar los componentes, las particiones del sistema, cuales son factibles de ser reutilizadas. Agruparlos por nodos y realizar un diagrama por cada nodo que se quiera modelar.
  - Identificar cada componente con su estereotipo correspondiente.
  - Considerar las relaciones entre componentes.

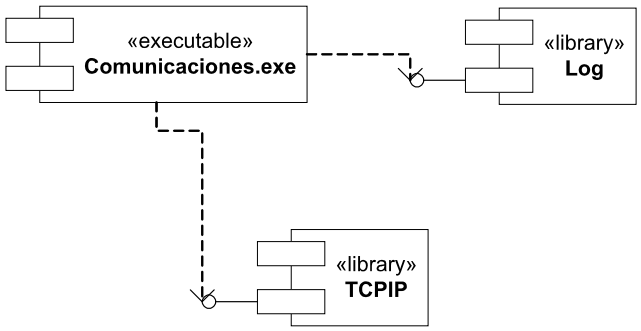


Figura 3.9 ejecutable que utiliza dos librerías

En esta Figura 3.9 se muestra un ejecutable que utiliza dos librerías, estas dos librerías disponen de su interface con el que ofrecen el acceso a sus servicios. Se puede ver que estas librerías son componentes que pueden ser reutilizados en otras partes del sistema.

- **Código fuente.** Se utiliza para documentar las dependencias de los diferentes ficheros de código fuente. Un ejecutable, o librería es una combinación de estos ficheros, y al mostrar la dependencia entre ellos obtenemos una visión de las partes necesarias para la creación del ejecutable o librería.

Al tener documentadas las relaciones se pueden realizar cambios en el código de un archivo teniendo en cuenta donde se utiliza, y que otros ficheros pueden verse afectados por su modificación.

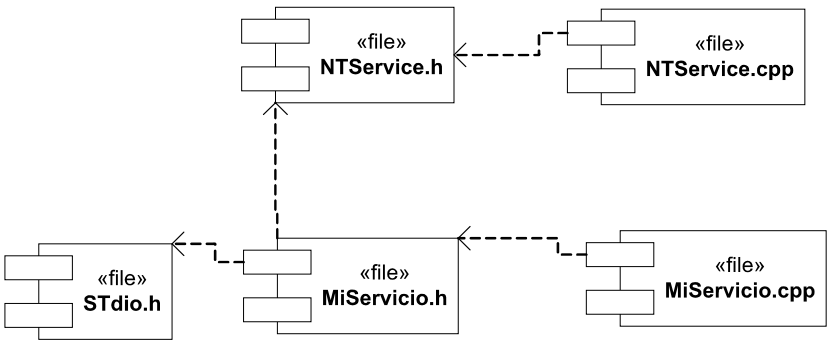


Figura 3.10 Relación entre ficheros del sistema

Aquí tenemos Figura 3.10 la relación entre los diferentes ficheros de un sistema. Cada fichero Cpp utiliza su fichero .h correspondiente, y MiServicio.h utiliza NTService.h u Stdio.h.

3.5.5.- DIAGRAMAS DE DESPLIEGUE

En el diagrama de despliegue se indica la situación física de los componentes lógicos desarrollados. Es decir se sitúa el software en el hardware que lo contiene. Cada Hardware se representa como un nodo.

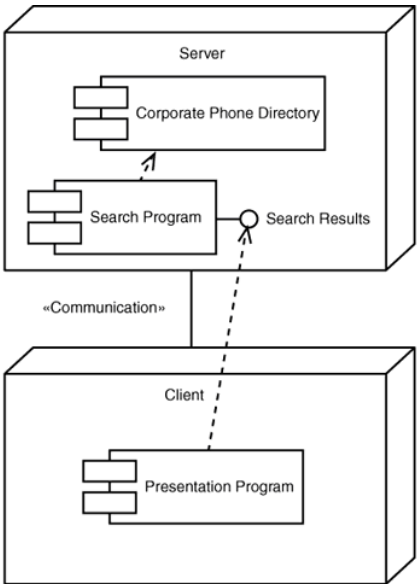


Figura 3.11 Diagrama de Despligue

Un nodo se representa como un cubo, un nodo es un elemento donde se ejecutan los componentes, representan el despliegue físico de estos componentes.

Aquí tenemos dos nodos, el cliente y el servidor, cada uno de ellos contiene componentes. El componente del cliente utiliza un interface de uno de los componentes del servidor. Se muestra la relación existente entre los dos Nodos. Esta relación podríamos asociarle un estereotipo para indicar que tipo de

conexión disponemos entre el cliente y el servidor, así como modificar su cardinalidad, para indicar que soportamos diversos clientes.

Como los componentes pueden residir en mas de un nodo podemos situar el componente de forma independiente, sin que pertenezca a ningún nodo, y relacionarlo con los nodos en los que se sitúa.

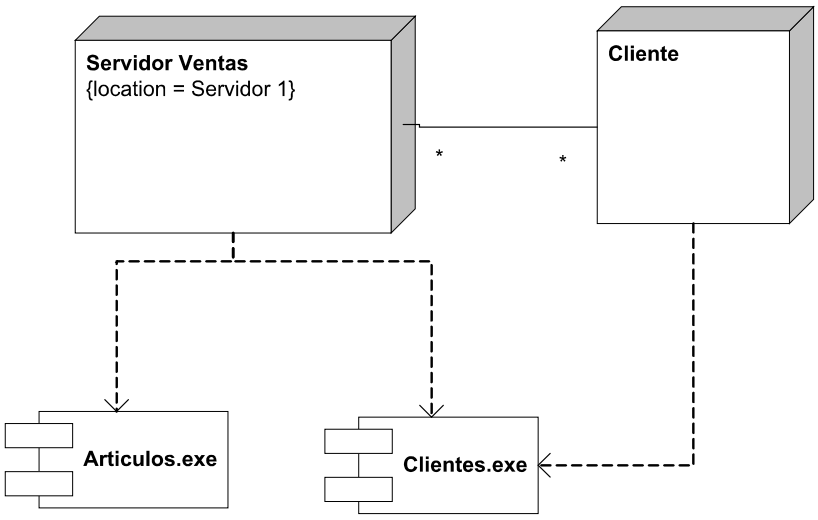


Figura 3.11 Componentes

3.5.6.- DIAGRAMA DE CASOS DE USO

Un Diagrama de Casos de Uso muestra la relación entre los actores y los casos de uso del sistema. Representa la funcionalidad que ofrece el sistema en lo que se refiere a su interacción externa. En el diagrama de casos de uso se representa también el sistema como una caja rectangular con el nombre en su interior. Los casos de uso están en el interior de la caja del sistema, y los actores fuera, y cada actor está unido a los casos de uso en los que participa mediante una línea. En la Figura 3.12 se muestra un ejemplo de Diagrama de Casos de Uso para un cajero automático.

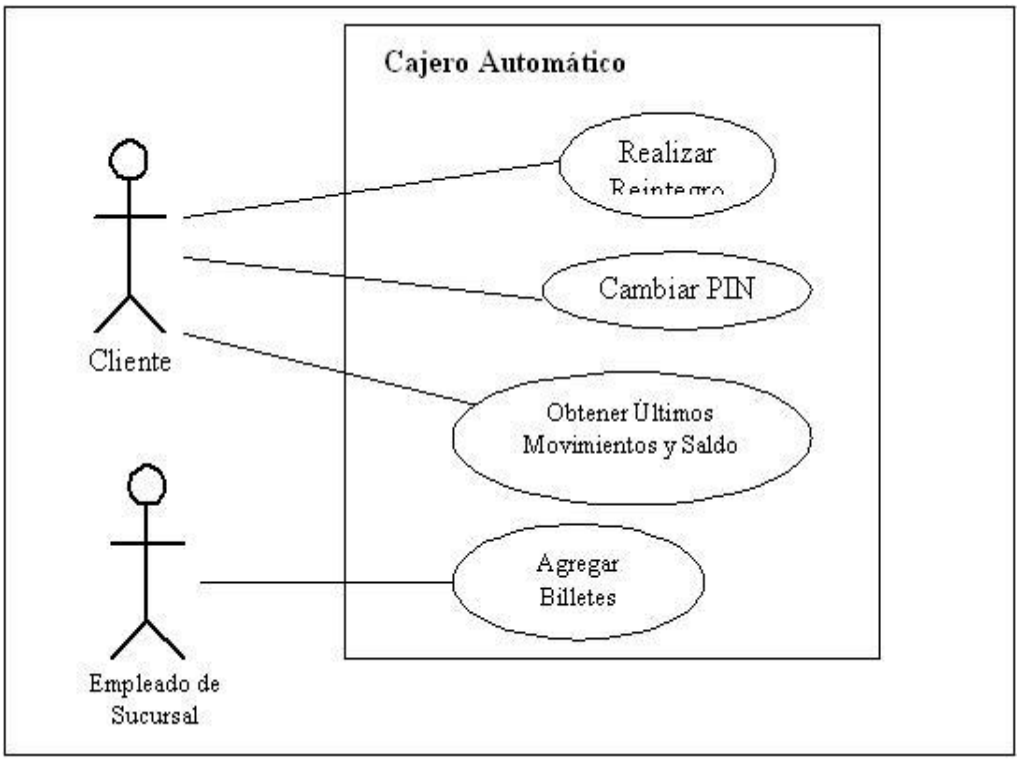


Figura 3.12 Diagrama de casos de uso

- **Elementos.** Los elementos que pueden aparecer en un Diagrama de Casos de Uso son: actores, casos de uso y relaciones entre casos de uso.
- **Actores.** Un actor es algo con comportamiento, como una persona (identificada por un rol), un sistema informatizado u organización, y que realiza algún tipo de interacción con el sistema.. Se representa mediante una figura humana dibujada con palotes. Esta representación sirve tanto para actores que son personas como para otro tipo de actores.
- **Casos de Uso.** Un caso de uso es una descripción de la secuencia de interacciones que se producen entre un actor y el sistema, cuando el actor usa el sistema para llevar a cabo una tarea específica. Expresa una unidad coherente de funcionalidad, y se representa en el Diagrama de Casos de Uso mediante una elipse con el nombre del caso de uso en su interior. El nombre del caso de uso debe reflejar la tarea específica que el actor desea llevar a cabo usando el sistema.

- **Modelado del contexto.** Se debe modelar la relación del sistema con los elementos externos, ya que son estos elementos los que forman el contexto del sistema.

Los pasos a seguir son Figura 3.13:

- Identificar los actores que interactúan con el sistema.
- Organizar a los actores.
- Especificar sus vías de comunicación con el sistema.

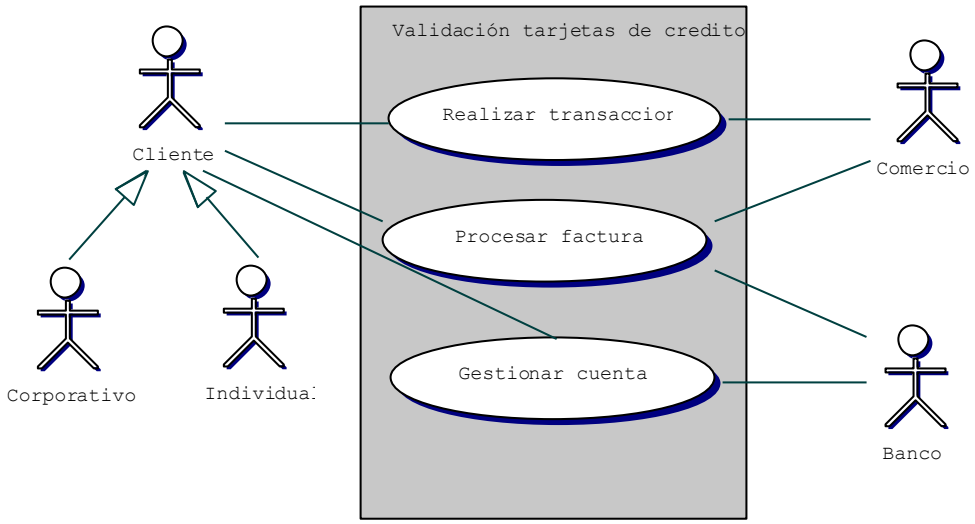


Figura 3.13 Relación del sistema con los elementos externos

- **Modelado de requisitos.** La función principal, o la mas conocida del diagrama de casos de uso es documentar los requisitos del sistema, o de una parte de el.

Los requisitos establecen un contrato entre el sistema y su exterior, definen lo que se espera que realice el sistema, sin definir su funcionamiento interno. Es el paso siguiente al modelado del contexto, no indica relaciones entre autores, tan solo indica cuales deben ser las funcionalidades (requisitos) del sistema. Se incorporan los casos de uso necesarios que no son visibles desde los usuarios del sistema.

Para modelar los requisitos es recomendable:

- Establecer su contexto, para lo que también podemos usar un diagrama de casos de uso.
- Identificar las necesidades de los elementos del contexto (Actores).
- Nombrar esas necesidades, y darles forma de caso de uso.
- Identificar que casos de uso pueden ser especializaciones de otros, o buscar especializaciones comunes para los casos de uso ya encontrados.

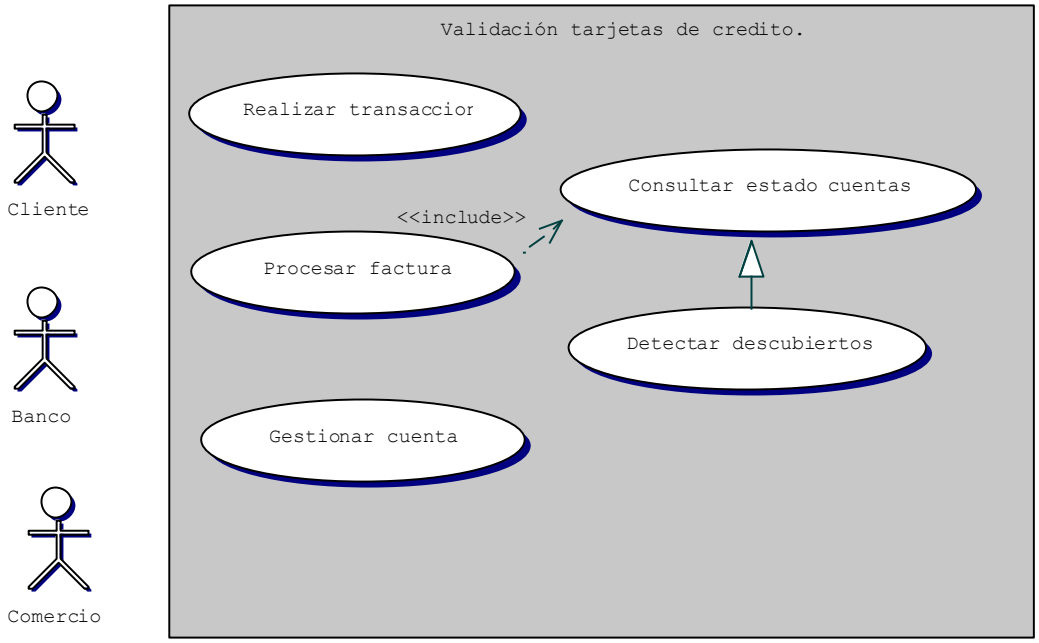


Figura 3.14 Modelamiento de los requisitos

Como podemos ver Figura 3.14 se incluyen nuevos casos de uso que no son visibles por ninguno de los actores del sistema, pero que son necesarios para el correcto funcionamiento.

- **Diagramas de Interacción.** En los diagramas de interacción se muestra un patrón de interacción entre objetos. Hay dos tipos de diagrama de interacción, ambos basados en la misma información, pero cada uno enfatizando un aspecto particular Diagramas de Secuencia y Diagramas de Colaboración.

3.5.7.- DIAGRAMAS DE SECUENCIA

El diagrama de secuencia forma parte del modelado dinámico del sistema. Se modelan las llamadas entre clases desde un punto concreto del sistema. Es útil para observar la vida de los objetos en sistema, identificar llamadas a realizar o posibles errores del modelado estático, que imposibiliten el flujo de información o de llamadas entre los componentes del sistema.

En el diagrama de secuencia se muestra el orden de las llamadas en el sistema. Se utiliza un diagrama para cada llamada a representar. Es imposible representar en un solo diagrama de secuencia todas las secuencias posibles del sistema, por ello se escoge un punto de partida. El diagrama se forma con los objetos que forman parte de la secuencia, estos se sitúan en la parte superior de la pantalla, normalmente en la izquierda se sitúa al que inicia la acción. De estos objetos sale una línea que indica su vida en el sistema. Esta línea simple se convierte en una línea gruesa cuando representa que el objeto tiene el foco del sistema, es decir cuando el esta activo.

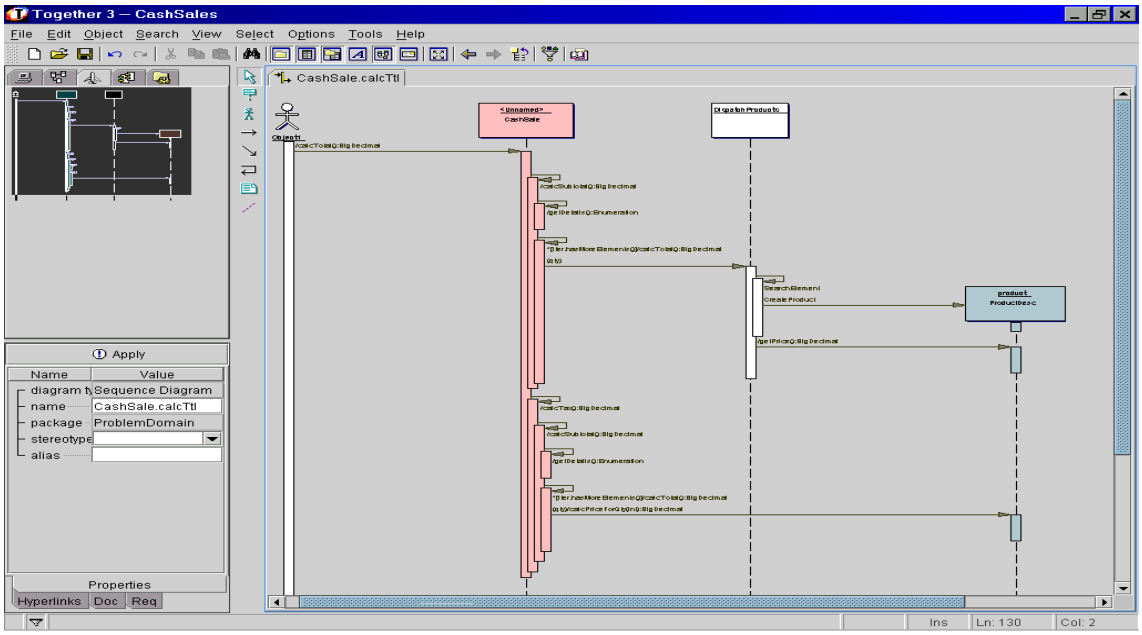


Figura 3.15 Diagrama de Secuencia

En esta Figura 3.15 pantalla tenemos tres objetos, y un actor, situado a la izquierda que es el que inicia la acción. Como podemos ver el objeto de mas a la derecha aparece mas abajo que los otros existentes. Esto se debe a que recibe una llamada de creación. Es decir el objeto no existe en el sistema hasta que recibe la primera petición.

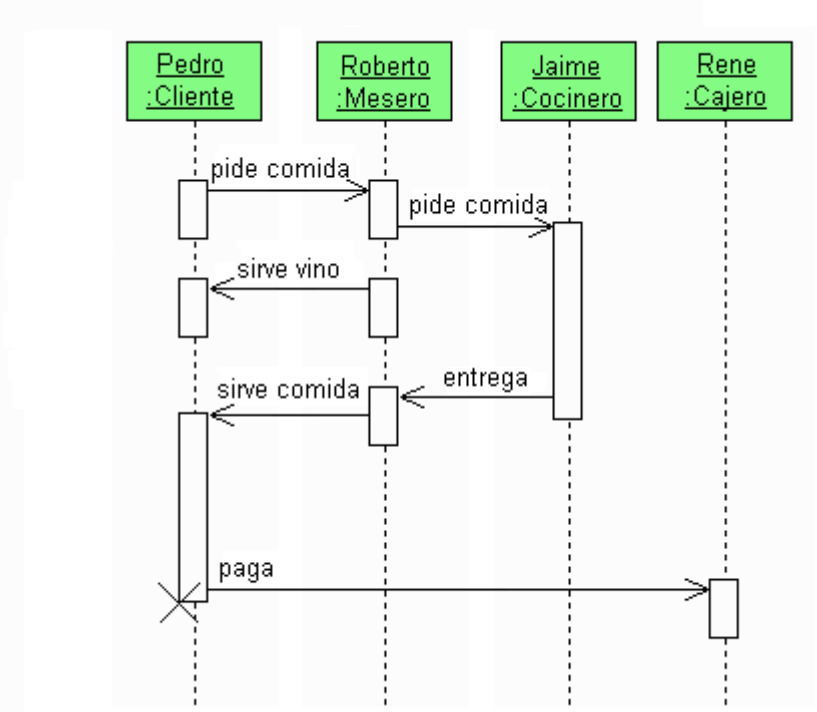


Figura 3.16 Diagrama de Secuencia

Figura 3.16 Este diagrama describe la secuencia (simplificada) de mensajes de un sistema de restaurante. El diagrama representa a un cliente pidiendo comida y pagando. Las líneas punteadas extendiéndose hacia abajo indican la **línea de tiempo** de cada objeto. Las flechas representan **mensajes** (estímulos) de un **actor** u **objeto** a otros objetos; en el ejemplo el cliente envía el mensaje de pago al cajero.

3.5.8.- DIAGRAMA DE COLABORACIÓN

Un Diagrama de Colaboración muestra una interacción organizada basándose en los objetos que toman parte en la interacción y los enlaces entre los mismos (en cuanto a la interacción se refiere). A diferencia de los Diagramas

de Secuencia, los Diagramas de Colaboración muestran las relaciones entre los roles de los objetos. La secuencia de los mensajes y los flujos de ejecución concurrentes deben determinarse explícitamente mediante números de secuencia.

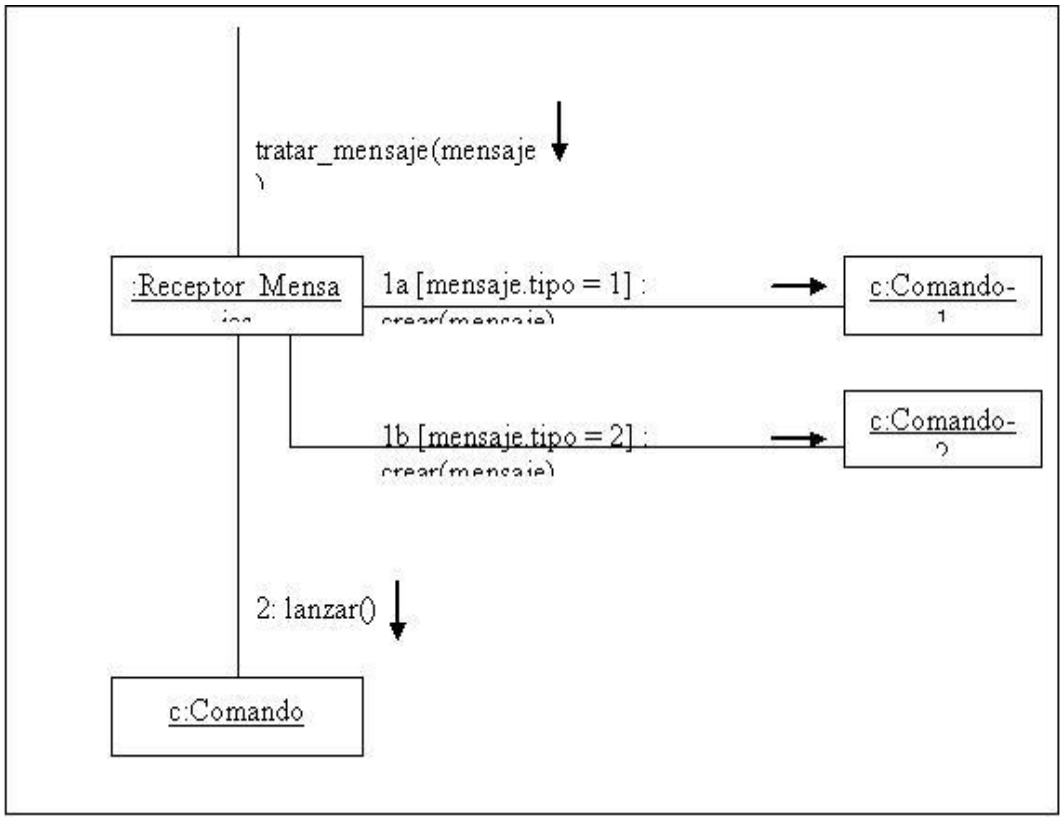


Figura 3.17 Diagrama de Colaboracion

En cuanto a la representación, un Diagrama de Colaboración muestra a una serie de objetos con los enlaces entre los mismos, y con los mensajes que se intercambian dichos objetos. Los mensajes son flechas que van junto al enlace por el que “circulan”, y con el nombre del mensaje y los parámetros (si los tiene) entre paréntesis. Cada mensaje lleva un número de secuencia que denota cuál es el mensaje que le precede, excepto el mensaje que inicia el diagrama, que no lleva número de secuencia.

Se pueden indicar alternativas con condiciones entre corchetes (por ejemplo [condición_de_test] : nombre_de_método()), tal y como aparece en el ejemplo de la Figura 3.17. También se puede mostrar el anidamiento de mensajes

con números de secuencia como 2.1, que significa que el mensaje con número de secuencia 2 no acaba de ejecutarse hasta que no se han ejecutado todos los 2. x

3.5.9.- DIAGRAMAS DE ESTADO

Un Diagrama de Estados muestra la secuencia de estados por los que pasa bien un caso de uso, bien un objeto a lo largo de su vida, o bien todo el sistema. En él se indican qué eventos hacen que se pase de un estado a otro y cuáles son las respuestas y acciones que genera.

En cuanto a la representación, un diagrama de estados es un grafo cuyos nodos son estados y cuyos arcos dirigidos son transiciones etiquetadas con los nombres de los eventos.

Un estado se representa como una caja redondeada con el nombre del estado en su interior. Una transición se representa como una flecha desde el estado origen al estado destino.

La caja de un estado puede tener 1 o 2 compartimentos. En el primer compartimento aparece el nombre del estado. El segundo compartimento es opcional, y en él pueden aparecer acciones de entrada, de salida y acciones internas. Una acción de entrada aparece en la forma entrada/acción_asociada donde acción_asociada es el nombre de la acción que se realiza al entrar en ese estado.

Cada vez que se entra al estado por medio de una transición la acción de entrada se ejecuta.

Una acción de salida aparece en la forma salida/acción_asociada. Cada vez que se sale del estado por una transición de salida la acción de salida se ejecuta. Una acción interna es una acción que se ejecuta cuando se recibe un determinado evento en ese estado, pero que no causa una transición a otro estado. Se indica en la forma nombre_de_evento/acción_asociada.

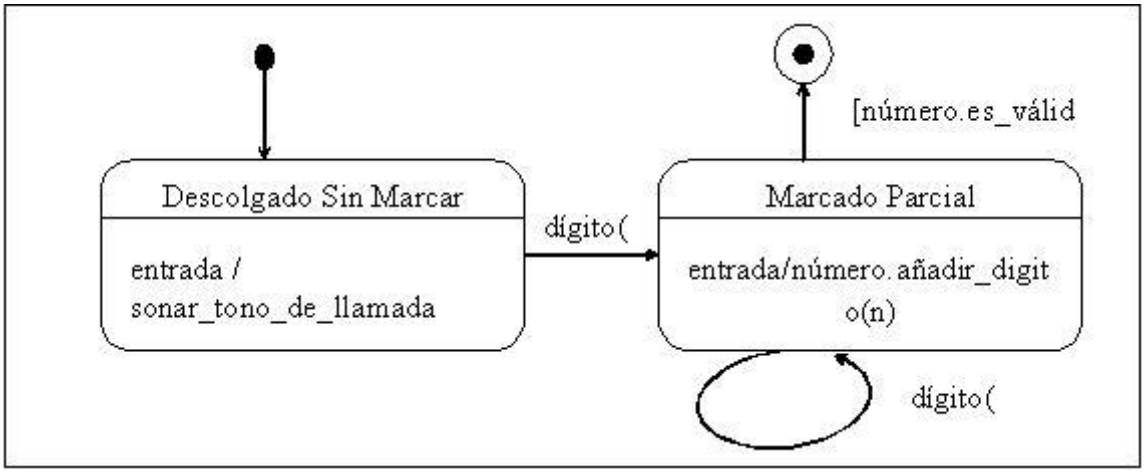


Figura 3.18 Diagrama de Estados.

Un diagrama de estados puede representar ciclos continuos, en la que hay un estado inicial de creación y un estado final de destrucción (finalización del caso de uso o destrucción del objeto). El estado inicial se muestra como un círculo sólido y el estado final como un círculo sólido rodeado de otro círculo. En realidad, los estados inicial y final son pseudoestados, pues un objeto no puede “estar” en esos estados, pero nos sirven para saber cuáles son las transiciones: inicial y final(es).

3.6.- NOTACION AVANZADA UML

3.6.1.- MODELADO DINAMICO

3.6.1.1.- Diagramas De Actividades

Vamos a recordar los diferentes modelos que sirven para representar el aspecto dinámico de un sistema:

- Diagramas de secuencia
- Diagramas de colaboración
- Diagramas de estados
- Diagramas de casos de uso
- Diagramas de actividades

Nos centraremos en los diagramas de actividades que sirven fundamentalmente para modelar el flujo de control entre actividades. La idea es generar una especie de diagrama Pert, en el que se puede ver el flujo de actividades que tienen lugar a lo largo del tiempo, así como las tareas concurrentes que pueden realizarse a la vez. El diagrama de actividades sirve para representar el sistema desde otra perspectiva, y de este modo complementa a los anteriores diagramas vistos. Gráficamente un diagrama de actividades será un conjunto de arcos y nodos. Desde un punto de vista conceptual, el diagrama de actividades muestra cómo fluye el control de unas clases a otras con la finalidad de culminar con un flujo de control total que se corresponde con la consecución de un proceso más complejo. Por este motivo, en un diagrama de actividades aparecerán acciones y actividades correspondientes a distintas clases.

Colaborando todas ellas para conseguir un mismo fin. Ejemplo: Hacer un pedido.

3.6.1.1.1.- Contenido del diagrama de actividades

Básicamente un diagrama de actividades contiene:

1) Estados de actividad
2) Estados de acción
3) Transiciones
4) Objetos

1) **Estados de actividad y estados de acción.** La representación de ambos es un rectángulo con las puntas redondeadas, en cuyo interior se representa bien una actividad o bien una acción. La forma de expresar tanto una actividad como una acción, no queda impuesta por UML, se podría utilizar lenguaje natural, una especificación formal de expresiones, un metalenguaje, etc. La idea central es la siguiente: “Un estado que

represente una acción es atómico, lo que significa que su ejecución se puede considerar instantánea y no puede ser interrumpida”

En la Figura 3.19, podemos ver ejemplos de estados de acción.

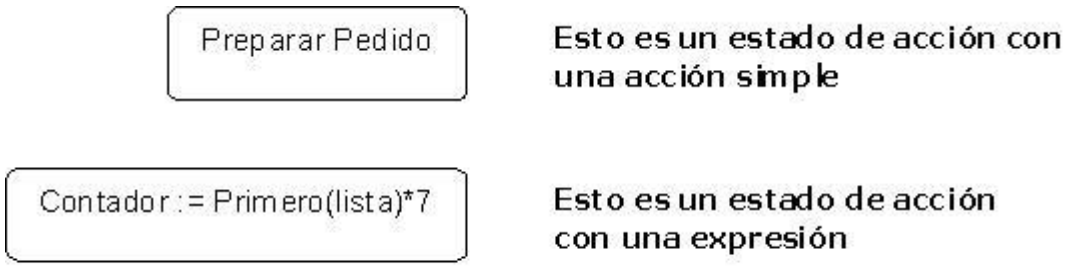


Figura 3.19 Estados de Acción

En cambio un estado de actividad, sí puede descomponerse en más sub-actividades representadas a través de otros diagramas de actividades. Además estos estados sí pueden ser interrumpidos y tardan un cierto tiempo en completarse. En los estados de actividad podemos encontrar otros elementos adicionales como son: acciones de entrada (entry) y de salida (exit) del estado en cuestión, así como definición de submáquinas, como podemos ver en la Figura 3.20

Figura 3.20 Estado de Actividad.

2) **Transiciones.** Las transiciones reflejan el paso de un estado a otro, bien sea de actividad o de acción. Esta transición se produce como resultado de la finalización del estado del que parte el arco dirigido que marca la transición. Como todo flujo de control debe empezar y terminar en algún momento, podemos indicar esto utilizando dos disparadores de inicio y fin tal y como queda reflejado en el ejemplo de la figura

Figura 3.21 Transiciones sin disparadores.

3) **Bifurcaciones.** Un flujo de control no tiene porqué ser siempre secuencial, puede presentar caminos alternativos. Para poder representar dichos caminos alternativos o bifurcación se utilizará como símbolo el rombo. Dicha bifurcación tendrá una transición de entrada y dos o más de salida. En cada transición de salida se colocará una expresión booleana que será evaluada una vez al llegar a la bifurcación, las guardas de la bifurcación han de ser excluyentes y contemplar todos los casos ya que de otro modo la ejecución del flujo de control quedaría interrumpida. Para poder cubrir todas las posibilidades se puede utilizar la palabra ELSE, para indicar una transición obligada a un determinado estado cuando el resto de guardas han fallado. En la Figura 3.22 podemos ver un ejemplo de bifurcación.

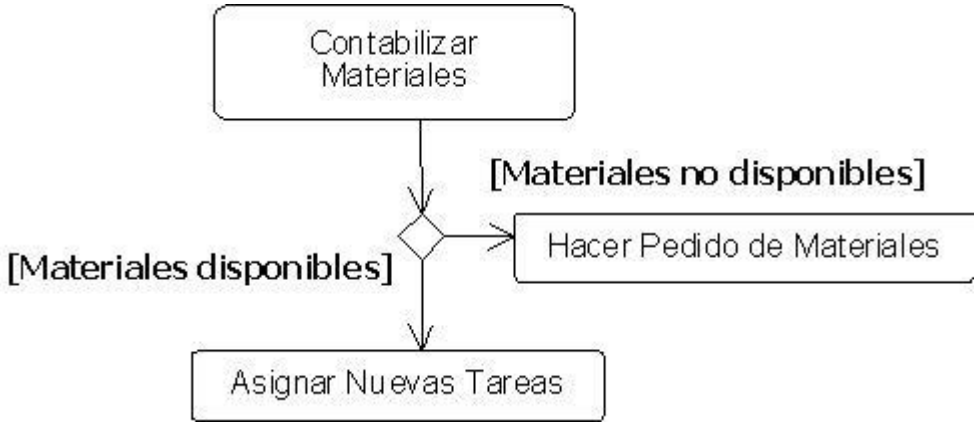


Figura 3.22 Bifurcación.

4) **División y unión.** No sólo existe el flujo secuencial y la bifurcación, también hay algunos casos en los que se requieren tareas concurrentes. UML representa gráficamente el proceso de división, que representa la concurrencia, y el momento de la unión de nuevo al flujo de control secuencial, por una línea horizontal ancha. En la Figura 3.23 podemos ver cómo se representa gráficamente.

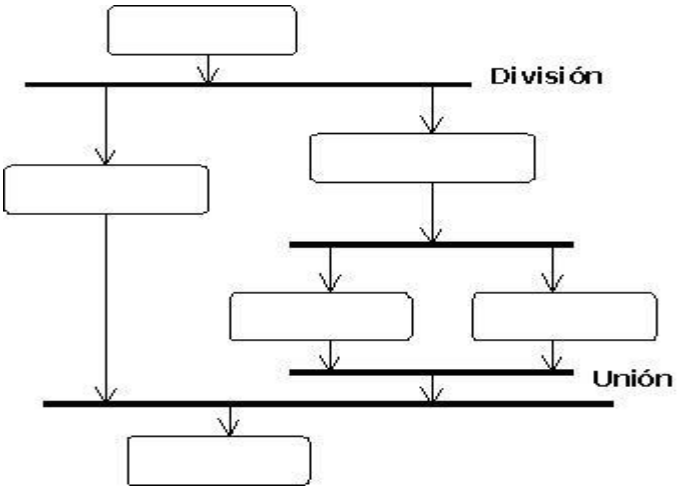


Figura 3.23 División y unión.

5) **Calles.** Cuando se modelan flujos de trabajo de organizaciones, es especialmente útil dividir los estados de actividades en grupos, cada grupo tiene un nombre concreto y se denominan calles. Cada calle representa a

la parte de la organización responsable de las actividades que aparecen en esa calle. Gráficamente quedaría como se muestra en la Figura 3.24

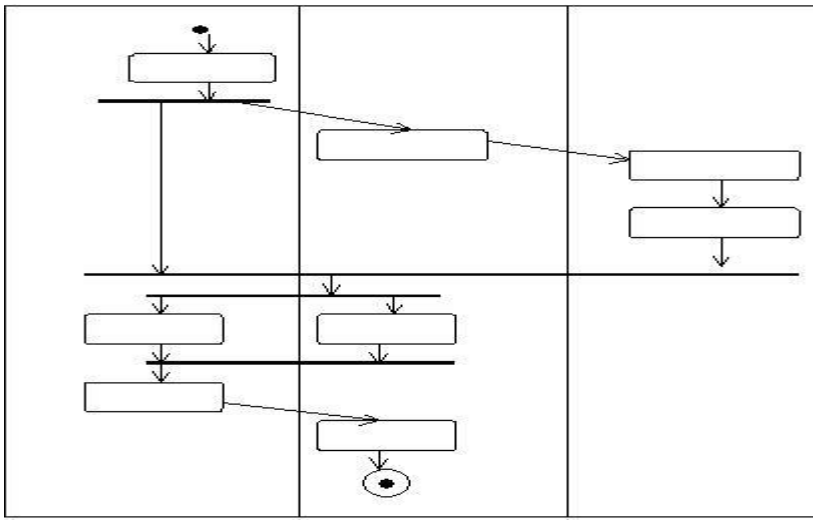


Figura 3.24 Calles

3.6.2.- MODELADO FÍSICO DE UN SISTEMA OO

Existen dos tipos de diagramas que sirven para modelar los aspectos físicos de un sistema orientado a objetos:

Diagramas de Componentes

Diagramas de Despliegue

Seguidamente veremos para qué sirve cada uno de ellos y cual es su representación gráfica.

3.6.2.1.- Diagramas De Componentes

Un diagrama de componentes muestra la organización y las dependencias entre un conjunto de componentes. Para todo sistema OO se han de construir una serie de diagramas que modelan tanto la parte estática (diagrama de clases), como dinámica (diagramas de secuencia, colaboración, estados y de actividades), pero llegado el momento todo esto se debe materializar en un sistema

implementado que utilizará partes ya implementadas de otros sistemas, todo esto es lo que pretendemos modelar con los diagramas de componentes.

3.6.2.1.1.- Algunos conceptos

Un diagrama de componentes muestra un conjunto de componentes y sus relaciones de manera gráfica a través del uso de nodos y arcos entre estos. Normalmente los diagramas de componentes contienen:

- Componentes
- Interfaces
- Relaciones de dependencia, generalización, asociaciones y realización.
- Paquetes o subsistemas
- Instancias de algunas clases.

Visto de otro modo un diagrama de componentes puede ser un tipo especial de diagrama de clases que se centra en los componentes físicos del sistema.

3.6.2.1.2.- Componentes

Los componentes pertenecen al mundo físico, es decir, representan un bloque de construcción al modelar aspectos físicos de un sistema.

Una característica básica de un componente es que: “debe definir una abstracción precisa con una interfaz bien definida, y permitiendo reemplazar fácilmente los componentes más viejos con otros más nuevos y compatibles.”

En UML todos los elementos físicos se modelan como componentes. UML proporciona una representación gráfica para estos como se puede ver en la Figura 3.25, en la que XXXX.dll, es el nombre del componente.

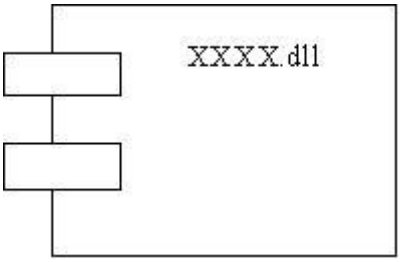


Figura 3.25 Representación de un componente

Cada componente debe tener un nombre que lo distinga de los demás. Al igual que las clases los componentes pueden enriquecerse con compartimentos adicionales que muestran sus detalles, como se puede ver en la Figura 3.26.

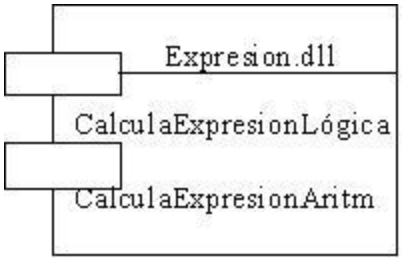


Figura 3.26 Representación extendida de un componente

Vamos a ver con más detalle cuáles son los parecidos y las diferencias entre los componentes y las clases.

- **PARECIDOS**
  - Ambos tienen nombre
  - Ambos pueden realizar un conjunto de interfaces.
  - Pueden participar en relaciones de dependencia, generalización y asociación.
  - Ambos pueden anidarse
  - Ambos pueden tener instancias
  - Ambos pueden participar en interacciones

- **DIFERENCIAS**
  - Las Clases Los Componentes

Representan abstracciones lógicas representan elementos físicos, es decir los componentes pueden estar en nodos y las clases no pueden tener atributos y operaciones indirectamente accesibles. Sólo tienen operaciones y estas son alcanzables a través de la interfaz del componente.

3.6.2.1.3.- Interfaces

Tanto los servicios propioS de una clase como los de un componente, se especifican a través de una Interfaz. Por ejemplo, todas las facilidades más conocidas de los sistemas operativos, basados en componentes (COM+, CORBA, etc.), utilizan las interfaces como lazo de unión entre unos componentes y otros.

La relación entre un componente y sus interfaces se puede representar de dos maneras diferentes, de forma icónica como se indica en la Figura 3.27, y de forma expandida como se muestra en la Figura 3.28.

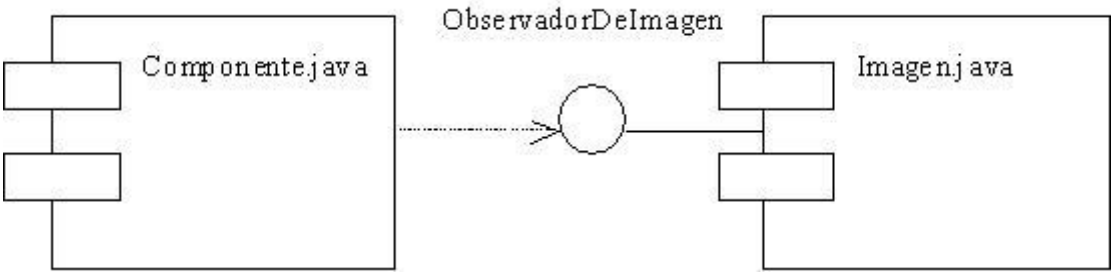


Figura 3 .27 Componentes e interfaces, formato icónico.

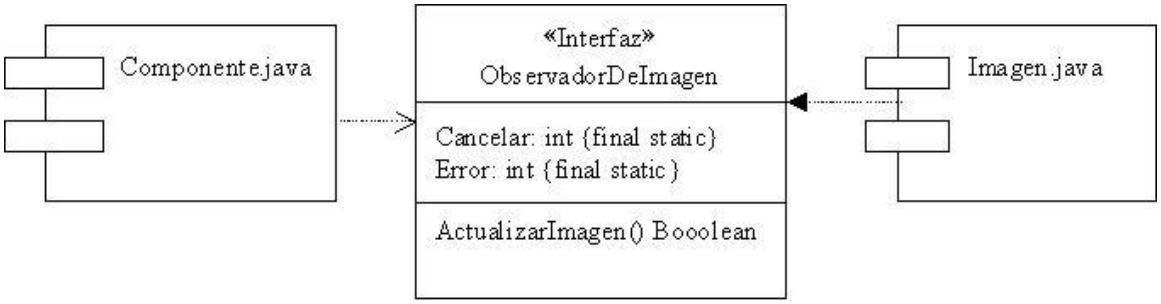


Figura 3.28 Componentes e interfaces, formato extendido.

3.6.2.1.4.- Tipos de componentes

Existen básicamente tres tipos de componentes:

1. Componentes de despliegue: componentes necesarios y suficientes para formar un sistema ejecutable, como pueden ser las bibliotecas dinámicas (DLLs) y ejecutables (EXEs).
2. Componentes producto del trabajo: estos componentes son básicamente productos que quedan al final del proceso de desarrollo. Consisten en cosas como archivos de código fuente y de datos a partir de los cuales se crean los componentes de despliegue.
3. Componentes de ejecución: son componentes que se crean como consecuencia de un sistema en ejecución. Es el caso de un objeto COM+ que se instancia a partir de una DLL.

3.6.2.1.5.- Organización de componentes

Los componentes se pueden agrupar en paquetes de la misma forma que se organizan las clases. Además se pueden especificar entre ellos relaciones de dependencia, generalización, asociación (incluyendo agregación), y realización.

3.6.2.1.6.- Estereotipos de componentes

UML define cinco estereotipos estándar que se aplican a los componentes:

- **Executable** Componente que se puede ejecutar en un nodo.
- **Library** Biblioteca de objetos estática o dinámica.
- **Table** Componentes que representa una tabla de una base de datos.
- **File** Componente que representa un documento que contiene código fuente o datos.
- **Document** Componente que representa un documento.

UML no especifica iconos predefinidos para estos estereotipos.

3.6.2.1.7.- Nodos

Al igual que los componentes los nodos pertenecen al mundo material. Vamos a definir un nodo como un elemento físico, que existe en tiempo de ejecución y representa un recurso computacional que generalmente tiene alguna memoria y, a menudo, capacidad de procesamiento. Los nodos sirven para modelar la topología del hardware sobre el que se ejecuta el sistema. Un nodo representa normalmente un procesador o un dispositivo sobre el que se pueden desplegar los componentes. Un nodo debe tener un nombre asignado que lo distinga del resto de nodos. Además los nodos se representan gráficamente como se indica en la Figura 3.29.

Figura 3.29 Nodos.

- **Nodos y componentes**

En muchos aspectos los nodos y los componentes tienen características parecidas. Vamos a ver con más detalle cuales son los parecidos y las diferencias entre los componentes y los nodos.

- **PARECIDOS**
- Ambos tienen nombre.
- Pueden participar en relaciones de dependencia, generalización y asociación.
- Ambos pueden anidarse
- Ambos pueden tener instancias
- Ambos pueden participar en interacciones

- **DIFERENCIAS**

- Los Nodos. Son los elementos donde se ejecutan los componentes.
- Los Componentes. Son los elementos que participan en la ejecución de un sistema.
- Representan el despliegue físico de los componentes.
- Representan el empaquetamiento físico de los elementos lógicos.
- La relación entre un nodo y los componentes que despliega se pueden representar mediante una relación de dependencia como se indica en la Figura 3.30
- Los nodos se pueden agrupar en paquetes igual que los las clases y los componentes.
- Los tipos de relación más común entre nodos es la asociación. Una asociación entre nodos viene a representar una conexión física entre nodos como se puede ver en la Figura 3.31.

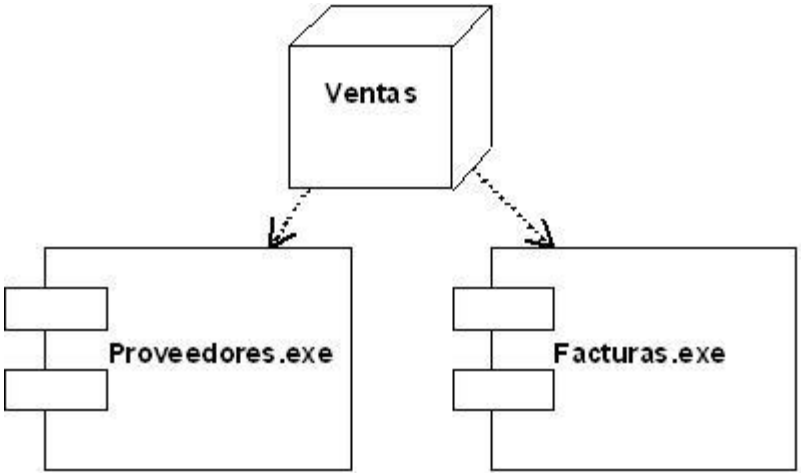


Figura 3.30 Relación entre nodos y componentes

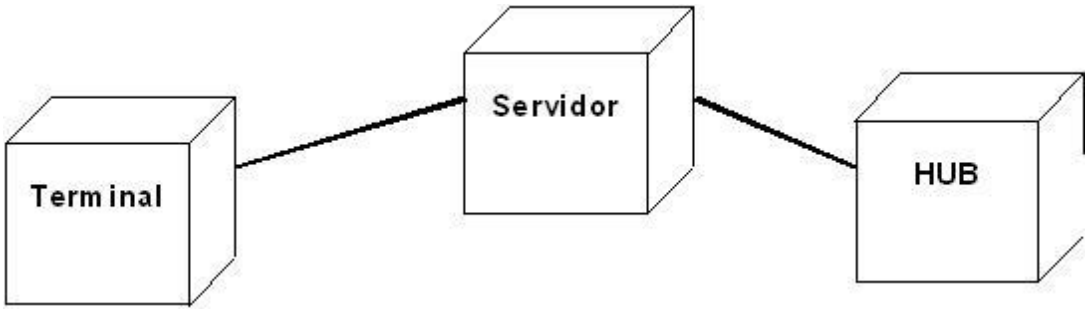


Figura 3.31 Conexiones entre nodos.

3.6.2.1.8.- Usos más comunes

a) Modelado de Código Fuente

Los diagramas de componentes se pueden utilizar para modelar la gestión de la configuración de los archivos de código fuente, tomando como productos de trabajo precisamente estos ficheros. Esto resulta bastante útil por ejemplo cuando se han implementado unas partes con Java otras con C, etc.

El resultado de esta implementación pueden ser multitud de ficheros ejecutables con características particulares, de manera que la mejor forma de controlarlos es estableciendo gestión de configuración.

Para poder llevar a cabo esta gestión con éxito será necesario definir los estereotipos de ficheros que se quieren tener bajo control así como las relaciones entre dichos tipos de ficheros.

Para modelar el código fuente de un sistema:

- Hay que identificar el conjunto de archivos de código fuente de interés y modelarlos como componentes estereotipados como archivos.
- Si el sistema es muy grande es necesario utilizar los paquetes para agrupar los archivos de código fuente.
- Es necesario identificar la versión del componente.

b) Modelado de una versión ejecutable y bibliotecas.

La utilización de los componentes para modelar versiones ejecutables se centra en la definición de todos los elementos que componen lo que se conoce como versión ejecutable, es decir la documentación, los ficheros que se entregan etc.

Para modelar una versión ejecutable es preciso:

- Identificar el conjunto de componentes que se pretende modelar.
- Identificar el estereotipo de cada componente del conjunto seleccionado.
- Para cada componente de este conjunto hay que considerar las relaciones con los vecinos. Esto implica definir las interfaces importadas por ciertos componentes y las exportadas por otros.

c) Modelado de una base de datos física

Para modelar una base de datos física es necesario:

• Identificar las clases del modelo que representan el esquema lógico de la base de datos.

- Seleccionar una estrategia para hacer corresponder las clases con tablas. Así como la distribución física de la/s base/s de datos.
- Para poder visualizar, especificar, construir y documentar dicha correspondencia es necesario crear un diagrama de componentes que tenga componentes estereotipados como tablas.
- Donde sea posible es aconsejable utilizar herramientas que ayuden a transformar diseño lógico en físico.

3.6.2.2.- Diagramas De Despliegue

3.6.2.2.1.- Técnicas más comunes de modelado

a) Modelado de un sistema empotrado

El desarrollo de un sistema empotrado es más que el desarrollo de un sistema software. Hay que manejar el mundo físico. Los diagramas de despliegue

son útiles para facilitar la comunicación entre los ingenieros de hardware y los de software.

Para modelar un sistema empotrado es necesario:

- Identificar los dispositivos y nodos propios del sistema.
- Proporcionar señales visuales, sobre todo para los dispositivos poco usuales.
- Modelar las relaciones entre esos procesadores y dispositivos en un diagrama de despliegue.
- Si es necesario hay que detallar cualquier dispositivo inteligente, modelando su estructura en un diagrama de despliegue más pormenorizado.

b) Modelado de un sistema cliente servidor

La división entre cliente y servidor en un sistema es complicada ya que implica tomar algunas decisiones sobre dónde colocar físicamente sus componentes software, qué cantidad de software debe residir en el cliente, etc. Para modelar un sistema cliente/servidor hay que hace lo siguiente:

- Identificar los nodos que representan los procesadores cliente y servidor del sistema.
- Destacar los dispositivos relacionados con el comportamiento del sistema.
- Proporcionar señales visuales para esos procesadores y dispositivos a través de estereotipos.
- Modelar la tipología de esos nodos mediante un diagrama de despliegue.

3.7.- ARQUITECTURA DEL SISTEMA

3.7.1.- ARQUITECTURA DE TRES NIVELES

La llamada “Arquitectura en Tres Niveles”, es la más común en sistemas de información que además de tener una interfaz de usuario contemplan la persistencia de los datos.

Una descripción de los tres niveles sería la siguiente:
Nivel 1: Presentación – ventanas, informes, etc.
Nivel 2: Lógica de la Aplicación – tareas y reglas que gobiernan el proceso.
Nivel 3: Almacenamiento – mecanismo de almacenamiento.

3.7.2.- ARQUITECTURA DE TRES NIVELES ORIENTADAS A OBJETOS

- **Descomposición del nivel de lógica de la aplicación**

En el diseño orientado a objetos, el nivel de lógica de la aplicación se descompone en sub-niveles que son los siguientes:

Objetos del Dominio: son clases que representan objetos del dominio. Por ejemplo en un problema de ventas, una “Venta” sería un objeto del dominio. Servicios: se hace referencia a funciones de interacción con la base de datos, informes, comunicaciones, seguridad, etc.

3.7.3.- ARQUITECTURA MULTI-NIVEL

La arquitectura de tres niveles puede pasar a llamarse de Múltiples Niveles si tenemos en cuenta el hecho de que todos los niveles de la arquitectura de tres niveles se pueden descomponer cada uno de ellos cada vez más. Por ejemplo el nivel de Servicios, se puede descomponer en servicios de alto y de bajo nivel, identificando como de alto nivel los servicios de generación de informes y como de bajo nivel los de manejo de ficheros de entrada y salida.

El motivo que lleva a descomponer la arquitectura del sistema en diferentes niveles es múltiple:

- Separación de la lógica de la aplicación en componentes separados que sean más fácilmente reutilizables.
- Distribución de niveles en diferentes nodos físicos de computación.
- Reparto de recursos humanos en diferentes niveles de la arquitectura.

- **Paquetes**

La forma que tiene UML de agrupar elementos en subsistemas es a través del uso de Paquetes, pudiéndose anidar los paquetes formando jerarquías de paquetes. De hecho un sistema que no tenga necesidad de ser descompuesto en subsistemas se puede considerar como con un único paquete que lo abarca todo. Gráficamente un paquete viene representado como se indica en la Figura 3.32.

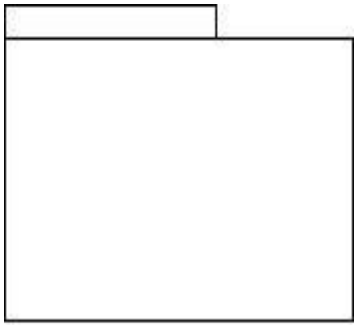


Figura 3.32 Representación de un paquete en UML.

En la Figura 3.33, vemos cómo se representa la arquitectura del sistema con la notación de paquetes.

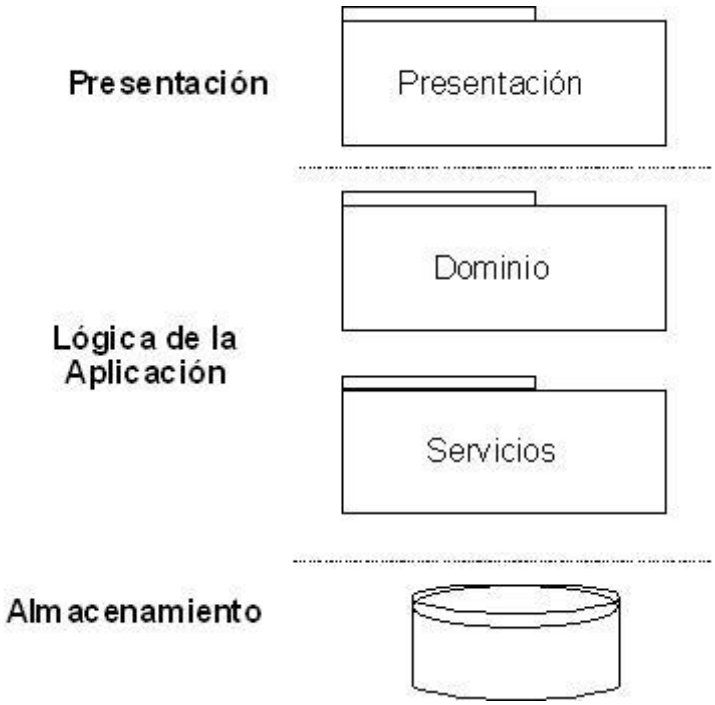


Figura 3.33 Arquitectura de un sistema utilizando paquetes.

- **Identificación de Paquetes**

Vamos a definir una serie de reglas que nos pueden ser de utilidad a la hora de agrupar los diferentes elementos en paquetes.

- Conviene agrupar elementos que proporcionen un mismo servicio.
- Los elementos que se agrupen en un mismo paquete han de presentar un alto grado de cohesión, es decir deben estar muy relacionados.
- Los elementos que estén en diferentes paquetes deben tener poca relación, es decir deben colaborar lo menos posible.

3.8.- DESARROLLO ORIENTADO A OBJETOS

3.8.1.- PROCESO DE DESARROLLO

El proceso a seguir para realizar desarrollo orientado a objetos es complejo, debido a la complejidad que nos vamos a encontrar al intentar desarrollar cualquier sistema software de tamaño medio-alto. El proceso está formado por una serie de actividades y subactividades, cuya realización se va repitiendo en el tiempo aplicado a distintos elementos.

En este apartado se va a presentar una visión general para poder tener una idea del proceso a alto nivel, y más adelante se verán los pasos que componen cada fase.

Las fases al nivel más alto son las siguientes:

- **Planificación y Especificación de Requisitos:** Planificación, definición de requisitos, construcción de prototipos, etc.
- **Construcción:** La construcción del sistema. Las fases dentro de esta etapa son las siguientes:
- **Diseño de Alto Nivel:** Se analiza el problema a resolver desde la perspectiva de los usuarios y de las entidades externas que van a solicitar servicios al sistema.
- **Diseño de Bajo Nivel**: El sistema se especifica en detalle, describiendo cómo va a funcionar internamente para satisfacer lo especificado en el Diseño de Alto Nivel.
- **Implementación**: Se lleva lo especificado en el Diseño de Bajo Nivel a un lenguaje de programación.
- **Pruebas**: Se llevan a cabo una serie de pruebas para corroborar que el software funciona correctamente y que satisface lo especificado en la etapa de Planificación y Especificación de Requisitos.
- **Instalación**: La puesta en marcha del sistema en el entorno previsto de uso.

De ellas, la fase de Construir es la que va a consumir la mayor parte del esfuerzo y del tiempo en un proyecto de desarrollo. Para llevarla a cabo se va adoptar un enfoque iterativo, tomando en cada iteración un subconjunto de los requisitos (agrupados según casos de uso) y llevándolo a través del diseño de alto y bajo nivel hasta la implementación y pruebas, tal y como se muestra en la Figura 3.34. El sistema va creciendo incrementalmente en cada ciclo. Con esta aproximación se consigue disminuir el grado de complejidad que se trata en cada ciclo, y se tiene pronto en el proceso una parte del sistema funcionando que se puede contrastar con el usuario / cliente.

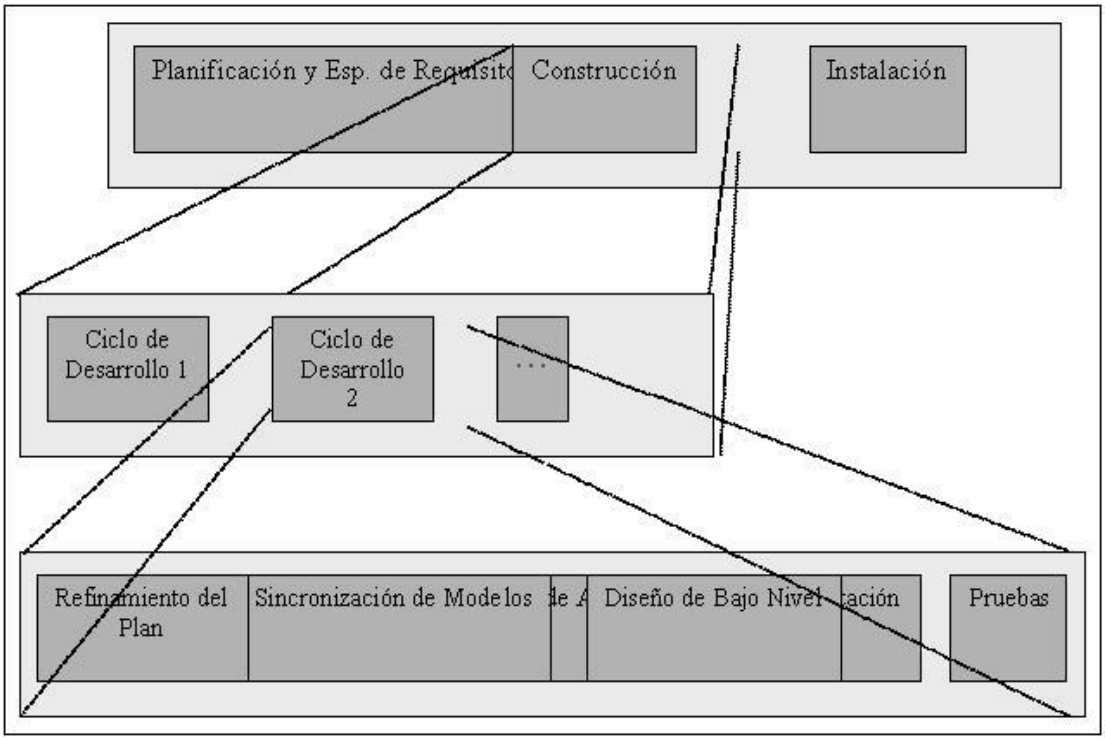


Figura 3.34 Desarrollo Iterativo en la Construcción

3.8.1.1.- Fase de Planificación y Especificación de Requisitos

Esta fase se corresponde con la Especificación de Requisitos tradicional ampliada con un Borrador de Modelo Conceptual y con una definición de Casos de Uso de alto nivel. En esta fase se decidiría si se aborda la construcción del

sistema mediante desarrollo orientado a objetos o no, por lo que, en principio, es independiente del paradigma empleado posteriormente.

- **Actividades**

Las actividades de esta fase son las siguientes:

- Definir el Plan-Borrador.
- Crear el Informe de Investigación Preliminar.
- Definir los Requisitos.
- Registrar Términos en el Glosario. (Continuado en posteriores fases)
- Implementar un Prototipo. (Opcional)
- Definir Casos de Uso (de alto nivel y esenciales).
- Definir el Modelo Conceptual-Borrador. (Puede retrasarse hasta una fase posterior)
- Definir la Arquitectura del Sistema-Borrador. (Puede retrasarse hasta una fase posterior)
- Refinar el Plan.

El orden no es estricto, lo normal es que las distintas actividades se solapen en el tiempo. Esto sucede también en las actividades de las fases de diseño, que se verán más adelante.

De estas actividades no se va a entrar en las que corresponden al campo de la planificación de proyectos software, como las correspondientes a creación de planes e informes preliminares.

- **Requisitos**

El formato del documento de Especificación de Requisitos no está definido en UML, pero se ha incluido este punto para resaltar que la actividad de definición de requisitos es un paso clave en la creación de cualquier producto software. Para

entender y describir los requisitos la técnica de casos de uso constituye una valiosa ayuda.

- **Casos de Uso**

Un Caso de Uso es un documento narrativo que describe a los actores utilizando un sistema para satisfacer un objetivo. Es una historia o una forma particular de usar un sistema. Los casos de uso son requisitos, en particular requisitos funcionales.

Nótese que UML no define un formato para describir un caso de uso. Tan sólo define la manera de representar la relación entre actores y casos de uso en un diagrama: El Diagrama de Casos de Uso, definido en la página 9. Sin embargo, un caso de uso individual no es un diagrama, es un documento de texto. En la siguiente sección se define el formato textual para la descripción de un caso de uso que se va a utilizar en este documento..

Un escenario es un camino concreto a través del caso de uso, una secuencia específica de acciones e interacciones entre los actores y el sistema. Más adelante se verá la representación de los escenarios correspondientes a un caso de uso por medio de Diagramas de Secuencia.

En un primer momento interesa abordar un caso de uso desde un nivel de abstracción alto, utilizando el formato de alto nivel. Cuando se quiere describir un caso de uso con más detalle se usa el formato expandido.

- **Casos de Uso de Alto Nivel**

El siguiente Caso de Uso de Alto Nivel describe el proceso de sacar dinero cuando se está usando un cajero automático:

Caso de Uso: Realizar Reintegro

Actores: Cliente

Tipo: primario.

Descripción: Un Cliente llega al cajero automático, introduce la tarjeta, se identifica y solicita realizar una operación de reintegro por una cantidad específica. El cajero le da el dinero solicitado tras comprobar que la operación puede realizarse. El Cliente coge el dinero y la tarjeta y se va.

En un caso de uso descrito a alto nivel la descripción es muy general, normalmente se condensa en dos o tres frases. Es útil para comprender el ámbito y el grado de complejidad del sistema.

- **Casos de Uso Expandidos**

Los casos de uso que se consideren los más importantes y que se considere que son los que más influencian al resto, se describen a un nivel más detallado: en el formato expandido.

La principal diferencia con un caso de uso de alto nivel está en que incluye un apartado de Curso Típico de Eventos, pero también incluye otros apartados como se ve en el siguiente ejemplo:

Caso de Uso: Realizar Reintegro

Actores: Cliente (iniciador)

Propósito: Realizar una operación de reintegro de una cuenta del banco

Visión General: Un Cliente llega al cajero automático, introduce la tarjeta, se identifica y solicita realizar una operación de reintegro por una cantidad específica. El cajero le da el dinero solicitado tras comprobar que la operación puede realizarse. El Cliente coge el dinero y la tarjeta y se va.

Tipo: primario y esencial

Referencias: Funciones: R1.3, R1.7

Curso Típico de Eventos:

Acción del Actor Respuesta del Sistema

1. Este caso de uso empieza cuando un Cliente introduce una tarjeta en el cajero.
2. Pide la clave de identificación.
3. Introduce la clave.
4. Presenta las opciones de operaciones disponibles.
5. Selecciona la operación de Reintegro.
6. Pide la cantidad a retirar.
7. Introduce la cantidad requerida.
8. Procesa la petición y da el dinero solicitado. Devuelve la tarjeta y genera un recibo.
9. Recoge la tarjeta.
10. Recoge el recibo.
11. Recoge el dinero y se va.
Cursos Alternativos:
• Línea 4: La clave es incorrecta. Se indica el error y se cancela la operación.
• Línea 8: La cantidad solicitada supera el saldo. Se indica el error y se cancela la operación.

El significado de cada apartado de este formato es como sigue:

- Caso de Uso: Nombre del Caso de Uso.
- Actores: Lista de actores (agentes externos), indicando quién inicia el caso de uso. Los actores son normalmente roles que un ser humano desempeña, pero puede ser cualquier tipo de sistema.
- Propósito: Intención del caso de uso.
- Visión General: Repetición del caso de uso de alto nivel, o un resumen similar.
- Tipo:
  1) primario, secundario u opcional (descritos más adelante).
  2) esencial o real (descritos más adelante).
- Referencias: Casos de uso relacionados y funciones del sistema que aparecen en los requisitos.

- Curso Típico de Eventos: Descripción de la interacción entre los actores y el sistema mediante las acciones numeradas de cada uno. Describe la secuencia más común de eventos, cuando todo va bien y el proceso se completa satisfactoriamente.

En caso de haber alternativas con grado similar de probabilidad se pueden añadir secciones adicionales a la sección principal, como se verá más adelante.

- Cursos Alternativos: Puntos en los que puede surgir una alternativa, junto con la descripción de la excepción.

- **Identificación de Casos de Uso**

La identificación de casos de uso requiere un conocimiento medio acerca de los requisitos, y se basa en la revisión de los documentos de requisitos existentes, y en el uso de la técnica de brainstorming entre los miembros del equipo de desarrollo.

Como guía para la identificación inicial de casos de uso hay dos métodos:

a) Basado en Actores

1. Identificar los actores relacionados con el sistema y/o la organización.
2. Para cada actor, identificar los procesos que inicia o en los que participa.

b) Basado en Eventos

1. Identificar los eventos externos a los que el sistema va a tener que responder.
2. Relacionar los eventos con actores y casos de uso.

Ejemplos de casos de uso:

- Pedir un producto.

- Matricularse en un curso de la facultad.
- Comprobar la ortografía de un documento en un procesador de textos.
- Comprar un libro en una tienda de libros en Internet

- **Identificación de los Límites del Sistema**

En la descripción de un caso de uso se hace referencia en todo momento al “sistema”. Para que los casos de uso tengan un significado completo es necesario que el sistema esté definido con precisión.

Al definir los límites del sistema se establece una diferenciación entre lo que es interno y lo que es externo al sistema. El entorno exterior se representa mediante los actores.

Ejemplos de sistemas son:
El hardware y software de un sistema informático.

- Un departamento de una organización.
- Una organización entera.

Si no se está haciendo reingeniería del proceso de negocio lo más normal es escoger como sistema el primero de los ejemplos: el hardware y el software del sistema que se quiere construir.

- **Tipos de Casos de Uso**

a) **Según Importancia**

Para establecer una primera aproximación a la priorización de casos de uso que identifiquemos los vamos a distinguir entre:

- Primarios: Representan los procesos principales, los más comunes, como Realizar Reintegro en el caso del cajero automático.

- Secundarios: Representan casos de uso menores, que van a necesitarse raramente, tales como Añadir Nueva Operación.
- Opcionales: Representan procesos que pueden no ser abordados en el presente proyecto.

b) Según el Grado de Compromiso con el Diseño

En las descripciones que se han visto anteriormente no se han hecho apenas compromisos con la solución, se han descrito los casos de uso a un nivel abstracto, independiente de la tecnología y de la implementación. Un caso de uso definido a nivel abstracto se denomina esencial. Los casos de uso definidos a alto nivel son siempre esenciales por naturaleza, debido a su brevedad y abstracción.
Por el contrario, un caso de uso real describe concretamente el proceso en términos del diseño real, de la solución específica que se va a llevar a cabo. Se ajusta a un tipo de interfaz específica, y se baja a detalles como pantallas y objetos en las mismas.

Como ejemplo de una parte de un Caso de Uso Real para el caso del reintegro en un cajero automático tenemos la siguiente descripción del Curso Típico de Eventos:

Acción del Actor Respuesta del Sistema

1) Este caso de uso empieza cuando un Cliente introduce una tarjeta en la ranura para tarjetas.
2) Pide el PIN (Personal Identification Number).
3) Introduce el PIN a través del teclado numérico.
4) Presenta las opciones de operaciones disponibles.
5) etc.

En principio, los casos de uso reales deberían ser creados en la fase de Diseño de Bajo Nivel y no antes. Sin embargo, en algunos proyectos se plantea la definición de interfaces en fases tempranas del ciclo de desarrollo, en base a que

son parte del contrato. En este caso se pueden definir algunos o todos los casos de uso reales, a pesar de que suponen tomar decisiones de diseño muy pronto en el ciclo de vida.

No hay una diferencia estricta entre un Caso de Uso Esencial y uno Real, el grado de compromiso con el diseño es un continuo, y una descripción específica de un caso de uso estará situada en algún punto de la línea entre Casos de Uso Esenciales y Reales, normalmente más cercano a un extremo que al otro, pero es raro encontrar Casos de Uso Esenciales o Reales puros.

- **Consejos Relativos a Casos de Uso**

a) Nombre

El nombre de un Caso de Uso debería ser un verbo, para enfatizar que se trata de un proceso, por ejemplo: Comprar Artículos o Realizar Pedido.

b) Alternativas equiprobables

Cuando se tiene una alternativa que ocurre de manera relativamente ocasional, se indica en el apartado Cursos Alternativos. Pero cuando se tienen distintas opciones, todas ellas consideradas normales se puede completar el Curso Típico de Eventos con secciones adicionales.

Así, si en un determinado número de línea hay una bifurcación se pueden poner opciones que dirigen el caso de uso a una sección que se detalla al final del Curso Típico de Eventos, en la siguiente forma:

Curso Típico de Eventos:
Sección: Principal

Acción del Actor Respuesta del Sistema

1. Este caso de uso empieza cuando Actor llega al sistema.
2. Pide la operación a realizar.
3. Escoge la operación A.
4. Presenta las opciones de pago.
5. Selecciona el tipo de pago:

a. Si se paga al contado ver sección Pago al Contado.

b. Si se paga con tarjeta ver sección Pago con Tarjeta.

6. Genera recibo.
7. Recoge el recibo y se va.

Cursos Alternativos:

• Líneas 3 y 5: Selecciona Cancelar. Se cancela la operación.

- Sección: Pago al Contado

Acción del Actor Respuesta del Sistema

1. Mete los billetes correspondientes
2. Coge los billetes y sigue pidiendo dinero hasta que la cantidad está bien
3. Devuelve el cambio.

Cursos Alternativos:

• Línea 3: No hay cambio suficiente. Se cancela la operación.

Sección: Pago con Tarjeta....

- **Construcción del Modelo de Casos de Uso**

Para construir el Modelo de Casos de Uso en la fase de Planificación y Especificación de Requisitos se siguen los siguientes pasos:

1) Después de listar las funciones del sistema, se definen los límites del sistema y se identifican los actores y los casos de uso.
2) Se escriben todos los casos de uso en el formato de alto nivel. Se categorizan como primarios, secundarios u opcionales.

3) Se dibuja el Diagrama de Casos de Uso.
4) Se detallan relaciones entre casos de uso, en caso de ser necesarias, y se ilustran tales relaciones en el Diagrama de Casos de Uso.
5) Los casos de uso más críticos, importantes y que conllevan un mayor riesgo, se describen en el formato expandido esencial. Se deja la definición en formato expandido esencial del resto de casos de uso para cuando sean tratados en posteriores ciclos de desarrollo, para no tratar toda la complejidad del problema de una sola vez.
6) Se crean casos de uso reales sólo cuando:
    - Descripciones más detalladas ayudan significativamente a incrementar la comprensión del problema.
    - El cliente pide que los procesos se describan de esta forma.
7) Ordenar según prioridad los casos de uso (este paso se va a ver a continuación).

- **Planificación de Casos de Uso según Ciclos de Desarrollo**

La decisión de qué partes del sistema abordar en cada ciclo de desarrollo se va a tomar basándose en los casos de uso. Esto es, a cada ciclo de desarrollo se le va a asignar la implementación de uno o más casos de uso, o versiones simplificadas de casos de uso. Se asigna una versión simplificada cuando el caso de uso completo es demasiado complejo para ser tratado en un solo ciclo (ver Figura 3.35).

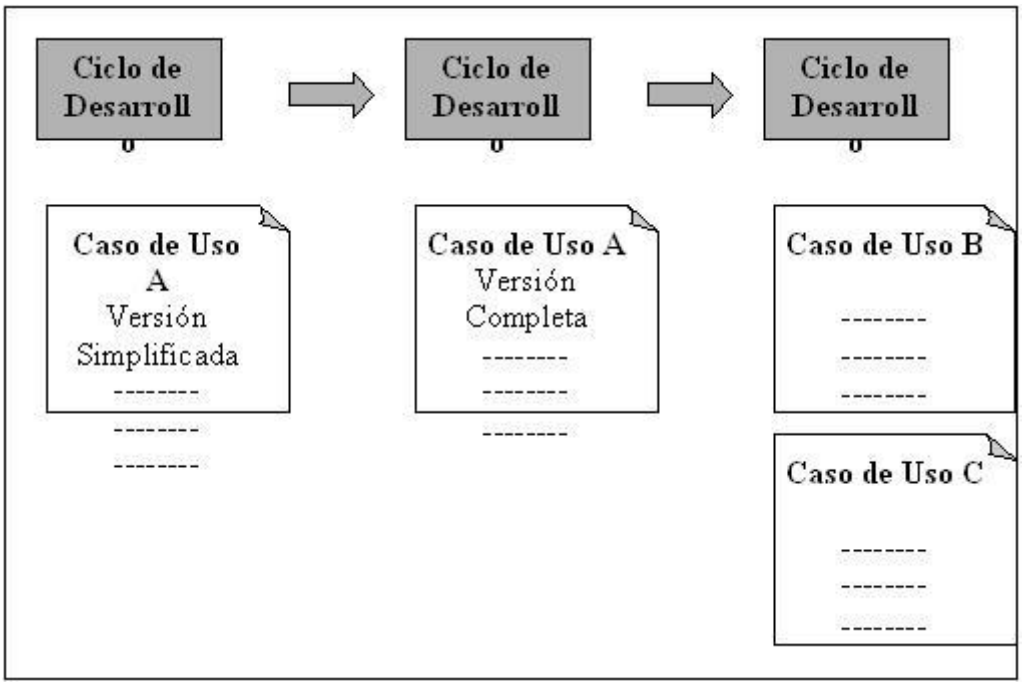


Figura 3.35 Ciclo de Desarrollo

Para tomar la decisión de qué casos de uso se van a tratar primero es necesario ordenarlos según prioridad. Las características de un caso de uso específico que van a hacer que un caso de uso tenga una prioridad alta son las siguientes:

a. Impacto significativo en el diseño de la arquitectura. Por ejemplo, si aporta muchas clases al modelo del dominio o requiere persistencia en los datos.
b. Se obtiene una mejor comprensión del diseño con un nivel de esfuerzo relativamente bajo.
c. Incluye funciones complejas, críticas en el tiempo o de nivel elevado de riesgo.
d. Implica bien un trabajo de investigación significante, o bien el uso de una tecnología nueva o arriesgada.
e. Representa un proceso de gran importancia en la línea de negocio.
f. Supone directamente un aumento de beneficios o una disminución de costes.

Para realizar la clasificación se puede asignar a cada caso de uso una valoración numérica de cada uno de estos puntos, para conseguir una puntuación total aplicando pesos a cada apartado. En la siguiente tabla 3.1 se muestra un ejemplo de tal tipo de clasificación:

Peso	3	2	4	1	3	4	Suma
Caso de Uso	a	b	c	d	e	f	
Realizar reintegro	5	4	1	0	5	2	50

Tabla 3.1.- Planificación de Casos de Uso según Ciclos de Desarrollo

- **Caso de Uso Inicialización**

Prácticamente todos los sistemas van a tener un caso de uso Inicialización. Aunque puede ser que no tenga una prioridad alta en la clasificación realizada según el punto anterior, normalmente va a interesar que sea desarrollado desde el principio. Inicialmente se desarrolla una versión simplificada, que se va completando en cada ciclo de desarrollo para satisfacer las necesidades de inicialización de los casos de uso que se tratan en dicho ciclo. Así se tiene un sistema en cada ciclo de desarrollo que puede funcionar.

3.8.1.2.- Fase de Construcción: Diseño de Alto Nivel

En la fase de Diseño de Alto Nivel de un ciclo de desarrollo se investiga sobre el problema, sobre los conceptos relacionados con el subconjunto de casos de uso que se esté tratando. Se intenta llegar a una buena comprensión del problema por parte del equipo de desarrollo, sin entrar en cómo va a ser la solución en cuanto a detalles de implementación.

Cuando el ciclo de desarrollo no es el primero, antes de la fase de Diseño de Alto Nivel hay una serie de actividades de planificación. Estas actividades consisten en actualizar los modelos que se tengan según lo que se haya implementado, pues siempre se producen desviaciones entre lo que se ha

analizado y diseñado y lo que finalmente se construye. Una vez se tienen los modelos acordes con lo implementado se empieza el nuevo ciclo de desarrollo con la fase de Diseño de Alto Nivel.

En esta fase se trabaja con los modelos de Diseño de Alto Nivel construidos en la fase anterior, ampliándolos con los conceptos correspondientes a los casos de uso que se traten en el ciclo de desarrollo actual.

- **Actividades**

Las actividades de la fase de Diseño de Alto Nivel son las siguientes:

1) Definir Casos de Uso Esenciales en formato expandido. (Si no están definidos)
2) Refinar los Diagramas de Casos de Uso.
3) Refinar el Modelo Conceptual.
4) Refinar el Glosario. (Continuado en posteriores fases)
5) Definir los Diagramas de Secuencia del Sistema.
6) Definir Contratos de Operación.
7) Definir Diagramas de Estados. (Opcional)

- **Modelo Conceptual**

Una parte de la investigación sobre el dominio del problema consiste en identificar los conceptos que lo conforman. Para representar estos conceptos se va a usar un Diagrama de Estructura Estática de UML, al que se va a llamar Modelo Conceptual.

En el Modelo Conceptual se tiene una representación de conceptos del mundo real, no de componentes software.

El objetivo de la creación de un Modelo Conceptual es aumentar la comprensión del problema. Por tanto, a la hora de incluir conceptos en el modelo,

es mejor crear un modelo con muchos conceptos que quedarse corto y olvidar algún concepto importante.

- **Identificación de Conceptos**

Para identificar conceptos hay que basarse en el documento de Especificación de Requisitos y en el conocimiento general acerca del dominio del problema.

En la Tabla 3.2 se muestran algunas categorías típicas, junto con ejemplos pertenecientes al dominio de los supermercados y al de la reserva de billetes de avión:

Otro consejo para identificar conceptos consiste en buscar sustantivos en los documentos de requisitos o, más concretamente, en la descripción de los casos de uso. No es un método infalible, pero puede servir de guía para empezar. Para poner nombre a los conceptos se puede usar la analogía con el cartógrafo, resumida en los siguientes tres puntos:

- Usar los nombres existentes en el territorio: Hay que usar el vocabulario del dominio para nombrar conceptos y atributos.
- Excluir características irrelevantes: Al igual que el cartógrafo elimina características no relevantes según la finalidad del mapa (por ejemplo datos de población en un mapa de carreteras), un Modelo Conceptual puede excluir conceptos en el dominio que no son pertinentes en base a los requisitos.
- No añadir cosas que no están ahí: Si algo no pertenece al dominio del problema no se añade al modelo.

- **Creación del Modelo Conceptual**

Para crear el Modelo Conceptual se siguen los siguientes pasos:

1) Hacer una lista de conceptos candidato usando la Lista de Categorías de Conceptos de la Tabla 3.2 y la búsqueda de sustantivos relacionados con los requisitos en consideración en este ciclo.
2) Representarlos en un diagrama.
3) Añadir las asociaciones necesarias para ilustrar las relaciones entre conceptos que es necesario conocer.
4) Añadir los atributos necesarios para contener toda la información que se necesite conocer de cada concepto.

- **Identificación de Asociaciones**

Una asociación es una relación entre conceptos que indica una conexión con sentido y que es de interés en el conjunto de casos de uso que se está tratando. Se incluyen en el modelo las asociaciones siguientes:

- Asociaciones para las que el conocimiento de la relación necesita mantenerse por un cierto período de tiempo (asociaciones “necesita-conocer”).
- Asociaciones derivadas de la Lista de Asociaciones Típicas que se muestra en la Tabla 3.3.

Tipo de Concepto Ejemplos
Objetos físicos o tangibles Avión
Terminal de Caja
Especificaciones, diseños o descripciones de cosas Especificación de Producto
Descripción de Vuelo
Lugares Supermercado
Aeropuerto
Transacciones Venta, Pago
Reserva
Líneas de una transacción Artículo de Venta
Roles de una persona Cajero
Piloto

Contenedores de otras cosas Supermercado, Cesta
Avión
Cosas en un contenedor Artículo
Pasajero
Otros ordenadores o sistemas electromecánicos externos a nuestro sistema Sistema de Autorización de Tarjetas de Crédito
Sistema Controlador de Tráfico Aéreo
Conceptos abstractos Hambre
Organizaciones Departamento de Ventas
Compañía Aérea Toto
Eventos Venta, Robo, Reunión
Vuelo, Accidente, Aterrizaje
Reglas y políticas Política de Devoluciones
Política de Cancelaciones
Catálogos Catálogo de Productos
Catálogo de Piezas
Archivos financieros, de trabajo, de contratos, de asuntos legales Recibo, Contrato de Empleo
Registro de Revisiones
Instrumentos y servicios financieros Línea de Crédito
Stock
Manuales, libros Manual del Empleado
Manual de Reparaciones

Tabla 3.2.- Lista de Conceptos Típicos

Categoría Ejemplos
A es una parte física de B Ala – Avión
A es una parte lógica de B Artículo en Venta –Venta
A está físicamente contenido en B Artículo – Estantería
Pasajero – Avión

A está lógicamente contenido en B Descripción de Artículo – Catálogo
A es una descripción de B Descripción de Artículo – Artículo
A es un elemento en una transacción o un informe B Trabajo de Reparación – Registro de Reparaciones
A es registrado/archivado/capturado en B Venta – Terminal de Caja
A es un miembro de B Cajero – Supermercado
Piloto – Compañía Aérea
A es una subunidad organizativa de B Sección – Supermercado
Mantenimiento – Compañía Aérea
A usa o gestiona B Cajero – Terminal de Caja
Piloto – Avión
A comunica con B Cliente – Cajero
Empleado de Agencia de Viajes – Pasajero
A está relacionado con una transacción B Cliente – Pago
Pasajero – Billete
A es una transacción relacionada con otra transacción B Pago – Venta
Reserva – Cancelación
A está junto a B Ciudad – Ciudad
A posee B Supermercado – Terminal de Caja
Compañía Aérea – Avión

Tabla 3.3.- Lista de Asociaciones Típicas

Una vez identificadas las asociaciones se representan en el Modelo Conceptual con la multiplicidad adecuada.

- **Identificación de Atributos**

Es necesario incorporar al Modelo Conceptual los atributos necesarios para satisfacer las necesidades de información de los casos de uso que se estén desarrollando en ese momento.

Los atributos deben tomar valor en tipos simples (número, texto, etc.), pues los tipos complejos deberían ser modelados como conceptos y ser relacionados mediante asociaciones.
Incluso cuando un valor es de un tipo simple es más conveniente representarlo como concepto en las siguientes ocasiones:

- Se compone de distintas secciones. Por ejemplo: un número de teléfono, el nombre de una persona, etc.
- Tiene operaciones asociadas, tales como validación. Ejemplo: NIF.
- Tiene otros atributos. Por ejemplo un precio de oferta puede tener fecha de fin.
- Es una cantidad con una unidad. Ejemplo: El precio, que puede estar en dólares o euros.

Una vez definidos los atributos se tiene ya un Modelo Conceptual. Este modelo no es un modelo definitivo, pues a lo largo del desarrollo se va refinando según se le añaden conceptos que se habían pasado por alto.

- **Glosario**

  En el glosario debe aparecer una descripción textual de cualquier elemento de cualquier modelo, para eliminar toda posible ambigüedad. Se ordena alfabéticamente por término.

  Un formato tipo para el glosario es el que se muestra en la Tabla 3.4.

Término	Categoría	Descripción

Tabla 3.4 Formato tipo de Glosario

- **Diagramas de Secuencia del Sistema**

Además de investigar sobre los conceptos del sistema y su estructura, también es preciso investigar en el Diseño de Alto Nivel sobre el comportamiento del sistema, visto éste como una caja negra. Una parte de la descripción del comportamiento del sistema se realiza mediante los Diagramas de Secuencia del Sistema.

En cada caso de uso se muestra una interacción de actores con el sistema. En esta interacción los actores generan eventos, solicitando al sistema operaciones. Por ejemplo, en el caso de una reserva de un billete de avión, el empleado de la agencia de viajes solicita al sistema de reservas que realice una reserva. El evento que supone esa solicitud inicia una operación en el sistema de reservas.

Los casos de uso representan una interacción genérica. Una instancia de un caso de uso se denomina escenario, y muestra una ejecución real del caso de uso, con las posibles bifurcaciones y alternativas resueltas de forma particular.

Un Diagrama de Secuencia de Sistema se representa usando la notación para diagramas de secuencia de UML. En él se muestra para un escenario particular de un caso de uso los eventos que los actores generan, su orden, y los eventos que se intercambian entre sistemas.

Para cada caso de uso que se esté tratando se realiza un diagrama para el curso típico de eventos, y además se realiza un diagrama para los cursos alternativos de mayor interés. En la Figura 3.36 se muestra el Diagrama de Secuencia del Sistema para el caso de uso Realizar Reintegro de un cajero automático.

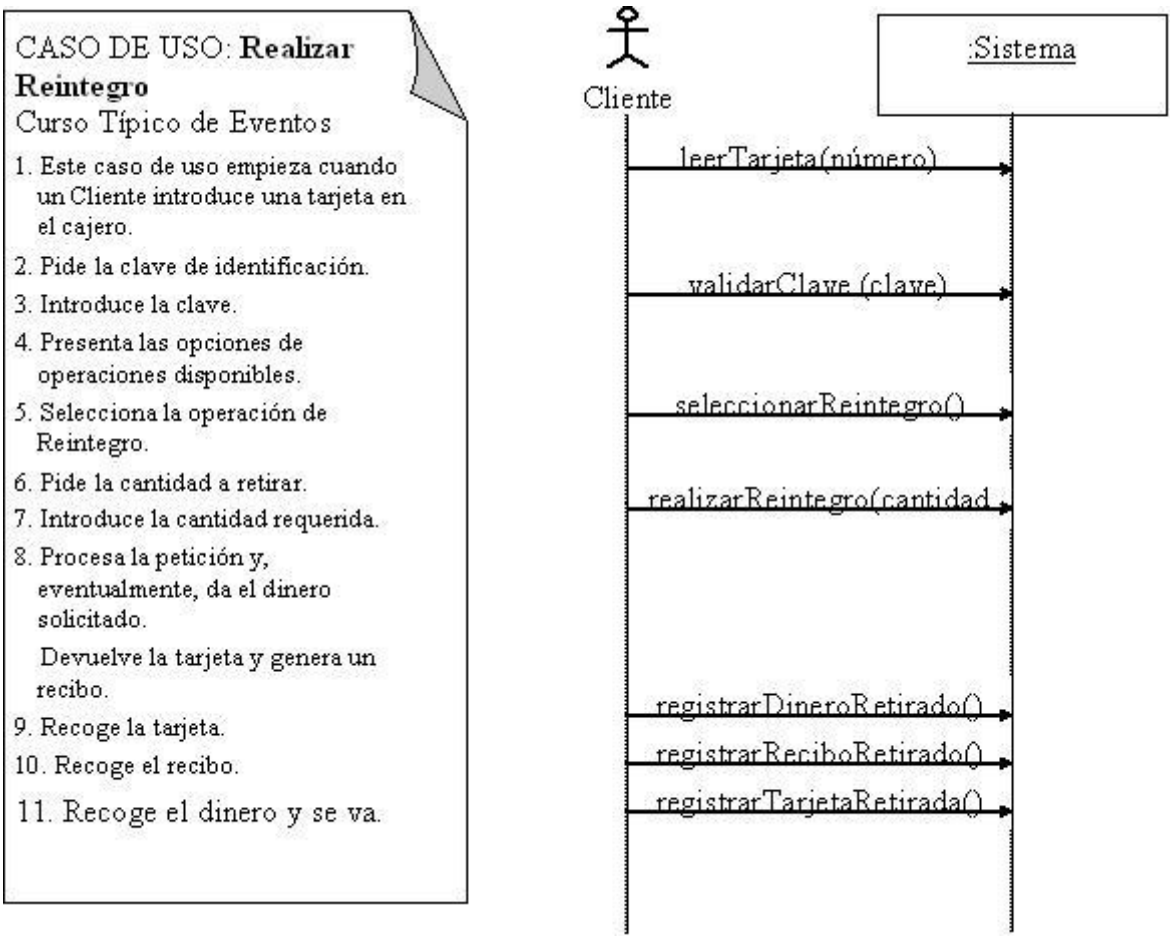


Figura 3.36 Ejemplo de Diagrama de Secuencia del Sistema

- **Construcción de un Diagrama de Secuencia del Sistema**

Para construir un Diagrama de Secuencia del Sistema para el curso típico de eventos de un caso de uso, se siguen los siguientes pasos:

1) Representar el sistema como un objeto con una línea debajo.
2) Identificar los actores que directamente operan con el sistema, y dibujar una línea para cada uno de ellos.
3) Partiendo del texto del curso típico de eventos del caso de uso, identificar los eventos (externos) del sistema que cada actor genera y representarlos en el diagrama.
4) Opcionalmente, incluir el texto del caso de uso en el margen del diagrama.

Los eventos del sistema deberían expresarse en base a la noción de operación que representan, en vez de en base a la interfaz particular. Por ejemplo, se prefiere “finalizarOperación” a “presionadaTeclaEnter”, porque captura la finalidad de la operación sin realizar compromisos en cuanto a la interfaz usada.

- **Contratos de Operaciones**

Una vez se tienen las Operaciones del Sistema identificadas en los Diagramas de Secuencia, se describe mediante contratos el comportamiento esperado del sistema en cada operación.

Un Contrato es un documento que describe qué es lo que se espera de una operación. Tiene una redacción en estilo declarativo, enfatizando en el qué más que en el cómo. Lo más común es expresar los contratos en forma de pre- y post-condiciones en torno a cambios de estado.

Se puede escribir un contrato para un método individual de una clase software, o para una operación del sistema completa. En este punto se verá únicamente éste último caso.

Un Contrato de Operación del Sistema describe cambios en el estado del sistema cuando una operación del sistema es invocada.

A continuación se ve un ejemplo de Contrato:

Contrato

Nombre: leerTarjeta (número_tarjeta: número)

Responsabilidades: Comprobar que la tarjeta numero_tarjeta es correcta y presentar las opciones disponibles.

Referencias Cruzadas: Funciones del Sistema: R1.2, R1.6, R1.7

Casos de Uso: Reingreso Notas:

Excepciones: Si la tarjeta es ilegible o no pertenece a los tipos de tarjetas aceptadas, indicar que ha habido un error.
Salida:
Pre-condiciones: No hay una operación activa.
Post-condiciones:
•Una nueva Operación se ha creado. (creación de instancia).
•La Operación se ha asociado con Cajero. (asociación formada).

La descripción de cada apartado de un contrato es como sigue: Nombre: Nombre de la operación y parámetros.
Responsabilidades: Una descripción informal de las responsabilidades que la operación debe desempeñar.
Referencias Cruzadas: Números de referencia en los requisitos de funciones del sistema, casos de uso, etc.
Notas: Comentarios de diseño, algoritmos, etc.
Excepciones: Casos excepcionales. Situaciones que debemos tener en cuenta que pueden pasar. Se indica también qué se hace cuando ocurre la excepción.
Salida: Salidas que no corresponden a la interfaz de usuario, como mensajes o registros que se envían fuera del sistema. (En la mayor parte de las operaciones del sistema este apartado queda vacío)
Pre-condiciones: Suposiciones acerca del estado del sistema antes de ejecutar la operación.
Post-condiciones: El estado del sistema después de completar la operación.

- **Construcción de un Contrato**

Los pasos a seguir para construir un contrato son los siguientes:

1) Identificar las operaciones del sistema a partir de los Diagramas de Secuencia del Sistema.
2) Para cada operación del sistema construir un contrato.

3) Empezar escribiendo el apartado de Responsabilidades, describiendo informalmente el propósito de la operación. Este es el apartado más importante del contrato.
4) A continuación rellenar el apartado de Post-condiciones, describiendo declarativamente los cambios de estado que sufren los objetos en el Modelo Conceptual. Puede ser que este apartado quede vacío si no cambia el valor de ningún dato de los maneja el sistema (por ejemplo en una operación del sistema que tan solo se encarga de sacar por pantalla algo al usuario).
5) Para describir las post-condiciones, usar las siguientes categorías:
- Creación y borrado de instancias.
- Modificación de atributos.
- Asociaciones formadas y retiradas.
6) Completar el resto de apartados en su caso.

- **Post-condiciones**

Las post-condiciones se basan en el Modelo Conceptual, en los cambios que sufren los elementos del mismo una vez se ha realizado la operación.

Es mejor usar el tiempo pasado o el pretérito perfecto al redactar una post-condición, para enfatizar que se trata de declaraciones sobre un cambio en el estado que ya ha pasado. Por ejemplo es mejor decir “se ha creado un nuevo Cliente” que decir “crear un Cliente”.

Cuando se ha creado un objeto, lo normal es que se haya asociado a algún otro objeto ya existente, porque si no queda aislado del resto del sistema. Por tanto, al escribir las post-condiciones hay que acordarse de añadir asociaciones a los objetos creados. Olvidar incluir estas asociaciones es el fallo más común cometido al escribir las post-condiciones de un contrato.

- **Diagramas de Estados**

Para modelar el comportamiento del sistema pueden usarse los Diagramas de Estados que define UML.

Se puede aplicar un Diagrama de Estados al comportamiento de los siguientes elementos:

- Una clase software.
- Un concepto.
- Un caso de uso.

En la fase de Diseño de Alto Nivel sólo se haría para los dos últimos tipos de elemento, pues una clase software pertenece al Diagrama de Clases de Diseño. Puesto que el sistema entero puede ser representado por un concepto, también se puede modelar el comportamiento del sistema completo mediante un Diagrama de Estados.

La utilidad de un Diagrama de Estados en esta fase reside en mostrar la secuencia permitida de eventos externos que pueden ser reconocidos y tratados por el sistema. Por ejemplo, no se puede insertar una tarjeta en un cajero automático si se está en el transcurso de una operación.

Para los casos de uso complejos se puede construir un Diagrama de Estados. El Diagrama de Estados del sistema sería una combinación de los diagramas de todos los casos de uso.

El uso de Diagramas de Estados es opcional. Tan solo los usaremos cuando consideremos que nos ayudan a expresar mejor el comportamiento del elemento descrito.

3.8.1.3.- Fase de Construcción: Diseño de Bajo Nivel

En la fase de Diseño de Bajo Nivel se crea una solución a nivel lógico para satisfacer los requisitos, basándose en el conocimiento reunido en la fase de Diseño de Alto Nivel. Los modelos más importantes en esta fase son el Diagrama de Clases de Diseño y los Diagramas de Interacción, que se realizan en paralelo y que definen los elementos que forman parte del sistema orientado a objetos que se va a construir (clases y objetos) y cómo colaboran entre sí para realizar las funciones que se piden al sistema, según éstas se definieron en los contratos de operaciones del sistema.

- **Actividades**

Las actividades que se realizan en la etapa de Diseño de Bajo Nivel son las siguientes:

- Definir los Casos de Uso Reales.
- Definir Informes e Interfaz de Usuario.
- Refinar la Arquitectura del Sistema.
- Definir los Diagramas de Interacción.
- Definir el Diagrama de Clases de Diseño. (en paralelo con los Diagramas de Interacción)
- Definir el Esquema de Base de Datos.

El paso de Refinar la Arquitectura del Sistema no tiene por qué realizarse en la posición 3, puede realizarse antes o después.

- **Casos de Uso Reales**

Un Caso de Uso Real describe el diseño real del caso de uso según una tecnología concreta de entrada y de salida y su implementación. Si el caso de uso implica una interfaz de usuario, el caso de uso real incluirá bocetos de las ventanas y detalles de la interacción a bajo nivel con los widgets (botón, lista seleccionable, campo editable, etc.) de la ventana.

Como alternativa a la creación de los Casos de Uso Reales, el desarrollador puede crear bocetos de la interfaz en papel, y dejar los detalles para la fase de implementación.

- **Diagramas de Interacción**

Los Diagramas de Interacción muestran el intercambio de mensajes entre instancias del modelo de clases para cumplir las post-condiciones establecidas en un contrato

Hay dos clases de Diagramas de Interacción:

1. Diagramas de Colaboración.
2. Diagramas de Secuencia.

De entre ambos tipos, los Diagramas de Colaboración tienen una mayor expresividad y mayor economía espacial (una interacción compleja puede ser muy larga en un Diagrama de Secuencia), sin embargo en ellos la secuencia de interacción entre objetos es más difícil de seguir que en un Diagrama de Secuencia. Ambos tipos de diagramas expresan la misma información, por lo que se puede usar cualquiera de los dos para expresar la interacción entre los objetos del sistema.

La creación de los Diagramas de Interacción de un sistema es una de las actividades más importantes en el desarrollo orientado a objetos, pues al construirlos se toman unas decisiones clave acerca del funcionamiento del futuro sistema. La creación de estos diagramas, por tanto, debería ocupar un porcentaje significativo en el esfuerzo dedicado al proyecto entero.

- **Creación de Diagramas de Interacción**

Para crear los Diagramas de Colaboración o de Secuencia se pueden seguir los siguientes consejos:

- Crear un diagrama separado para cada operación del sistema en desarrollo en el ciclo de desarrollo actual.
- Para cada evento del sistema, hacer un diagrama con él como mensaje inicial.

  Usando los apartados de responsabilidades y de post-condiciones del contrato de operación, y la descripción del caso de uso como punto de partida, diseñar un sistema de objetos que interaccionan para llevar a cabo las tareas requeridas.
- Si el diagrama se complica, dividirlo en dos diagramas más pequeños. Para ello se termina la secuencia de mensajes en un mensaje determinado, y en el segundo diagrama se comienza con el mensaje que terminó el primero. Debe indicarse en el primer diagrama que el resto de la interacción se detalla en el segundo.

El comportamiento dinámico del sistema que se describe en un Diagrama de Interacción debe ser acorde con la estructura estática del sistema que se refleja en el Diagrama de Clases de Diseño. Es aconsejable realizar un Diagrama de Clases de Diseño borrador antes de comenzar con los Diagramas de Interacción. La capacidad de realizar una buena asignación de responsabilidades a los distintos objetos es una habilidad clave, y se va adquiriendo según aumenta la experiencia en el desarrollo orientado a objetos.

Responsabilidad es como un contrato u obligación de una clase o tipo. Las responsabilidades están ligadas a las obligaciones de un objeto en cuanto a su comportamiento. Básicamente, estas responsabilidades pueden ser de tipo Conocer o de tipo Hacer:

- Conocer:
  - Conocer datos privados encapsulados.
  - Conocer los objetos relacionados.
  - Conocer las cosas que puede calcular o derivar.

- Hacer:
  - Hacer algo él mismo.
  - Iniciar una acción en otros objetos.
  - Controlar y coordinar actividades en otros objetos.

Por ejemplo, puedo decir que “un Recibo es responsable de calcular el total” (tipo hacer), o que “una Transacción es responsable de saber su fecha” (tipo conocer). Las responsabilidades de tipo “conocer” se pueden inferir normalmente del Modelo Conceptual.

Una responsabilidad no es lo mismo que un método, pero los métodos se implementan para satisfacer responsabilidades.

- **Diagrama de Clases de Diseño**

Un Diagrama de Clases de Diseño muestra la especificación para las clases software de una aplicación. Incluye la siguiente información:

- Clases, asociaciones y atributos.
- Interfaces, con sus operaciones y constantes.
- Métodos.
- Navegabilidad.
- Dependencias.

A diferencia del Modelo Conceptual, un Diagrama de Clases de Diseño muestra definiciones de entidades software más que conceptos del mundo real. En la Figura 3.37 se muestra un ejemplo de Diagrama de Clases de Diseño sencillo.

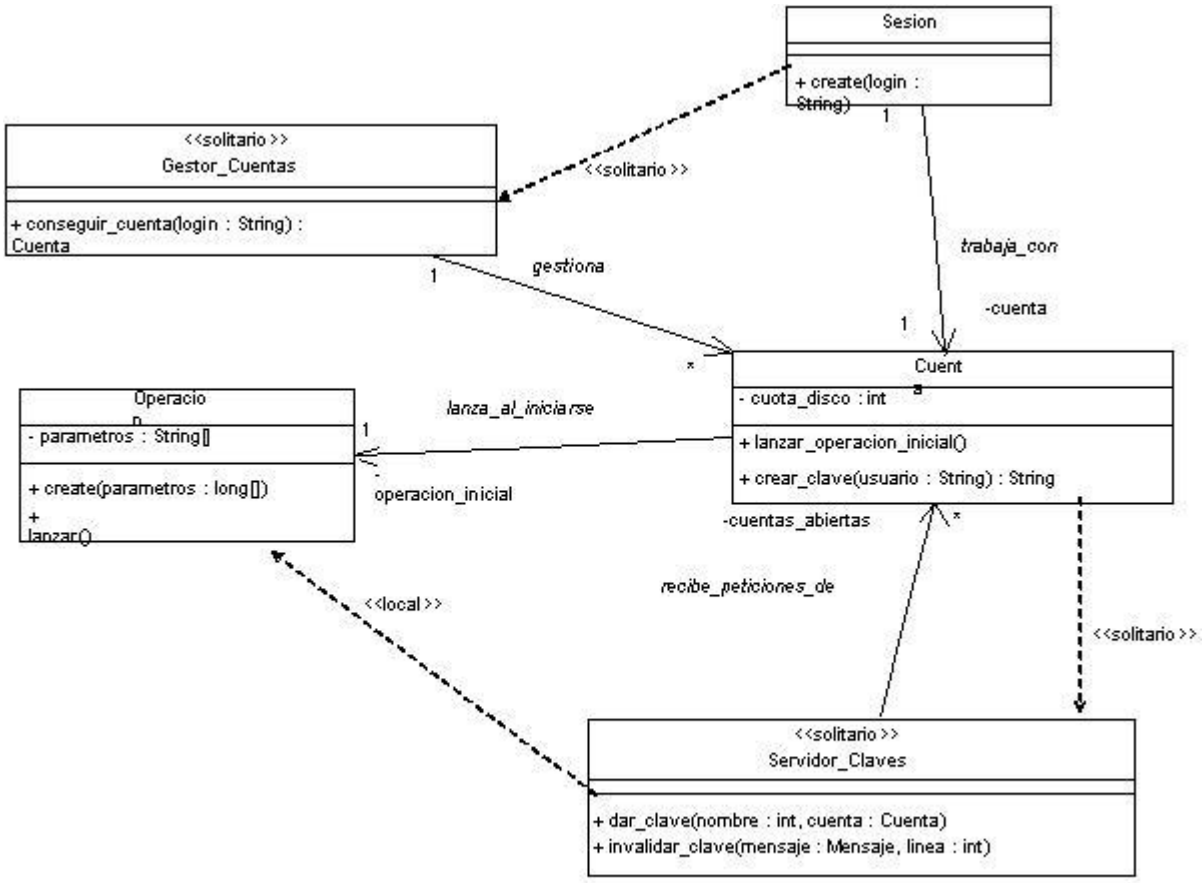


Figura 3.37 Ejemplo de Diagrama de Clases de Diseño

- **Relaciones de Dependencia para Representar Visibilidad entre Clases**

Cuando una clase conoce a otra por un medio que no es a través de un atributo (una asociación con la navegabilidad adecuada), entonces es preciso indicar esta situación por medio de una dependencia.

Un objeto debe conocer a otro para poder llamar a uno de sus métodos, se dice entonces que el primer objeto tiene “visibilidad” sobre el segundo. La visibilidad más directa es por medio de atributo, cuando hay una asociación entre ambas clases y se puede navegar de la primera a la segunda (un atributo de la primera es un puntero a un objeto de la segunda). Hay otros tres tipos de visibilidad que hay que representar en el diagrama de clases mediante relaciones de dependencia:

- Parámetro: Cuando a un método de una clase se le pasa como parámetro un objeto de otra clase, se dice que la primera tiene visibilidad de parámetro sobre la segunda. La relación de dependencia entre ambas clases se etiqueta con el estereotipo <> (<> en inglés).
- Local: Cuando en un método de una clase se define una variable local que es un objeto de otra clase, se dice que la primera tiene visibilidad local sobre la segunda. La relación de dependencia entre ambas clases se etiqueta con el estereotipo <>.
- Global: Cuando hay una variable global en el sistema, instancia de una clase A, y un método de una clase B llama a un método de A, se dice que la clase B tiene visibilidad global sobre la clase A. La relación de dependencia entre ambas clases se etiqueta con el estereotipo <>.

No es necesario representar la relación de dependencia entre clases que ya están relacionadas por medio de una asociación, que se trata de una "dependencia" más fuerte. Las relaciones de dependencia se incluyen tan solo para conocer qué elementos hay que revisar cuando se realiza un cambio en el diseño de un elemento del sistema.

a) Solitario: Caso Particular de Visibilidad global

El uso de variables globales no se aconseja por los efectos laterales que se pueden presentar, pero hay un caso en el que sí hay cierta globalidad: las clases que sólo van a tener una instancia. Suele tratarse de clases que se han creado en el Diseño de Bajo Nivel, que no aparecían en el Modelo Conceptual.

Varias clases de nuestro sistema pueden querer llamar a los métodos de la única instancia de una clase de ese tipo, entonces sí se considera que es beneficioso que se pueda acceder a esa instancia como un valor accesible de forma global. Una solución elegante para este caso se implementa mediante un método de clase para obtener esa única instancia (los métodos de clase los permiten algunos lenguajes orientados a objetos, por ejemplo Java), en vez de

definirla directamente como una variable global. Para indicar que una clase sólo va a tener una instancia, se etiqueta la clase con el estereotipo <> (<> en inglés), y las relaciones de dependencia entre las clases que la usan y se etiquetan también en vez de <>.

A continuación se muestra un ejemplo del código en Java de una clase solitario:

```
public class Solitario {
// se define la instancia como atributo de clase (static)
Solitario static instancia := null;
// método de clase que devuelve la instancia
public static Solitario dar_instancia() {
if (instancia = = null) {
// si no está creada la instancia la crea
instancia := new Solitario();
}
return instancia;
}
... // otros métodos de la clase
}
```

Cuando otra clase quiere llamar a un método de la instancia incluye el siguiente código:

```
variable Solitario;
variable = Solitario.dar_instancia();
variable.método(); // llamada al método que necesitamos
```

- **Construcción de un Diagrama de Clases de Diseño**

Normalmente se tiene una idea de un Diagrama de Clases, con una asignación de responsabilidades inicial. En caso de que no se tenga dicho Diagrama de Clases Borrador, puede seguirse la siguiente estrategia:

1) Identificar todas las clases participantes en la solución software. Esto se lleva a cabo analizando los Diagramas de Interacción.
2) Representarlas en un diagrama de clases.
3) Duplicar los atributos que aparezcan en los conceptos asociados del Modelo Conceptual.
4) Añadir los métodos, según aparecen en los Diagramas de Interacción.
5) Añadir información de tipo a los atributos y métodos.
6) Añadir las asociaciones necesarias para soportar la visibilidad de atributos requerida.
7) Añadir flechas de navegabilidad a las asociaciones para indicar la dirección de visibilidad de los atributos.
8) Añadir relaciones de dependencia para indicar visibilidad no correspondiente a atributos.

Algunos de estos pasos se van realizando según se vayan completando los Diagramas de Interacción correspondientes. No existe precedencia entre la realización del Diagrama de Clases de Diseño y los Diagramas de Interacción. Ambos tipos de diagramas se realizan en paralelo, y unas veces se trabaja primero más en el de clases y otras veces se trabaja primero más en los de interacción.

No todas las clases que aparecían en el Modelo Conceptual tienen por qué aparecer en el Diagrama de Clases de Diseño. De hecho, tan solo se incluirán aquellas clases que tengan interés en cuanto a que se les ha asignado algún tipo de responsabilidad en el diseño del sistema. No hay, por tanto, un transición directa entre el Modelo Conceptual y el Diagrama de Clases de Diseño, debido a

que ambos se basan en enfoques completamente distintos: el primero en comprensión de un dominio, y el segundo en una solución software.

En el Diagrama de Clases de Diseño se añaden los detalles referentes al lenguaje de programación que se vaya a usar. Por ejemplo, los tipos de los atributos y parámetros se expresarán en el lenguaje de implementación escogido.

- **Navegabilidad**

La navegabilidad es una propiedad de un rol (un extremo de una asociación) que indica que es posible “navegar” unidireccionalmente a través de la asociación, desde objetos de la clase origen a objetos de la clase destino.

Como se vio en la parte II, se representa en UML mediante una flecha. La navegabilidad implica visibilidad, normalmente visibilidad por medio de un atributo en la clase origen. En la implementación se traducirá en la clase origen como un atributo que sea una referencia a la clase destino.

Las asociaciones que aparezcan en el Diagrama de Clases deben cumplir una función, deben ser necesarias, si no es así deben eliminarse.

Las situaciones más comunes en las que parece que se necesita definir una asociación con navegabilidad de A a B son:

- A envía un mensaje a B.
- A crea una instancia B.
- A necesita mantener una conexión con B.

- **Visibilidad de Atributos y Métodos**

Los atributos y los métodos deben tener una visibilidad asignada, que puede ser:

- Visibilidad pública.

- Visibilidad protegida.
- Visibilidad privada.

También puede ser necesario incluir valores por defecto, y todos los detalles ya cercanos a la implementación que sean necesarios para completar el Diagrama de Clases.

- **Otros Aspectos en el Diseño del Sistema**

En los puntos anteriores se ha hablado de decisiones de diseño a un nivel de granularidad fina, con las clases y objetos como unidades de decisión. En el diseño de un sistema es necesario también tomar decisiones a un nivel más alto sobre la descomposición de un sistema en subsistemas y sobre la arquitectura del sistema. Esta parte del Diseño de Bajo Nivel es lo que se denomina Diseño del Sistema. Estas decisiones no se toman de forma distinta en un desarrollo orientado a objetos a como se llevan a cabo en un desarrollo tradicional. Por tanto, no se va a entrar en este documento en cómo se realiza esta actividad.

Sí hay que tener en cuenta que las posibles divisiones en subsistemas tienen que hacerse en base a las clases definidas en el Diagrama de Clases del Diseño.

3.8.1.4.- Fases de Implementación y Pruebas

Una vez se tiene completo el Diagrama de Clases de Diseño, se pasa a la implementación en el lenguaje de programación elegido.

El programa obtenido se depura y prueba, y ya se tiene una parte del sistema funcionando que se puede probar con los futuros usuarios, e incluso poner en producción si se ha planificado una instalación gradual.

Una vez se tiene una versión estable se pasa al siguiente ciclo de desarrollo para incrementar el sistema con los casos de uso asignados a tal ciclo.

IV. DESARROLLO DEL SISTEMA WORKFLOW

4.1.- FASE DE ANÁLISIS

4.1.1.- ESPECIFICACIÓN DE REQUISITOS SOFTWARE

4.1.1.1.- Introducción

Este documento es una especificación de requisitos software (ERS) para el Sistema de Gestión de Seguimiento de Procesos, la documentación ha sido elaborada en base a manuales de procesos. Esta especificación de requisitos se ha estructurado tomando en cuenta las directrices dadas por el standard “IEEE Recommended Practice for Software Requirements Specification ANSI/IEEE 830 1998”

4.1.1.2.- Propósito

El objetivo del presente documento es definir de manera clara y precisa todas las funcionalidades y restricciones del sistema a desarrollarse, el cual deberá: Automatizar los flujos de trabajos de empresas cuyos procesos hayan sido racionalizados, lo que implica que deberá soportar: gestión de departamentos, consulta de grupos de usuarios, gestión de usuarios, gestión de procesos, gestión de tareas, gestión de asignación de tareas a usuarios, desarrollo de tareas, monitoreo de procesos, cancelar procesos, cambio de clave de acceso, modificar datos personales.

4.1.1.3.- Ámbito del Sistema

El sistema recibirá el nombre de SiGeSPro (Sistema de Gestión de Seguimiento de Procesos).

La situación de partida es la racionalización de procesos dentro de la empresa, o en determinado departamento o área de la misma, a través de manuales de procesos, los mismos que servirán de base para el desarrollo del sistema.

Tomando en cuenta que en cada departamento se maneja considerable cantidad de procesos, existe la necesidad de realizarlos en forma más rápida, es por esto que el sistema a realizarse proveerá de una herramienta que permita graficar estos procesos y asigne actividades para automatizarlos y a través del sistema de monitoreo realizar el seguimiento de los mismos y de esta manera acortar tiempos, costos y recursos dentro de la organización; así como también que sea un sistema que funcione simultáneamente y de manera autónoma en cada uno de los departamentos de la organización y que tenga capacidad de integración.

La carga del sistema dependerá de la cantidad de procesos que se realicen en cada organización

4.1.1.4.- Definiciones, Acrónimos y Abreviaturas

4.1.1.4.1.- Definiciones

Diagramador	Persona que elabora un esquema gráfico o mapa de proceso en una explicación visual de cómo el proceso fluye y se conecta.
Administrador	Núcleo central del sistema.
Jefe Departamental	Persona que da inicio a los procesos.
Usuario General	Persona que recibe y realiza la carga de actividades a este asignadas.

4.1.1.4.2.- Acrónimos

ERS	Especificación de requisitos de

	Software.
ARS	Análisis de requisitos del Sistema.

4.1.1.4.3.-Abreviaturas

SiGeSPro	Sistema de Gestión de Seguimiento de Procesos

4.1.1.5.- Referencias

IEEE Recommended Practice for Software Requirements Specification. ANSI/IEEE std 830, 1998.

4.1.1.6.- Visión General del Documento

Este documento consta de tres secciones, esta sección es la introducción y proporciona una visión general de la ERS. En la sección 2 se da una descripción general del sistema, con el fin de conocer las principales funciones que debe realizar, los datos asociados y los factores, restricciones y dependencias que afectan al desarrollo, en la sección 3 se detallan los requisitos que debe satisfacer el sistema.

4.1.1.7.- Descripción General

En esta sección nos presenta una descripción general del sistema, con el fin de conocer las principales funciones que debe realizar, los datos asociados, las restricciones impuestas, y cualquier factor que pueda afectar al desarrollo del mismo.

4.1.1.8.- Perspectiva del Producto

El sistema en esta versión, no interactuará con ningún otro sistema informático.

4.1.1.9.- Funciones del Sistema

En términos generales, el sistema deberá proporcionar soporte a las siguientes tareas de gestión:

- Gestión de Departamentos.
- Consulta de Grupos de Usuario.
- Gestión de Usuarios.
- Gestión de Procesos.
- Gestión de Tareas.
- Iniciar Proceso.
- Gestión de Asignación de Tareas a Usuarios.
- Ejecución de Tareas Asignadas.
- Monitoreo de Procesos.
- Cancelar Procesos.
- Cambio de Clave de Acceso.
- Modificar Datos Personales.

4.1.1.9.1.- Gestión de Departamentos

Permitirá realizar las funciones de alta, modificación, consulta (Individual o general) de departamentos.

Para el alta de un departamento, el sistema permitirá generar un código de departamento, y permitirá ingresar el nombre y una descripción.

Para modificar los datos de un departamento, se seleccionará de la lista de departamentos, luego de verificar que los datos corresponden al departamento a modificar, solo se podrán modificar los campos que el sistema permita.

Para la consulta individual de departamentos, se selecciona de la lista, y se despliega la información.

La consulta general permitirá el conocer la lista de todos los departamentos del sistema.

4.1.1.9.2.- Consulta de Grupos de Usuarios

Permitirá realizar las funciones de consulta de grupos de usuario.

La consulta general permitirá conocer la lista de todos los grupo de usuario del sistema.

La consulta de roles de usuario permitirá conocer el papel que desempeña un usuario del sistema.

4.1.1.9.3.- Gestión de Usuarios

Permitirá realizar las funciones de alta, modificación, consulta (Individual o general) de usuarios y roles de usuario.

Para el alta de un usuario, el sistema permitirá ingresar el nombre, apellido, número de cédula de identidad, dirección, teléfono, nombre de usuario de acceso al sistema, clave de acceso al sistema, el departamento al que pertenecerá, el tipo o grupo de usuario, correo electrónico, además el sistema generará el código del usuario, y tomará la fecha de ingreso del sistema.

Para modificar los datos de un usuario, se seleccionará de la lista de usuarios, luego de verificar que los datos corresponden al usuario a modificar, solo se podrán modificar los campos que el sistema permita.

Para la consulta individual de usuarios, se selecciona de la lista, y se despliega la información.

La consulta general permitirá el conocer la lista de todos los usuarios del sistema.

4.1.1.9.4.- Gestión de Procesos

Permitirá realizar las funciones de creación, consulta de un proceso.

Para el alta o creación de un Proceso se deberá graficar el proceso dibujando el inicio de proceso, las actividades, las transiciones, el fin de proceso ingresando el nombre de cada elemento que lo conforma, una vez dibujado el proceso, el sistema permitirá ingresar el nombre del proceso, además se tomará del sistema, la fecha y hora de creación, y se asignará automáticamente un código, un estado.

Para realizar consultas de los procesos existentes, el usuario diagramador deberá ingresar al entorno web con el mismo login y password que para el diagramador de procesos.

Para las consultas Individuales de un Proceso, se seleccionará uno de la lista de procesos, y desplegará la información del Proceso en cuestión junto con las tareas que a este pertenezcan.

La consulta general permitirá el conocer todos los Procesos existentes en la organización.

4.1.1.9.5.- Gestión de Tareas

Permitirá realizar las funciones de alta.

Para el alta de una tarea el sistema toma automáticamente los nombres de cada una de las actividades que se graficaron dentro del diagramador, el código del proceso, toma además del sistema la fecha y hora de creación, y genera un código y un estado.

Una consulta de una tarea se encuentra inmersa en las consultas individuales de los procesos.

4.1.1.9.6.- Iniciar Proceso

Permite dar el alta a un proceso en ejecución.

Para el alta de un proceso en ejecución, el sistema permitirá seleccionar un proceso de la lista de procesos existentes, ingresará una observación del mismo y el sistema tomará el departamento desde el cual se inicia el proceso.

4.1.1.9.7.- Gestión de Asignación de Tareas a Usuarios

Permitirá realizar las funciones de alta, modificación de tareas a usuarios.

Para el alta de una nueva asignación de tarea a un usuario el sistema tomará las tareas correspondientes al proceso que se dio inicio, la fecha y hora de asignación, la fecha y hora de inicio, fecha y hora de fin, permitirá ingresar al usuario al que se le desee asignar cada una de dichas tareas, la descripción, y generará el código y estado.

Para modificar los datos de una asignación de tarea o reasignar la misma, se selecciona un proceso de la lista de procesos en ejecución, dentro de esta se escoge una tarea de la lista de tareas, luego de verificar que los datos correspondan a la tarea y que no se encuentre en estado Finalizado, se modificarán los campos que el sistema permita.

4.1.1.9.8.- Ejecución de Tareas Asignadas

Permitirá realizar las funciones de actualización, consulta de las tareas en ejecución.

Para la actualización del estado de las tareas se selecciona una tarea de la lista de tareas asignadas a un usuario pendientes, se actualizan los campos que el sistema permita.

La consulta general permitirá conocer la carga de actividades para cada usuario.

4.1.1.9.9.- Monitoreo de Procesos

Permite la consulta de los procesos en ejecución.

Para la consulta individual de los seguimientos de tareas, se selecciona una de la lista, y se despliega la información.

La consulta general permitirá el conocer la lista de todos los seguimientos de tareas por usuario existentes.

4.1.1.9.10.- Cancelar Procesos

Permite cancelar un proceso en ejecución.

Para cancelar un proceso el sistema permitirá seleccionar el proceso de la lista de procesos en ejecución y cambiar su estado.

4.1.1.9.11.- Cambio de Clave de Acceso

Permite el modificar la clave de ingreso al sistema.

Para cambiar la clave el sistema permitirá ingresar la clave actual, la nueva clave y la confirmación de la nueva clave.

4.1.1.9.12.- Modificar Datos Personales

Permite modificar a sí mismo los datos de usuario.

Para modificar a sí mismo los datos, el sistema desplegará la información del usuario, solo se podrán modificar los campos que el sistema permita.

4.1.1.10.- Características de los Usuarios

El sistema de información deberá proporcionar una interfaz de usuario fácil de aprender y sencillo de manejar, además deberá presentar un alto grado de usabilidad.

4.1.1.11.- Restricciones

Dado que existe ya una red instalada con plataforma cliente-servidor, la restricción de hardware y software es que el sistema se desarrolle para esta plataforma.

4.1.1.12.- Suposiciones

Los requisitos aquí descritos son estables dado que fueron realizados en base a un prototipo.

4.1.1.13.- Dependencias

El sistema a desarrollar no tiene dependencia respecto a otros sistemas. Seguirá una arquitectura cliente-servidor, por lo que la disponibilidad del sistema dependerá de la conexión entre las máquinas en la que residirá el programa cliente y la máquina servidora de datos.

4.1.1.14.- Requisitos Específicos

En este apartado se describirá los requisitos funcionales que deberán ser satisfechos por el sistema.

4.1.1.15.- Requisitos Funcionales

4.1.1.15.1.- Gestión de Departamentos

El sistema deberá permitir:

1) Req(01) Ingresar los datos de un departamento.
2) Req(02) Modificar los datos de un departamento.
3) Req(03) Consultar en forma general los departamento.
4) Req(04) Consultar individualmente departamentos.

4.1.1.15.2.- Gestión de Grupos de Usuarios

El sistema deberá permitir:

5) Req(05) Consultar en forma general los grupos de usuarios.
6) Req(06) Consultar roles de usuarios**.**

4.1.1.15.3.- Gestión de Usuarios

El sistema deberá permitir:

7) Req(07) Ingresar los datos de un usuario.
8) Req(08) Modificar los datos de un usuario.
9) Req(09) Consultar en forma general los usuarios.
10) Req(10) Consultar usuarios de manera individual.

4.1.1.15.4.- Gestión de Procesos

El sistema deberá permitir:

11) Req(11) Ingresar un nuevo Proceso.

12) Req(12) Consultar en forma general los Procesos de la organización.
13) Req(13) Consultar en forma individual los Procesos de la organización.

4.1.1.15.5.- Gestión de Tareas

El sistema deberá permitir:

14) Req(14) Ingresar los datos de una tarea.

4.1.1.15.6.- Iniciar Proceso

El sistema deberá permitir:

15) Req(15) Iniciar un proceso.

4.1.1.15.7.- Gestión de Asignación de Tareas a Usuarios

El sistema deberá permitir.

16) Req(16) Ingresar una nueva asignación de tarea a usuario.
17) Req(17) Modificar los datos de una tarea asignada.

4.1.1.15.8.- Ejecución de Tareas Asignadas

El sistema deberá permitir.

18) Req(18) Actualizar estado de tareas.
19) Req(19) Consultar carga de actividades para cada usuario.

4.1.1.15.9.- Monitoreo de Procesos

El sistema deberá permitir:

20) Req(20) Consultar en forma general los estados en los que se encuentran los procesos.
21) Req(21) Consultar en forma individual los estados de los procesos.

4.1.1.15.10.- Cancelar Procesos

El sistema deberá permitir:

22)Req(22) Cancelar un proceso en ejecución.

4.1.1.15.11.- Cambiar Clave

El sistema deberá permitir:

23)Req(23) Cambiar la clave de acceso al sistema.

4.1.1.15.12.- Modificar Datos Personales

El sistema deberá permitir:

24)Req(24) Modificar los datos personales de un usuario.

4.1.1.16.- Requisitos de Interfaces Externas

4.1.1.16.1.- Interfaces de Usuario

La interfaz del sistema debe ser orientada a ventanas, y el manejo del programa se realizará a través del teclado y mouse.

4.1.1.16.2.- Interfaces Hardware

Se trabajara en plataforma cliente _ servidor.

4.1.1.16.3.- Interfaces Software.

De momento, no habrá ninguna interfaz software con sistemas externos.

4.1.1.16.4.- Interfaces de Comunicación

La conexión de la red se la hará utilizando una topología en estrella y siguiendo la norm2a IEEE 802.4.

4.1.1.17.- Requisitos de Rendimiento

No se ha definido.

4.1.1.18.- Requisitos de Desarrollo

El ciclo de vida para desarrollar el producto es el de refinamiento sucesivo o de mejora interactiva.

4.1.1.19.- Requisitos Tecnológicos

La aplicación cliente se ejecutará sobre un PC con la siguiente configuración:

1) Procesador Pentium de 700 Mhz.
2) Memoria 128 MB
3) Espacio libre en disco 100 Mb
4) Tarjeta Ethernet

Todos los PC se conectarán a un servidor el cual se localizará en el centro de cómputo de la Organización.

El sistema operativo en la que se debe ejecutar la aplicación será el Windows 2000 Professional o a su vez Windows XP.

Para el acceso a la base de datos se utilizará JDBC para java y un archivo config para php.

4.1.1.20.- Atributos

4.1.1.20.1.- Seguridad

Cuando un usuario del sistema intente conectarse, deberá ingresar su identificación y su clave, la cual será entregada por el administrador del sistema. En función de los atributos asignados por el administrador, a un usuario, se

activaran las opciones a las que puede acceder y utilizar. Si el identificador introducido junto con la clave no corresponde, se le indicará un mensaje de error.

Los tipos de usuarios que se van a contemplar, y las labores que corresponden a cada uno de ellos son:

- **Administrador del sistema**

Se encargará de definir los perfiles de los diferentes usuarios, que podrán acceder al sistema, los departamentos y los grupos a los que pertenecerán, consulta de procesos, consultar roles de usuarios, modificar sus datos personales, cambiar su clave de acceso al sistema.

- **Diagramadores**

Serán los encargados de la gestión de Procesos y gestión de tareas, consulta de procesos, modificar sus datos personales, cambiar su clave de acceso al sistema.

- **Jefes Departamentales**

Se encargará de dar inicio a la ejecución de un proceso, de asignar tareas a usuarios, reasignar tareas, monitorear el estado del los procesos, cancelar procesos, también podrá ingresar usuarios al sistema dentro del departamento que se encuentre a cargo, modificar sus datos personales, cambiar su clave de acceso al sistema.

- **Usuarios Generales**

Se encarga de realizar las tareas a este asignadas, modificar sus datos personales, cambiar su clave de acceso al sistema.

4.1.1.21.- Actividades de gestión de requisitos

Las matrices que se presentan a continuación, se utilizará de base para el desarrollo de los diferentes módulos del sistema, para lo cual se consideraran aquellos de mayor prioridad, cómo primarios, por cuanto sirven de base para la ejecución de otros procesos.

1 Gestión de Departamentos	
Número	1
Nombre	Gestión de Departamentos
Origen	Manual de Procesos
Estabilidad	Alta Media Baja X
Prioridad	Primario Secundario Opcional X
Fecha de Creación	
Fecha de Cambio	

1.1	Ingresar los datos de un departamento.
1.2	Modificar los datos de un departamento.
1.3	Consultar en forma general los departamento.
1.4	Consultar individualmente departamentos.

2 Gestión de Grupos de Usuario	
Número	2
Nombre	Gestión de Grupos de Usuario
Origen	Manual de Procesos
Estabilidad	Alta Media Baja X
Prioridad	Primario Secundario Opcional X
Fecha de Creación	
Fecha de Cambio	

2.5	Consultar en forma general los grupos de usuarios.
2.6	Consultar roles de usuarios.

3 Gestión de Usuarios	
Número	3
Nombre	Gestión de Usuarios
Origen	Manual de Procesos
Estabilidad	Alta Media Baja X
Prioridad	Primario Secundario Opcional X
Fecha de Creación	
Fecha de Cambio	

3.7	Ingresar los datos de un usuario.
3.8	Modificar los datos de un usuario.
3.9	Consultar en forma general los usuarios.
3.10	Consultar en forma individual los datos de un usuario.

4 Gestión de Procesos	
Número	4
Nombre	Gestión de Procesos
Origen	Manual de Procesos
Estabilidad	Alta Media Baja X
Prioridad	Primario Secundario Opcional X
Fecha de Creación	
Fecha de Cambio	

4.11	Ingresar un nuevo Proceso.
4.12	Consultar en forma general los Procesos de la organización.
4.13	Consultar en forma individual los Procesos de la organización.

5 Gestión de Tareas	
Número	5
Nombre	Gestión de Tareas
Origen	Manual de Procesos
Estabilidad	Alta Media Baja X
Prioridad	Primario Secundario Opcional X
Fecha de Creación	
Fecha de Cambio	

5.14	Ingresar una nueva tarea.

6 Iniciar Proceso			
Número	6		
Nombre	Iniciar Proceso		
Origen	Manual de Procesos		
Estabilidad	Alta	Media	Baja
	X		
Prioridad	Primario	Secundario	Opcional
	X		
Fecha de Creación			
Fecha de Cambio			

6.15	Iniciar un proceso.

7 Gestión de Asignación de Tareas a Usuario	
Número	7
Nombre	Gestión de Asignación de Tareas a Usuario
Origen	Manual de Procesos
Estabilidad	Alta Media Baja X
Prioridad	Primario Secundario Opcional X
Fecha de Creación	
Fecha de Cambio	

7.16	Ingresar una nueva asignación de tarea a usuario.
7.17	Modificar los datos de una asignación de tarea.

8 Ejecución de Tareas Asignadas	
Número	8
Nombre	Ejecución de Tareas Asignadas
Origen	Manual de Procesos
Estabilidad	Alta Media Baja X
Prioridad	Primario Secundario Opcional X
Fecha de Creación	
Fecha de Cambio	

8.18	Actualizar estado de tarea.
8.19	Consultar carga de actividades para cada usuario.

9 Monitoreo de Procesos	
Número	9
Nombre	Monitoreo de Procesos
Origen	Secretaría Académica
Estabilidad	Alta Media Baja X
Prioridad	Primario Secundario Opcional X
Fecha de Creación	
Fecha de Cambio	

9.20	Consultar en forma general los estados de los procesos.
9.21	Consultar en forma individual los estados de los procesos.

10 Cancelar Procesos	
Número	10
Nombre	Cancelar Procesos
Origen	Manual de Procesos
Estabilidad	Alta Media Baja X
Prioridad	Primario Secundario Opcional X
Fecha de Creación	
Fecha de Cambio	

10.22	Cancelar un proceso en ejecución.

11 Cambiar Clave	
Número	11
Nombre	Cambiar Clave
Origen	Manual de Procesos
Estabilidad	Alta Media Baja X
Prioridad	Primario Secundario Opcional X
Fecha de Creación	
Fecha de Cambio	

11.23	Cambiar la clave de acceso al sistema.

12 Modificar Datos Personales			
Número	12		
Nombre	Modificar Datos Personales		
Origen	Manual de Procesos		
Estabilidad	Alta	Media	Baja
	X		
Prioridad	Primario	Secundario	Opcional
	X		
Fecha de Creación			
Fecha de Cambio			

12.24	Modificar los datos personales de un usuarios.

4.1.2.- Estimación del SiGeSPro

4.1.2.1.- Introducción

El presente documento presenta la estimación del SiGeSPro (Sistema de Gestión de Seguimiento de Procesos) aplicando la técnica de puntos de función, documento que servirá de base para la planificación del proyecto y posterior seguimiento del mismo, con la finalidad de poder comparar lo real con lo planificado.

4.1.2.2.- Propósito

El objetivo del presente documento es el de calcular costos y tiempo del proyecto SiGeSPro (Sistema de Gestión de Seguimiento de Procesos), utilizando la técnica puntos de función.

El análisis de los puntos de función se desarrolla considerando cinco parámetros básicos externos del sistema.

1) Entrada (EI External Input).
2) Salida (EO External Output).
3) Consulta (EQ External Query).
4) Grupos de Datos Lógicos Internos (ILF).
5) Grupos de datos Lógicos Externos (EIF).

4.1.2.3.- Definición de parámetros.

Los componentes se clasifican en dos tipos de funciones que son de Datos y Transacciones.

4.1.2.4.- Función de datos

4.1.2.4.1.- Ficheros lógicos internos (ILF)

Los datos lógicos internos para SIGESPRO que se han identificado son:

Fichero lógico interno	Número de ficheros
Departamentos	1
Grupos de Usuarios	1
Procesos	1
Usuarios	1
Tareas	1
Iniciar Proceso	1
Asignación de Tareas a Usuarios	1

Una vez identificado el número de ILF, es necesario valorar su complejidad, para ello se hace necesario descubrir para cada ILF, el número de datos requeridos (DET), y el número de subgrupos identificables (RET). Obteniéndose los siguientes resultados.

Fichero lógico interno	Numero de DET	Numero de RET	Complejidad
Departamentos	2	1	Baja
Grupos de Usuarios	3	1	Baja
Usuarios	12	1	Baja
Procesos	3	1	Baja
Tareas	4	1	Baja
Iniciar Proceso	7	1	Baja
Asignación de Tareas a Usuarios	8	1	Baja

De allí tenemos que el ILF es igual a:

Complejidad	Valor
Baja	7
Media	0
Alta	0

4.1.2.4.2.- Ficheros de interfaz externos (EIF)

Los archivos de Interfaz externos hacen referencia a grupos de datos que son mantenidos por otras aplicaciones, en nuestro caso no existe teniendo que EIF es igual a cero.

Ficheros de Interfaz externo = 0

4.1.2.5.- Función de transacciones

Aquí tenemos 3 tipos de funciones que corresponden a entradas internas (EI), Salidas externas (EO) y las consultas externas (EQ).

Entradas internas

Entrada externa	Función	Número de entradas
Datos Departamentos	Alta/Modificación	3
Datos Grupos de Usuarios	-	0
Datos Procesos	Alta	1
Datos Usuarios	Alta/Modificación	2
Datos Tareas	Alta	1
Datos Inicio Proceso	Alta/Modificación	2
Datos Asignación de Tareas a Usuarios	Alta/ Modificación	2

Con los datos obtenidos pasamos a determinar la complejidad.

Entrada externa	No FTR	No DET	Complejidad
Datos Departamentos	1	2	BAJA
Datos Grupos de Usuarios	1	3	BAJA
Datos Procesos	3	3	MEDIA
Datos Usuarios	3	12	ALTA
Datos Tareas	3	4	MEDIA
Datos Inicio Proceso	5	7	MEDIA
Datos Asignación de Tareas a Usuarios	4	8	MEDIA

Con los datos de la tabla anterior, en la que determinamos la cantidad de entradas de complejidad alta, media y baja, procedemos con éstas a multiplicar por su correspondiente peso.

Entradas internas

Complejidad	Peso	Cantidad	Total
ALTA	6	1	6
MEDIA	4	4	16
BAJA	3	2	6

Salidas externas (EO).

Entrada externa	Función	Número de entradas
Consulta Departamentos	Pantalla/Papel	2
Consulta Grupos de Usuarios	Pantalla/Papel	2
Consulta Procesos	Pantalla/Papel	2
Consulta Usuarios	Pantalla/Papel	2
Consulta Tareas	Pantalla/Papel	2
Consulta Monitoreo Proceso	Pantalla/Papel	2

Consulta Asignación de Tareas a Usuarios	Pantalla/Papel	2
Consulta Ejecución de Tareas Asignadas	Pantalla	1

Con los datos obtenidos pasamos a determinar la complejidad.

Entrada externa	**No FTR**	**No DET**	**Complejidad**
Consulta de Departamentos	1	2	BAJA
Consulta de Grupos de Usuarios	1	3	BAJA
Consulta de Usuarios por Grupo de Usuario	2	12	ALTA
Consulta de Procesos	1	4	BAJA
Consulta de Procesos en Ejecución	2	7	MEDIA
Consulta de tareas por proceso	2	8	MEDIA

Con los datos de la tabla anterior, en la que determinamos cuantas entradas de complejidad alta, media y baja, procedemos con estas a multiplicar por su correspondiente peso.

Salidas externas

Complejidad	**Peso**	**Cantidad**	**Total**
ALTA	7	1	7
MEDIA	5	2	10
BAJA	4	3	12

Consultas externas

Entrada externa	**Función**	**Número de entradas**
Consulta Individual Departamentos	Pantalla/Papel	2
Consulta Roles de Usuarios	Pantalla/Papel	2
Consulta Individual Procesos	Pantalla/Papel	2

Consulta Individual Usuarios	Pantalla/Papel	2
Consulta Individual Monitoreo Proceso	Pantalla/Papel	2
Consulta Asignación de Tareas a Usuarios	Pantalla/Papel	2
Consulta Ejecución de Tareas Asignadas	Pantalla	1

Con los datos obtenidos pasamos a determinar la complejidad.

Para la entrada

Entrada externa	**No FTR**	**No DET**	**Complejidad**
Consulta Individual Departamentos	1	1	BAJA
Consulta Roles de Usuarios	1	1	BAJA
Consulta Individual Procesos	2	2	BAJA
Consulta Individual Usuarios	3	2	BAJA
Consulta Individual Monitoreo Proceso	3	2	BAJA
Consulta Asignación de Tareas a Usuarios	4	2	BAJA
Consulta Ejecución de Tareas Asignadas	4	2	BAJA

Para la salida

Entrada externa	**No FTR**	**No DET**	**Complejidad**
Consulta Individual Departamentos	1	2	BAJA
Consulta Roles de Usuarios	1	3	BAJA
Consulta Individual Procesos	2	12	ALTA
Consulta Individual Usuarios	3	4	MEDIA
Consulta Individual Monitoreo Proceso	3	7	MEDIA
Consulta Asignación de Tareas a Usuarios	4	8	MEDIA

Consulta Ejecución de Tareas Asignadas	4	8	MEDIA

Tomando la mayor de ambas tenemos**.**

Entrada externa	**C. Entrada**	**C. Salida**	**Complejidad**
Consulta Individual Departamentos	BAJA	BAJA	BAJA
Consulta Roles de Usuarios	BAJA	BAJA	BAJA
Consulta Individual Procesos	BAJA	ALTA	ALTA
Consulta Individual Usuarios	BAJA	MEDIA	MEDIA
Consulta Individual Monitoreo Proceso	BAJA	MEDIA	MEDIA
Consulta Asignación de Tareas a Usuarios	BAJA	MEDIA	MEDIA
Consulta Ejecución de Tareas Asignadas	BAJA	MEDIA	MEDIA
Consulta Individual Departamentos	BAJA	BAJA	BAJA

Con los datos de la tabla anterior, en la que determinamos cuantas entradas de complejidad alta, media y baja, procedemos con estas a multiplicar por su correspondiente peso.

Consultas externas

Complejidad	**Peso**	**Cantidad**	**Total**
ALTA	6	1	6
MEDIA	4	4	16
BAJA	3	3	9

4.1.3.- MODELO CONCEPTUAL

4.1.3.1.- Lista de Conceptos, Definiciones

Empresa. Unidad de organización dedicada a actividades industriales, mercantiles o de prestación de servicios con fines lucrativos.

Departamentos. División administrativa.

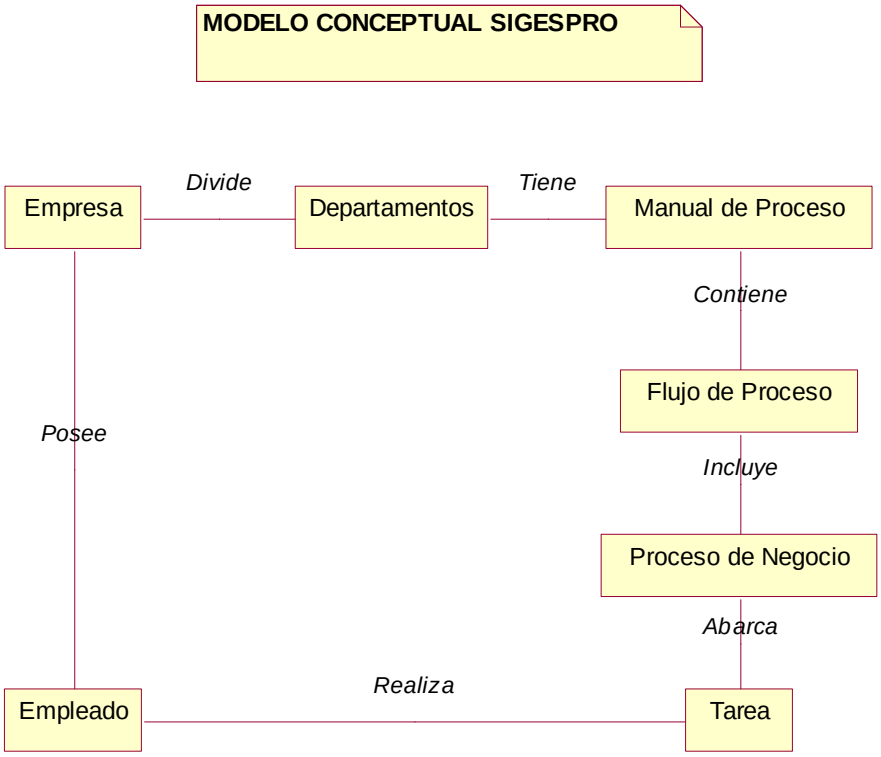
Manual de Procesos. Libro donde se recopila los procedimientos existentes dentro de una empresa y que se deben seguir para la consecución de los objetivos.

Flujos de Proceso. Es la secuencia de acciones y pasos usados en procesos de negocios.

Proceso de Negocio. Conjunto de uno o más procedimientos o actividades directamente ligadas, que colectivamente realizan un objetivo del negocio.

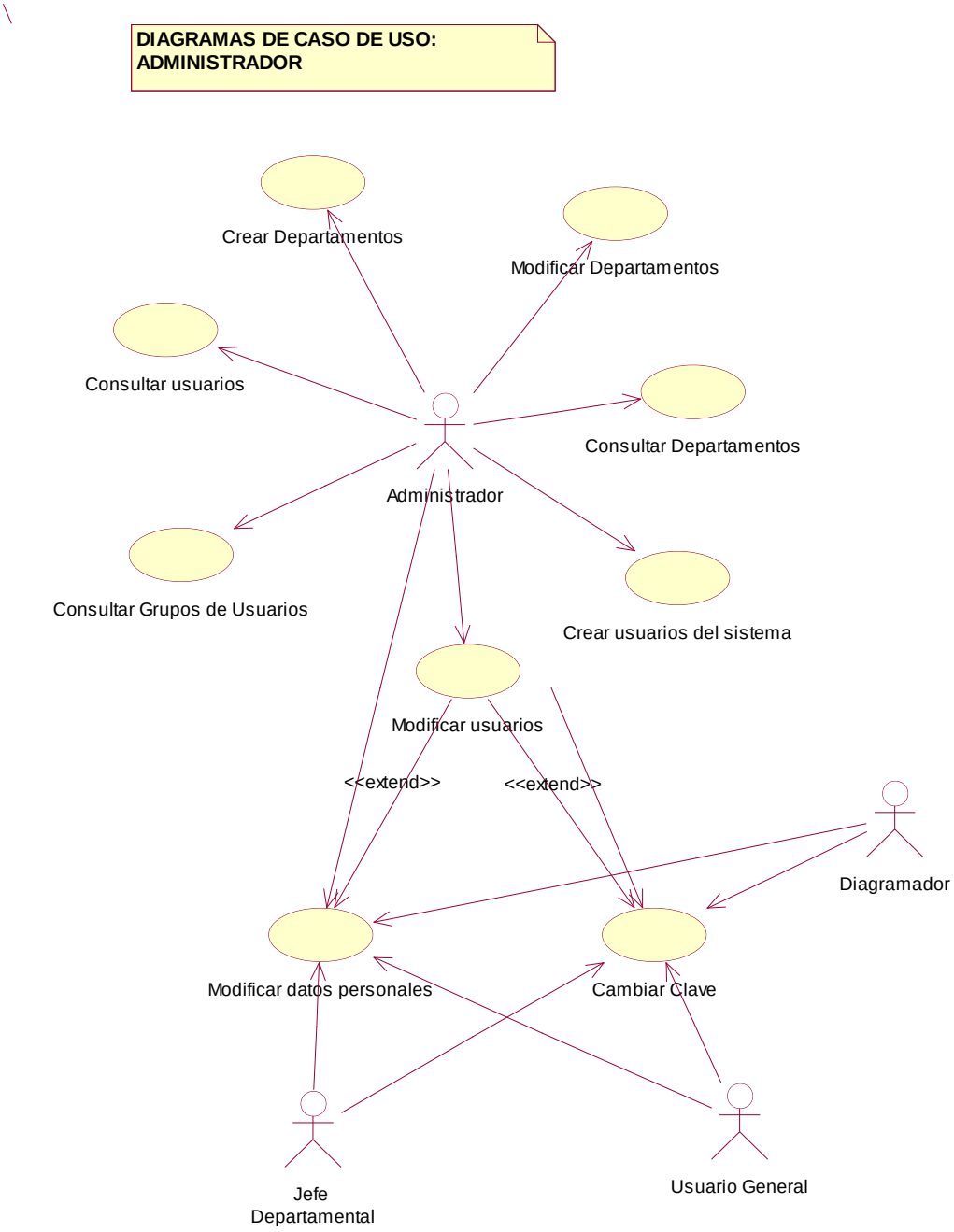
Documentos, información o tareas. son los elementos que son distribuidos a los empleados para que actúen.

Empleado. Persona que realiza actividades de cualquier tipo para una empresa.

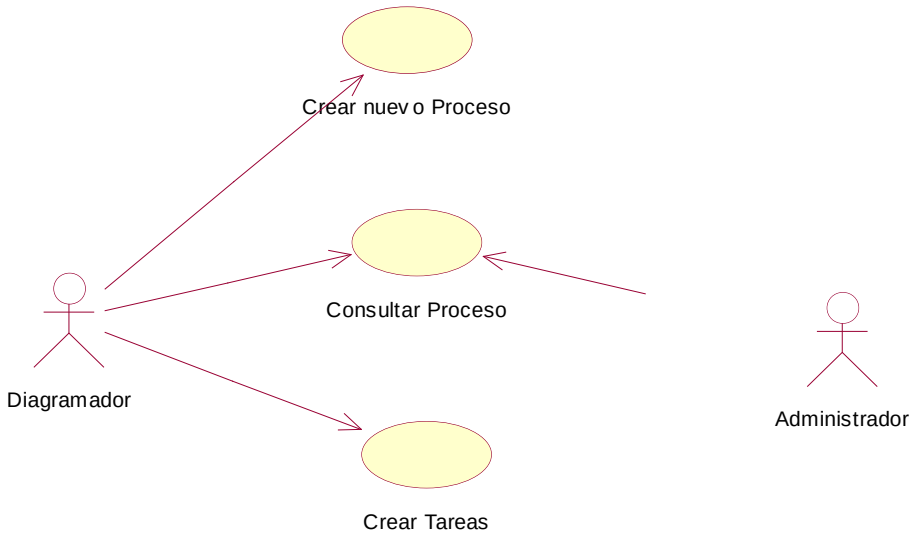

MODELO CONCEPTUAL SIGESPRO
Empresa
Divide
Departamentos
Tiene
Manual de Proceso
Contiene
Flujo de Proceso
Posee
Incluye
Proceso de Negocio
Abarca
Realiza
Empleado
Tarea

4.2.- FASE DE DISEÑO

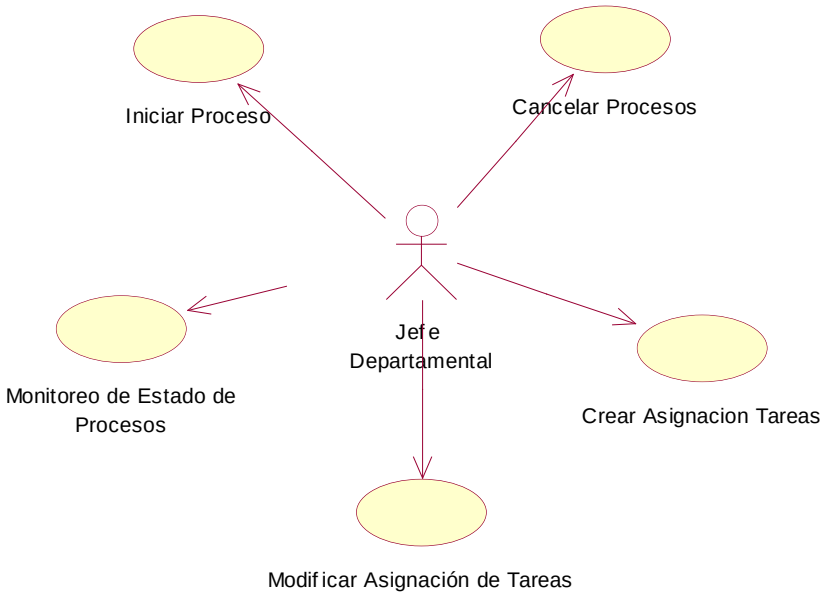
4.2.1.- DIAGRAMAS DE CASOS DE USO



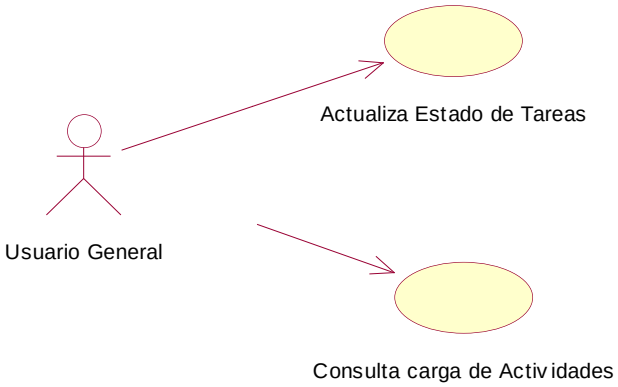
DIAGRAMAS DE CASO DE USO:
DIAGRAMADOR



DIAGRAMAS DE CASO DE USO:
JEFE DEPARTAMENTAL



DIAGRAMAS DE CASO DE USO
USUARIO GENERAL



4.2.1.1.- Definición de Casos de Uso de Alto nivel

NOMBRE DEL CASO DE USO: CREAR DEPARTAMENTOS
ACTOR: Administrador
TIPO: Primario
DESCRIPCIÓN: El administrador ingresa los datos del departamento, el sistema valida datos, almacena y termina el caso de uso.

NOMBRE DEL CASO DE USO: MODIFICAR DEPARTAMENTOS
ACTOR: Administrador
TIPO: Primario
DESCRIPCIÓN: El administrador selecciona al usuario que desea modificar de la lista de usuarios, el sistema lo modifica y termina el caso de uso.

NOMBRE DEL CASO DE USO: CONSULTAR EN FORMA GENERAL DEPARTAMENTOS
ACTOR: Administrador
TIPO: Primario
DESCRIPCIÓN: El administrador selecciona lista general de departamentos, el sistema despliega la lista, termina el caso de uso.

NOMBRE DEL CASO DE USO: CONSULTAR EN FORMA INDIVIDUAL DEPARTAMENTOS
ACTOR: Administrador
TIPO: Primario
DESCRIPCIÓN: El administrador selecciona un departamento específico, el sistema despliega la información, termina el caso de uso.

NOMBRE DEL CASO DE USO: CONSULTAR EN FORMA GENERAL GRUPOS DE USUARIOS
ACTOR: Administrador
TIPO: Primario

DESCRIPCIÓN: El administrador selecciona lista general de grupos de usuarios, el sistema despliega la lista, termina el caso de uso.

NOMBRE DEL CASO DE USO: CONSULTAR ROLES DE USUARIOS
ACTOR: Administrador
TIPO: Primario
DESCRIPCIÓN: El administrador ingresa apellido del usuario, el sistema despliega rol, termina el caso de uso.

NOMBRE DEL CASO DE USO: CREAR USUARIOS DEL SISTEMA
ACTOR: Administrador
TIPO: Primario
DESCRIPCIÓN: El administrador ingresa los datos personales del usuario, el sistema valida datos, almacena y termina el caso de uso.

NOMBRE DEL CASO DE USO: MODIFICAR USUARIOS
ACTOR: Administrador
TIPO: Primario
DESCRIPCIÓN: El administrador selecciona al usuario que desea modificar de la lista de usuarios, el sistema lo modifica y termina el caso de uso.

NOMBRE DEL CASO DE USO: CONSULTAR EN FORMA GENERAL USUARIOS
ACTOR: Administrador
TIPO: Primario
DESCRIPCIÓN: El administrador selecciona lista general de usuarios, el sistema despliega la lista, termina el caso de uso.

NOMBRE DEL CASO DE USO: CONSULTAR EN FORMA INDIVIDUAL USUARIOS
ACTOR: Administrador
TIPO: Primario

DESCRIPCIÓN: El administrador selecciona un usuario específico, el sistema despliega la información, termina el caso de uso.

NOMBRE DEL CASO DE USO: CREAR NUEVO PROCESO
ACTOR: Diagramador
TIPO: Primario
DESCRIPCIÓN: El diagramador grafica el flujo de proceso en el diseñador, el sistema valida los datos almacena, termina el caso de uso.

NOMBRE DEL CASO DE USO: CONSULTAR PROCESOS EN FORMA GENERAL
ACTOR: Diagramador
TIPO: Primario
DESCRIPCIÓN: El diagramador selecciona la lista general de procesos, el sistema despliega la lista, termina el caso de uso.

NOMBRE DEL CASO DE USO: CONSULTAR INDIVIDUALMENTE PROCESOS
ACTOR: Diagramador
TIPO: Primario
DESCRIPCIÓN: El diagramador selecciona individualmente los procesos, el sistema despliega la información, termina el caso de uso.

NOMBRE DEL CASO DE USO: CREAR TAREAS
ACTOR: Diagramador
TIPO: Primario
DESCRIPCIÓN: El diagramador ingresa los datos de las tareas del proceso, el sistema valida datos almacena, termina el caso de uso.

NOMBRE DEL CASO DE USO: INICIAR PROCESO
ACTOR: Jefe Departamental
TIPO: Primario

DESCRIPCIÓN: El jefe departamental ingresa los datos del inicio de un nuevo proceso, el sistema valida datos almacena, termina el caso de uso.

NOMBRE DEL CASO DE USO: CREAR ASIGNACIÓN DE TAREAS A USUARIOS
ACTOR: Jefe Departamental
TIPO: Primario
DESCRIPCIÓN: El jefe departamental ingresa los datos de la nueva asignación de tarea a usuario, el sistema valida datos almacena, termina el caso de uso.

NOMBRE DEL CASO DE USO: MODIFICAR ASIGNACIÓN DE TAREAS A USUARIOS
ACTOR: Jefe Departamental
TIPO: Primario
DESCRIPCIÓN: El jefe departamental selecciona la asignación de tarea que desea modificar de la lista de asignaciones de tareas, el sistema la modifica, termina el caso de uso.

NOMBRE DEL CASO DE USO: ACTUALIZAR ESTADO DE TAREA
ACTOR: Usuario General
TIPO: Primario
DESCRIPCIÓN: El usuario general selecciona la tarea que va actualizar del listado de tareas, el sistema modifica, termina el caso de uso.

NOMBRE DEL CASO DE USO: CONSULTAR CARGA DE ACTIVIDADES GENERAL
ACTOR: Usuario General
TIPO: Primario
DESCRIPCIÓN: El usuario general selecciona la lista general de carga de actividades asignadas a este, el sistema despliega la lista, termina el caso de uso.

NOMBRE DEL CASO DE USO: CONSULTAR EN FORMA GENERAL LOS ESTADOS DE LOS PROCESOS

ACTOR: Jefe Departamental

TIPO: Primario

DESCRIPCIÓN: El jefe departamental selecciona lista general de monitoreos de los estados de los procesos, el sistema despliega la lista, termina el caso de uso.

NOMBRE DEL CASO DE USO: CONSULTAR EN FORMA INDIVIDUAL LOS ESTADOS DE LOS PROCESOS.

ACTOR: Jefe Departamental

TIPO: Primario

DESCRIPCIÓN: El jefe departamental selecciona de la lista de monitoreos de los estados de procesos un determinado proceso, el sistema despliega la información, termina el caso de uso.

NOMBRE DEL CASO DE USO: CANCELAR PROCESO EN EJECUCION

ACTOR: Jefe Departamental

TIPO: Primario

DESCRIPCIÓN: El jefe departamental selecciona el proceso en ejecución que va actualizar del listado de procesos, el sistema modifica, termina el caso de uso.

NOMBRE DEL CASO DE USO: CAMBIAR CLAVE

ACTOR: Usuario del Sistema (Administrador, Diagramador, Jefe Departamental, Usuario General)

TIPO: Primario

DESCRIPCIÓN: El usuario del sistema ingresa la clave actual y la nueva, el sistema modifica clave, termina el caso de uso.

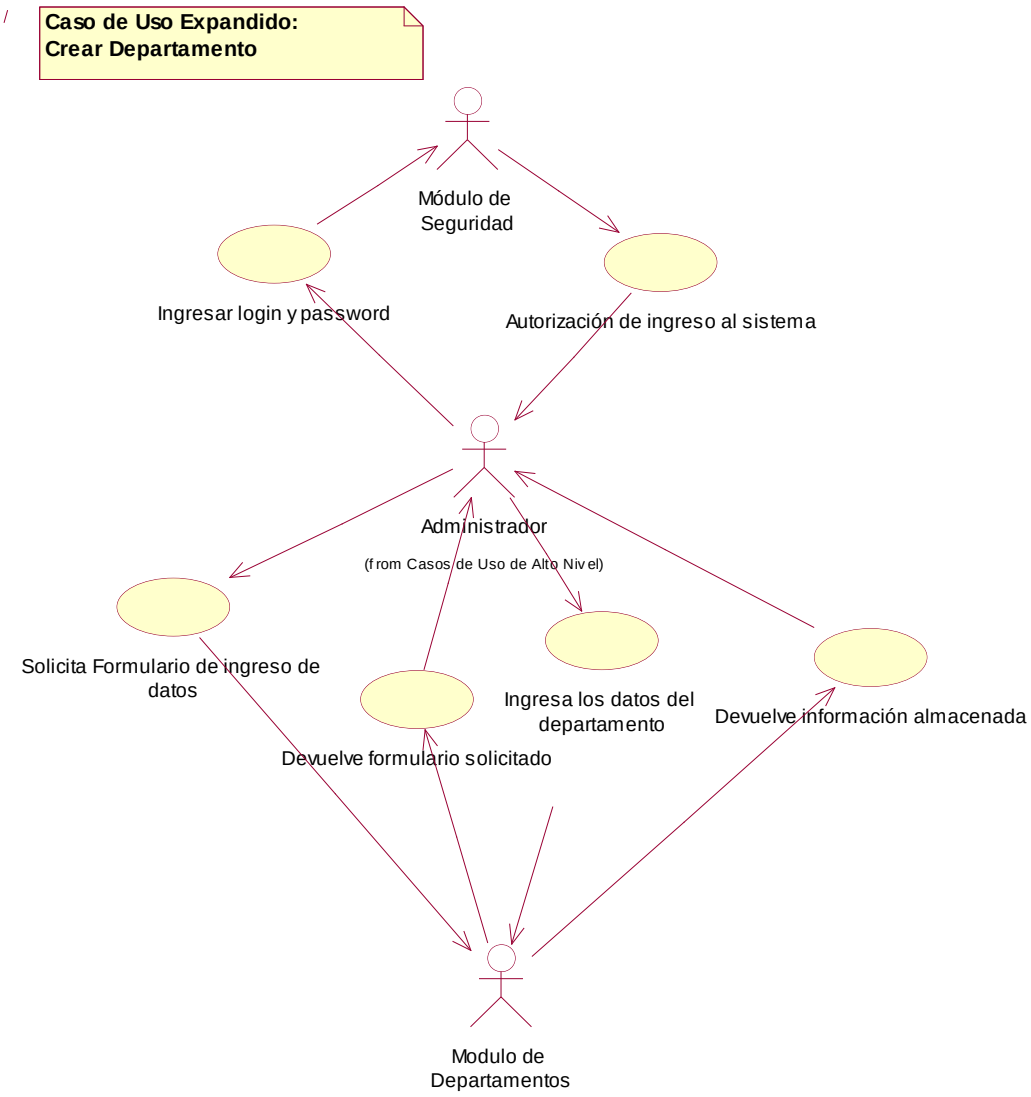
NOMBRE DEL CASO DE USO: MODIFICAR DATOS PERSONALES

ACTOR: Usuario del Sistema (Administrador, Diagramador, Jefe Departamental, Usuario General)

TIPO: Primario

DESCRIPCIÓN: El usuario del sistema ingresa la clave actual y la nueva, el sistema modifica clave, termina el caso de uso.

4.2.1.2.- Definición de Casos de Uso Expandidos, Diagramas de Secuencia, Contratos de Operación, Diagramas de Colaboración



NOMBRE DEL CASO DE USO: CREAR DEPARTAMENTOS.

PROPÓSITO: Establecer departamentos en el sistema.

ACTOR (Iniciador): Administrador.

TIPO: Primario Real.

VISION GENERAL: El administrador ingresa su login y password, el sistema autoriza el ingreso al sistema, el administrador solicita el formulario para el ingreso de datos del departamento, el sistema despliega el formulario solicitado, el administrador ingresa los datos en el formulario requeridos para la creación del nuevo departamento, el sistema devuelve información almacenada y termina el caso de uso.

CURSO TIPICO DE EVENTOS

ACTOR	**SISTEMA**
1. Ingresar login y password	2. Autorizar ingreso al sistema
3.Solicitar formulario ingreso departamento	4.Devolver formulario solicitado
5. Ingresar los datos del departamento	6. Devolver información almacenada

CURSOS ALTERNATIVOS:

2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

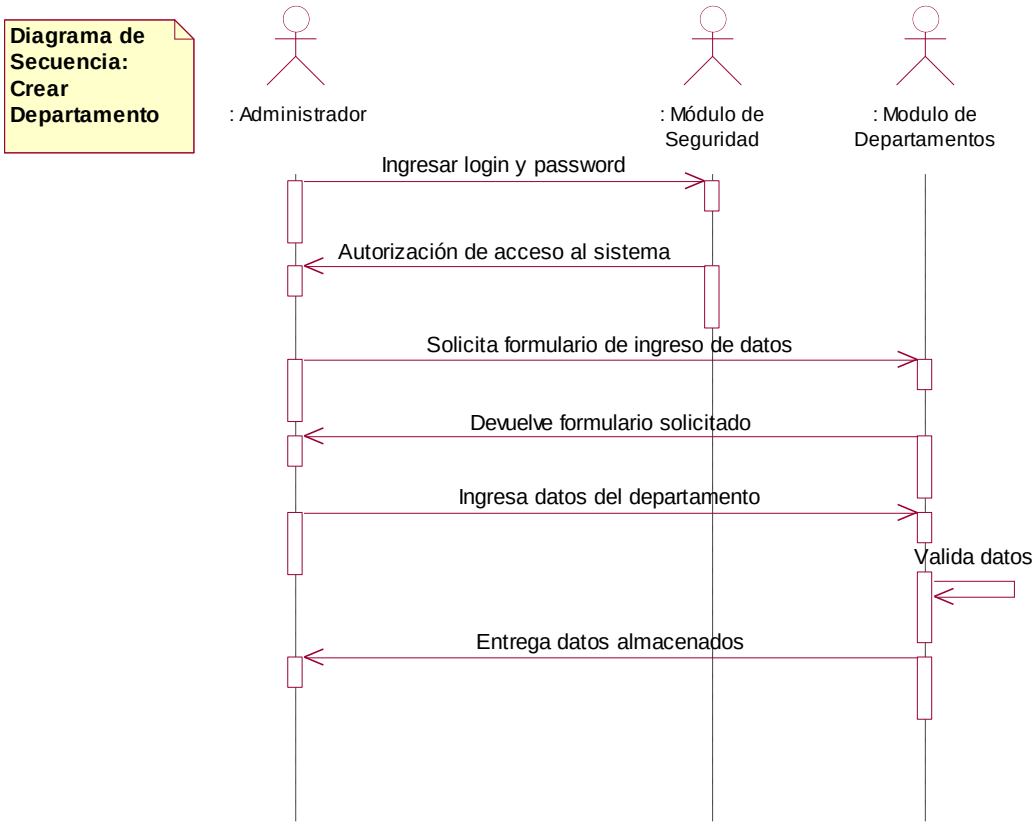
2* Login y/o password no validos. “Termina caso de uso”

4* No existe formulario de ingreso de datos del departamento. ”Termina caso de uso”.

6* No presenta mensaje de información almacenada exitosamente. “Termina caso de uso”.

6* Presenta mensaje de ingresar datos requeridos. “Termina caso de uso”.

DIAGRAMA DE SECUENCIA



CONTRATO DE OPERACION

NOMBRE DEL CONTRATO: Ingresar login y password.

PROPÓSITO: Acceder al sistema para creación de nuevos departamentos (transacción).

TIPO: Sistema.

SALIDA: Acceso al sistema

EXCEPCIONES: Login y/o password no validos.

No existe formulario de ingreso de login y password.

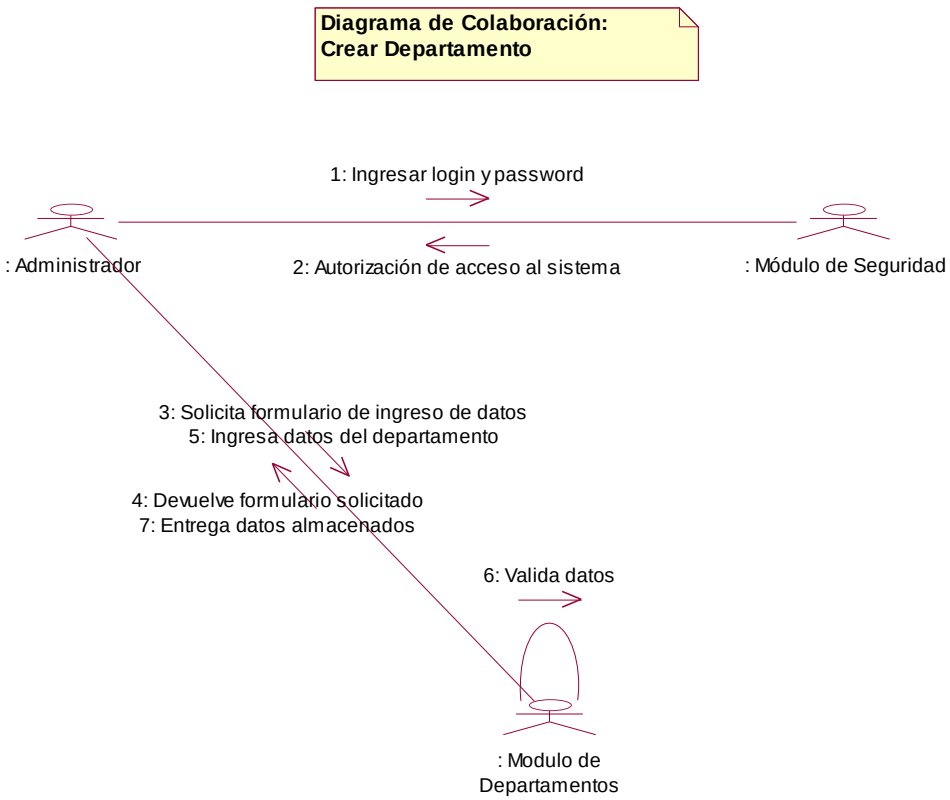
PRE-CONDICIONES: Relación usuario - interfaz

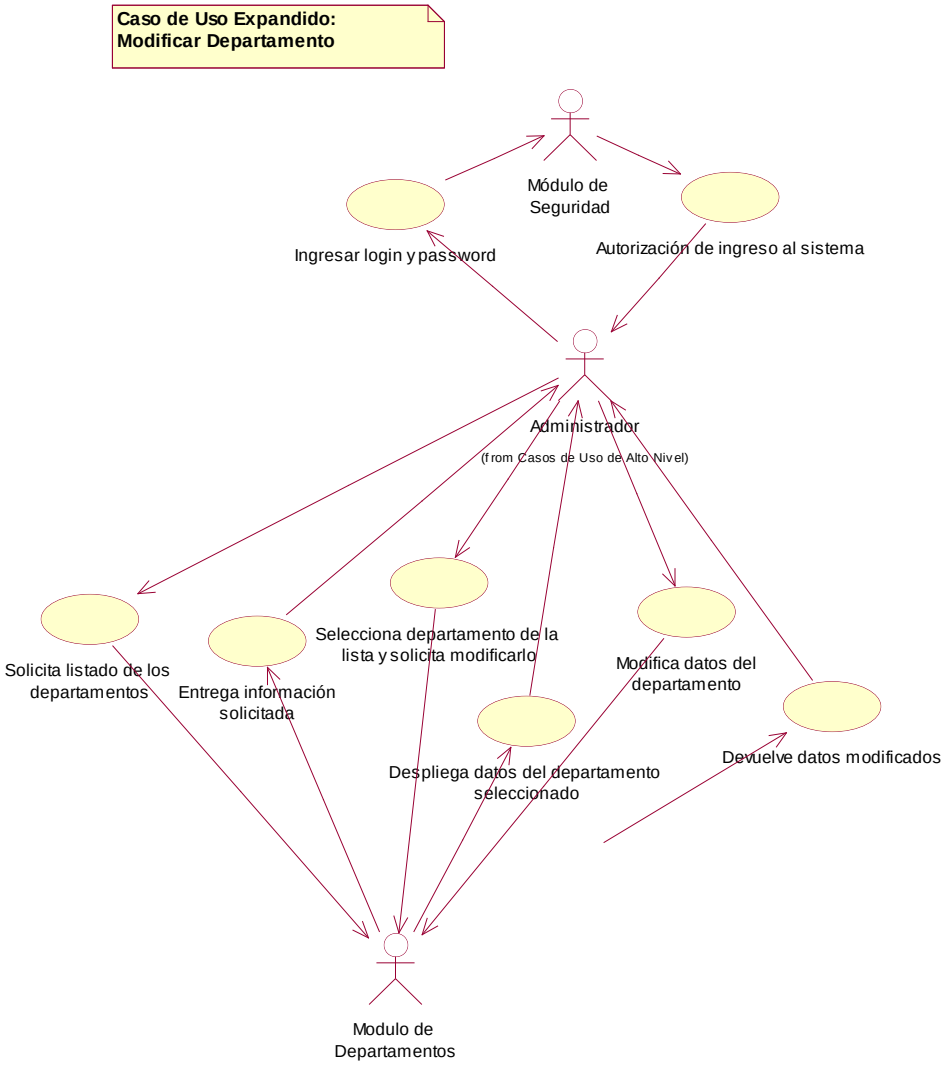
POST-CONDICIONES:

NOMBRE DEL CONTRATO: Solicitar formulario departamento.
PROPÓSITO: Acceder al formulario de ingreso de datos para creación de departamentos.
TIPO: Sistema.
SALIDA: Formulario ingreso de datos de departamentos.
EXCEPCIONES: No existe formulario de ingreso de datos del departamento.
PRE-CONDICIONES: Relación usuario – interfaz (formulario).
POST-CONDICIONES:

NOMBRE DEL CONTRATO: Ingresar datos del departamento.
PROPÓSITO: Ingresar al sistema datos para creación de nuevos departamentos(transacción).
TIPO: Sistema.
SALIDA: Formulario ingreso de datos.
EXCEPCIONES: No presenta mensaje de información almacenada exitosamente.
Presenta mensaje de ingresar datos requeridos.
PRE-CONDICIONES: Relación usuario - interfaz
POST-CONDICIONES:

DIAGRAMA DE COLABORACION





NOMBRE DEL CASO DE USO: MODIFICAR DEPARTAMENTOS DEL SISTEMA.

PROPÓSITO: Cambiar datos de departamentos en el sistema.

ACTOR (Iniciador): Administrador.

TIPO: Primario Real.

VISION GENERAL: El administrador ingresa su login y password, el sistema autoriza el ingreso al sistema, el administrador solicita el listado de departamentos existentes en el sistema, el sistema devuelve información solicitada, el administrador escoge el departamento y solicita desplegar datos, el sistema presenta datos de departamento a modificar, el administrador modifica datos de

departamento, el sistema devuelve datos de departamento modificado termina el caso de uso.

CURSO TIPICO DE EVENTOS

ACTOR	SISTEMA
1. Ingresar login y password.	2. Autorizar ingreso al sistema.
3. Solicitar listado de departamento existentes.	4. Presentar listado de departamento existentes.
5.Seleccionar departamento y Solicitar modificarlo	6.Presentar datos de departamento a modificar
7.Modificar datos de departamento	8. Devolver datos de departamento modificado

CURSOS ALTERNATIVOS:

2* Login y/o password no válidos. “Termina caso de uso”.

2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

4* No existe departamentos en el sistema. ”Termina caso de uso”.

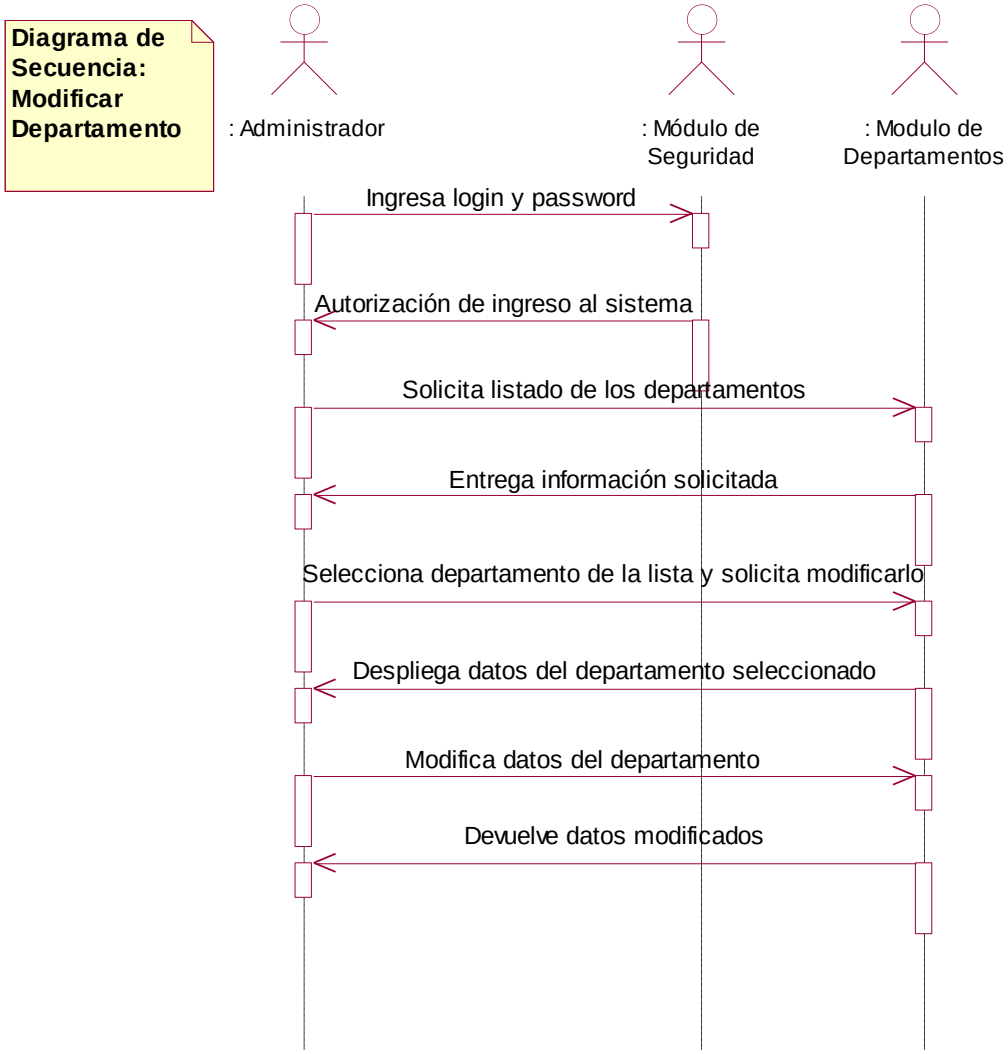
4* No presenta formulario con el listado de departamentos. “Termina caso de uso”.

6* No presenta datos de departamento a modificar ”Termina caso de uso”.

8* No presenta mensaje de departamento modificado. “Termina caso de uso”.

8* No se puede modificar registro. “Termina caso de uso”.

DIAGRAMA DE SECUENCIA



CONTRATO DE OPERACIÓN

NOMBRE DEL CONTRATO: Ingresar login y password.

PROPÓSITO: Acceder al sistema para modificación de departamento (transacción).

TIPO: Sistema.

SALIDA: Autorización de ingreso al sistema.

EXCEPCIONES: Login y/o password no válidos.

No existe formulario de ingreso de login y password.

PRE-CONDICIONES: Relación usuario - interfaz

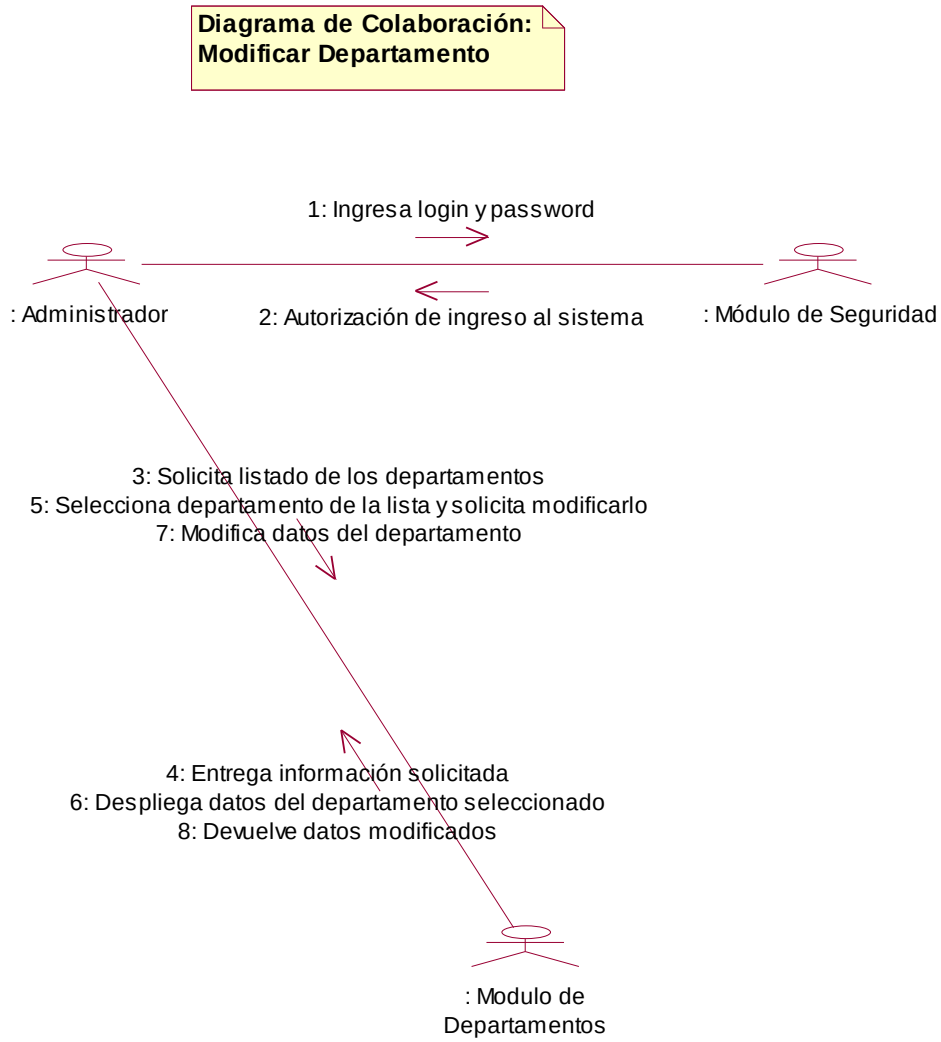
POST-CONDICIONES:

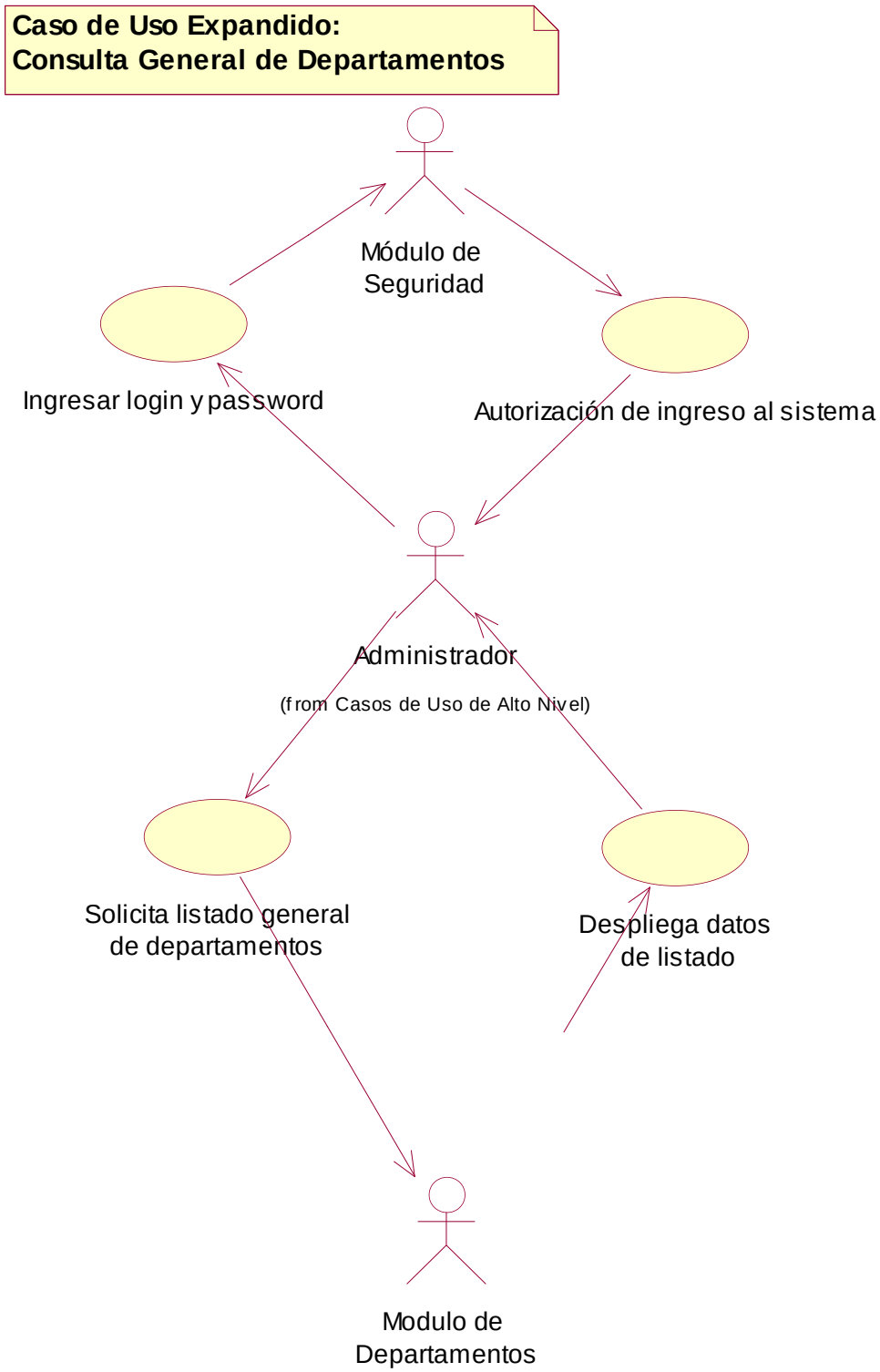
NOMBRE DEL CONTRATO: Solicitar listado de departamentos.
PROPÓSITO: Obtener una lista de los departamentos existentes (transacción).
TIPO: Sistema.
SALIDA: Lista de departamentos existentes.
EXCEPCIONES: No existen departamentos en el sistema.
No presenta formulario con el listado de departamentos.
PRE-CONDICIONES: Relación usuario - interfaz
POST-CONDICIONES:

NOMBRE DEL CONTRATO: Seleccionar departamento y solicitar modificarlo.
PROPÓSITO: Escoger departamento para la modificación y solicitar su modificación (transacción).
TIPO: Sistema.
SALIDA: Formulario de actualización de datos de departamento.
EXCEPCIONES: No presenta datos de departamento a .
PRE-CONDICIONES: Relación usuario - interfaz
POST-CONDICIONES:

NOMBRE DEL CONTRATO: Modificar datos de departamento.
PROPÓSITO: Cambiar datos de departamento.
TIPO: Sistema.
SALIDA: Formulario de usuarios
EXCEPCIONES: No presenta mensaje de departamento modificado.
No se puede modificar registro.
PRE-CONDICIONES: Relación usuario - interfaz
POST-CONDICIONES:

DIAGRAMA DE COLABORACIÓN




Caso de Uso Expandido:
Consulta General de Departamentos
Módulo de
Seguridad
Ingresar login y password
Autorización de ingreso al sistema
Administrador
(from Casos de Uso de Alto Nivel)
Solicita listado general
de departamentos
Despliega datos
de listado
Modulo de
Departamentos

NOMBRE DEL CASO DE USO: CONSULTAR DEPARTAMENTOS DEL SISTEMA EN FORMA GENERAL.

PROPÓSITO: Informarse datos de departamentos en el sistema en forma general

ACTOR (Iniciador): Administrador.

TIPO: Primario Real.

VISION GENERAL: El administrador ingresa su login y password, el sistema autoriza el ingreso al sistema, el administrador solicita listado general de departamentos existentes en el sistema, el sistema devuelve y despliega listado información solicitada termina caso de uso.

CURSO TIPICO DE EVENTOS

ACTOR	SISTEMA
1. Ingresar login y password.	2. Autorizar ingreso al sistema.
3. Solicitar listado general de departamentos.	4. Presentar listado de departamentos.

CURSOS ALTERNATIVOS:

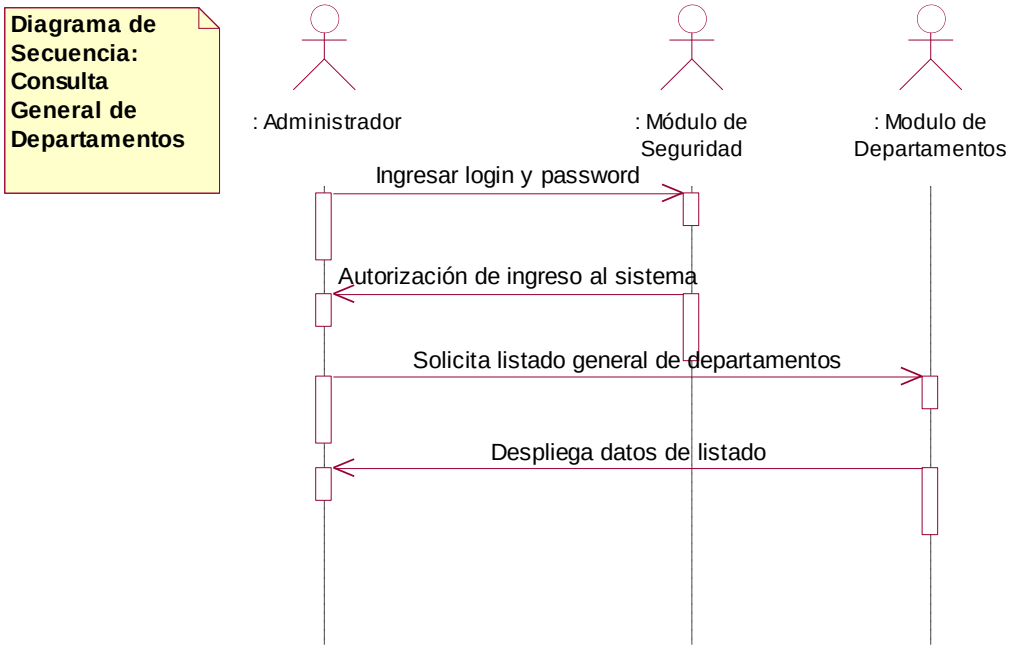
2* Login y/o password inválidos. “Termina caso de uso”.

2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

4* No existen departamentos en el sistema. ”Termina caso de uso”.

4* No presenta formulario con el listado de departamentos. “Termina caso de uso”.

DIAGRAMA DE SECUENCIA



CONTRATO DE OPERACIÓN

NOMBRE DEL CONTRATO: Ingresar login y password.

PROPÓSITO: Acceder al sistema para la consulta general de departamentos.

TIPO: Sistema.

SALIDA: Autorización de ingreso al sistema.

EXCEPCIONES: Login y/o password inválidos.
No existe formulario de ingreso de login y password.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Solicitar listado general de departamentos.

PROPÓSITO: Desplegar listado de todos los departamentos del sistema.

TIPO: Sistema.

SALIDA: Listado general de departamentos.

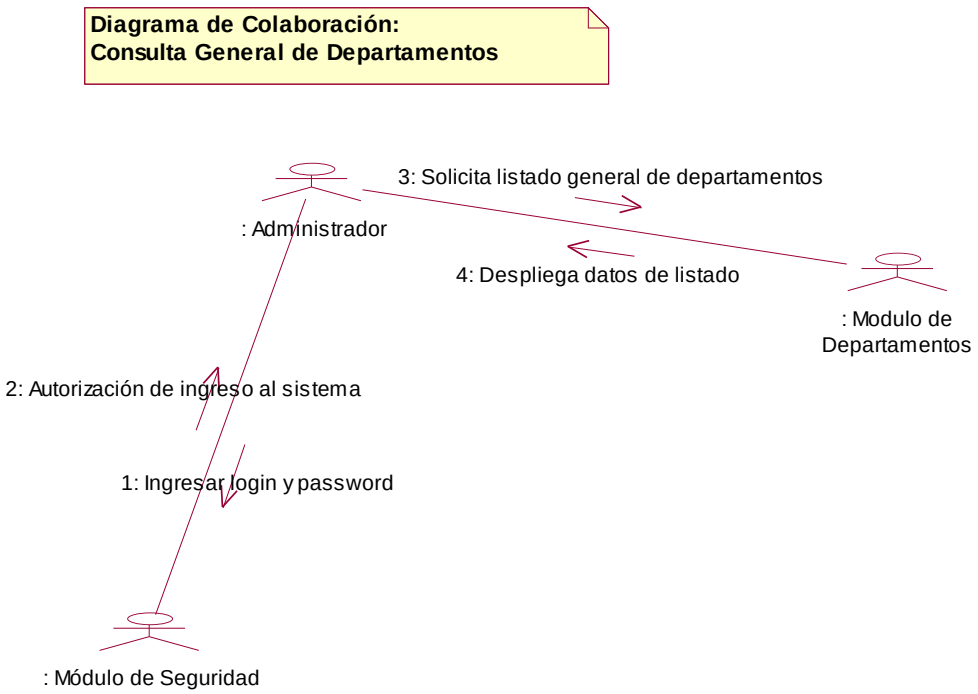
EXCEPCIONES: No existen departamentos en el sistema.

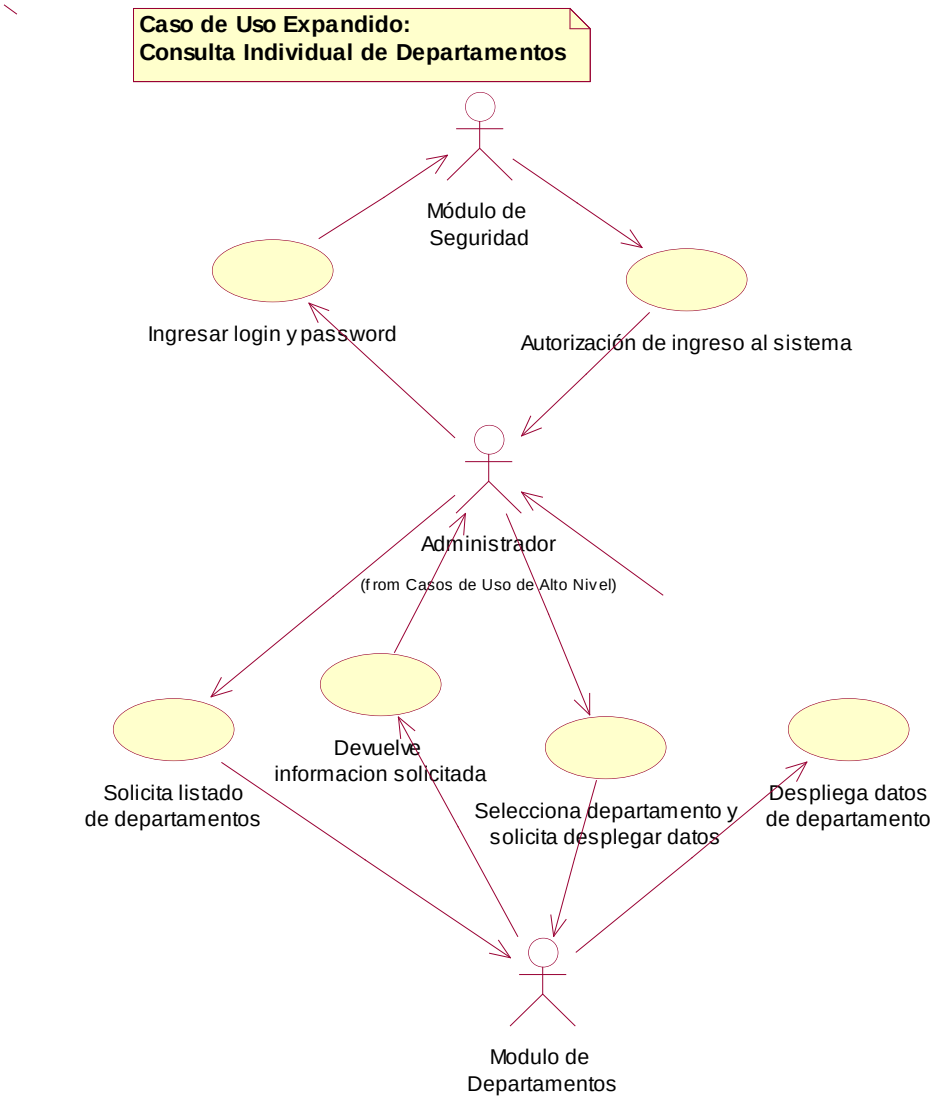
No presenta formulario con el listado de departamentos.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

DIAGRAMA DE COLABORACION





NOMBRE DEL CASO DE USO: CONSULTAR DEPARTAMENTOS DEL SISTEMA EN FORMA INDIVIDUAL.

PROPÓSITO: Conocer datos de departamentos del sistema en forma individual.

ACTOR (Iniciador): Administrador.

TIPO: Primario Real.

VISION GENERAL: El administrador ingresa su login y password, el sistema autoriza el ingreso al sistema, el administrador solicita listado de departamentos existentes en el sistema, el sistema devuelve y presenta información solicitada, el

administrador escoge un departamento y solicita desplegar datos, el sistema presenta datos de departamento termina el caso de uso.

CURSO TIPICO DE EVENTOS

ACTOR	SISTEMA
1. Ingresar login y password.	2. Autorizar ingreso al sistema.
3. Solicitar listado de departamentos.	4. Presentar listado de departamentos.
5. Escoger usuario y Solicitar desplegar datos de departamento.	6. Presentar datos de departamento.

CURSOS ALTERNATIVOS:

2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

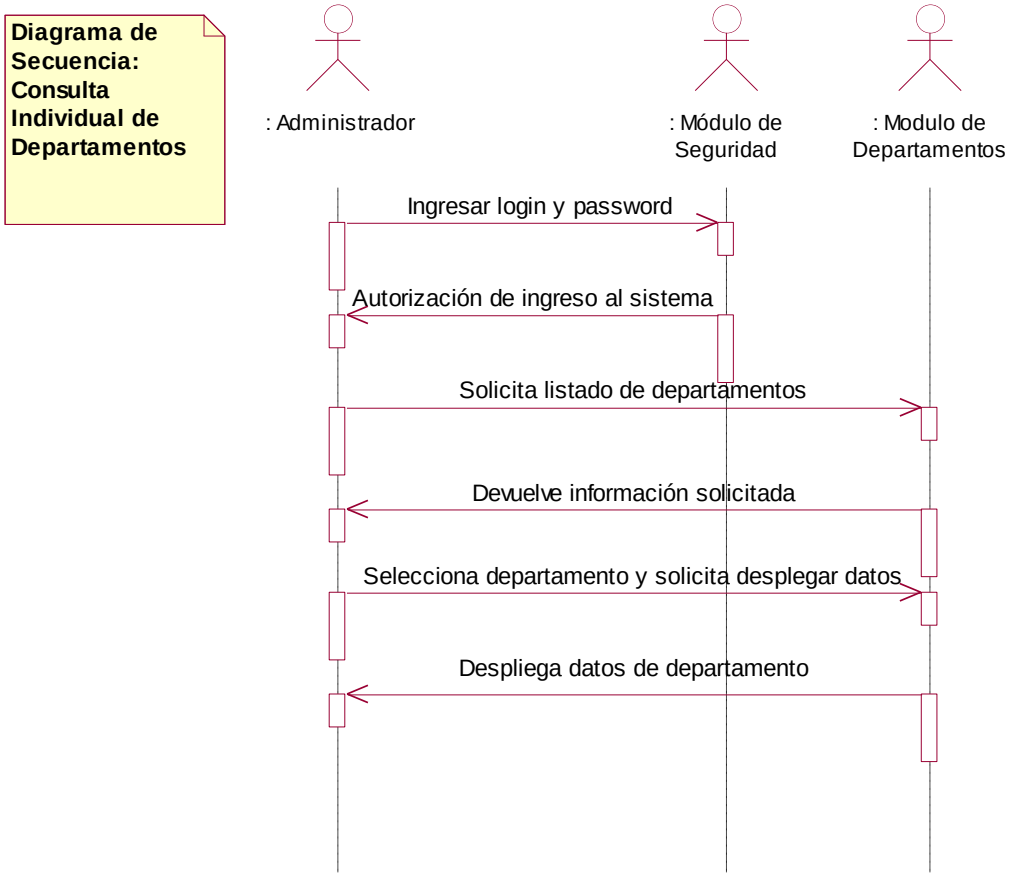
2* Login y/o password incorrecto. “Termina caso de uso”.

4* No existen departamentos. ”Termina caso de uso”.

4* No presenta formulario con el listado de departamentos. “Termina caso de uso”.

6* No presenta datos. ”Termina caso de uso”.

DIAGRAMA DE SECUENCIA



CONTRATO DE OPERACIÓN

NOMBRE DEL CONTRATO: Ingresar login y password.

PROPÓSITO: Acceder al sistema para la consulta individual de departamentos.

TIPO: Sistema.

SALIDA: Autorización de ingreso al sistema.

EXCEPCIONES: Login y/o password incorrectos.

No existe formulario de ingreso de login y password.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Solicitar listado de departamentos.

PROPÓSITO: Observar los departamentos que posee el sistema.

TIPO: Sistema.

SALIDA: Listado de departamentos

EXCEPCIONES: No existen departamentos.
No presenta formulario con el listado de departamentos.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Seleccionar departamento y solicitar despliegue datos

PROPÓSITO: Elegir un departamento para que se despliegue la información del mismo.

TIPO: Sistema.

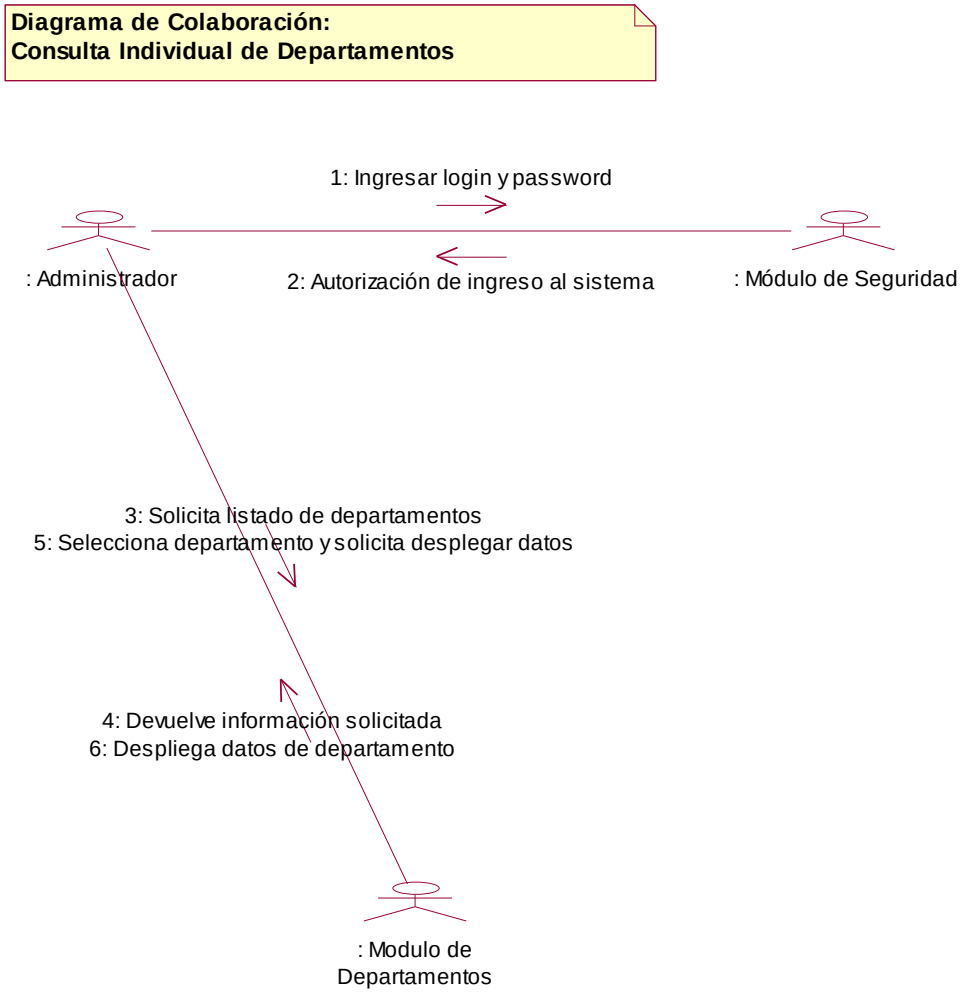
SALIDA: Datos del departamento solicitado.

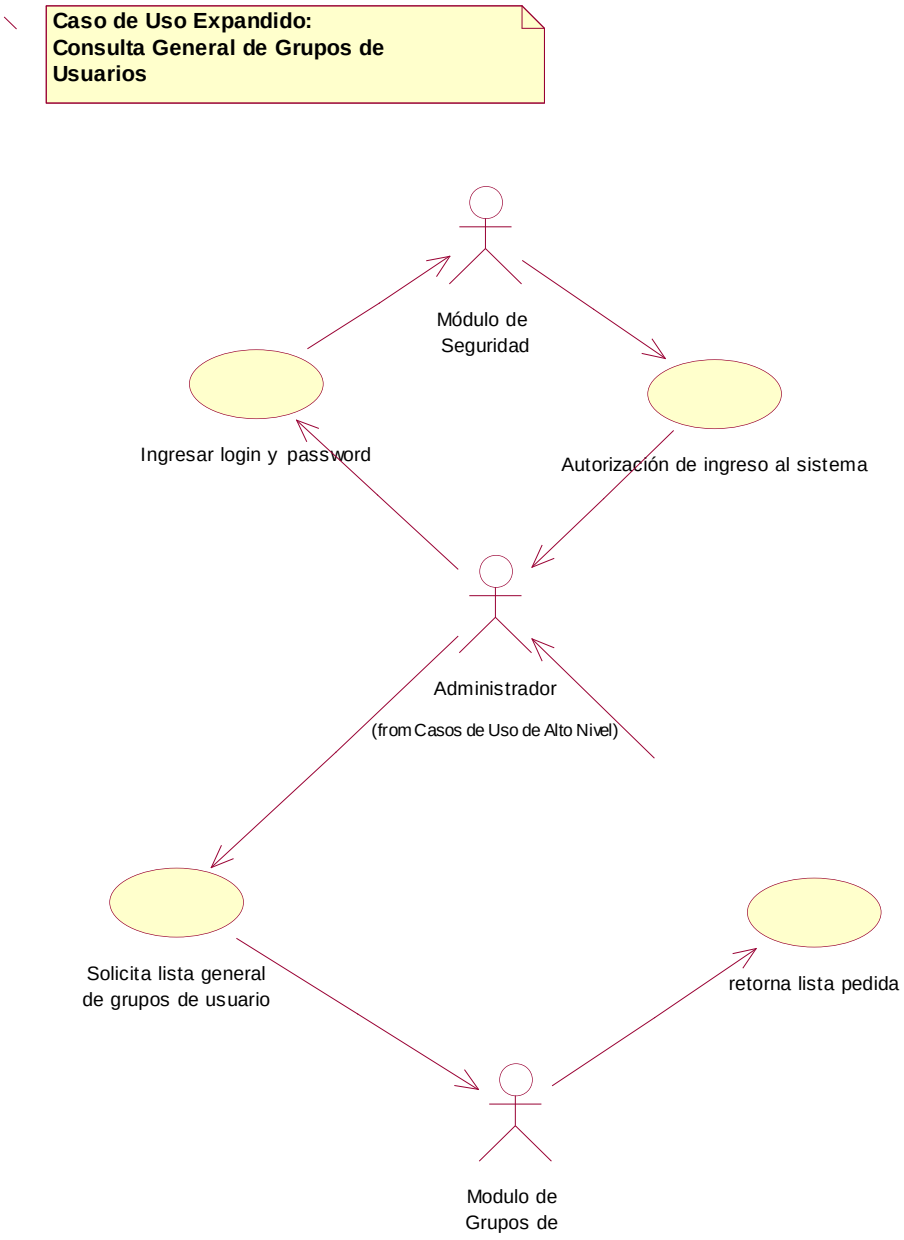
EXCEPCIONES: No presenta datos.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

DIAGRAMA DE COLABORACION





NOMBRE DEL CASO DE USO: CONSULTAR GRUPOS DE USUARIOS DEL SISTEMA EN FORMA GENERAL.

PROPÓSITO: Informarse de los grupos de usuarios en el sistema en forma general

ACTOR (Iniciador): Administrador.

TIPO: Primario Real.

VISION GENERAL: El administrador ingresa su login y password, el sistema autoriza el ingreso al sistema, el administrador solicita listado general de grupos de usuarios existentes en el sistema, el sistema despliega listado de la información solicitada termina caso de uso.

CURSO TIPICO DE EVENTOS

ACTOR	SISTEMA
1. Ingresar login y password.	2. Autorizar ingreso al sistema.
3. Solicitar listado general de grupos de usuarios.	4. Presentar listado de grupos de usuarios.

CURSOS ALTERNATIVOS:

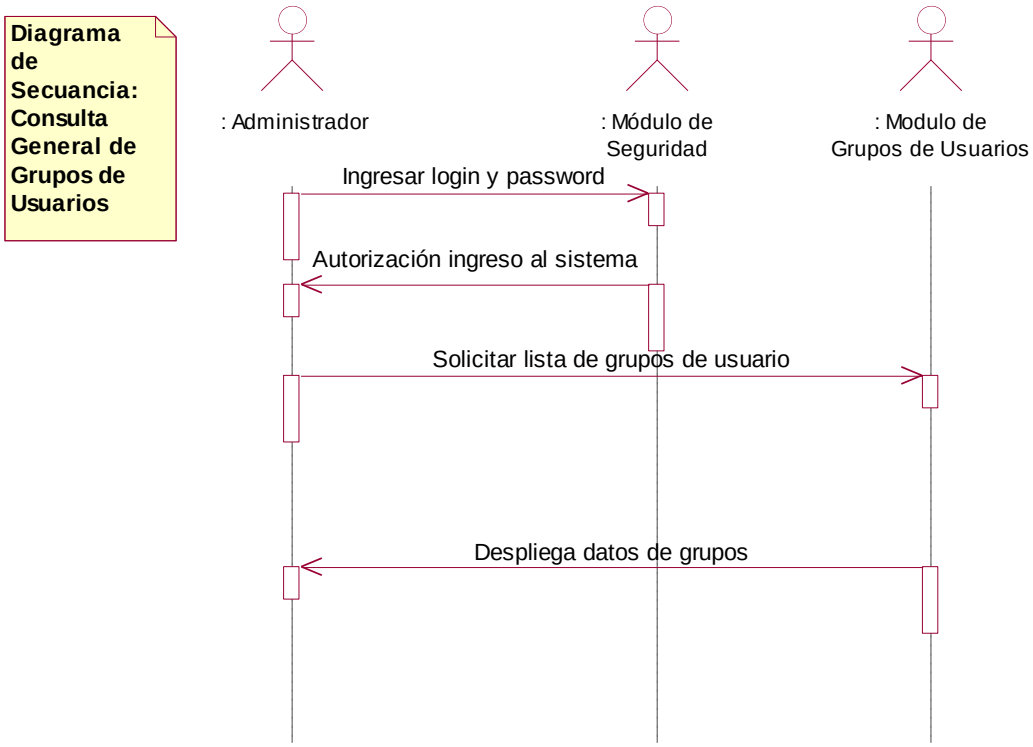
2* Login y/o password inválidos. “Termina caso de uso”.

2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

4* No existen grupos de usuarios en el sistema. ”Termina caso de uso”.

4* No presenta formulario con el listado de grupos de usuarios. “Termina caso de uso”.

DIAGRAMA DE SECUENCIA



CONTRATO DE OPERACIÓN

NOMBRE DEL CONTRATO: Ingresar login y password.

PROPÓSITO: Acceder al sistema para la consulta general de grupos de usuarios.

TIPO: Sistema.

SALIDA: Autorización de ingreso al sistema.

EXCEPCIONES: Login y/o password inválidos.
No existe formulario de ingreso de login y password.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Solicitar listado general de grupos de usuarios.

PROPÓSITO: Desplegar listado de todos los grupos de usuarios del sistema.

TIPO: Sistema.

SALIDA: Listado general de grupos de usuarios.

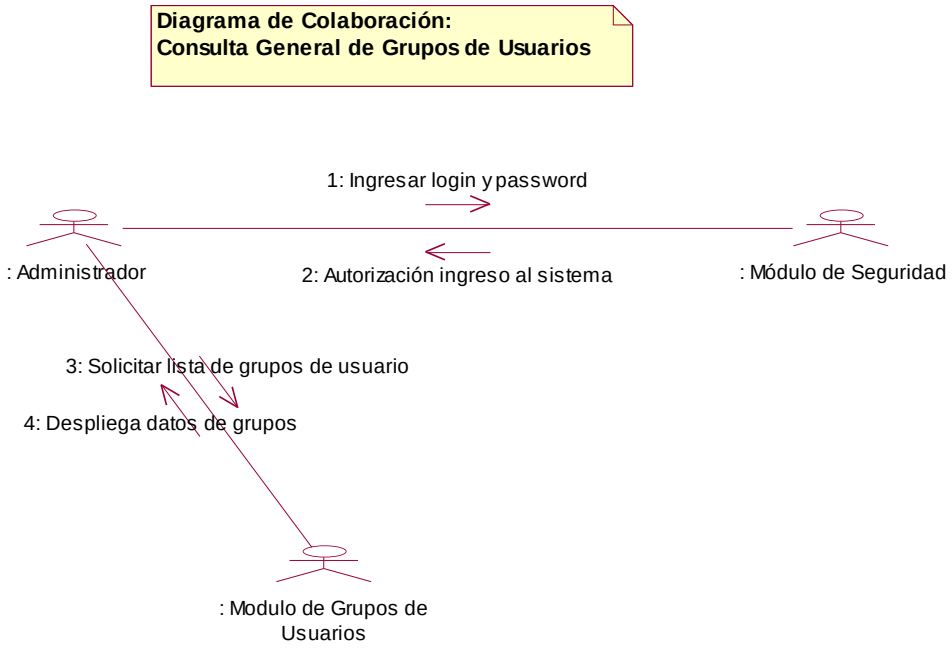
EXCEPCIONES: No existen grupos de usuarios en el sistema.

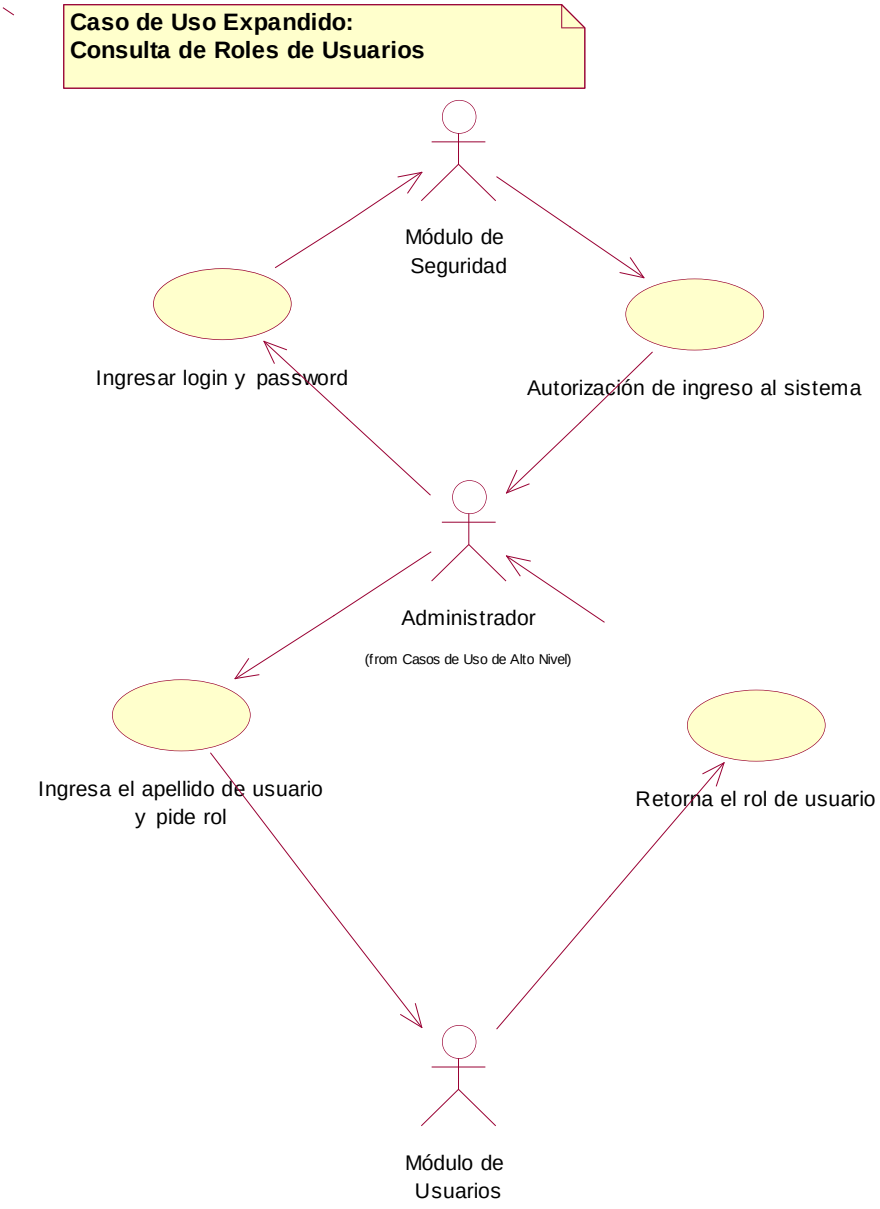
No presenta formulario con el listado de grupos de usuarios.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

DIAGRAMA DE COLABORACIÓN





NOMBRE DEL CASO DE USO: CONSULTAR ROLES DE USUARIOS.
PROPÓSITO: Informarse de los grupos de usuarios en el sistema en forma general
ACTOR (Iniciador): Administrador.
TIPO: Primario Real.
VISION GENERAL: El administrador ingresa su login y password, el sistema autoriza el ingreso al sistema, el administrador ingresa el apellido de un usuario

del sistema y pide el rol que desempeña, el sistema despliega la información solicitada con el rol de dicho usuario termina caso de uso.

CURSO TIPICO DE EVENTOS

ACTOR	SISTEMA
1. Ingresar login y password.	2. Autorizar ingreso al sistema.
3. Ingresar apellido de usuario del sistema y pedir rol.	4. Desplegar información de rol de usuario.

CURSOS ALTERNATIVOS:

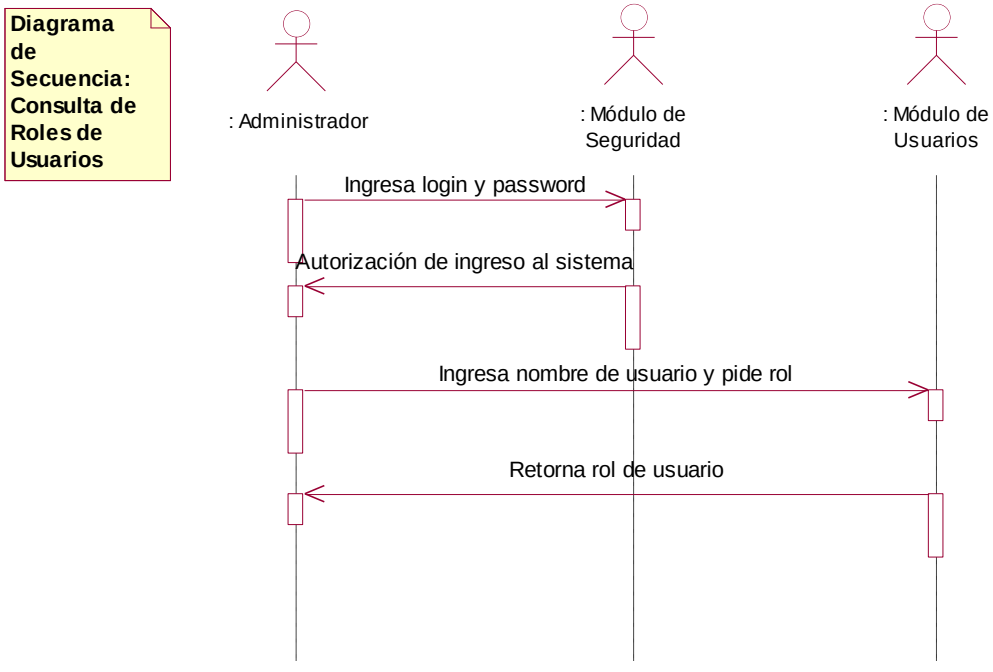
2* Login y/o password inválidos. “Termina caso de uso”.

2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

4* No existe usuario con apellido ingresado o no existe grupos de usuario. ”Termina caso de uso”.

4* No presenta formulario con la información de rol de usuarios. “Termina caso de uso”.

DIAGRAMA DE SECUENCIA



CONTRATO DE OPERACIÓN

NOMBRE DEL CONTRATO: Ingresar login y password.

PROPÓSITO: Acceder al sistema para la consulta de roles de usuarios.

TIPO: Sistema.

SALIDA: Autorización de ingreso al sistema.

EXCEPCIONES: Login y/o password inválidos.
No existe formulario de ingreso de login y password.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Ingresar apellido de usuario y pedir rol.

PROPÓSITO: Desplegar información de rol de usuario.

TIPO: Sistema.

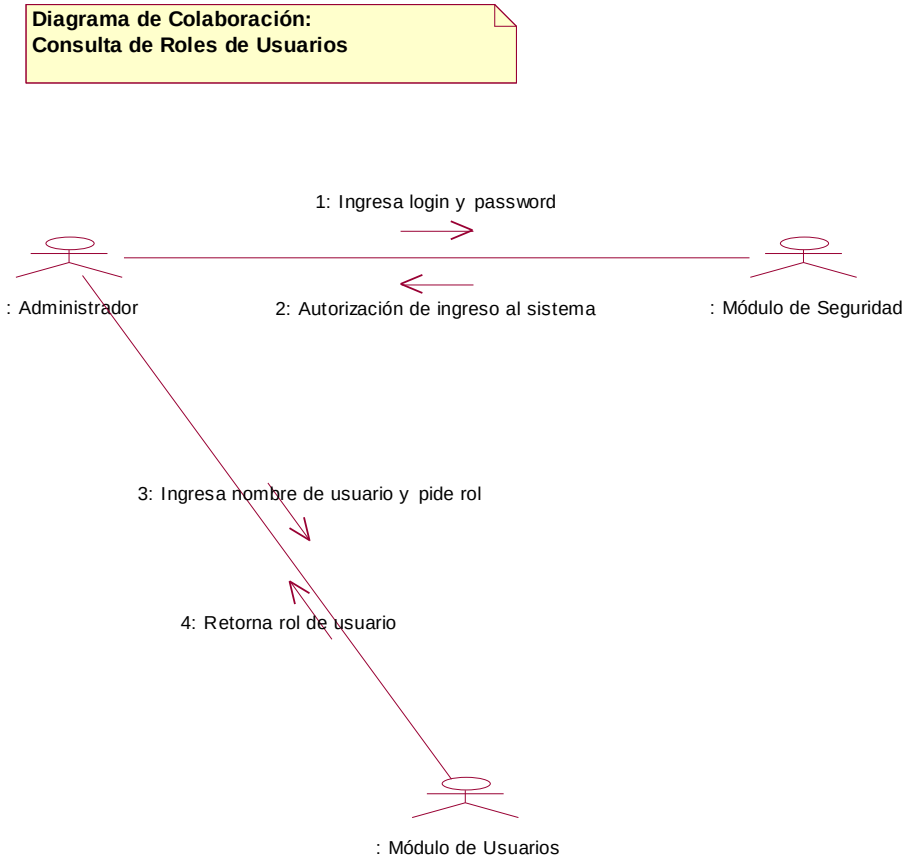
SALIDA: Información de roles de usuarios.

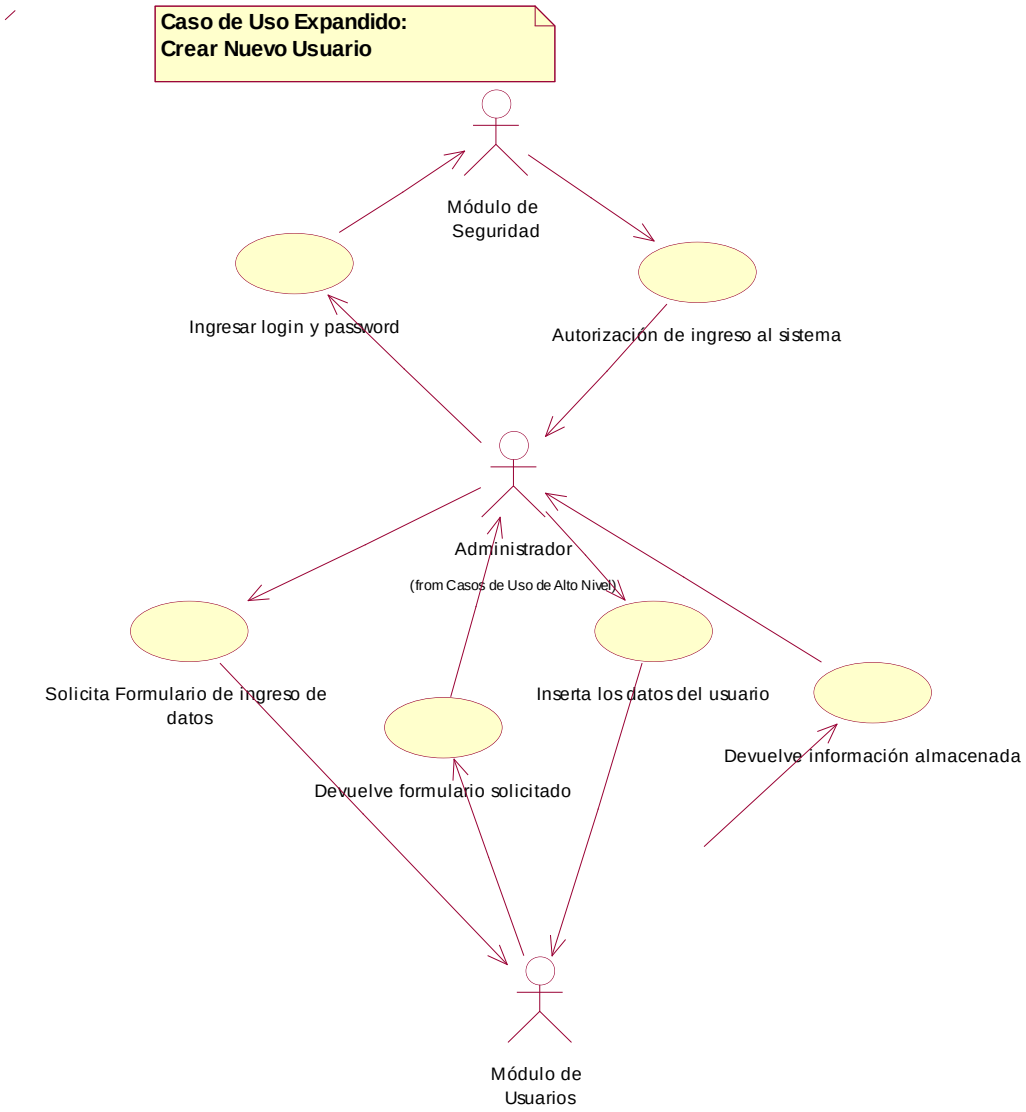
EXCEPCIONES: No existe usuario con apellido ingresado o no existe grupos de usuario.
No presenta formulario con la información de rol de usuarios.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

DIAGRAMA DE COLABORACIÓN





NOMBRE DEL CASO DE USO: CREAR USUARIOS DEL SISTEMA.

PROPÓSITO: Establecer usuarios en el sistema.

ACTOR (Iniciador): Administrador.

TIPO: Primario Real.

VISION GENERAL: El administrador ingresa su login y password, el sistema autoriza el ingreso al sistema, el administrador solicita el formulario para el ingreso de datos del usuario, el sistema despliega el formulario solicitado, el administrador ingresa los datos en el formulario requeridos para la creación del

nuevo usuario, el sistema devuelve información almacenada y termina el caso de uso.

CURSO TIPICO DE EVENTOS

ACTOR	SISTEMA
1. Ingresar login y password.	2. Autorizar ingreso al sistema.
3. Solicitar formulario.	4. Presentar formulario de ingreso de datos de usuario.
5. Ingresar datos del usuario.	6. Presentar mensaje de información del usuario almacenada.

CURSOS ALTERNATIVOS:

2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

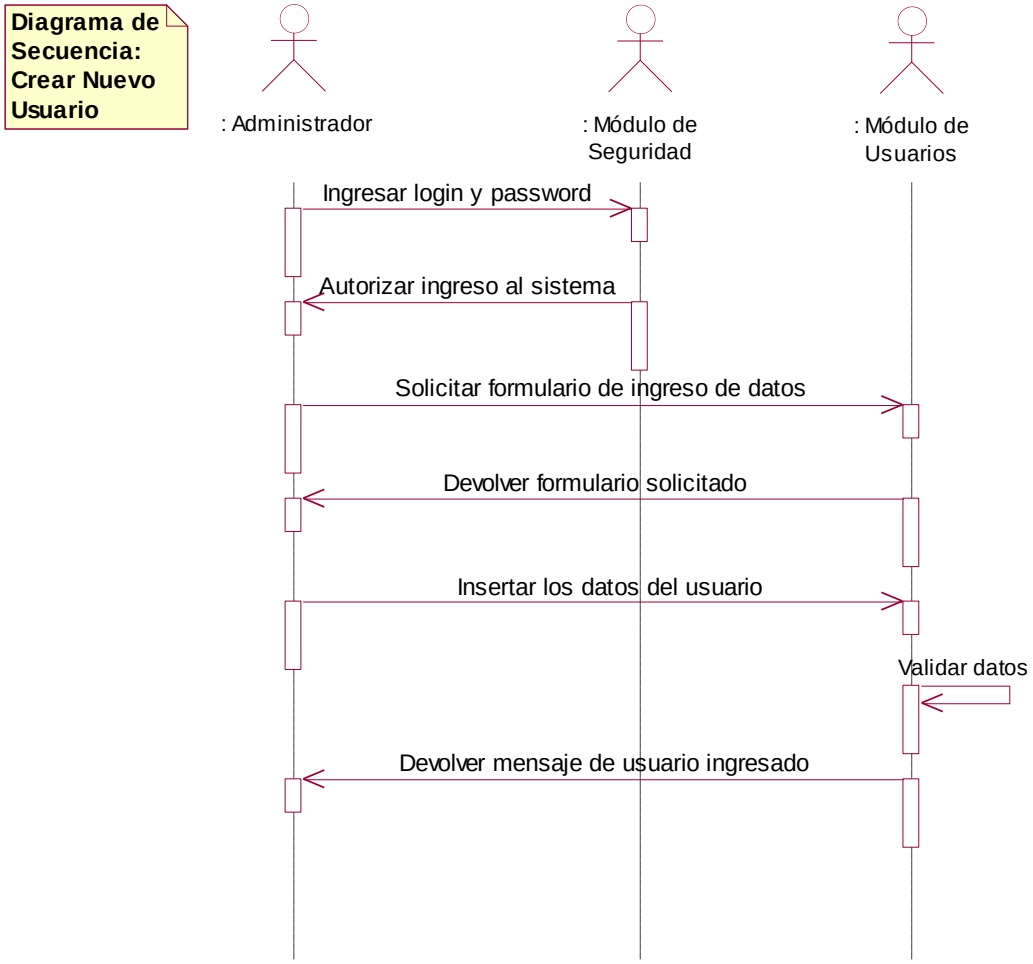
2* Login y/o password no válidos. ”Termina caso de uso”.

4* No existe formulario de ingreso de datos del usuario. ”Termina caso de uso”.

6* No presenta mensaje de información almacenada exitosamente. “Termina caso de uso”.

6* Presenta mensaje de ingresar datos requeridos. “Termina caso de uso”.

DIAGRAMA DE SECUENCIA



CONTRATO DE OPERACION

NOMBRE DEL CONTRATO: Ingresar login y password.

PROPÓSITO: Acceder al sistema para creación de nuevos usuarios (transacción).

TIPO: Sistema.

SALIDA: Autorización de ingreso al sistema.

EXCEPCIONES: Login y/o password no válidos

No existe formulario de ingreso de login y password.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Solicitar formulario de ingreso de datos.

PROPÓSITO: Acceder al formulario de ingreso de datos para creación de usuarios.

TIPO: Sistema.

SALIDA: Formulario ingreso de datos de usuarios.

EXCEPCIONES: No existe formulario de ingreso de datos del usuario.

PRE-CONDICIONES: Relación usuario – interfaz (formulario).

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Insertar datos de usuario.

PROPÓSITO: Ingresar al sistema datos para creación de nuevos usuarios (transacción).

TIPO: Sistema.

SALIDA: Datos almacenados exitosamente.

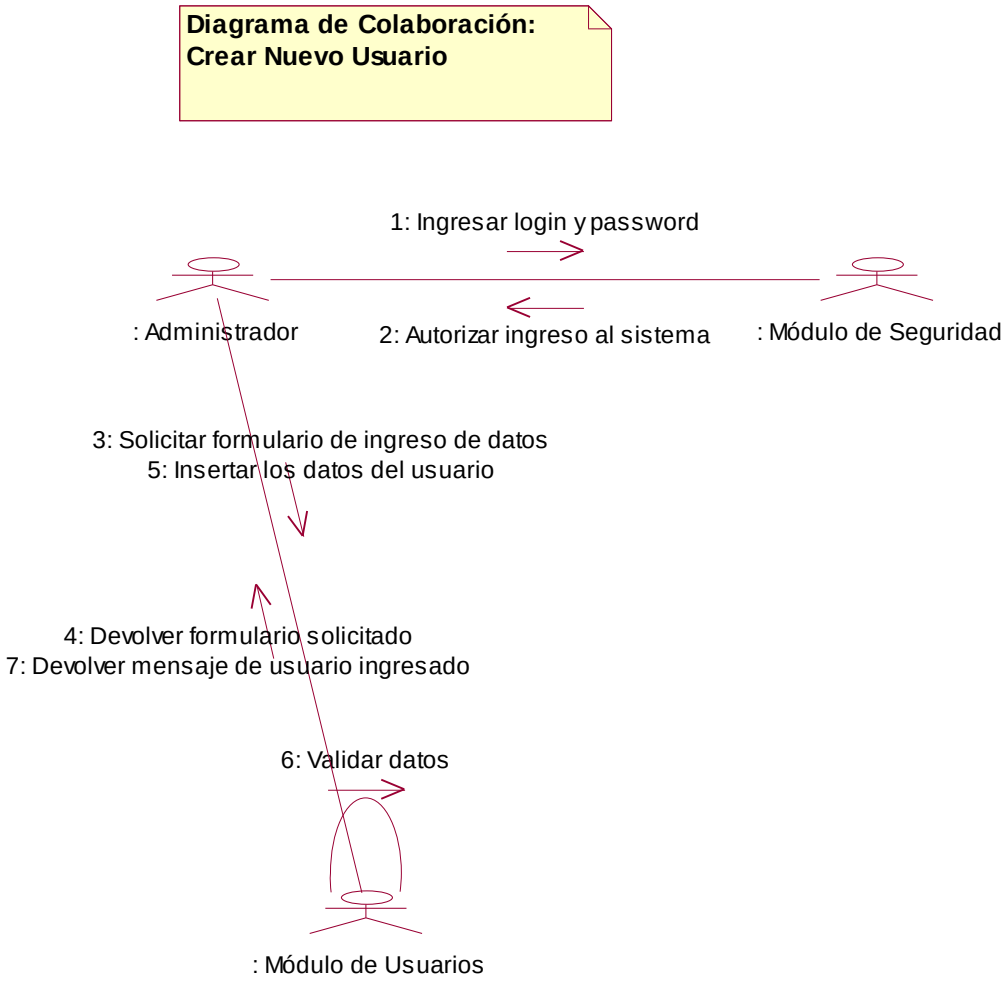
EXCEPCIONES: No presenta mensaje de información almacenada exitosamente.

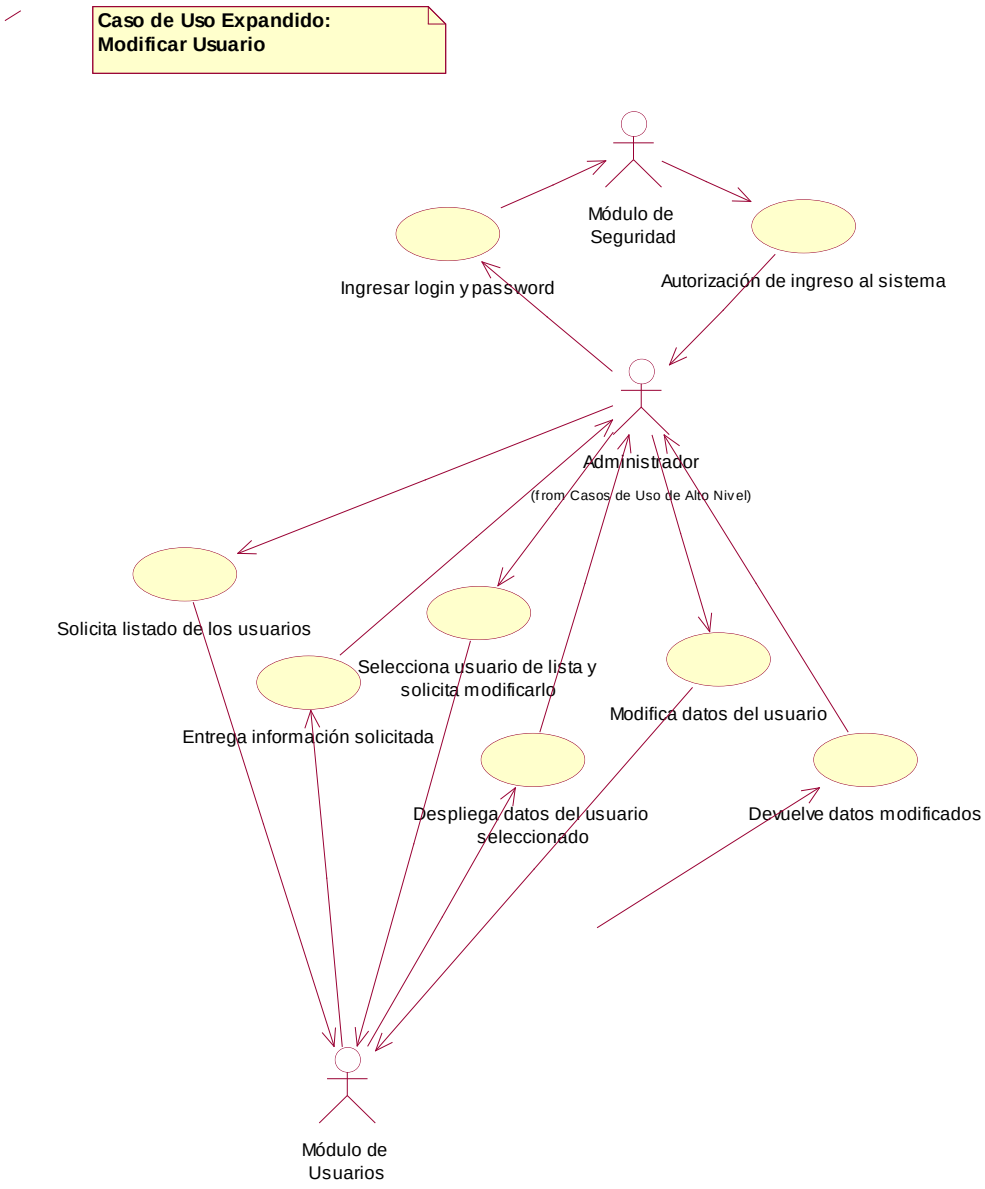
Presenta mensaje de ingresar datos requeridos.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

DIAGRAMA DE COLABORACION





NOMBRE DEL CASO DE USO: MODIFICAR USUARIOS DEL SISTEMA.

PROPÓSITO: Cambiar datos de usuarios en el sistema.

ACTOR (Iniciador): Administrador.

TIPO: Primario Real.

VISION GENERAL: El administrador ingresa su login y password, el sistema autoriza el ingreso al sistema, el administrador solicita el formulario de usuarios existentes en el sistema, el sistema devuelve información solicitada, el

administrador escoge al usuario y solicita desplegar datos, el sistema presenta datos de usuario a modificar, el administrador modifica datos de usuario el sistema devuelve datos de usuario modificado termina el caso de uso.

CURSO TIPICO DE EVENTOS

ACTOR	SISTEMA
1. Ingresar login y password.	2. Autorizar ingreso al sistema.
3. Solicitar formulario de usuarios.	4. Presentar formulario de usuarios.
5.Escoger usuario y Solicitar datos de usuario a modificar	6.Presentar datos de usuario a modificar
7.Modificar datos de usuario	8. Devolver datos de usuario modificado

CURSOS ALTERNATIVOS:

2* Login y/o password no válidos. “Termina caso de uso”.

2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

4* No existe usuarios en el sistema. ”Termina caso de uso”.

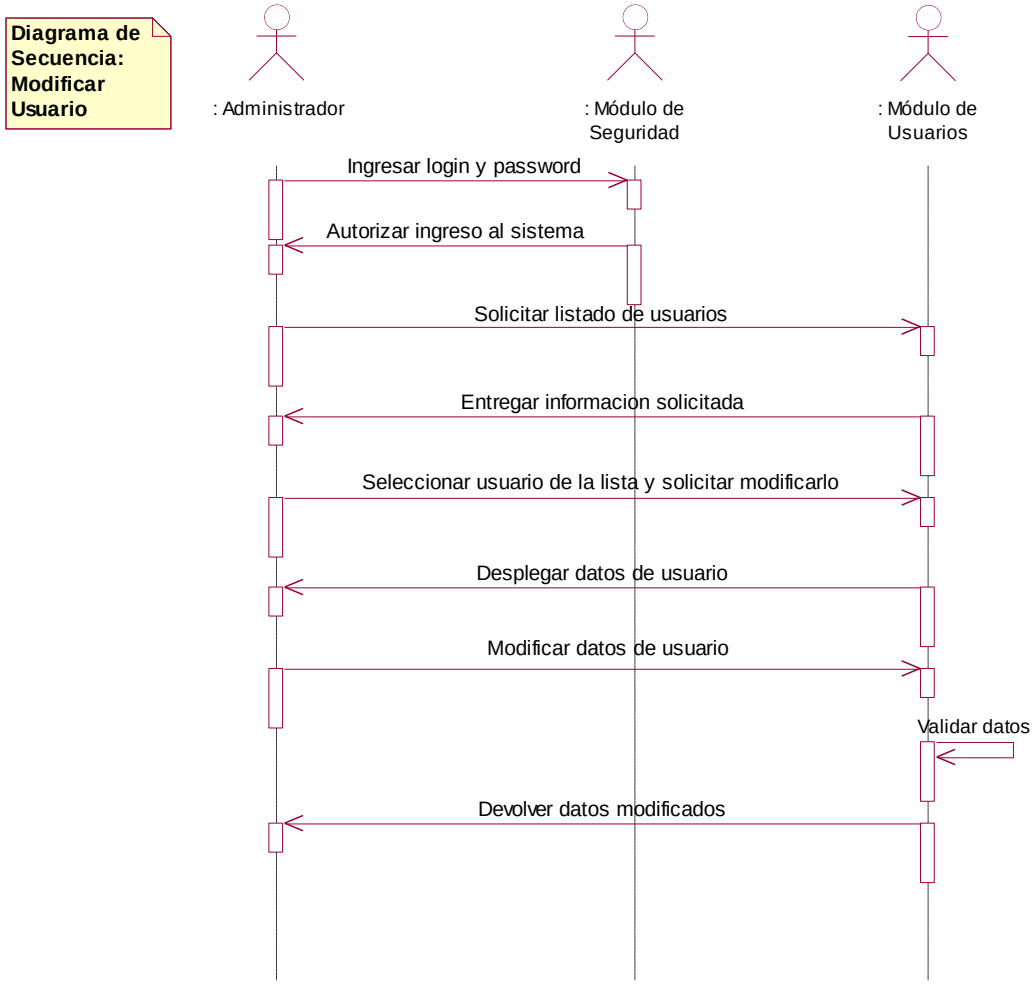
4* No presenta formulario con el listado de usuarios. “Termina caso de uso”.

6* No presenta datos de usuario a modificar ”Termina caso de uso”.

8* No presenta mensaje de usuario modificado. “Termina caso de uso”.

8* No se puede modificar registro. “Termina caso de uso”.

DIAGRAMA DE SECUENCIA



CONTRATO DE OPERACIÓN

NOMBRE DEL CONTRATO: Ingresar login y password.

PROPÓSITO: Acceder al sistema para modificar usuarios existentes en el sistema (transacción).

TIPO: Sistema.

SALIDA: Autorización de ingreso al sistema.

EXCEPCIONES: Login y/o password no válidos.
No existe formulario de ingreso de login y password.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Solicitar listado de usuarios existentes.

PROPÓSITO: Obtener la lista de los usuarios existentes (transacción).

TIPO: Sistema.

SALIDA: Lista de usuarios existentes.

EXCEPCIONES: No existen usuarios en el sistema.
No presenta formulario con el listado de usuarios.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Escoger usuario de la lista y solicita modificarlo.

PROPÓSITO: Elegir usuario para la modificación y solicitar al sistema la modificación (transacción).

TIPO: Sistema.

SALIDA: Formulario de actualización de datos de usuario.

EXCEPCIONES: No presenta datos de usuario a modificar.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Modificar datos de usuario.

PROPÓSITO: Cambiar datos de usuario.

TIPO: Sistema.

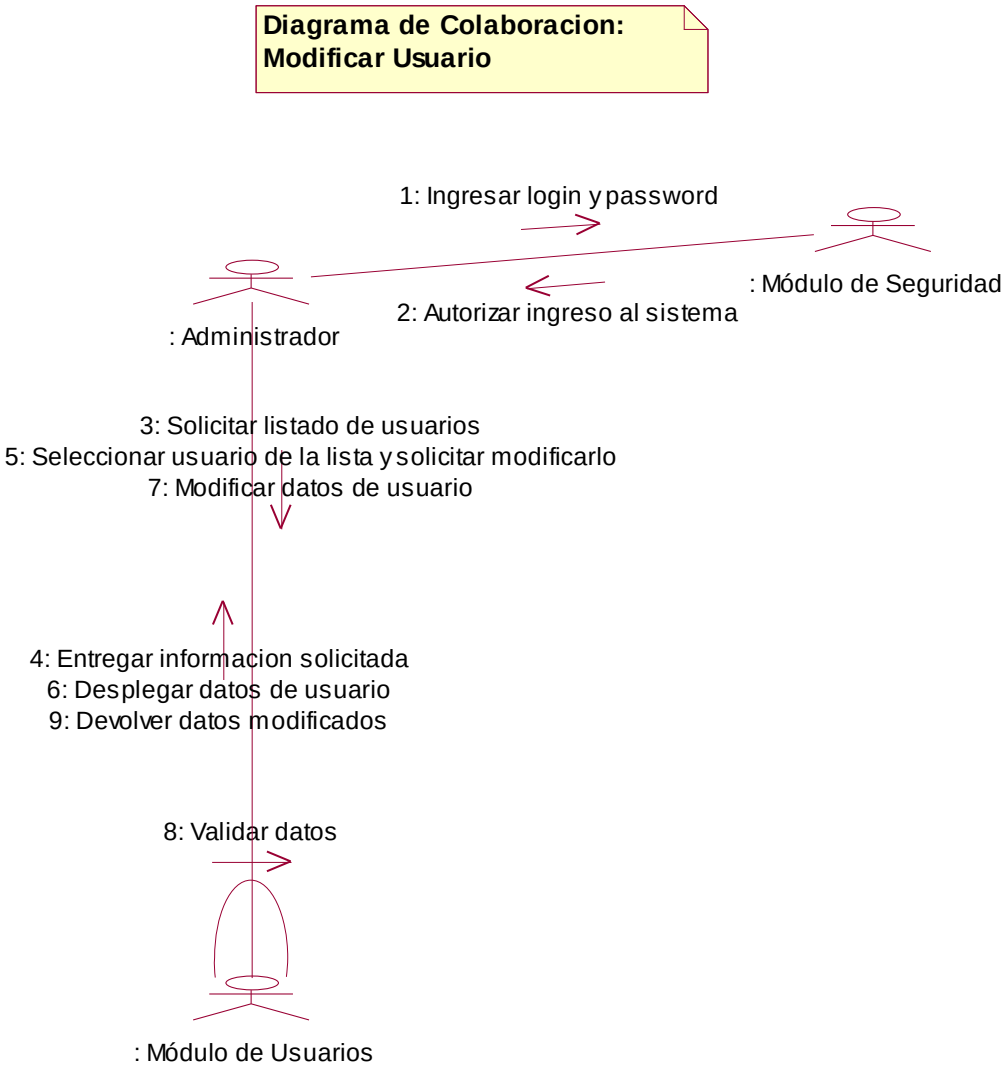
SALIDA: Mensaje de datos actualizados correctamente.

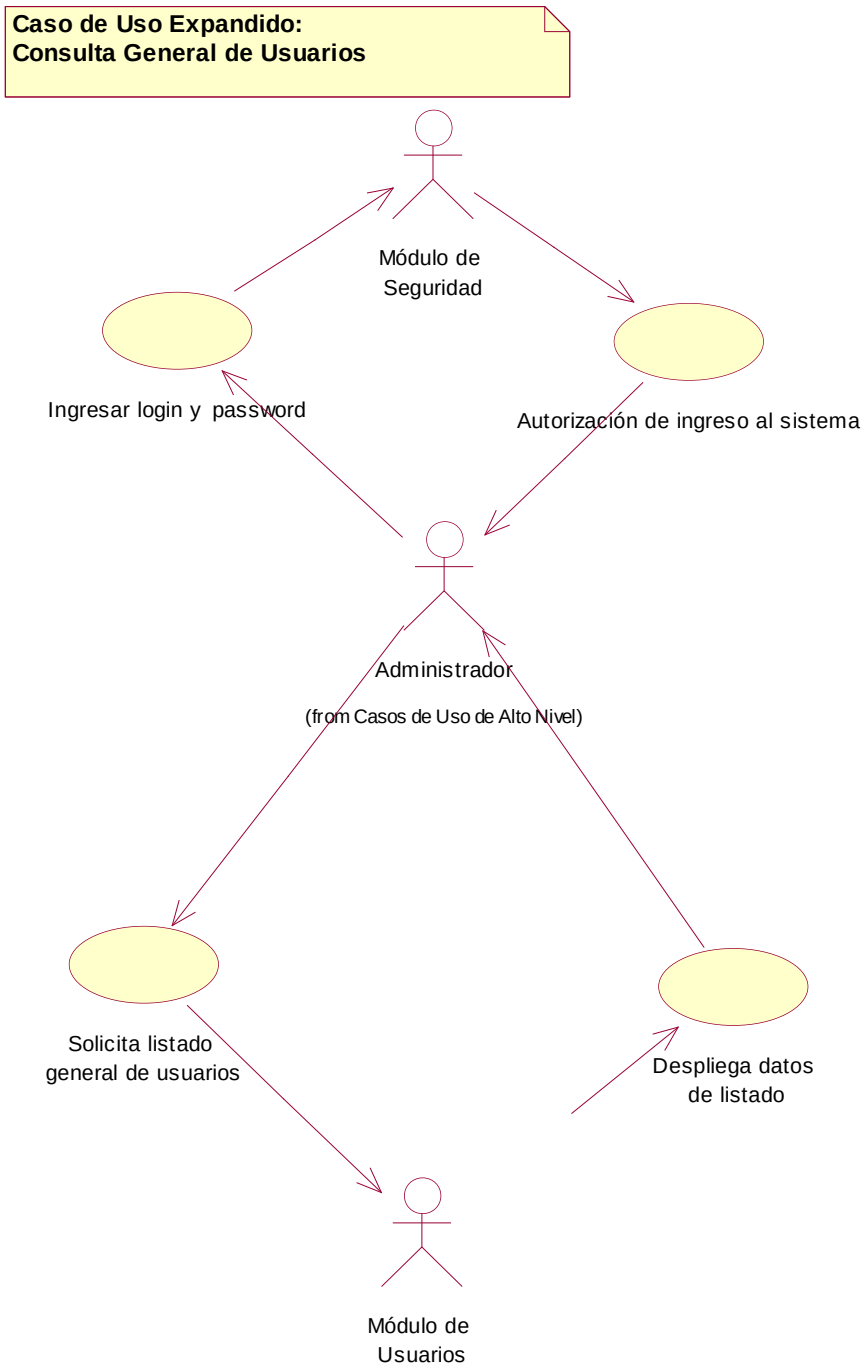
EXCEPCIONES: No presenta mensaje de usuario modificado.
No se puede modificar registro.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

DIAGRAMA DE COLABORACIÓN





NOMBRE DEL CASO DE USO: CONSULTAR USUARIOS DEL SISTEMA EN FORMA GENERAL.

PROPÓSITO: Informarse datos de usuarios en el sistema en forma general

ACTOR (Iniciador): Administrador.

TIPO: Primario Real.

VISION GENERAL: El administrador ingresa su login y password, el sistema autoriza el ingreso al sistema, el administrador solicita listado general de usuarios existentes en el sistema, el sistema devuelve y despliega listado información solicitada termina caso de uso.

CURSO TIPICO DE EVENTOS

ACTOR	SISTEMA
1. Ingresar login y password.	2. Autorizar ingreso al sistema.
3. Solicitar listado general de usuarios.	4. Presentar listado de usuarios.

CURSOS ALTERNATIVOS:

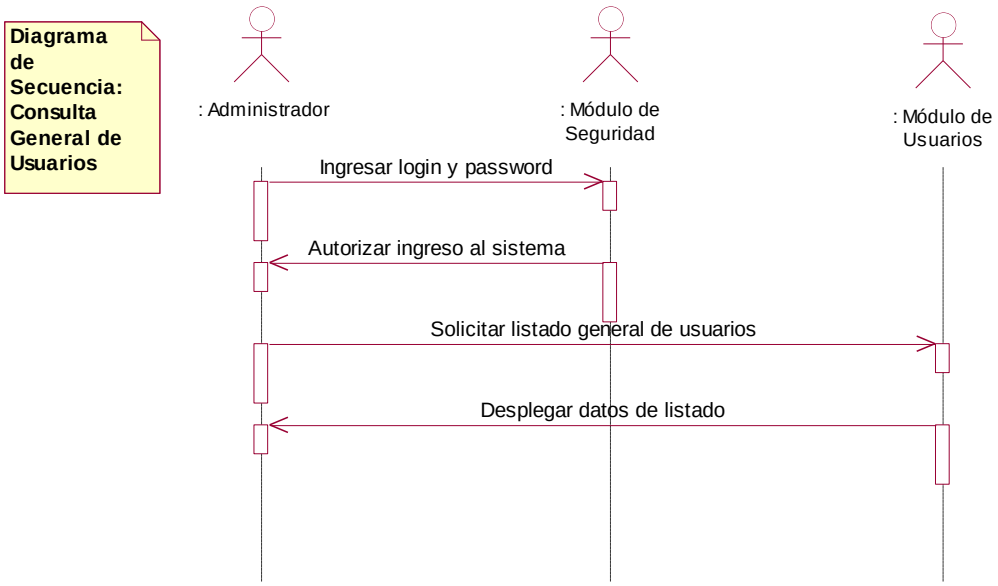
2* Login y/o password inválidos. “Termina caso de uso”.

2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

4* No existen usuarios en el sistema. ”Termina caso de uso”.

4* No presenta formulario con el listado de usuarios. “Termina caso de uso”.

DIAGRAMA DE SECUENCIA



CONTRATO DE OPERACIÓN

NOMBRE DEL CONTRATO: Ingresar login y password.

PROPÓSITO: Acceder al sistema para la consulta general de usuarios.

TIPO: Sistema.

SALIDA: Autorización de ingreso al sistema.

EXCEPCIONES: Login y/o password inválidos.
No existe formulario de ingreso de login y password.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Solicitar listado general de usuarios.

PROPÓSITO: Desplegar listado de todos los usuarios del sistema.

TIPO: Sistema.

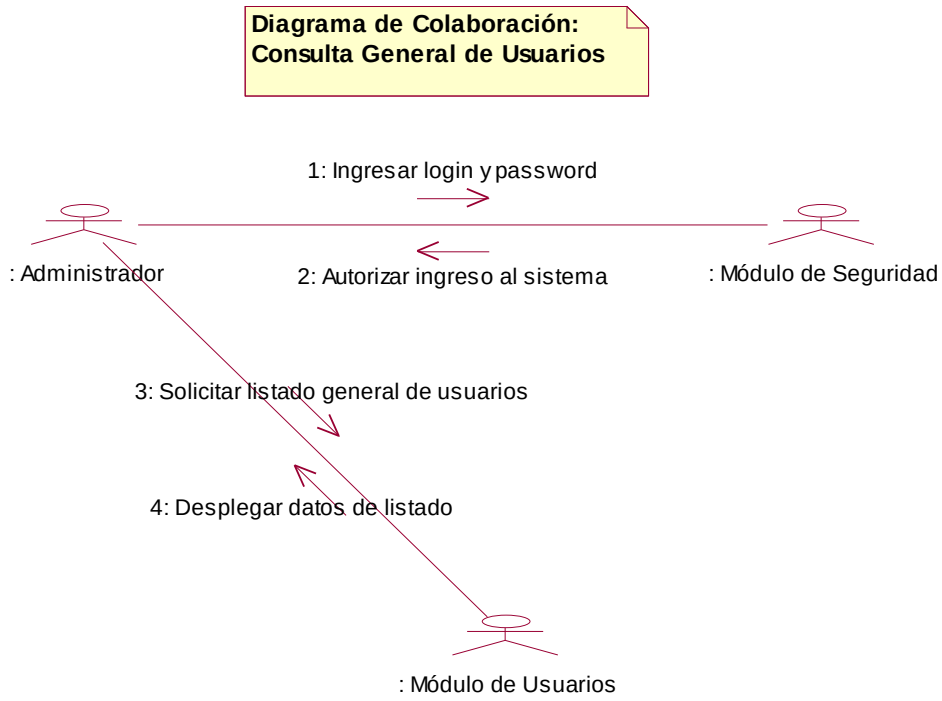
SALIDA: Listado general de usuarios.

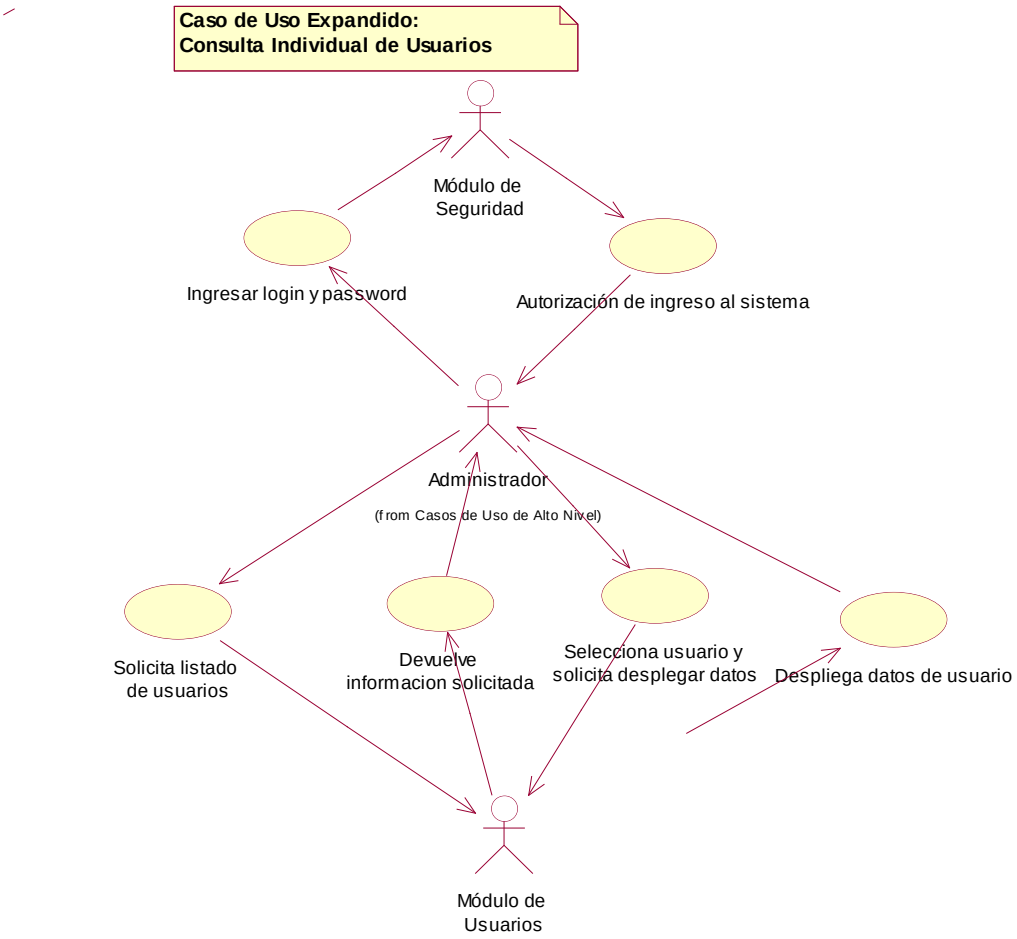
EXCEPCIONES: No existen usuarios en el sistema.
No presenta formulario con el listado de usuarios.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

DIAGRAMA DE COLABORACION





NOMBRE DEL CASO DE USO: CONSULTAR USUARIOS DEL SISTEMA EN FORMA INDIVIDUAL.

PROPÓSITO: Conocer datos de usuarios del sistema en forma individual.

ACTOR (Iniciador): Administrador.

TIPO: Primario Real.

VISION GENERAL: El administrador ingresa su login y password, el sistema autoriza el ingreso al sistema, el administrador solicita listado de usuarios existentes en el sistema, el sistema devuelve y presenta información solicitada, el administrador escoge al usuario y solicita desplegar datos, el sistema presenta datos de usuario termina el caso de uso.

CURSO TIPICO DE EVENTOS

ACTOR	SISTEMA
1. Ingresar login y password.	2. Autorizar ingreso al sistema.
3. Solicitar listado de usuarios.	4. Presentar listado de usuarios.
5. Escoger usuario y Solicitar desplegar datos de usuario.	6. Presentar datos de usuario.

CURSOS ALTERNATIVOS:

2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

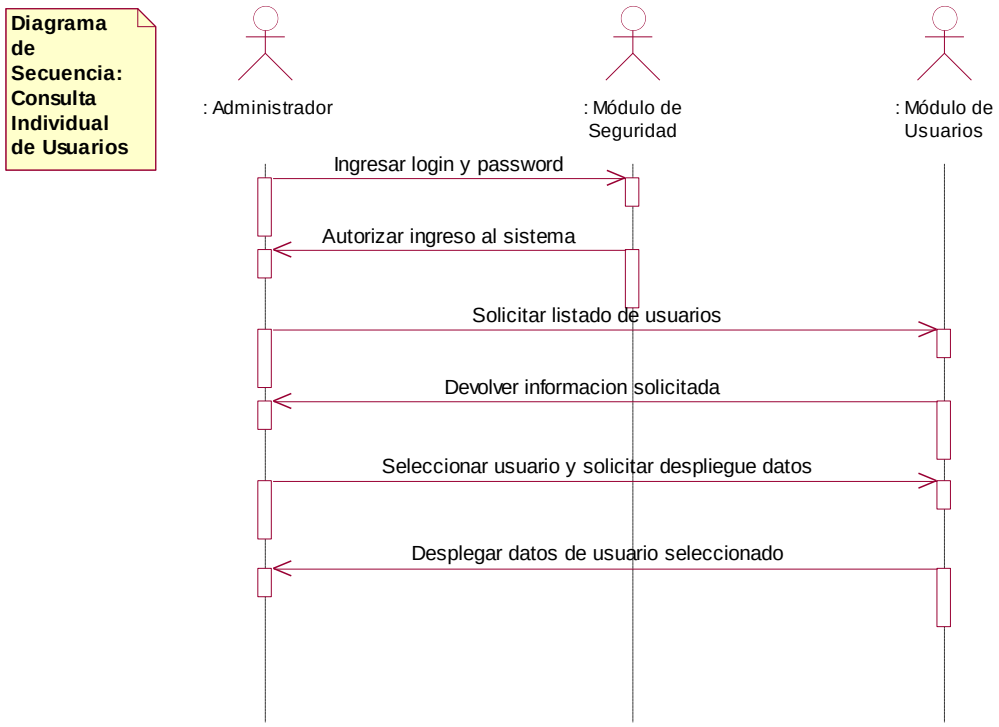
2* Login y/o password incorrecto. “Termina caso de uso”.

4* No existen usuarios. ”Termina caso de uso”.

4* No presenta formulario con el listado de usuarios. “Termina caso de uso”.

6* No presenta datos de usuario. ”Termina caso de uso”.

DIAGRAMA DE SECUENCIA



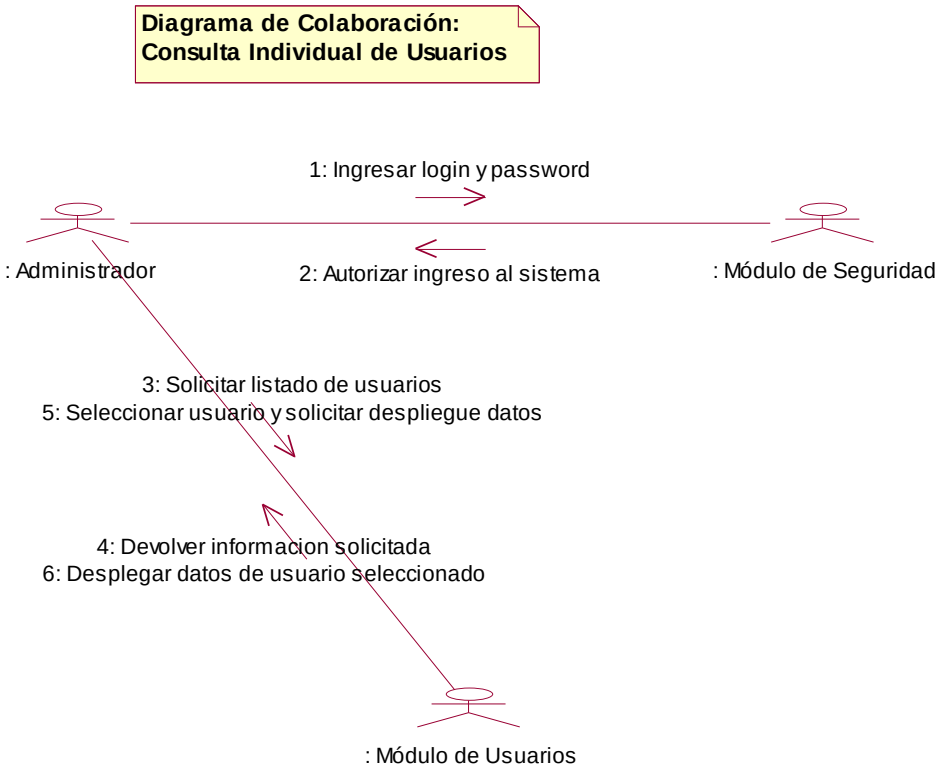
CONTRATO DE OPERACIÓN

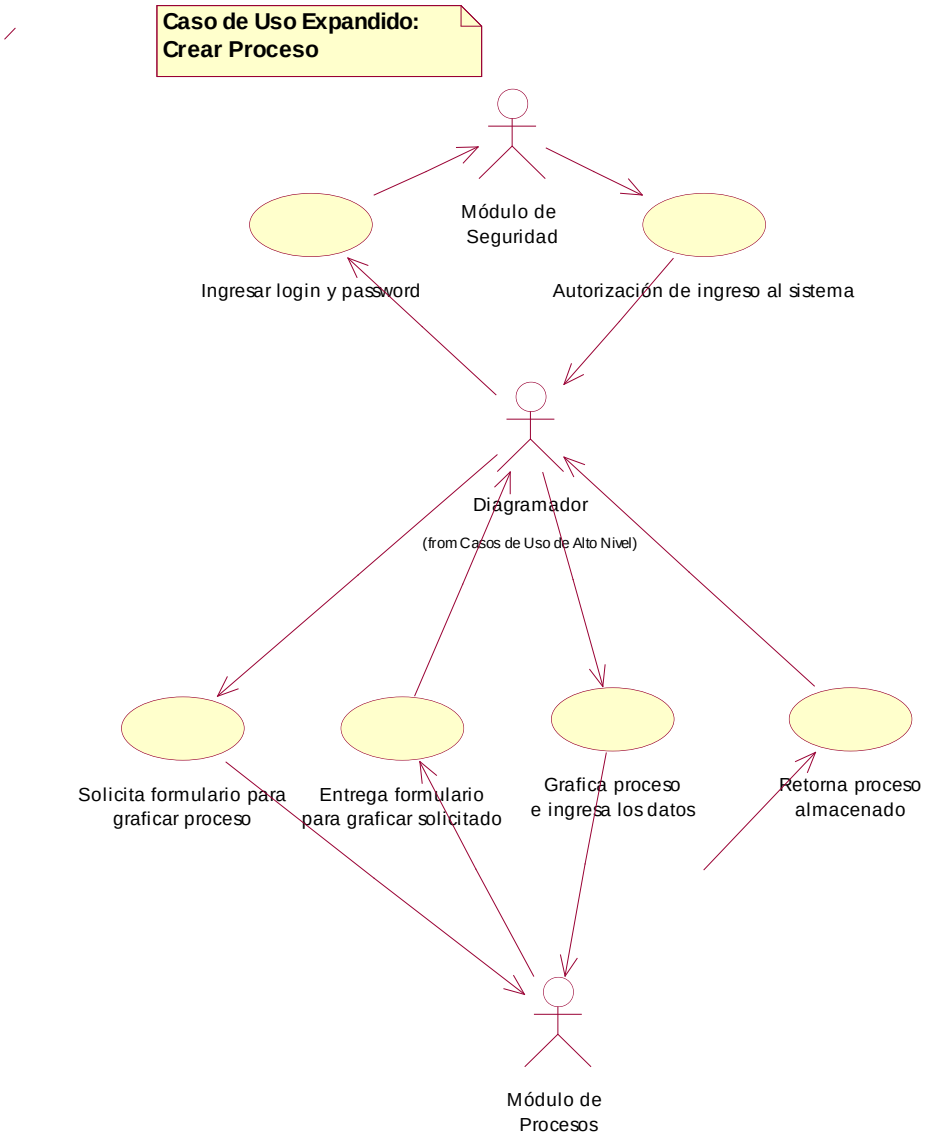
NOMBRE DEL CONTRATO: Ingresar login y password.
PROPÓSITO: Acceder al sistema para la consulta individual de usuarios.
TIPO: Sistema.
SALIDA: Autorización de ingreso al sistema.
EXCEPCIONES: Login y/o password incorrectos.
No existe formulario de ingreso de login y password.
PRE-CONDICIONES: Relación usuario - interfaz
POST-CONDICIONES:

NOMBRE DEL CONTRATO: Solicitar listado de usuarios..
PROPÓSITO: Observar los usuarios que posee el sistema.
TIPO: Sistema.
SALIDA: Listado de usuarios
EXCEPCIONES: No existen usuarios.
No presenta formulario con el listado de usuarios.
PRE-CONDICIONES: Relación usuario - interfaz
POST-CONDICIONES:

NOMBRE DEL CONTRATO: Seleccionar usuario y solicitar despliegue datos
PROPÓSITO: Elegir un usuario para que se despliegue la información del mismo.
TIPO: Sistema.
SALIDA: Datos del usuario solicitado.
EXCEPCIONES: No presenta datos de usuario.
PRE-CONDICIONES: Relación usuario - interfaz
POST-CONDICIONES:

DIAGRAMA DE COLABORACION





NOMBRE DEL CASO DE USO: CREAR PROCESO

PROPÓSITO: Implantar un nuevo proceso por parte del diagramador.

ACTOR (Iniciador): Diagramador.

TIPO: Primario Real.

VISION GENERAL: El diagramador ingresa su login y password, el sistema autoriza el ingreso al sistema, el diagramador solicita el formulario para graficar procesos, el sistema entrega formulario para graficar solicitado, el diagramador grafica el proceso e ingresa datos, el sistema retorna proceso almacenado, termina el caso de uso.

CURSO TIPICO DE EVENTOS

ACTOR	SISTEMA
1. Ingresar login y password.	2. Autorizar ingreso al sistema.
3. Solicitar formulario para graficar procesos.	4. Entregar formulario para graficar.
5. Graficar proceso e ingresar datos.	6. Presentar proceso y datos almacenados.

CURSOS ALTERNATIVOS:

2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

2* Login y/o password no válidos. ”Termina caso de uso”.

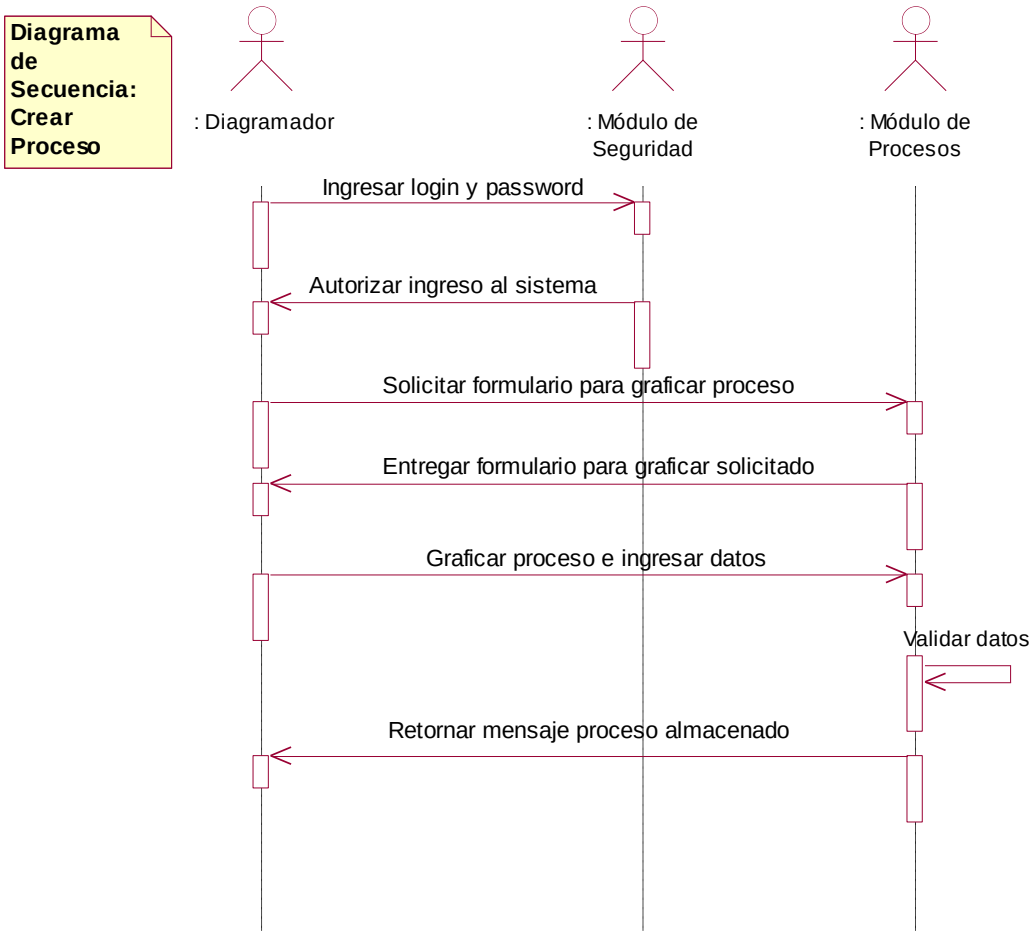
4* No existe formulario de ingreso de datos del proceso. ”Termina caso de uso”.

6* No presenta mensaje de información almacenada exitosamente. “Termina caso de uso”.

6* Presenta mensaje de ingresar datos requeridos. “Termina caso de uso”.

6* Presenta mensaje de incoherencias en el gráfico. “Termina caso de uso”.

DIAGRAMA DE SECUENCIA



CONTRATO DE OPERACIÓN

NOMBRE DEL CONTRATO: Ingresar login y password.

PROPÓSITO: Acceder al sistema para creación de nuevos procesos (transacción).

TIPO: Sistema.

SALIDA: Autorización de ingreso al sistema.

EXCEPCIONES: Login y/o password incorrectos.

No existe formulario de ingreso de login y password.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Solicitar formulario para graficar procesos.

PROPÓSITO: Acceder al formulario de graficación e ingreso de datos para la creación de procesos.

TIPO: Sistema.

SALIDA: Formulario para graficar e ingresar de datos de procesos.

EXCEPCIONES: No presenta formulario para graficar procesos.

PRE-CONDICIONES: Relación usuario – interfaz (formulario).

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Graficar proceso e Ingresar datos.

PROPÓSITO: Graficar e Ingresar al sistema datos para creación de nuevos procesos.

TIPO: Sistema.

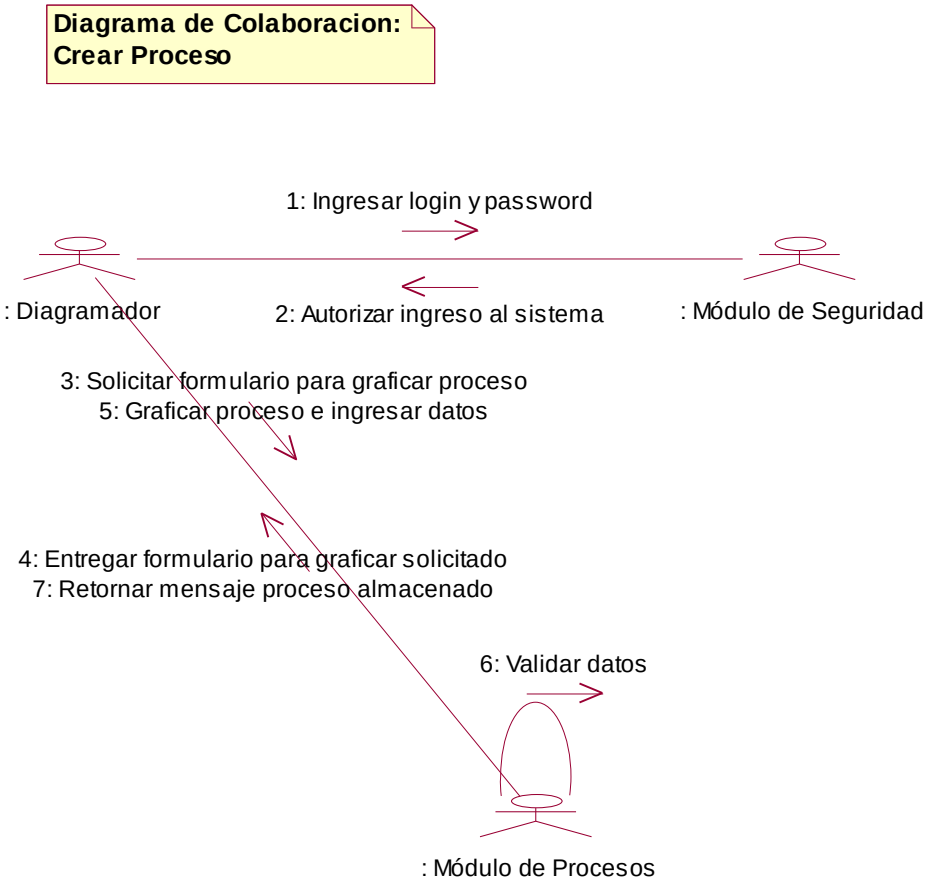
SALIDA: Presenta gráfico almacenado.

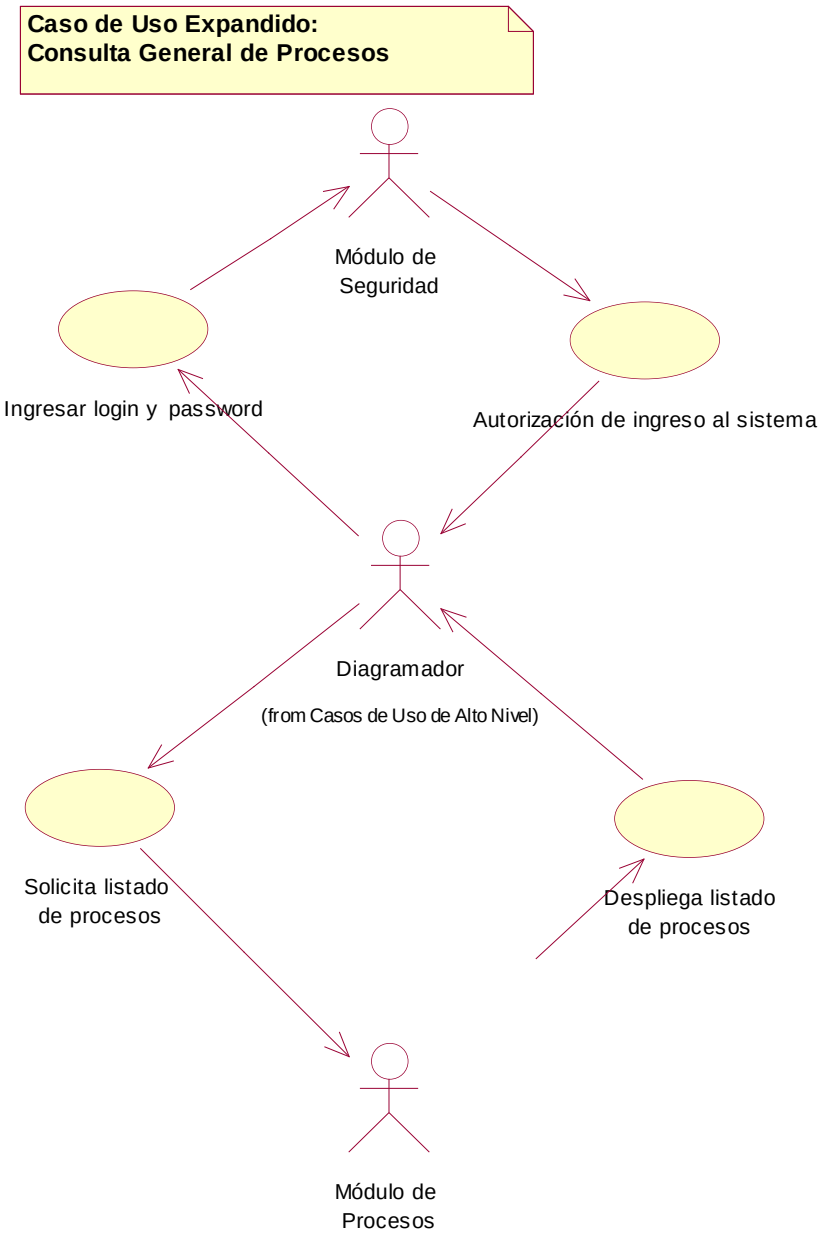
EXCEPCIONES: No presenta mensaje de información almacenada exitosamente.
Presenta mensaje de ingresar datos requeridos.
Presenta mensaje de incoherencias en el gráfico.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

DIAGRAMA DE COLABORACIÓN





NOMBRE DEL CASO DE USO: CONSULTAR PROCESOS EN FORMA GENERAL.

PROPÓSITO: Informarse acerca de procesos en el sistema en forma general

ACTOR (Iniciador): Diagramador.

TIPO: Primario Real.

VISION GENERAL: El diagramador ingresa su login y password, el sistema autoriza el ingreso al sistema, el diagramador solicita listado de procesos existentes en el sistema, el sistema devuelve y despliega listado de procesos y termina caso de uso.

CURSO TIPICO DE EVENTOS

ACTOR	SISTEMA
1. Ingresar login y password.	2. Autorizar ingreso al sistema.
3. Solicitar listado de procesos.	4. Presentar listado de procesos.

CURSOS ALTERNATIVOS:

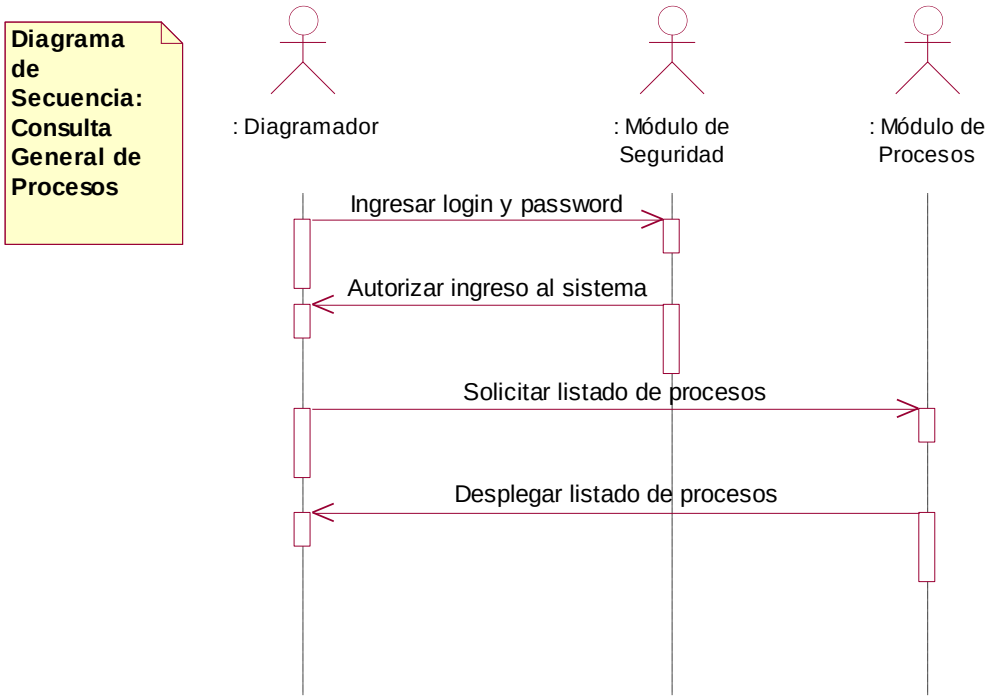
2* Login y/o password incorrectos“Termina caso de uso”.

2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

4* No existen procesos en el sistema. ”Termina caso de uso”.

4* No presenta formulario con el listado de procesos. “Termina caso de uso”.

DIAGRAMA DE SECUENCIA



CONTRATO DE OPERACIÓN

NOMBRE DEL CONTRATO: Ingresar login y password.

PROPÓSITO: Acceder al sistema para la consulta general de procesos.

TIPO: Sistema.

SALIDA: Autorización de ingreso al sistema.

EXCEPCIONES: Login y/o password incorrectos.
No existe formulario de ingreso de login y password.

PRE-CONDICIONES: Relación usuario – interfaz.

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Solicitar listado de procesos.

PROPÓSITO: Desplegar lista de procesos del sistema.

TIPO: Sistema.

SALIDA: Listado general de procesos.

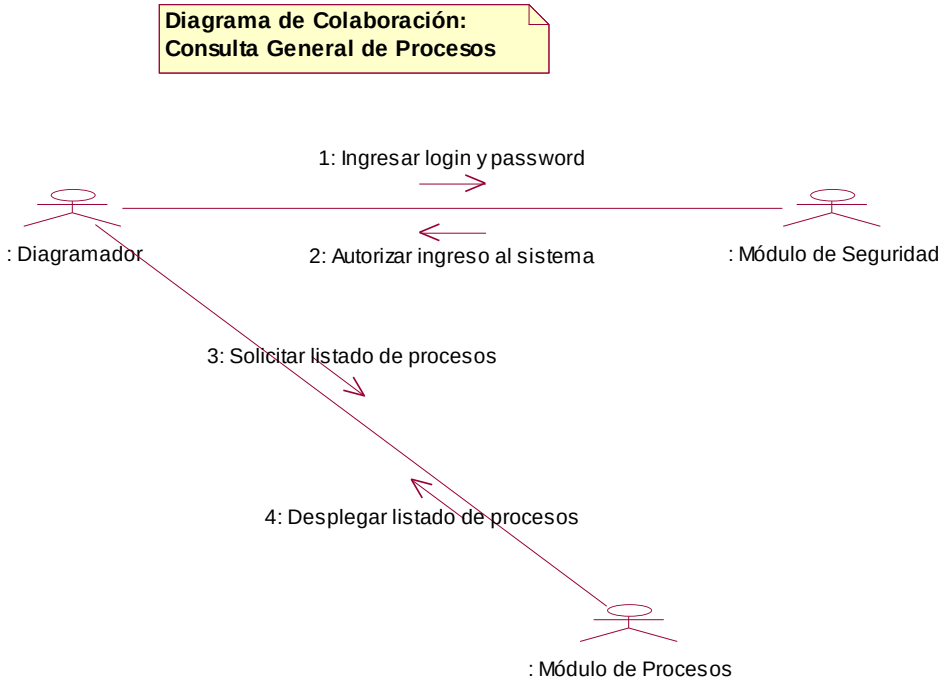
EXCEPCIONES: No existen procesos dentro del sistema.

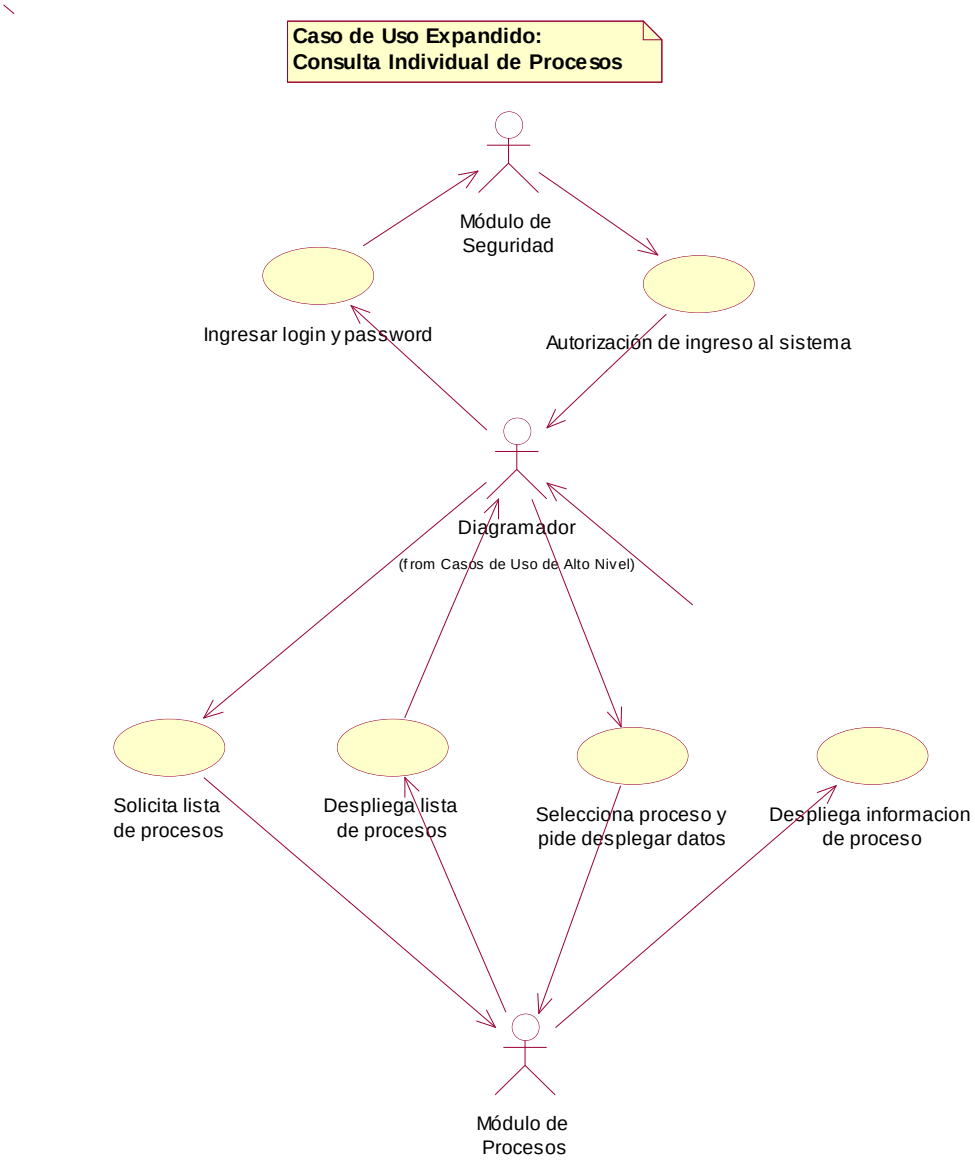
No presenta formulario con el listado de usuarios.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

DIAGRAMA DE COLABORACIÓN





NOMBRE DEL CASO DE USO: CONSULTAR PROCESOS EN FORMA INDIVIDUAL.

PROPÓSITO: Conocer datos de procesos del sistema en forma individual.

ACTOR (Iniciador): Diagramador.

TIPO: Primario Real.

VISION GENERAL: El diagramador ingresa su login y password, el sistema autoriza el ingreso al sistema, el diagramador solicita listado de procesos existentes en el sistema, el sistema devuelve y presenta listado de procesos, el

diagramador escoge el proceso y solicita desplegar datos, el sistema presenta datos del proceso y termina el caso de uso.

CURSO TIPICO DE EVENTOS

ACTOR	SISTEMA
1. Ingresar login y password.	2. Autorizar ingreso al sistema.
3. Solicitar listado de procesos	4. Presentar listado de procesos.
5. Seleccionar proceso y Solicitar desplegar datos.	6. Presentar datos de proceso.

CURSOS ALTERNATIVOS:

2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

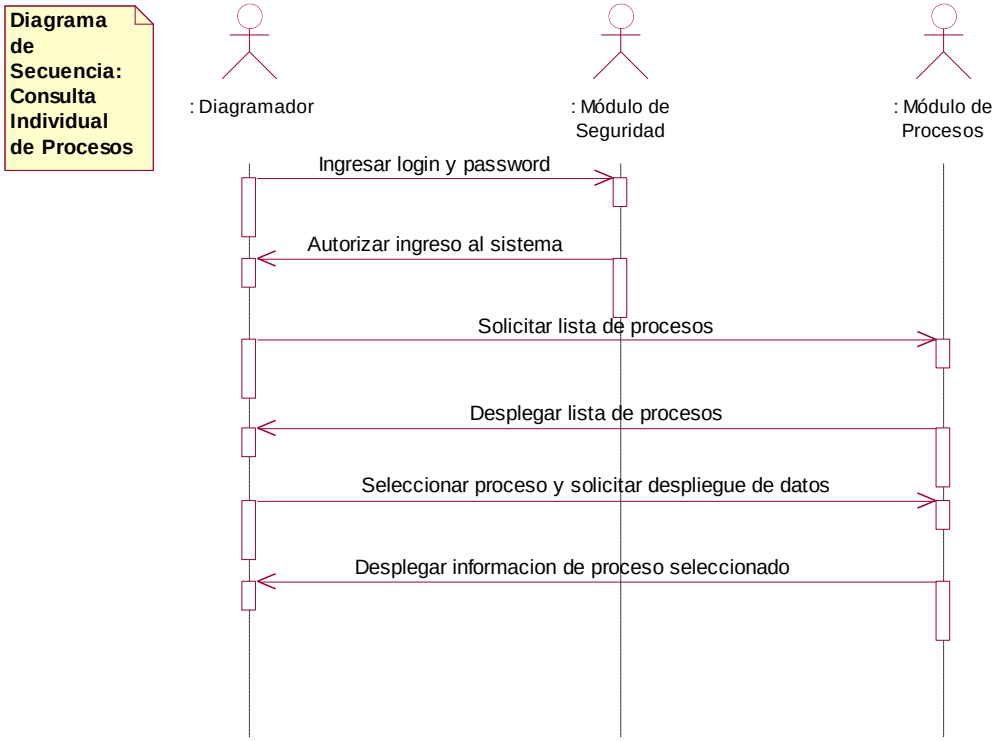
2* Login y/o password incorrecto. “Termina caso de uso”.

4* No existen procesos dentro del sistema. ”Termina caso de uso”.

4* No presenta formulario con el listado de procesos. “Termina caso de uso”.

6* No presenta datos. ”Termina caso de uso”.

DIAGRAMA DE SECUENCIA



CONTRATO DE OPERACIÓN

NOMBRE DEL CONTRATO: Ingresar login y password.

PROPÓSITO: Acceder al sistema para la consulta individual de procesos.

TIPO: Sistema.

SALIDA: Formulario ingreso de login y password

EXCEPCIONES: No existe formulario de ingreso login y password.
Login y/o password incorrecto.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Solicitar lista de procesos.

PROPÓSITO: Desplegar los procesos que posee el sistema.

TIPO: Sistema.

SALIDA: Lista de procesos.

EXCEPCIONES: No existe procesos dentro del sistema.

No presenta formulario con el listado de procesos.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Seleccionar proceso y solicitar desplegar información.

PROPÓSITO: Elegir un proceso para que se despliegue la información del mismo.

TIPO: Sistema.

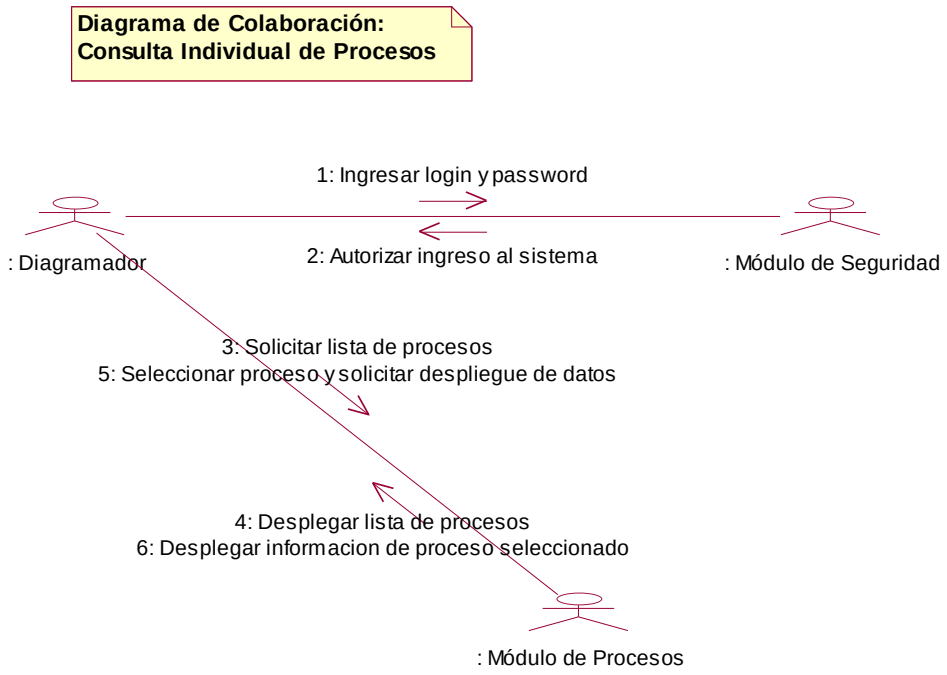
SALIDA: Formulario de información de proceso

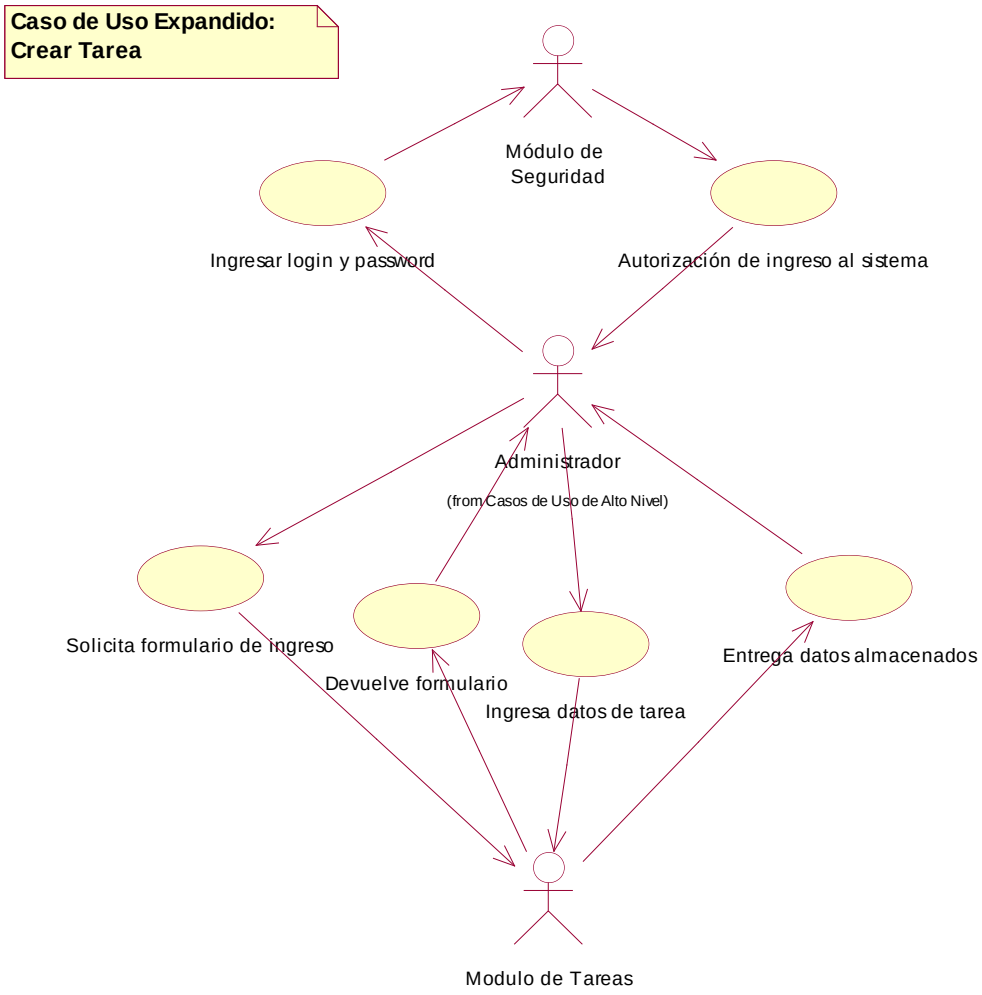
EXCEPCIONES: No presenta datos.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

DIAGRAMA DE COLABORACIÓN





NOMBRE DEL CASO DE USO: CREAR TAREAS.

PROPÓSITO: Implantar tareas en el sistema por parte del diagramador.

ACTOR (Iniciador): Diagramador.

TIPO: Primario Real.

VISION GENERAL: El diagramador ingresa su login y password, el sistema autoriza el ingreso al sistema, el diagramador ingresa datos de tarea, el sistema presenta datos de tarea almacenados, termina el caso de uso.

CURSO TIPICO DE EVENTOS

ACTOR	**SISTEMA**

1. Ingresar login y password.	2. Autorizar ingreso al sistema.
3. Solicitar formulario de ingreso de tareas.	4. Presentar formulario de ingreso.
5. Ingresar datos de tarea.	6. Presentar datos de tarea almacenada.

CURSOS ALTERNATIVOS:

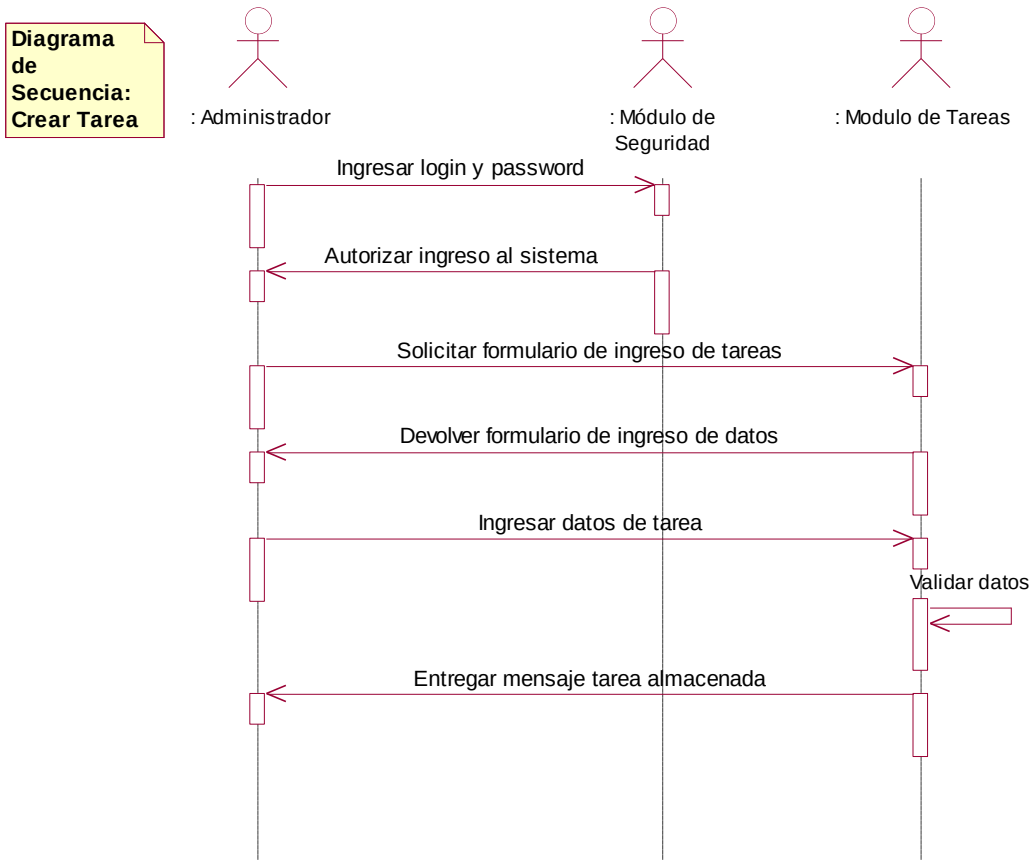
2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

2* Login y/o password no válidos. ”Termina caso de uso”.

4* No existe formulario de ingreso de datos de tarea. ”Termina caso de uso”.

6* Presenta mensaje de ingresar datos requeridos. “Termina caso de uso”.

DIAGRAMA DE SECUENCIA



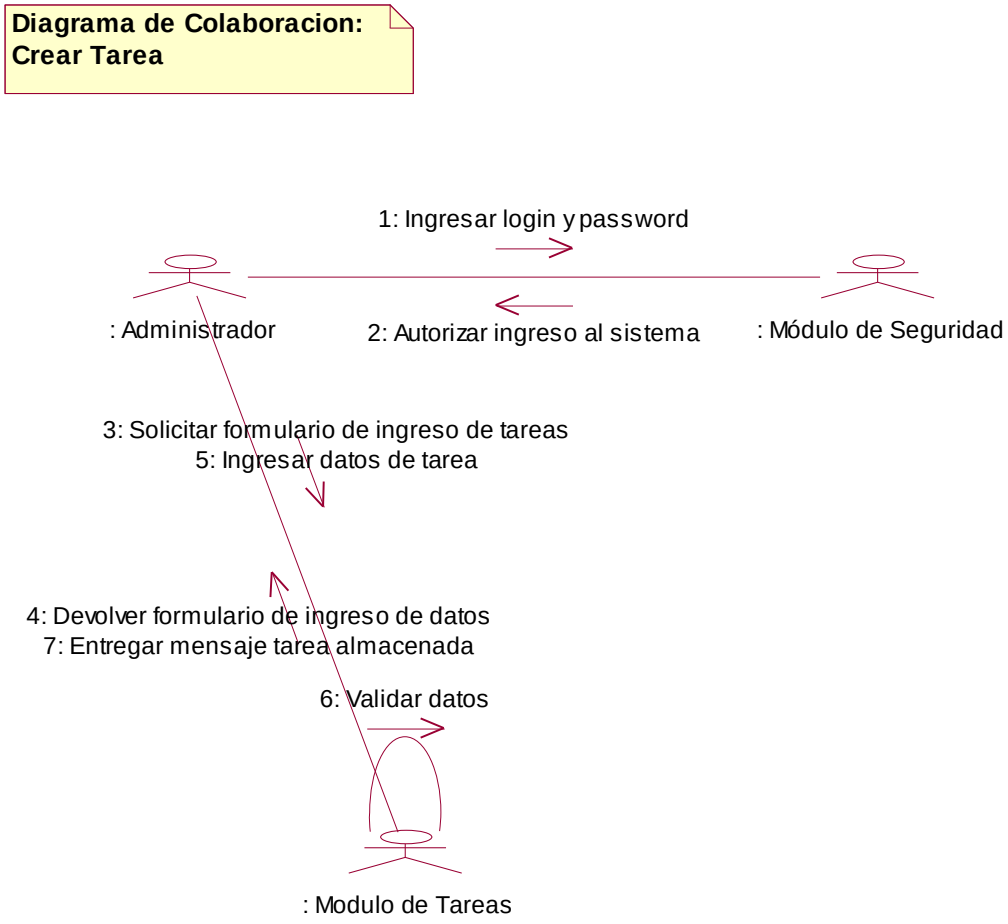
CONTRATO DE OPERACIÓN

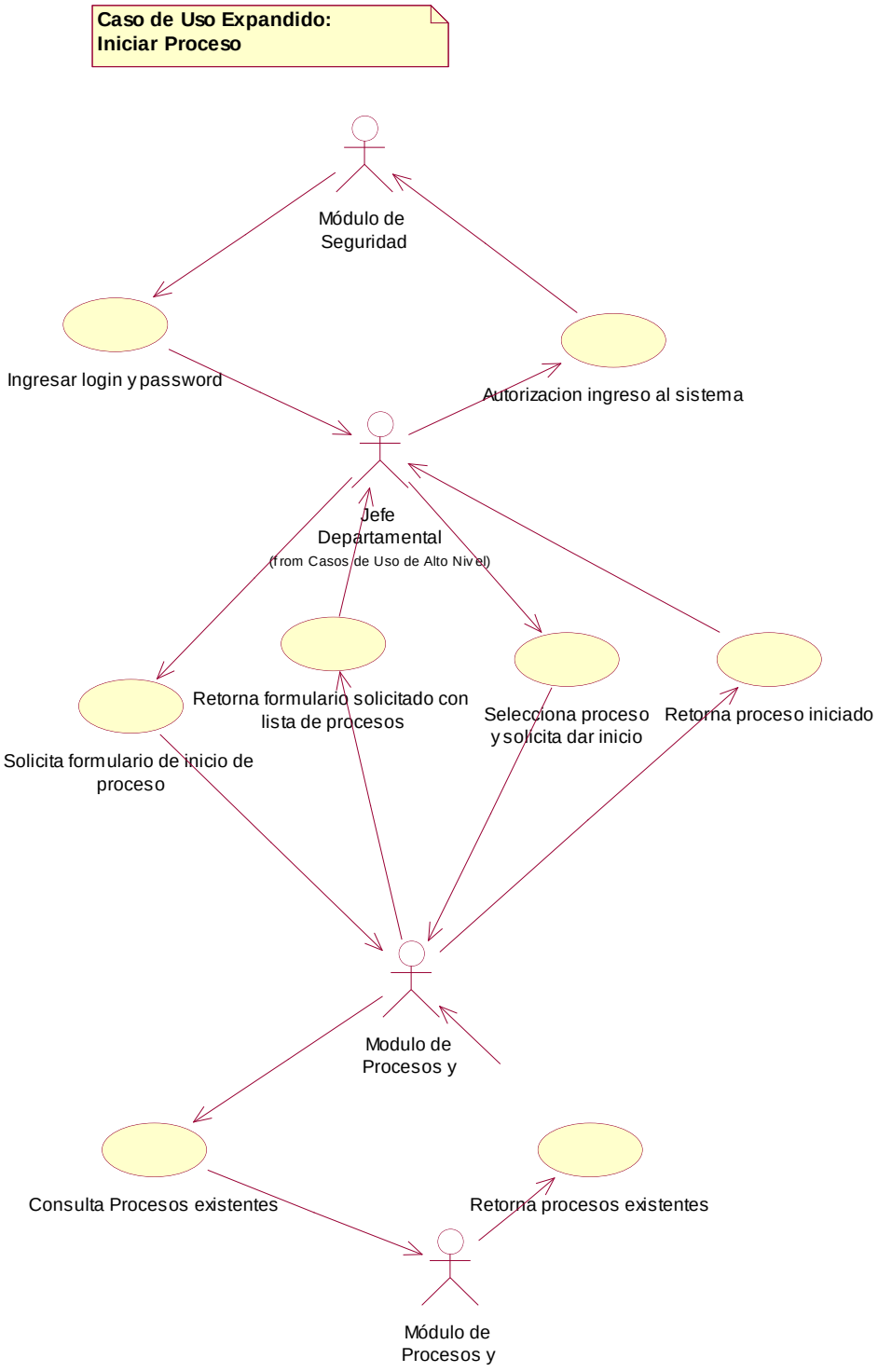
NOMBRE DEL CONTRATO: Ingresar login y password.
PROPÓSITO: Acceder al sistema para creación de nuevas tareas.
TIPO: Sistema.
SALIDA: Autorización de ingreso al sistema.
EXCEPCIONES: No existe formulario de ingreso login y password.
Login y/o password no válidos.
PRE-CONDICIONES: Relación usuario - interfaz
POST-CONDICIONES:

NOMBRE DEL CONTRATO: Solicitar formulario de ingreso de tareas.
PROPÓSITO: Acceder al formulario de ingreso de tareas para creación de una de ellas.
TIPO: Sistema.
SALIDA: Formulario ingreso de tareas.
EXCEPCIONES: No existe formulario de ingreso de tareas.
PRE-CONDICIONES: Relación usuario – interfaz (formulario).
POST-CONDICIONES:

NOMBRE DEL CONTRATO: Ingresar datos de tarea
PROPÓSITO: Ingresar al sistema datos para creación de nuevas tareas(transacción).
TIPO: Sistema.
SALIDA: Tarea almacenada exitosamente.
EXCEPCIONES: Presenta mensaje de ingresar datos requeridos.
PRE-CONDICIONES: Relación usuario – interfaz (formulario).
POST-CONDICIONES:

DIAGRAMA DE COLABORACIÓN





NOMBRE DEL CASO DE USO: INICIAR PROCESO.

PROPÓSITO: Arrancar con un nuevo proceso.

ACTOR (Iniciador): Jefe Departamental.

TIPO: Primario Real.

VISION GENERAL: El jefe departamental ingresa su login y password, el sistema autoriza el ingreso al sistema, el jefe departamental solicita formulario de inicio de proceso, el sistema retorna el formulario solicitado con listado de procesos, el jefe departamental selecciona un proceso y solicita dar inicio, el sistema retorna proceso iniciado termina caso de uso.

CURSO TIPICO DE EVENTOS

ACTOR	SISTEMA
1. Ingresar login y password.	2. Autorizar ingreso al sistema.
3. Solicitar formulario de inicio de proceso.	4. Retornar formulario solicitado con listado de procesos.
5. Seleccionar proceso y solicitar dar inicio.	6. Retornar proceso iniciado.

CURSOS ALTERNATIVOS:

2* Login y/o password inválidos. “Termina caso de uso”.

2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

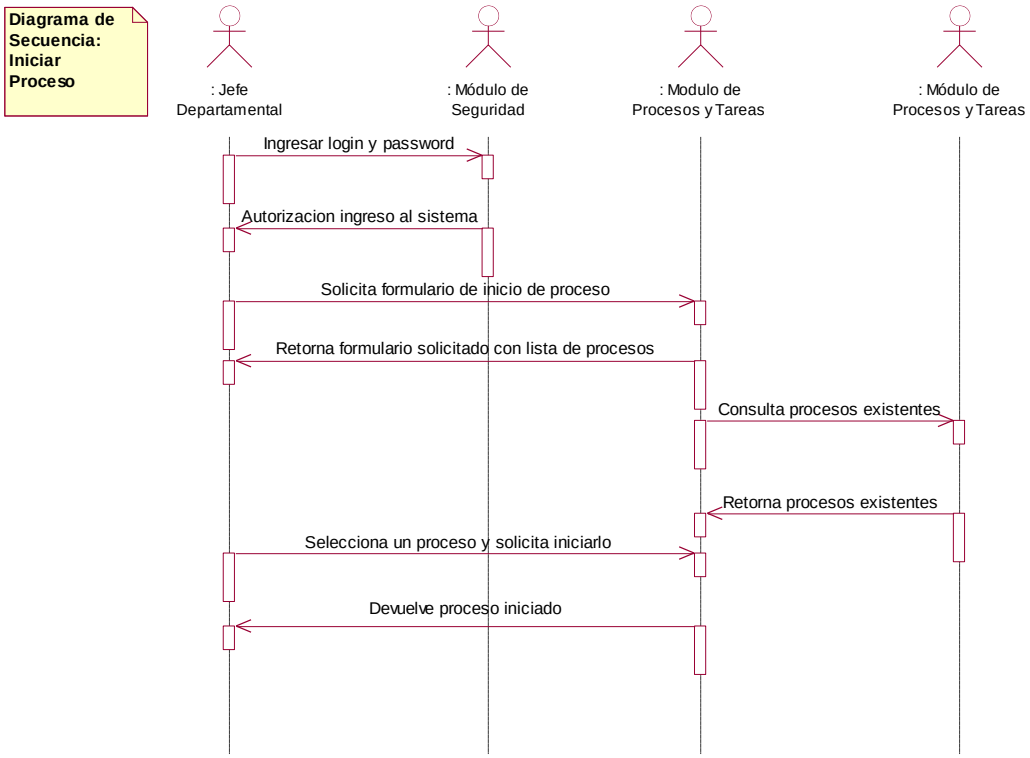
4* No existe listado de procesos en el sistema. ”Termina caso de uso”.

4* No presenta formulario de inicio de proceso. “Termina caso de uso”.

6* No presenta mensaje de proceso iniciado. “Termina caso de uso”.

6* No da inicio al proceso. “Termina caso de uso”.

DIAGRAMA DE SECUENCIA



CONTRATO DE OPERACIÓN

NOMBRE DEL CONTRATO: Ingresar login y password.

PROPÓSITO: Acceder al sistema para dar inicio a un proceso.

TIPO: Sistema.

SALIDA: Autorización de ingreso al sistema.

EXCEPCIONES: Login y/o password inválidos.
No existe formulario de ingreso de login y password.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Solicitar formulario de inicio de proceso.

PROPÓSITO: Desplegar formulario con listado de todos los procesos del sistema.

TIPO: Sistema.

SALIDA: Listado general de procesos para dar inicio.

EXCEPCIONES: No existe listado de procesos en el sistema.

No presenta formulario de inicio de proceso.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Seleccionar proceso y solicitar dar inicio.

PROPÓSITO: Desplegar mensaje de proceso iniciado correctamente.

TIPO: Sistema.

SALIDA: Proceso iniciado.

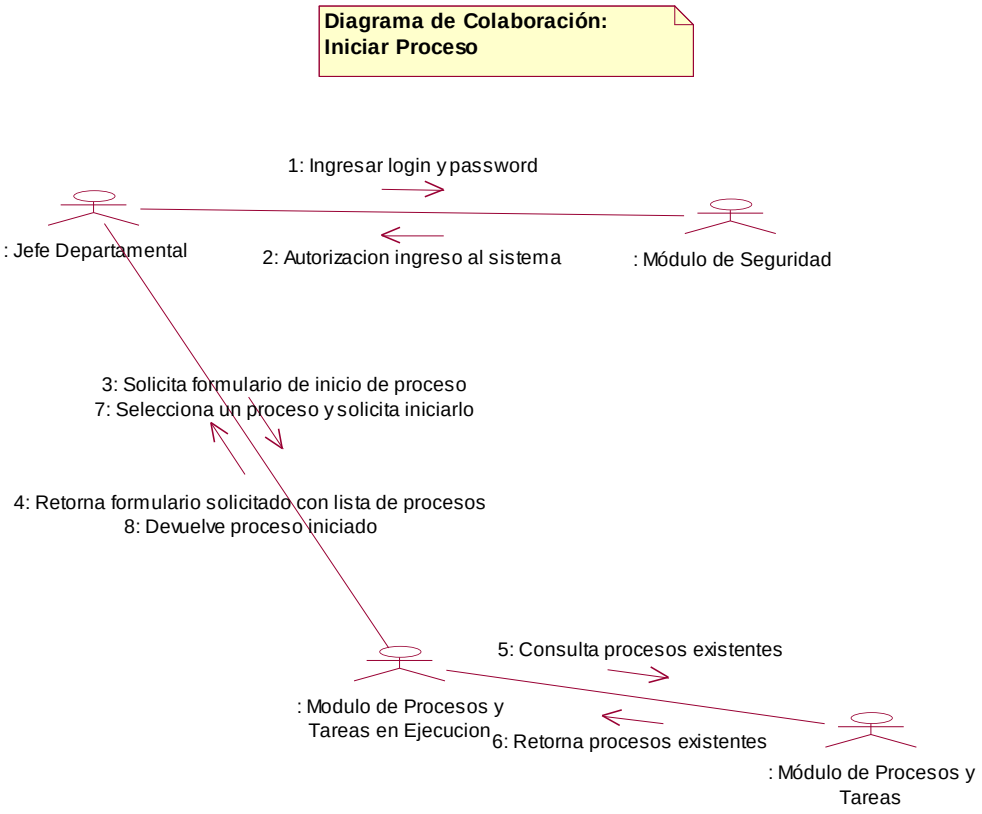
EXCEPCIONES: No presenta mensaje de proceso iniciado.

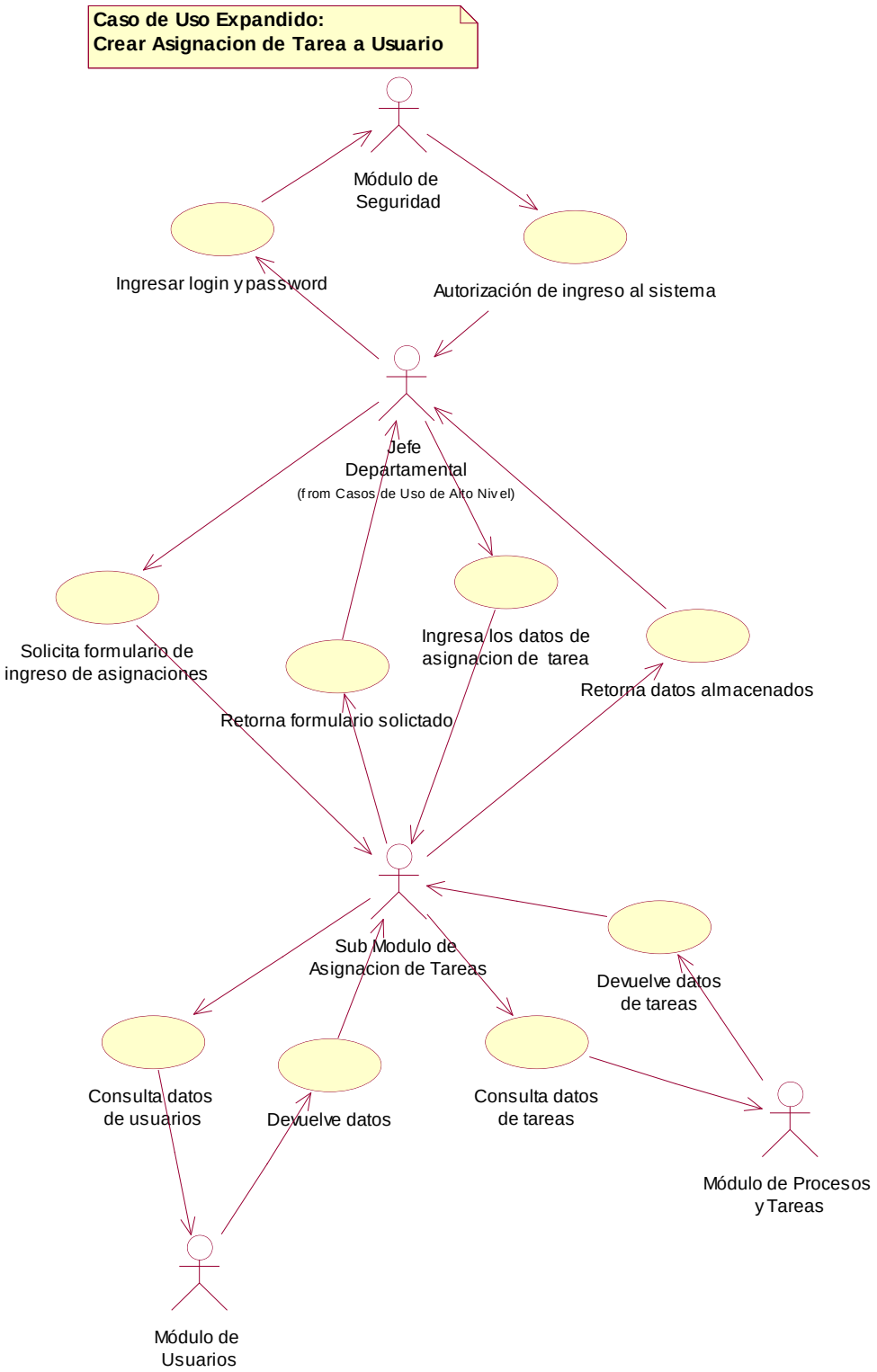
No da inicio al proceso.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

DIAGRAMA DE COLABORACION




Caso de Uso Expandido:
Crear Asignacion de Tarea a Usuario
Módulo de
Seguridad
Ingresar login y password
Autorización de ingreso al sistema
Jefe
Departamental
(from Casos de Uso de Alto Nivel)
Solicita formulario de
ingreso de asignaciones
Retorna formulario solictado
Ingresa los datos de
asignacion de tarea
Retorna datos almacenados
Sub Modulo de
Asignacion de Tareas
Devuelve datos
de tareas
Consulta datos
de usuarios
Devuelve datos
Consulta datos
de tareas
Módulo de Procesos
y Tareas
Módulo de
Usuarios

NOMBRE DEL CASO DE USO: CREAR ASIGNACIONES DE TAREAS A USUARIOS.

PROPÓSITO: Implantar asignaciones de tareas a usuarios en el sistema por parte del jefe departamental.

ACTOR (Iniciador): Jefe Departamental.

TIPO: Primario Real.

VISION GENERAL: El jefe departamental ingresa su login y password, el sistema autoriza el ingreso al sistema, el jefe departamental solicita el formulario de ingreso de asignaciones de tareas, el sistema retorna formulario de ingreso, el jefe departamental ingresa datos de asignaciones de tarea, el sistema presenta datos asignaciones de tarea almacenados, termina el caso de uso.

CURSO TIPICO DE EVENTOS

ACTOR	SISTEMA
1. Ingresar login y password.	2. Autorizar ingreso al sistema.
3. Solicitar formulario de ingreso de asignaciones de tareas.	4. Presentar formulario de ingreso.
5. Ingresar datos de asignaciones de tarea.	6. Presentar datos de asignaciones tarea almacenada.

CURSOS ALTERNATIVOS:

2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

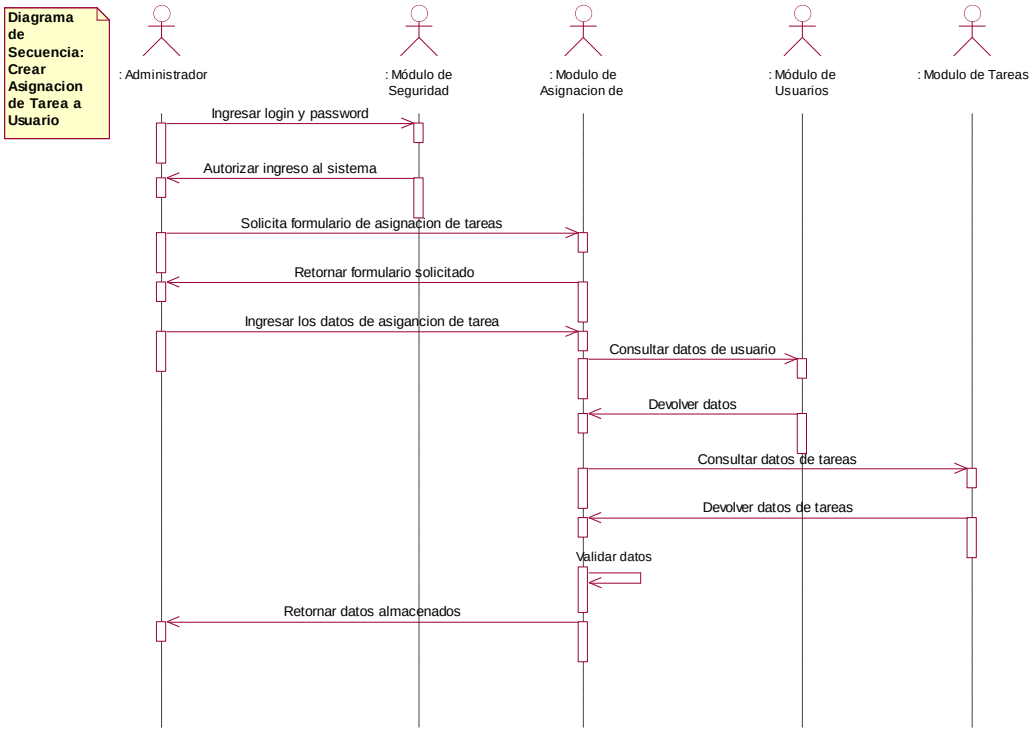
2* Login y/o password no válidos. ”Termina caso de uso”.

4* No existe formulario de ingreso de asignaciones. ”Termina caso de uso”.

6* No presenta mensaje de información almacenada exitosamente. “Termina caso de uso”.

6* Presenta mensaje de ingresar datos requeridos. “Termina caso de uso”.

DIAGRAMA DE SECUENCIA



CONTRATO DE OPERACIÓN

NOMBRE DEL CONTRATO: Ingresar login y password.

PROPÓSITO: Acceder al sistema para la asignación de tareas a usuarios

TIPO: Sistema.

SALIDA: Autorización de ingreso al sistema

EXCEPCIONES: No existe formulario de ingreso de login y password.
Login y/o password no válidos.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Solicitar formulario de asignación de tareas.

PROPÓSITO: Acceder al formulario de asignación de tareas

TIPO: Sistema.

SALIDA: Formulario de asignación de tareas.

EXCEPCIONES: No existe formulario de ingreso de asignaciones.

PRE-CONDICIONES: Relación usuario – interfaz.

POST-CONDICIONES

NOMBRE DEL CONTRATO: Ingresar datos de asignación de tareas.

PROPÓSITO: Ingresar al sistema datos para la asignación de tareas

TIPO: Sistema.

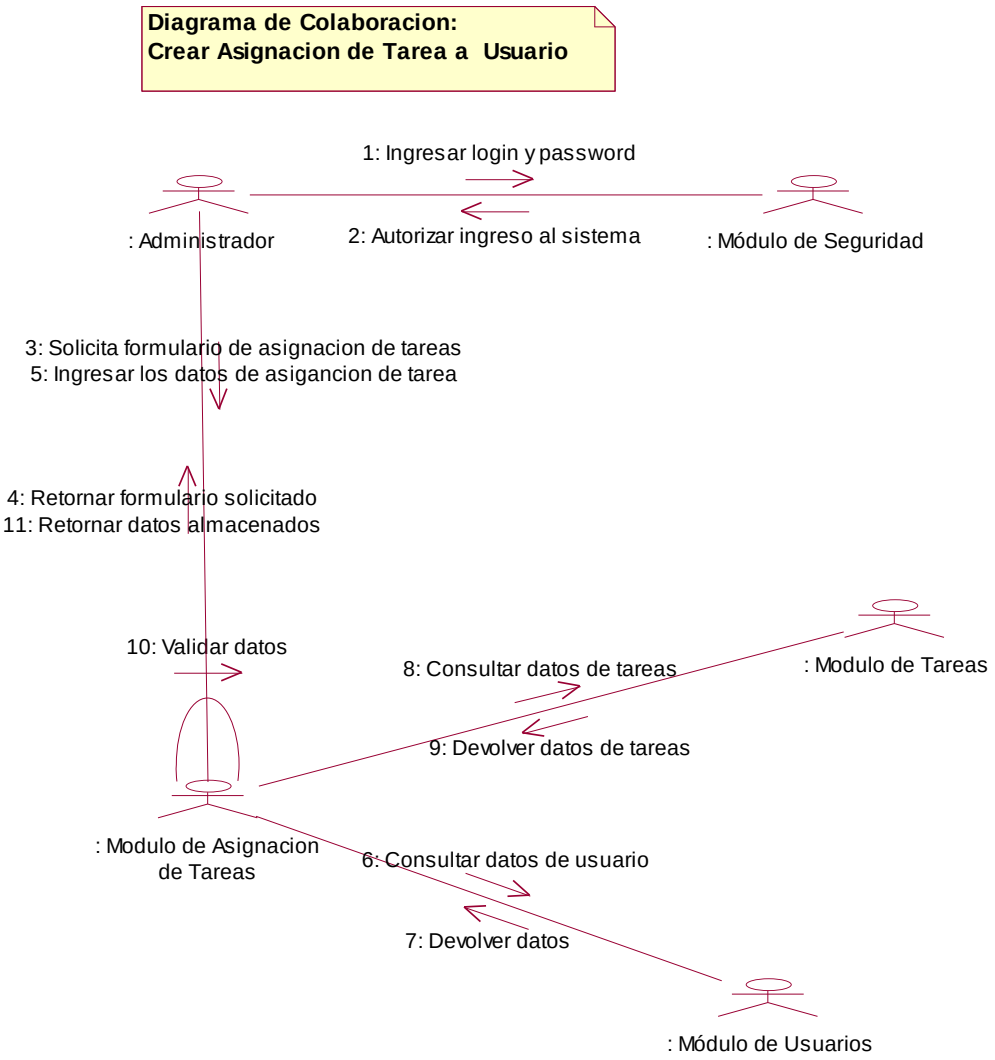
SALIDA: Tareas asignadas exitosamente.

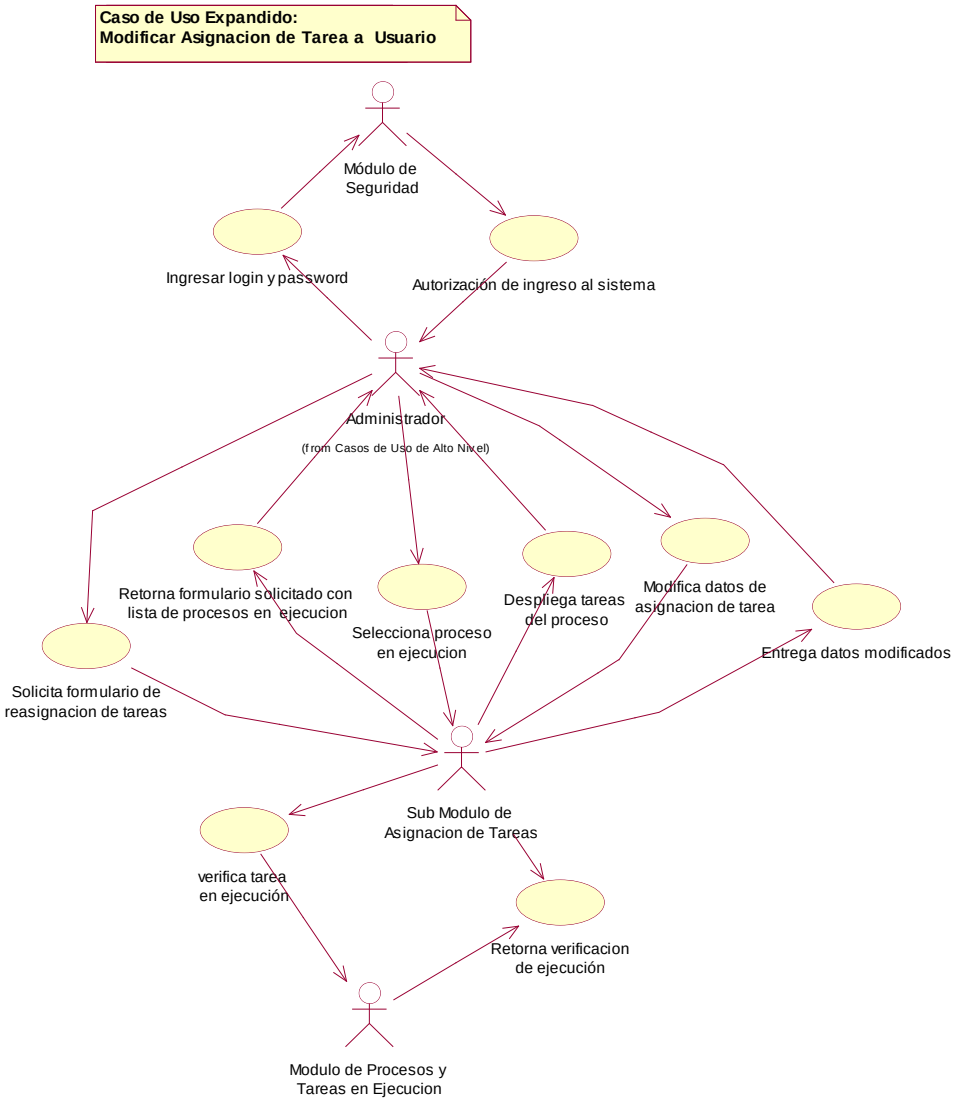
EXCEPCIONES: No presenta mensaje de información almacenada exitosamente.
Presenta mensaje de ingresar datos requeridos.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

DIAGRAMA DE COLABORACIÓN





NOMBRE DEL CASO DE USO: MODIFICAR ASIGNACION DE TAREAS A USUARIOS.

PROPÓSITO: Cambiar datos de asignación tareas a usuarios

ACTOR (Iniciador): Jefe Departamental.

TIPO: Primario Real.

VISION GENERAL: El jefe departamental ingresa su login y password, el sistema autoriza el ingreso al sistema, el jefe departamental solicita el formulario de reasignación de tareas, el sistema devuelve formulario solicitado con listado de procesos en ejecución, el jefe departamental selecciona proceso en ejecución, el sistema despliega las tareas de dicho proceso, el jefe departamental modifica

datos de asignación, el sistema devuelve datos de asignación de tarea modificada termina el caso de uso.

CURSO TIPICO DE EVENTOS

ACTOR	SISTEMA
1. Ingresar login y password.	2. Autorizar ingreso al sistema.
3. Solicitar formulario de reasignación de tareas.	4. Presentar formulario de reasignación de tarea con lista de procesos en ejecución.
5. Seleccionar Proceso en ejecución.	6. Presentar tareas de dicho proceso
7. Modificar datos de asignación.	8. Devolver datos de asignaciones de tarea modificada.

CURSOS ALTERNATIVOS:

2* Login y/o password no válidos. “Termina caso de uso”.

2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

4* No existe procesos iniciados y ejecutándose en el sistema. ”Termina caso de uso”.

4* No presenta formulario con el listado de procesos en ejecución. “Termina caso de uso”.

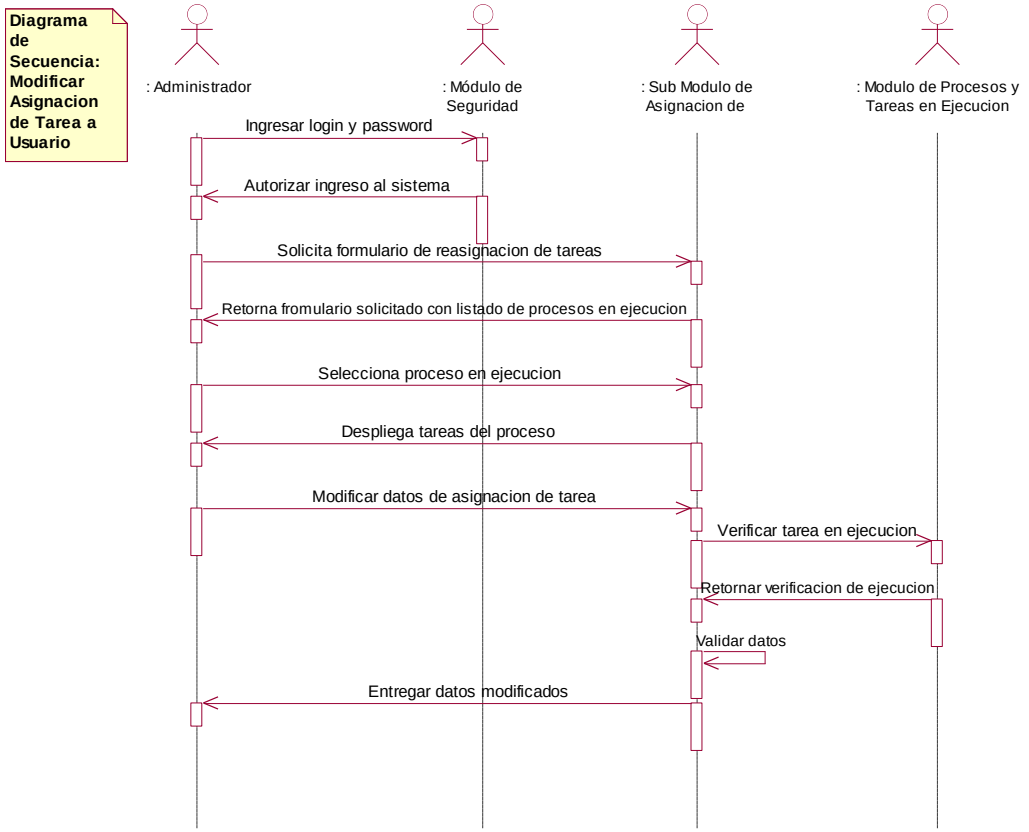
6* No presenta tareas a reasignar del proceso. ”Termina caso de uso”.

6* El proceso no tiene tareas para reasignar. “Termina caso de uso”.

8* No presenta mensaje de tarea reasignada. “Termina caso de uso”.

8* No se puede reasignar tareas las tareas fueron finalizadas. “Termina caso de uso”.

DIAGRAMA DE SECUENCIA



CONTRATO DE OPERACIÓN

NOMBRE DEL CONTRATO: Ingresar login y password.

PROPÓSITO: Acceder al sistema para la reasignación de tareas a usuarios.

TIPO: Sistema.

SALIDA: Autorización de ingreso al sistema

EXCEPCIONES: Login y/o password no válidos.
No existe formulario de ingreso de login y password.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Solicitar formulario de reasignación de tareas.

PROPÓSITO: Desplegar información de procesos en ejecución con sus respectivas tareas para la reasignación.

TIPO: Sistema.

SALIDA: Formulario de reasignación de tarea junto con listado de procesos en ejecución.

EXCEPCIONES: No existe procesos iniciados y ejecutándose en el sistema.
No presenta formulario con el listado de procesos en ejecución.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Seleccionar proceso en ejecución.

PROPÓSITO: Elegir un proceso en ejecución para que se despliegue las tareas del mismo para la reasignación.

TIPO: Sistema.

SALIDA: Tareas del proceso seleccionado para la reasignación.

EXCEPCIONES: No presenta tareas a reasignar del proceso.
El proceso no tiene tareas para reasignar.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Modificar datos de asignación de tarea

PROPÓSITO: Cambiar datos de tarea.

TIPO: Sistema.

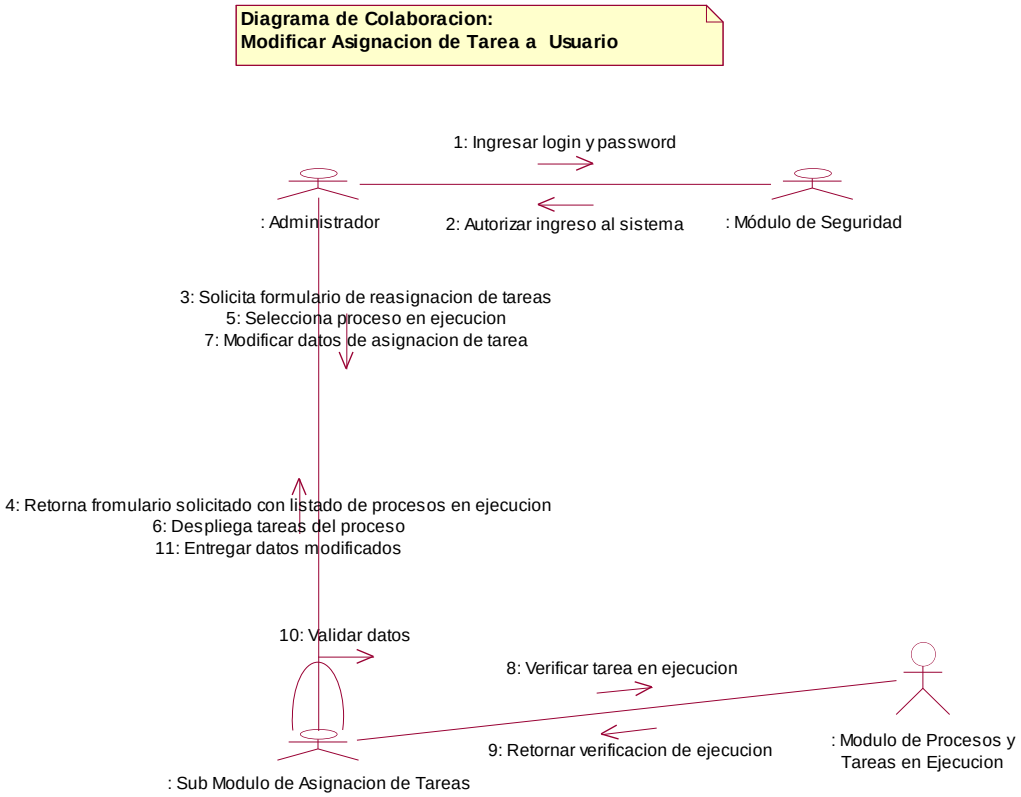
SALIDA: Mensaje. Tarea reasignada exitosamente.

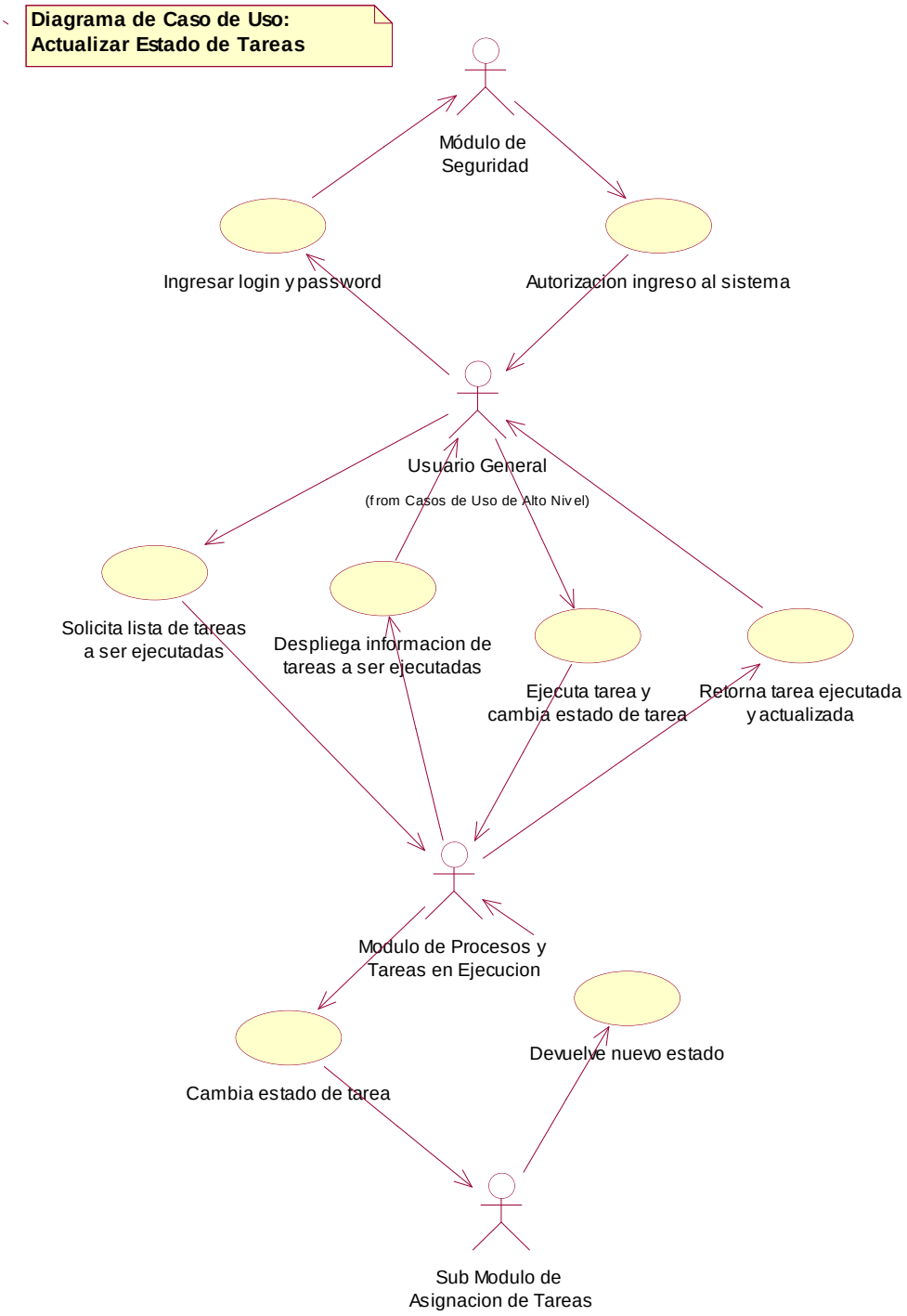
EXCEPCIONES: No presenta mensaje de tarea reasignada.
No se puede reasignar tareas las tareas fueron finalizadas.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

DIAGRAMA DE COLABORACIÓN





NOMBRE DEL CASO DE USO: ACTUALIZAR ESTADO DE TAREAS.

PROPÓSITO: Cambiar el estado de una tarea ejecutada

ACTOR (Iniciador): Usuario General.

TIPO: Primario Real.

VISION GENERAL: El usuario general ingresa su login y password, el sistema autoriza el ingreso al sistema, el usuario general solicita lista de tareas a ser ejecutadas, el sistema despliega información de tareas a ser ejecutadas, el usuario general ejecuta la tarea y cambia el estado de la misma, el sistema retorna tarea ejecutada y estado actualizado termina el caso de uso.

CURSO TIPICO DE EVENTOS

ACTOR	SISTEMA
1. Ingresar login y password.	2. Autorizar ingreso al sistema.
3. Solicitar lista de tareas a ser ejecutadas.	4. Desplegar información de tareas a ser ejecutadas.
5. Ejecutar la tarea y cambiar el estado de la misma.	6. Retornar tarea ejecutada y estado actualizado.

CURSOS ALTERNATIVOS:

2* Login y/o password no válidos. “Termina caso de uso”.

2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

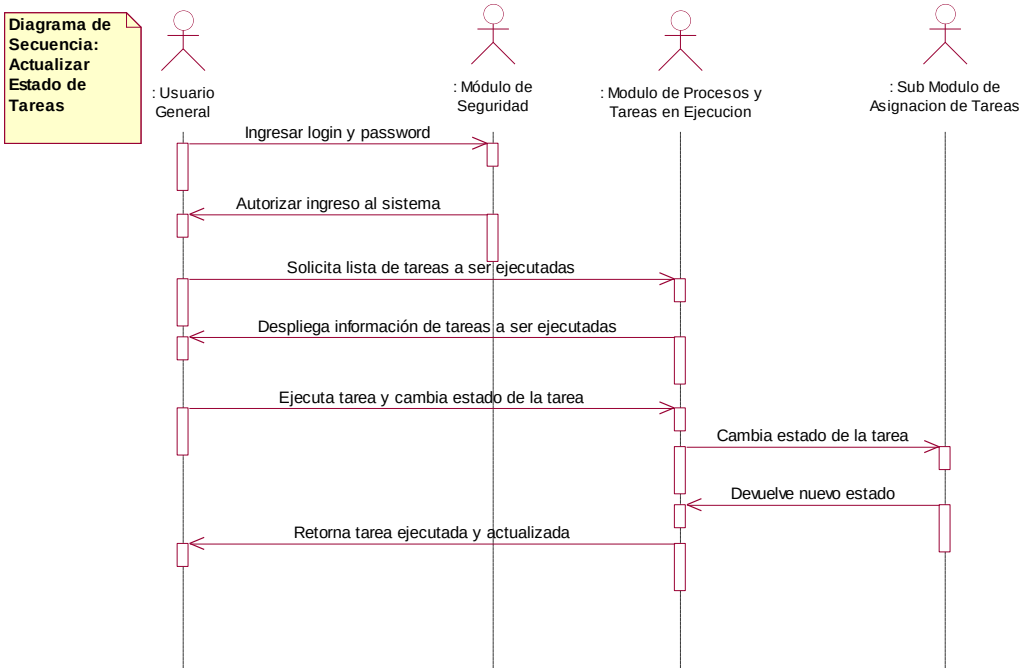
4* No existen carga de actividades para ese usuario. ”Termina caso de uso”.

4* No presenta formulario con el listado de tareas a ser ejecutadas. “Termina caso de uso”.

6* No presenta estado de tarea actualizado. ”Termina caso de uso”.

6* No se actualiza el estado de la tarea. “Termina caso de uso”.

DIAGRAMA DE SECUENCIA



CONTRATO DE OPERACIÓN

NOMBRE DEL CONTRATO: Ingresar login y password.

PROPÓSITO: Acceder al sistema para la reasignación de tareas a usuarios.

TIPO: Sistema.

SALIDA: Autorización de ingreso al sistema

EXCEPCIONES: Login y/o password no válidos.

No existe formulario de ingreso de login y password.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Solicitar lista de tareas a ser ejecutadas.

PROPÓSITO: Desplegar listado general de carga de actividades.

SALIDA: Listado general de carga de actividades.

EXCEPCIONES: No existe carga de actividades para ese usuario.

No presenta formulario con el listado de tareas a ser ejecutadas.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Ejecutar la tarea y cambiar el estado de la misma

PROPÓSITO: Cambiar estado de tarea.

TIPO: Sistema.

SALIDA: Mensaje. Estado actualizado exitosamente.

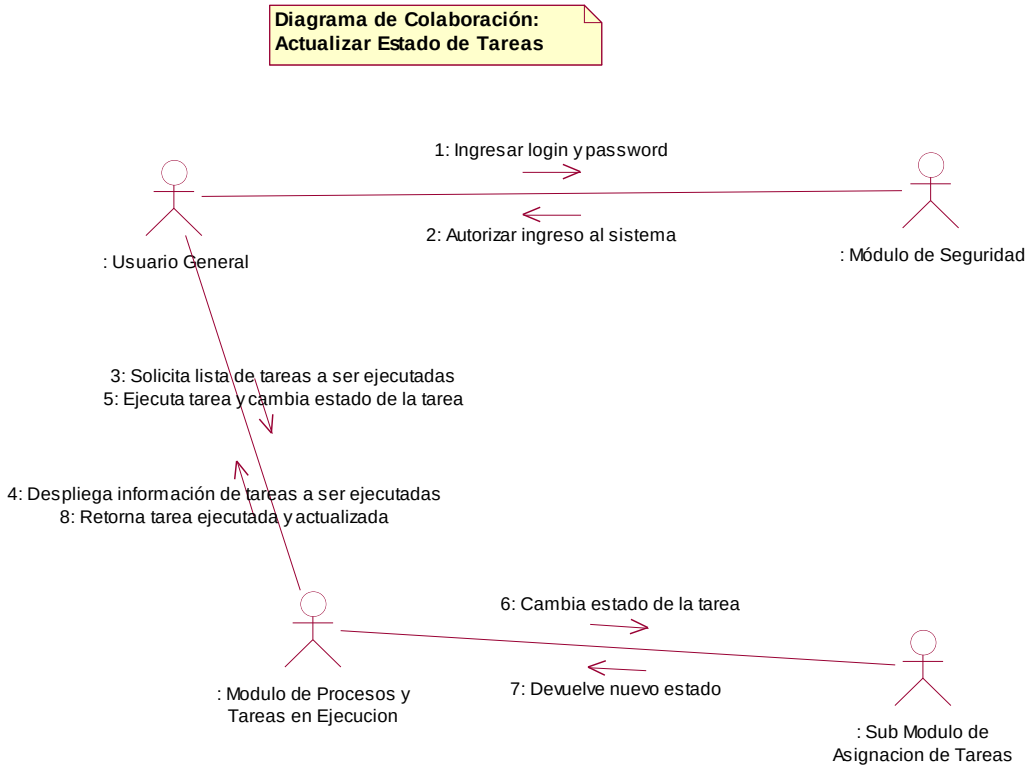
EXCEPCIONES: No presenta estado de tarea actualizado.

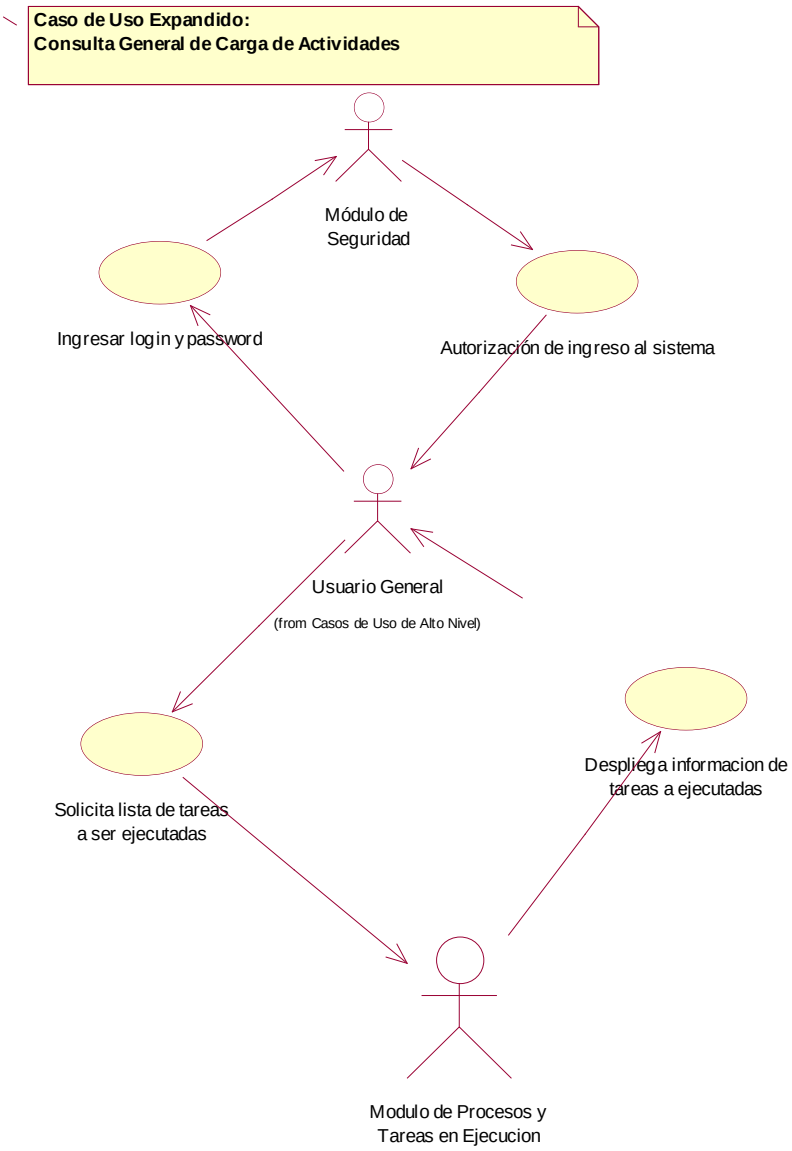
No se actualiza el estado de la tarea.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

DIAGRAMA DE COLABORACIÓN





NOMBRE DEL CASO DE USO: CONSULTAR CARGA DE ACTIVIDADES EN FORMA GENERAL.

PROPÓSITO: Conocer datos en forma general de las cargas de actividades que posee cada usuario.

ACTOR (Iniciador): Usuario General.

TIPO: Primario Real.

VISION GENERAL: El usuario general solicita ingresar su login y password, el sistema autoriza el ingreso, el usuario general solicita listado de tareas a ser

ejecutadas, el sistema despliega datos de la carga de actividades y termina el caso de uso.

CURSO TIPICO DE EVENTOS

ACTOR	SISTEMA
1. Ingresar login y password.	2. Autorizar ingreso al sistema.
3. Solicitar lista de tareas a ser ejecutadas.	4. Desplegar información de carga de actividades.

CURSOS ALTERNATIVOS:

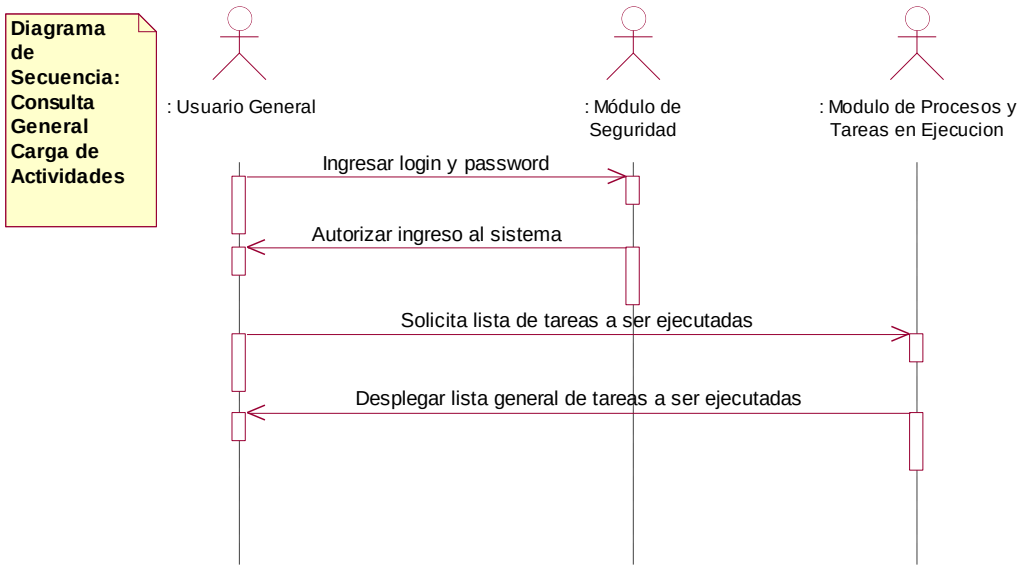
2* Login y/o password incorrectos“Termina caso de uso”.

2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

4* No existen carga de actividades para ese usuario. ”Termina caso de uso”.

4* No presenta formulario con el listado de tareas a ser ejecutadas. “Termina caso de uso”.

DIAGRAMA DE SECUENCIA



CONTRATO DE OPERACIÓN

NOMBRE DEL CONTRATO: Ingresar login y password.

PROPÓSITO: Acceder al sistema para la consulta general de carga de actividades.

TIPO: Sistema.

SALIDA: Autorización de ingreso al sistema.

EXCEPCIONES: Login y/o password incorrectos.

No existe formulario de ingreso de login y password.

PRE-CONDICIONES: Relación usuario – interfaz.

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Solicitar lista de tareas a ser ejecutadas.

PROPÓSITO: Desplegar listado general de carga de actividades.

SALIDA: Listado general de carga de actividades.

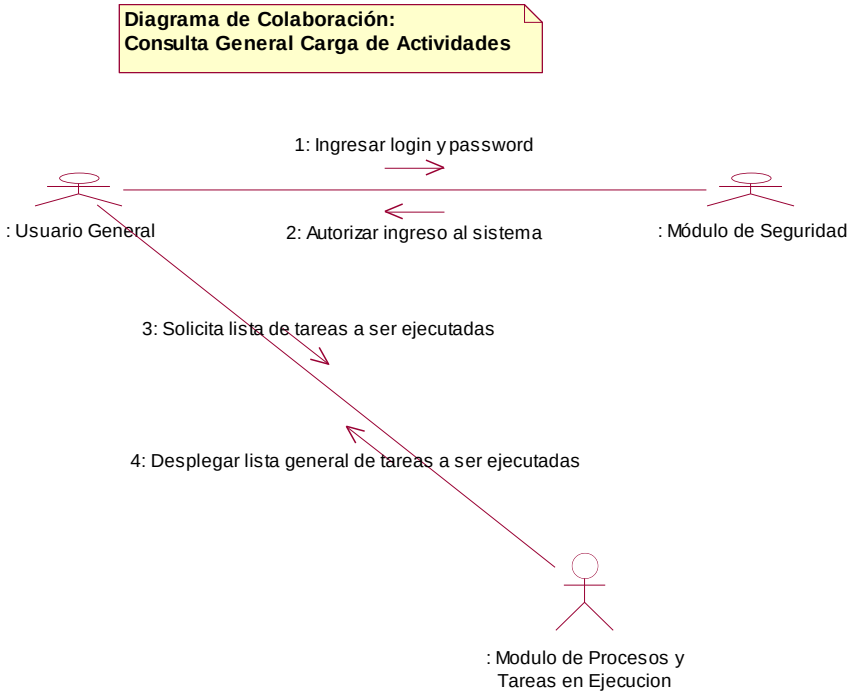
EXCEPCIONES: No existe carga de actividades para ese usuario.

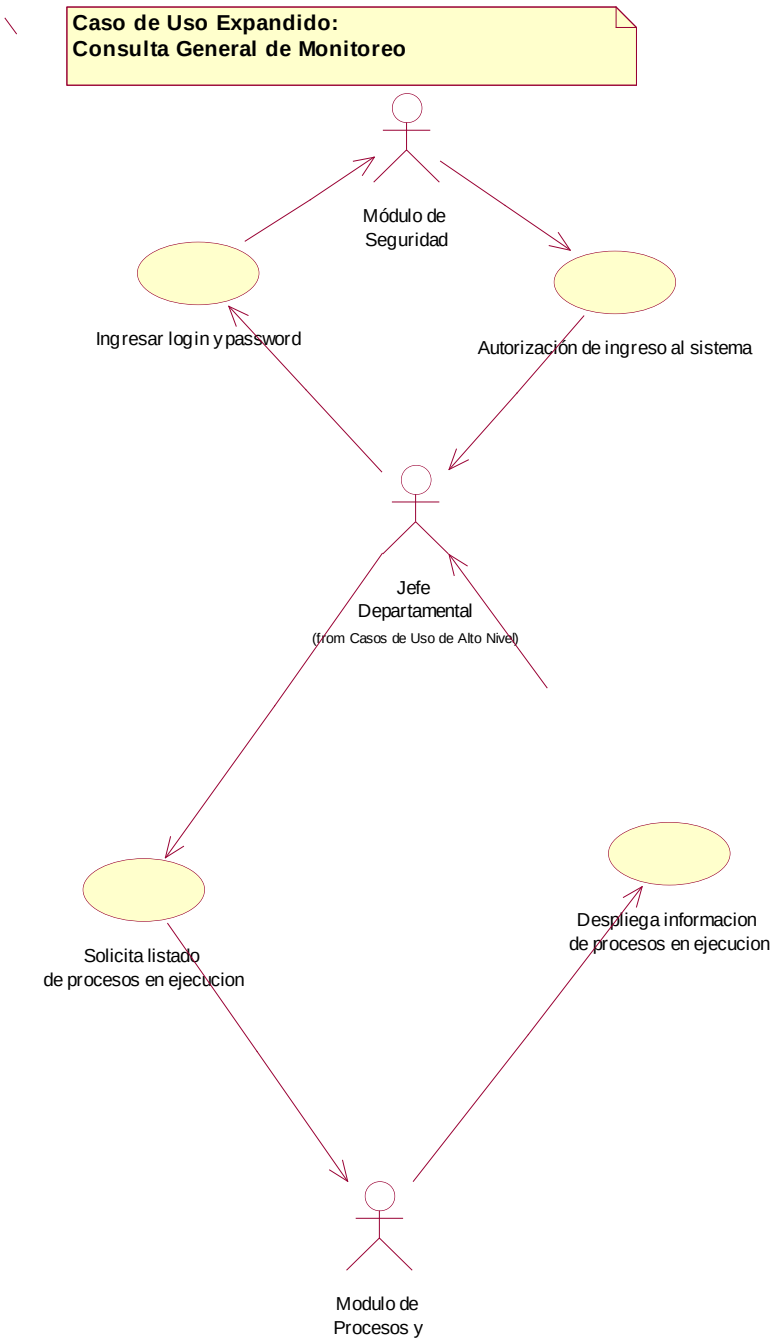
No presenta formulario con el listado de tareas a ser ejecutadas.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

DIAGRAMA DE COLABORACIÓN





NOMBRE DEL CASO DE USO: CONSULTAR EN FORMA GENERAL LOS ESTADOS DE LOS PROCESOS.

ROPÓSITO: Monitorear todos los estados por los que atraviesa un determinado proceso.

ACTOR (Iniciador): Jefe Departamental.

TIPO: Primario Real.

VISION GENERAL: El jefe departamental ingresa su login y password, el sistema autoriza ingreso al sistema, el jefe departamental pide desplegar la lista de todos los procesos en ejecución, el sistema despliega datos de procesos en forma general, termina caso de uso.

CURSO TIPICO DE EVENTOS

ACTOR	SISTEMA
1. Ingresar login y password.	2. Autorizar ingreso al sistema.
3. Solicitar despliegue lista de procesos en ejecución.	4. Presentar datos de procesos.

CURSOS ALTERNATIVOS:

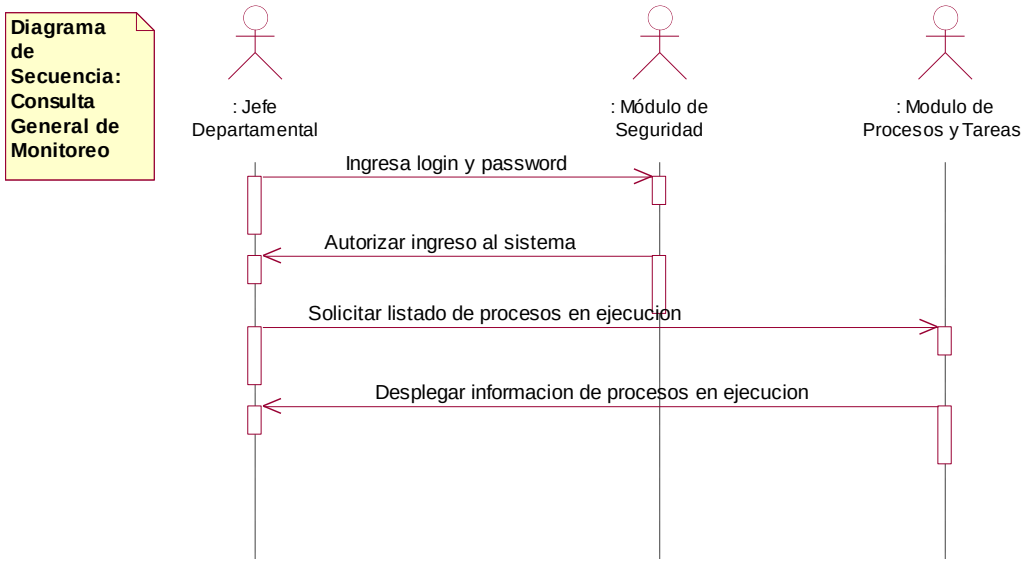
2* Login y/o password incorrectos“Termina caso de uso”.

2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

4* No existen procesos en ejecución en el sistema. ”Termina caso de uso”.

4* No presenta formulario con el listado de procesos en ejecución. “Termina caso de uso”.

DIAGRAMA DE SECUENCIA



CONTRATO DE OPERACIÓN

NOMBRE DEL CONTRATO: Ingresar login y password.

PROPÓSITO: Acceder al sistema para el monitoreo general de procesos.

TIPO: Sistema.

SALIDA: Autorización de ingreso al sistema.

EXCEPCIONES: No existe formulario de ingreso login y password.
Login y /o password incorrectos.

PRE-CONDICIONES: Relación usuario – interfaz.

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Solicitar listado de procesos en ejecución.

PROPÓSITO: Desplegar listado general de procesos.

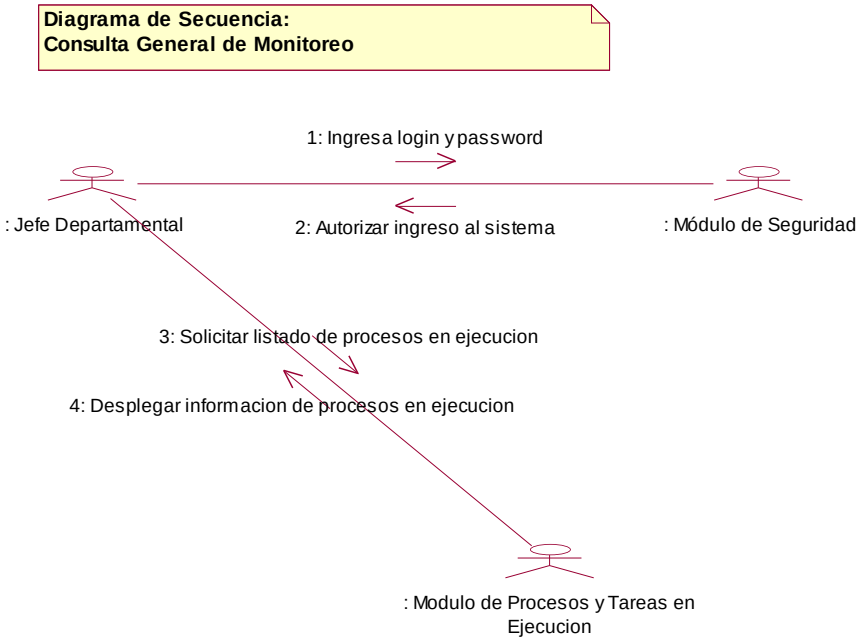
SALIDA: Listado general de procesos.

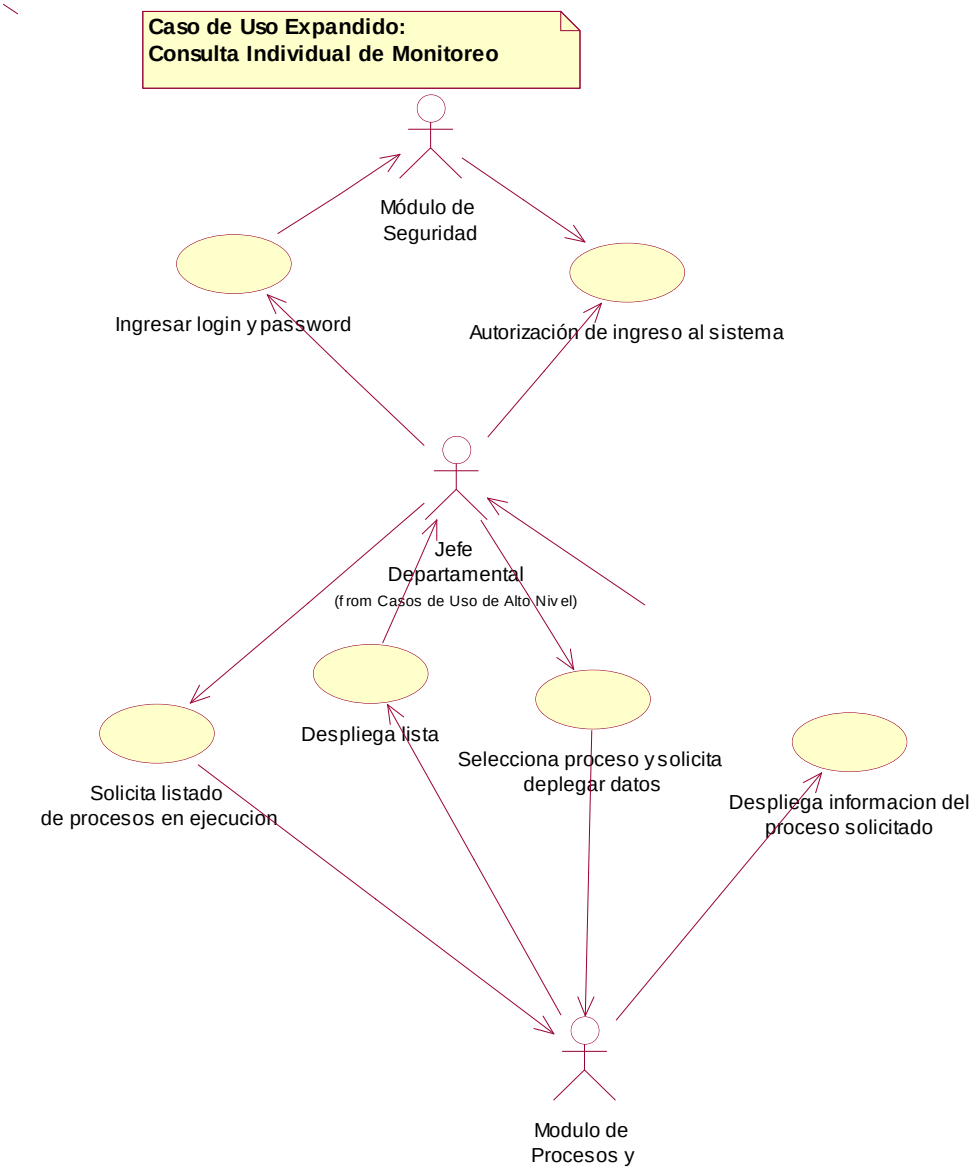
EXCEPCIONES: No existen procesos en ejecución en el sistema.
No presenta formulario con el listado de procesos en ejecución.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

DIAGRAMA DE COLABORACIÓN





NOMBRE DEL CASO DE USO: CONSULTAR EN FORMA INDIVIDUAL LOS ESTADOS DE LOS PROCESOS.

PROPÓSITO: Informarse acerca del desarrollo de un proceso en forma individual.

ACTOR (Iniciador): Jefe Departamental

TIPO: Primario Real.

VISION GENERAL: El jefe departamental ingresa el login y el password, el sistema autoriza ingreso al sistema, el jefe departamental solicita el listado de procesos en ejecución, el sistema despliega el listado, el jefe departamental

selecciona un proceso y solicita desplegar datos, el sistema despliega la información del proceso solicitado, termina caso de uso

CURSO TIPICO DE EVENTOS

ACTOR	SISTEMA
1.Ingresar login y password	2. Autorizar ingreso al sistema.
3. Solicitar listado de procesos en ejecución.	4. Presentar listado de los procesos.
5. Seleccionar un proceso del listado y solicita desplegar datos.	6. Presentar información del proceso solicitado.

CURSOS ALTERNATIVOS:

2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

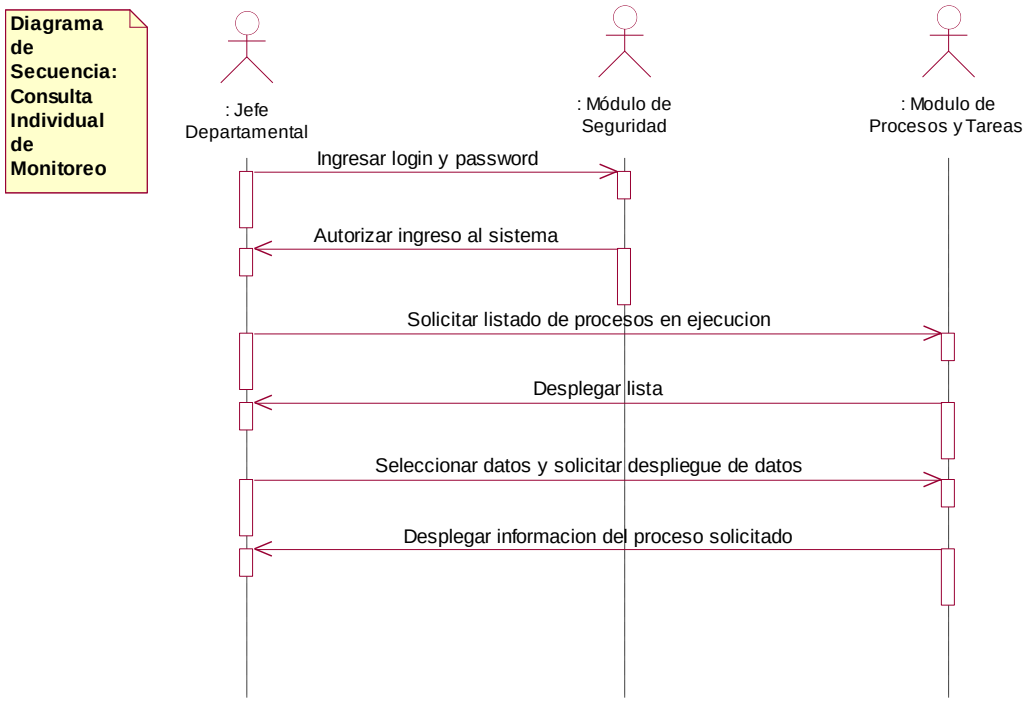
2* Login y/o password incorrecto. “Termina caso de uso”.

4* No existen procesos en ejecución. ”Termina caso de uso”.

4* No presenta el listado de procesos en ejecución. “Termina caso de uso”.

6* No presenta datos del proceso seleccionado. ”Termina caso de uso”.

DIAGRAMA DE SECUENCIA



CONTRATO DE OPERACIÓN

NOMBRE DEL CONTRATO: Ingresar login y password.

PROPÓSITO: Acceder al sistema para el monitoreo individual de un proceso

TIPO: Sistema.

SALIDA: Autorización de ingreso al sistema.

EXCEPCIONES: No existe formulario de ingreso de login y password.
Login y/o password incorrecto.

PRE-CONDICIONES: Relación usuario – interfaz.

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Solicitar listado de procesos en ejecución.

PROPÓSITO: Desplegar listado general de monitoreos a procesos.

SALIDA: Listado general de seguimiento o monitoreo de procesos.

EXCEPCIONES: No existen procesos en ejecución.

No presenta el listado de procesos en ejecución.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Seleccionar un proceso y solicitar desplegar información.

PROPÓSITO: Elegir monitoreo de un determinado proceso para que se despliegue la información del mismo.

TIPO: Sistema.

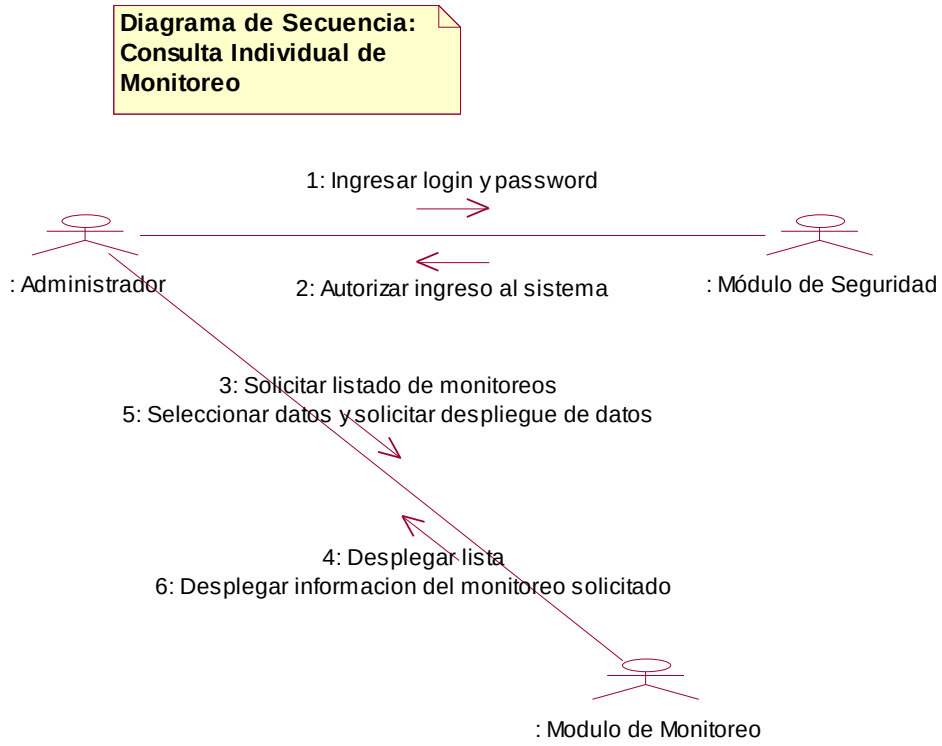
SALIDA: Formulario de información del monitoreo seleccionado.

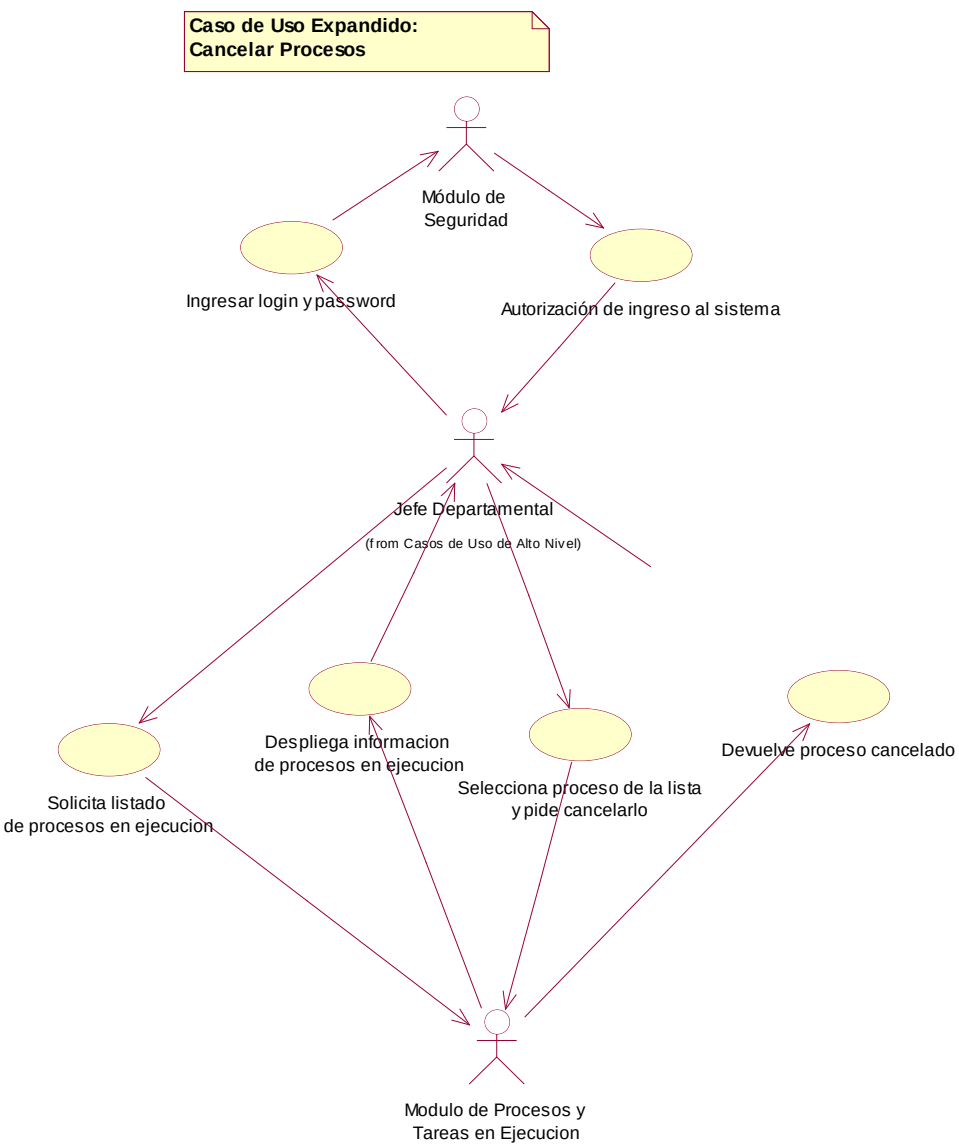
EXCEPCIONES: No presenta datos del proceso seleccionado.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

DIAGRAMA DE COLABORACIÓN





NOMBRE DEL CASO DE USO: CANCELAR PROCESOS EN EJECUCION.

PROPÓSITO: Interrumpir un proceso que se encuentre ejecutándose.

ACTOR (Iniciador): Jefe Departamental

TIPO: Primario Real.

VISION GENERAL: El jefe departamental ingresa el login y el password, el sistema autoriza ingreso al sistema, el jefe departamental solicita el listado de

procesos en ejecución, el sistema despliega el listado de procesos en ejecución, el jefe departamental selecciona un proceso y pide cancelarlo, el sistema devuelve proceso cancelado, termina caso de uso

CURSO TIPICO DE EVENTOS

ACTOR	SISTEMA
1.Ingresar login y password	2. Autorizar ingreso al sistema.
3. Solicitar listado de procesos en ejecución.	4. Presentar listado de los procesos en ejecución.
5. Seleccionar un proceso del listado y pedir cancelar proceso.	6. Devolver proceso cancelado.

CURSOS ALTERNATIVOS:

2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

2* Login y/o password incorrecto. “Termina caso de uso”.

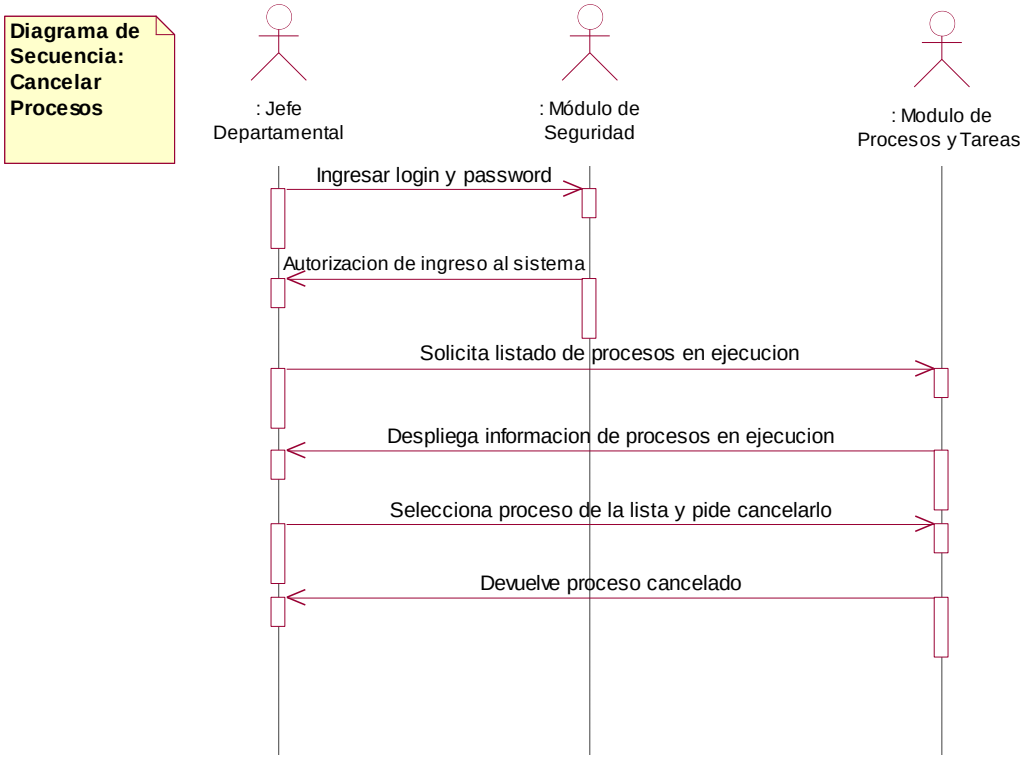
4* No existen procesos en ejecución. ”Termina caso de uso”.

4* No presenta el listado de procesos en ejecución. “Termina caso de uso”.

6* No presenta mensaje de proceso cancelado. ”Termina caso de uso”.

6* No cancela el proceso. “Termina caso de uso”.

DIAGRAMA DE SECUENCIA



CONTRATO DE OPERACIÓN

NOMBRE DEL CONTRATO: Ingresar login y password.

PROPÓSITO: Acceder al sistema para el cancelar un proceso en ejecución.

TIPO: Sistema.

SALIDA: Autorización de ingreso al sistema.

EXCEPCIONES: No existe formulario de ingreso de login y password.
Login y/o password incorrecto.

PRE-CONDICIONES: Relación usuario – interfaz.

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Solicitar listado de procesos en ejecución.

PROPÓSITO: Desplegar listado general de procesos en ejecución.

SALIDA: Listado general de procesos en ejecución.

EXCEPCIONES: No existen procesos en ejecución.

No presenta el listado de procesos en ejecución.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Seleccionar un proceso y solicitar cancelar proceso.

PROPÓSITO: Elegir un determinado proceso y pedir que se cancele el mismo.

TIPO: Sistema.

SALIDA: Mensaje. Proceso cancelado.

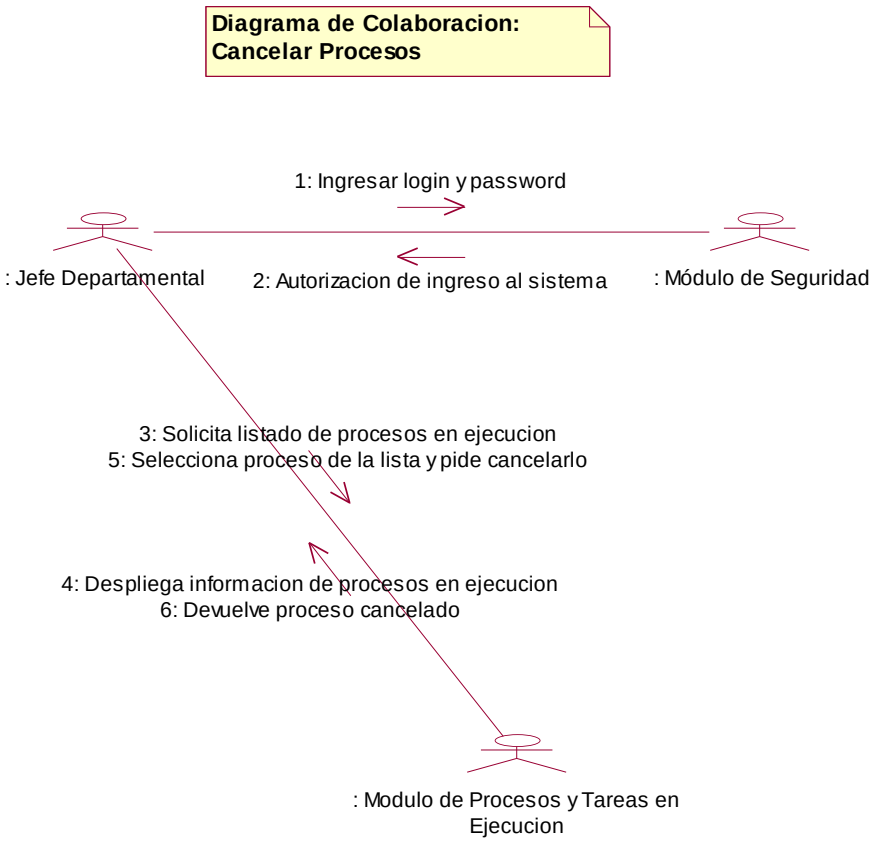
EXCEPCIONES: No presenta mensaje de proceso cancelado.

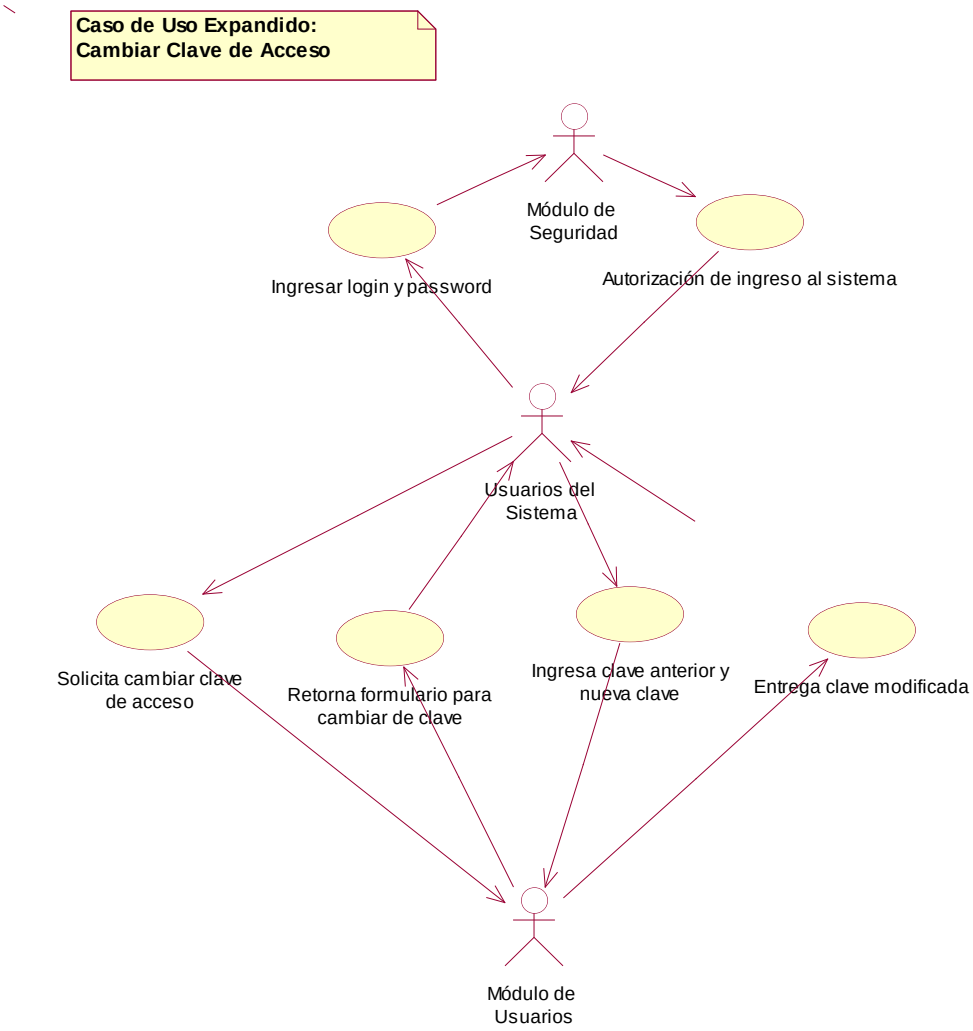
No cancela el proceso.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

DIAGRAMA DE COLABORACIÓN





NOMBRE DEL CASO DE USO: CAMBIAR CLAVE DE ACCESO AL SISTEMA.

PROPÓSITO: Reemplazar la clave provisional de acceso al sistema.

ACTOR (Iniciador): Usuarios del Sistema (Administrador, Diagramador, Jefe Departamental, Usuario General)

TIPO: Primario Real.

VISION GENERAL: Los usuarios del sistema ingresan el login y el password, el sistema autoriza ingreso al sistema, los usuarios del sistema solicitan cambiar clave de acceso, el sistema retorna el formulario de cambio de clave, los usuarios

del sistema ingresan la clave anterior y la nueva clave, el sistema entrega clave modificada, termina caso de uso

CURSO TIPICO DE EVENTOS

ACTOR	SISTEMA
1.Ingresar login y password	2. Autorizar ingreso al sistema.
3. Solicitar cambiar clave de acceso.	4. Retornar el formulario de cambio de clave.
5. Ingresar la clave anterior y la nueva clave.	6. Entregar clave modificada.

CURSOS ALTERNATIVOS:

2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

2* Login y/o password incorrecto. “Termina caso de uso”.

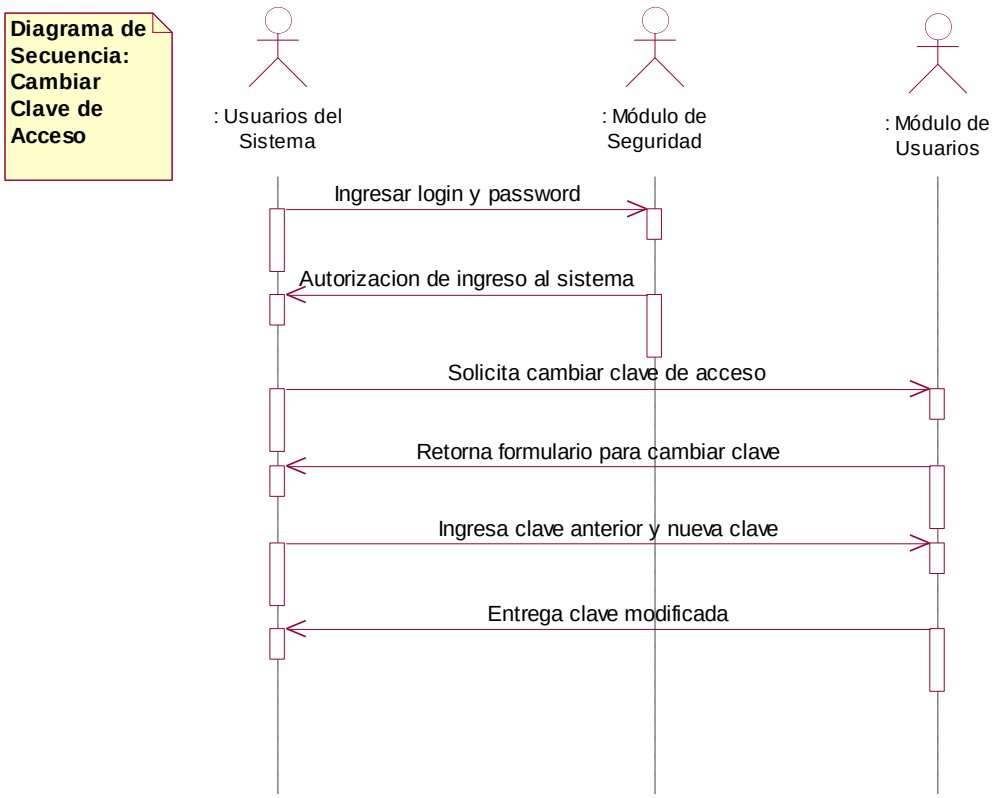
4* No presenta el formulario de cambio de clave. “Termina caso de uso”.

6* No presenta mensaje de clave modificada. ”Termina caso de uso”.

6* No modifica la clave. Clave anterior incorrecta. “Termina caso de uso”.

6* No modifica la clave. Confirmación de nueva clave es errónea. “Termina caso de uso”.

DIAGRAMA DE SECUENCIA



CONTRATO DE OPERACIÓN

NOMBRE DEL CONTRATO: Ingresar login y password.

PROPÓSITO: Acceder al sistema para el cambiar la clave de acceso al sistema.

TIPO: Sistema.

SALIDA: Autorización de ingreso al sistema.

EXCEPCIONES: No existe formulario de ingreso de login y password.
Login y/o password incorrecto.

PRE-CONDICIONES: Relación usuario – interfaz.

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Solicitar cambiar clave de acceso.

PROPÓSITO: Presentar el formulario de sustitución de la clave provisional de acceso al sistema.

SALIDA: Formulario de cambio de clave de acceso.

EXCEPCIONES: No presenta el formulario de cambio de clave.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Ingresar clave anterior y nueva clave.

PROPÓSITO: Ingresar la nueva clave de acceso al sistema.

TIPO: Sistema.

SALIDA: Mensaje. Clave modificada exitosamente.

EXCEPCIONES: No presenta mensaje de clave modificada.

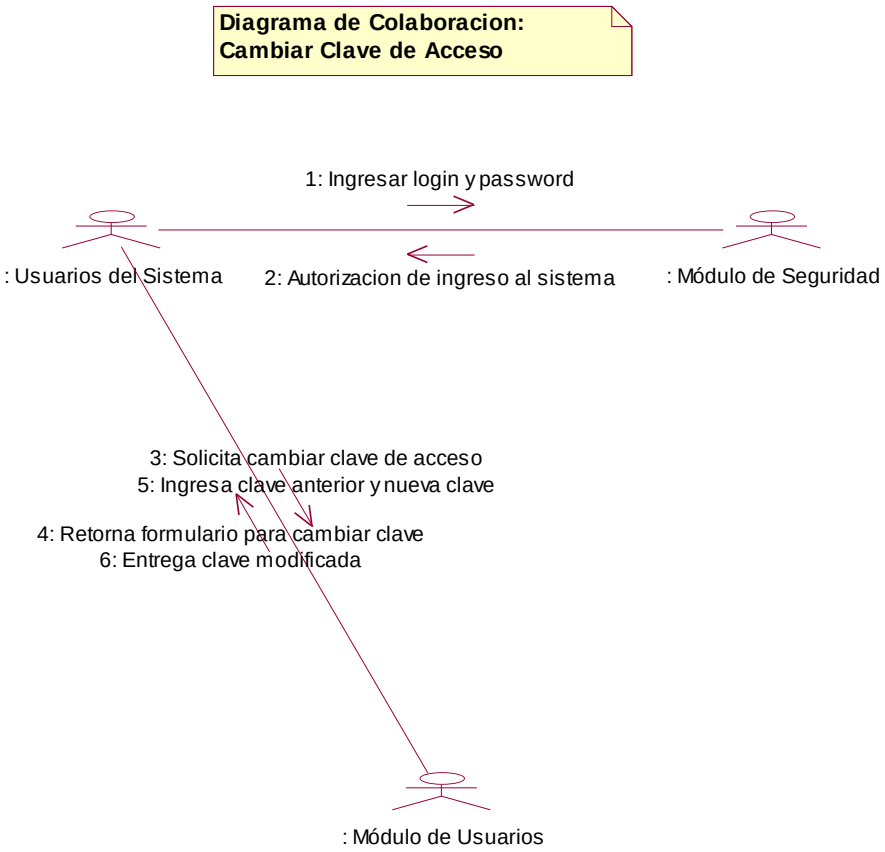
No modifica la clave. Clave anterior incorrecta.

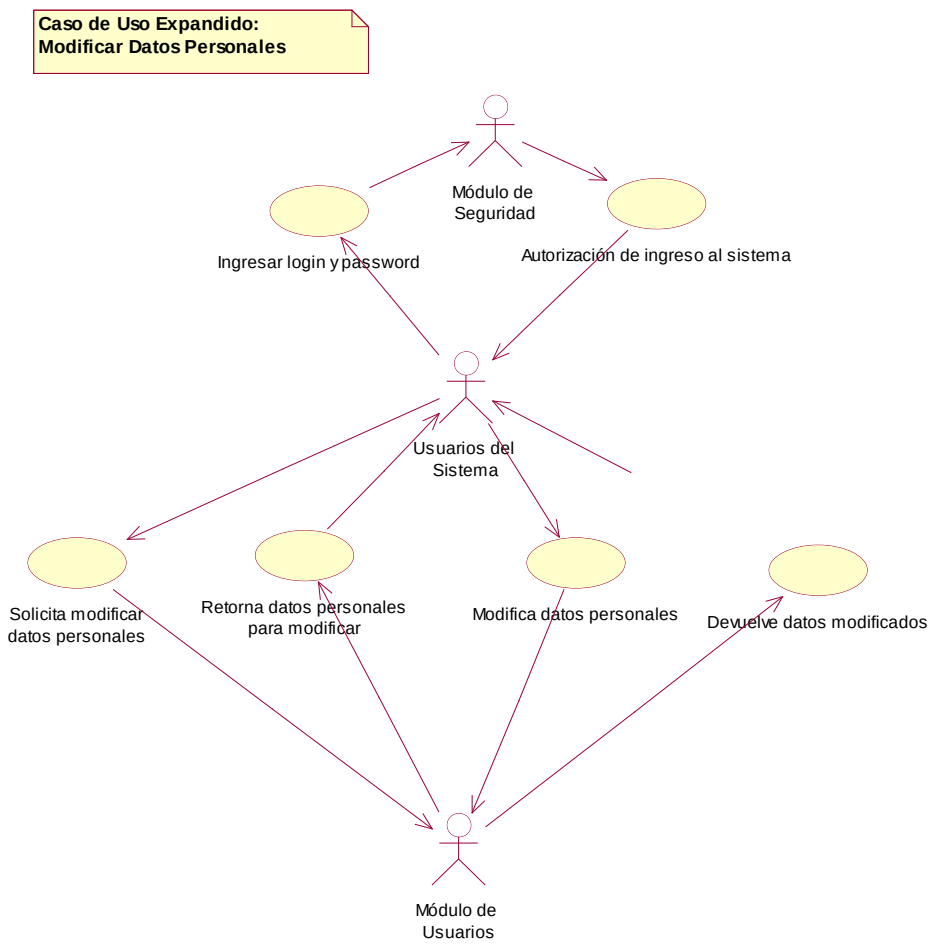
No modifica la clave. Confirmación de nueva clave es errónea.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

DIAGRAMA DE COLABORACIÓN





NOMBRE DEL CASO DE USO: MODIFICAR DATOS PERSONALES.

PROPÓSITO: Cambiar datos personales de los usuarios del sistema.

ACTOR (Iniciador): Usuarios del Sistema (Administrador, Diagramador, Jefe Departamental, Usuario General).

TIPO: Primario Real.

VISION GENERAL: Los usuarios del sistema ingresan su login y password, el sistema autoriza el ingreso al sistema, los usuarios del sistema solicitan el formulario de sus datos personales para modificar, el sistema retorna el formulario con los datos personales para modificar, los usuarios del sistema modifican sus datos personales, el sistema devuelve los datos personales modificados termina el caso de uso.

CURSO TIPICO DE EVENTOS

ACTOR	SISTEMA
1. Ingresar login y password.	2. Autorizar ingreso al sistema.
3. Solicitar formulario de sus datos personales.	4. Retornar el formulario con los datos personales para modificar.
5.Modificar sus datos personales	6. Devolver los datos personales modificados

CURSOS ALTERNATIVOS:

2* Login y/o password no válidos. “Termina caso de uso”.

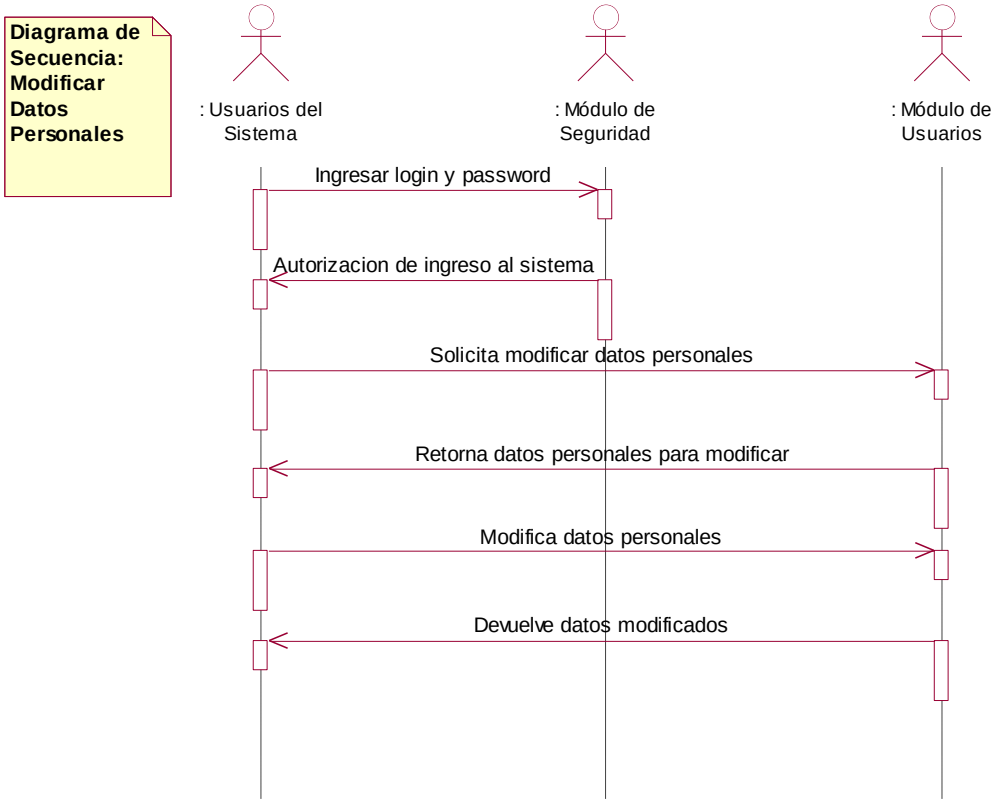
2* No existe formulario de ingreso de login y password. “Termina caso de uso”.

4* No presenta formulario con datos personales del usuario. “Termina caso de uso”.

6* No modifica datos personales de usuario. ”Termina caso de uso”.

6* No presenta mensaje de datos personales modificados. “Termina caso de uso”.

DIAGRAMA DE SECUENCIA



CONTRATO DE OPERACIÓN

NOMBRE DEL CONTRATO: Ingresar login y password.

PROPÓSITO: Acceder al sistema para que un usuario modifique sus datos personales (transacción).

TIPO: Sistema.

SALIDA: Autorización de ingreso al sistema.

EXCEPCIONES: Login y/o password no válidos.
No existe formulario de ingreso de login y password.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Solicitar formulario de sus datos personales.

PROPÓSITO: Obtener los datos personales para modificarlos (transacción).

TIPO: Sistema.

SALIDA: Formulario con los datos personales del usuario para modificarlo.

EXCEPCIONES: No presenta formulario con datos personales del usuario.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

NOMBRE DEL CONTRATO: Modificar sus datos personales.

PROPÓSITO: Cambiar datos personales de usuario.

TIPO: Sistema.

SALIDA: Mensaje de datos personales actualizados correctamente.

EXCEPCIONES: No modifica datos personales de usuario.

No presenta mensaje de datos personales modificados.

PRE-CONDICIONES: Relación usuario - interfaz

POST-CONDICIONES:

DIAGRAMA DE COLABORACIÓN

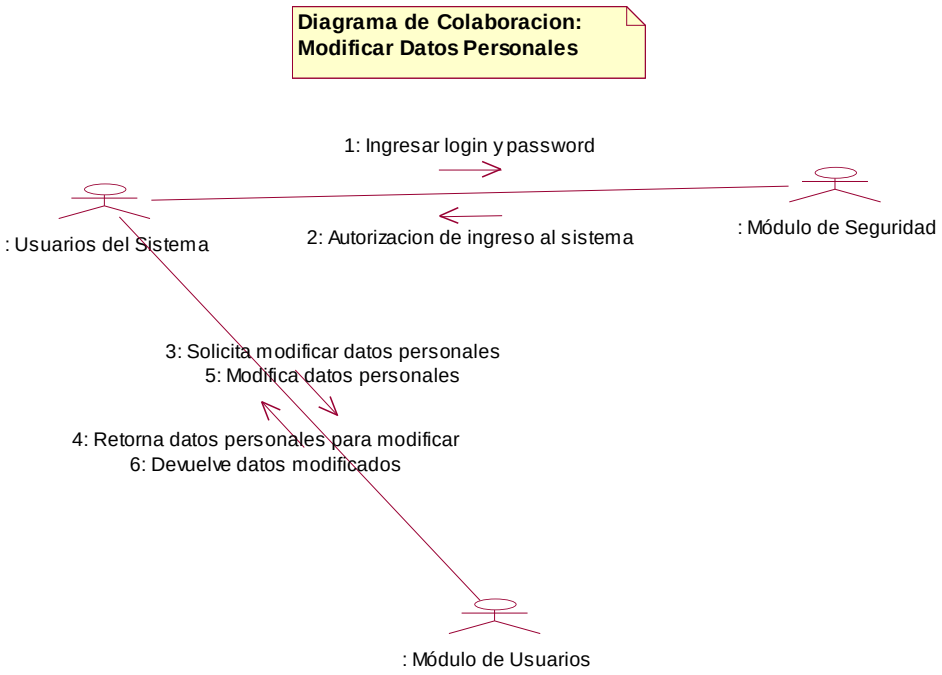
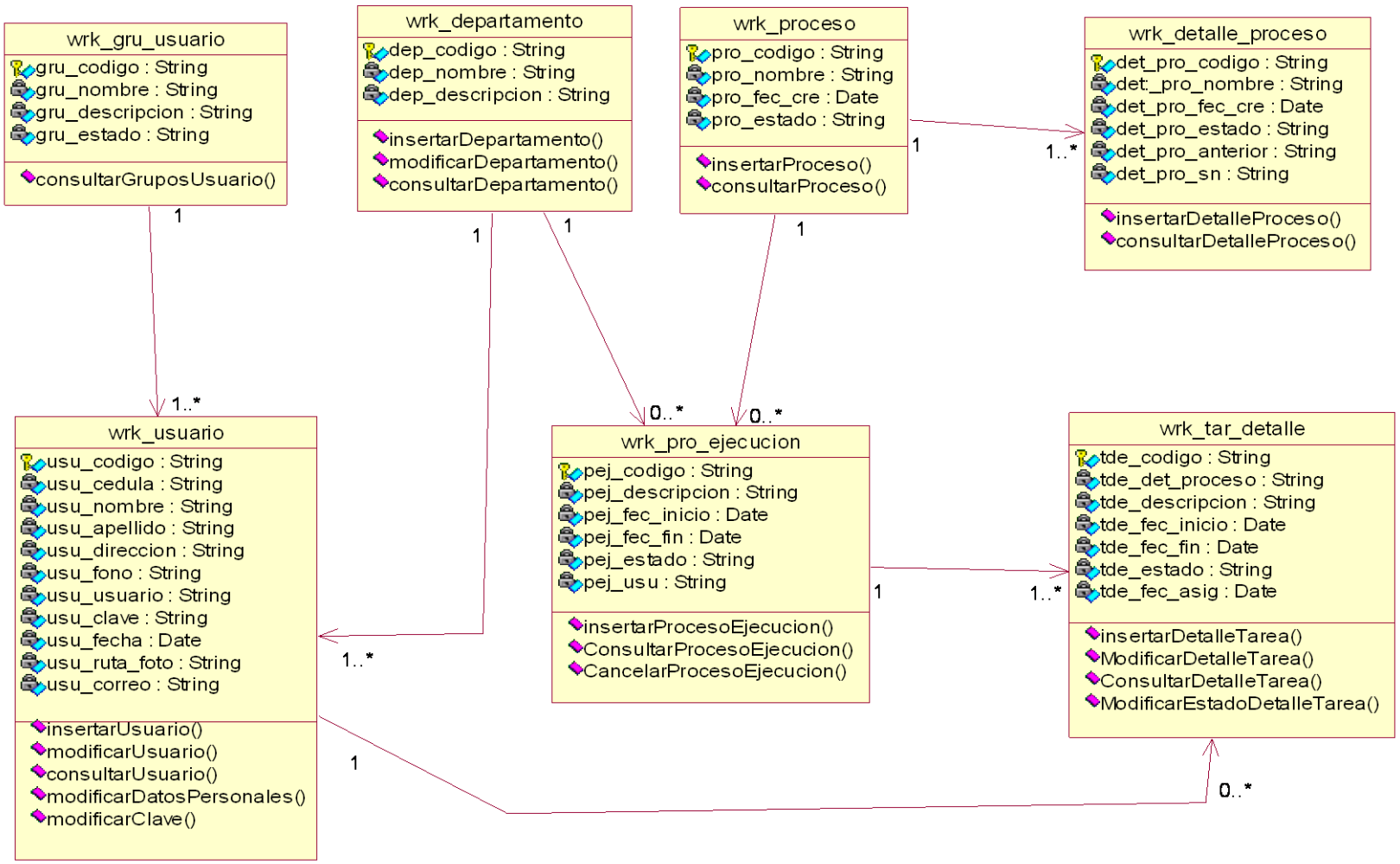
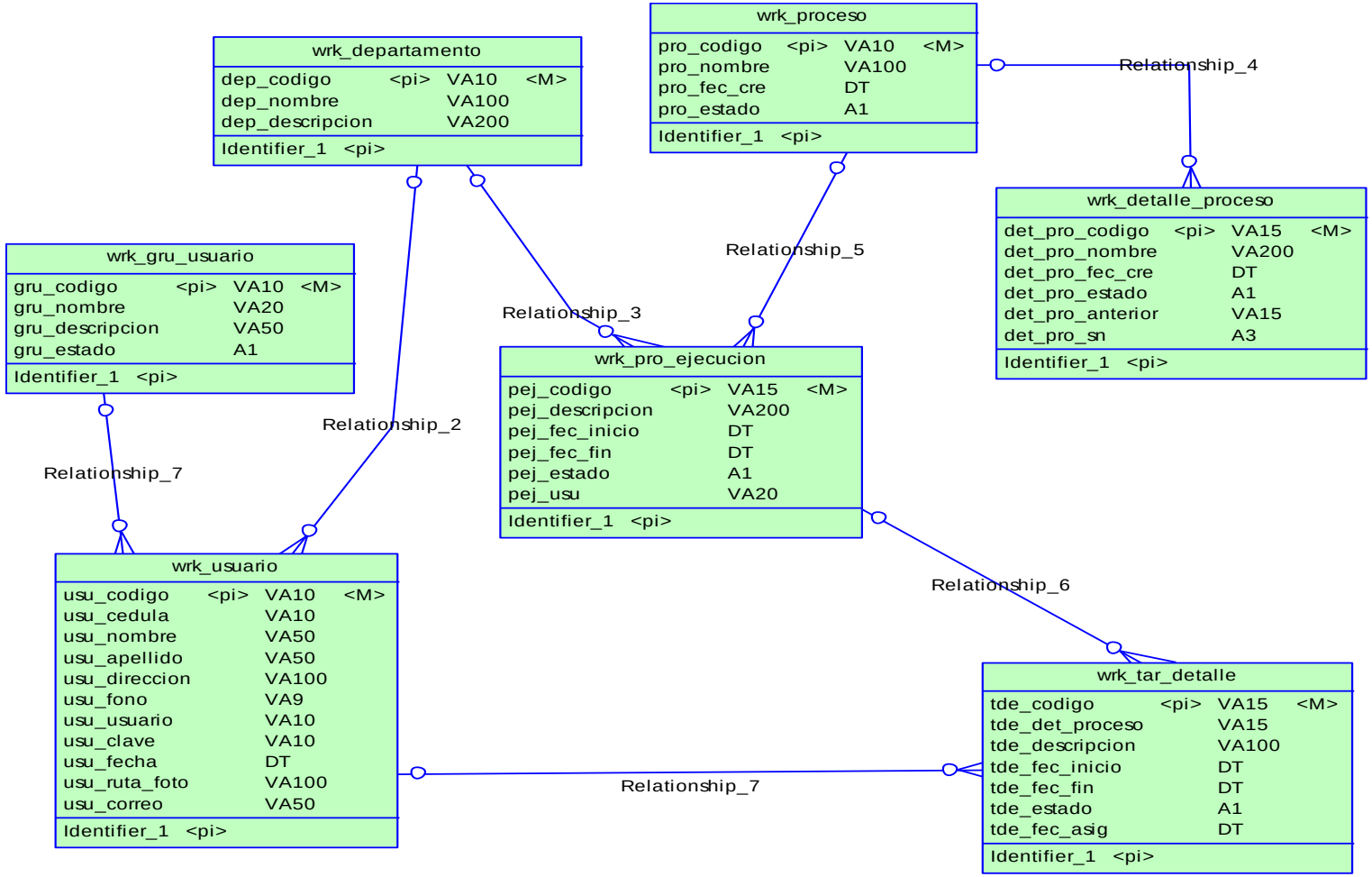


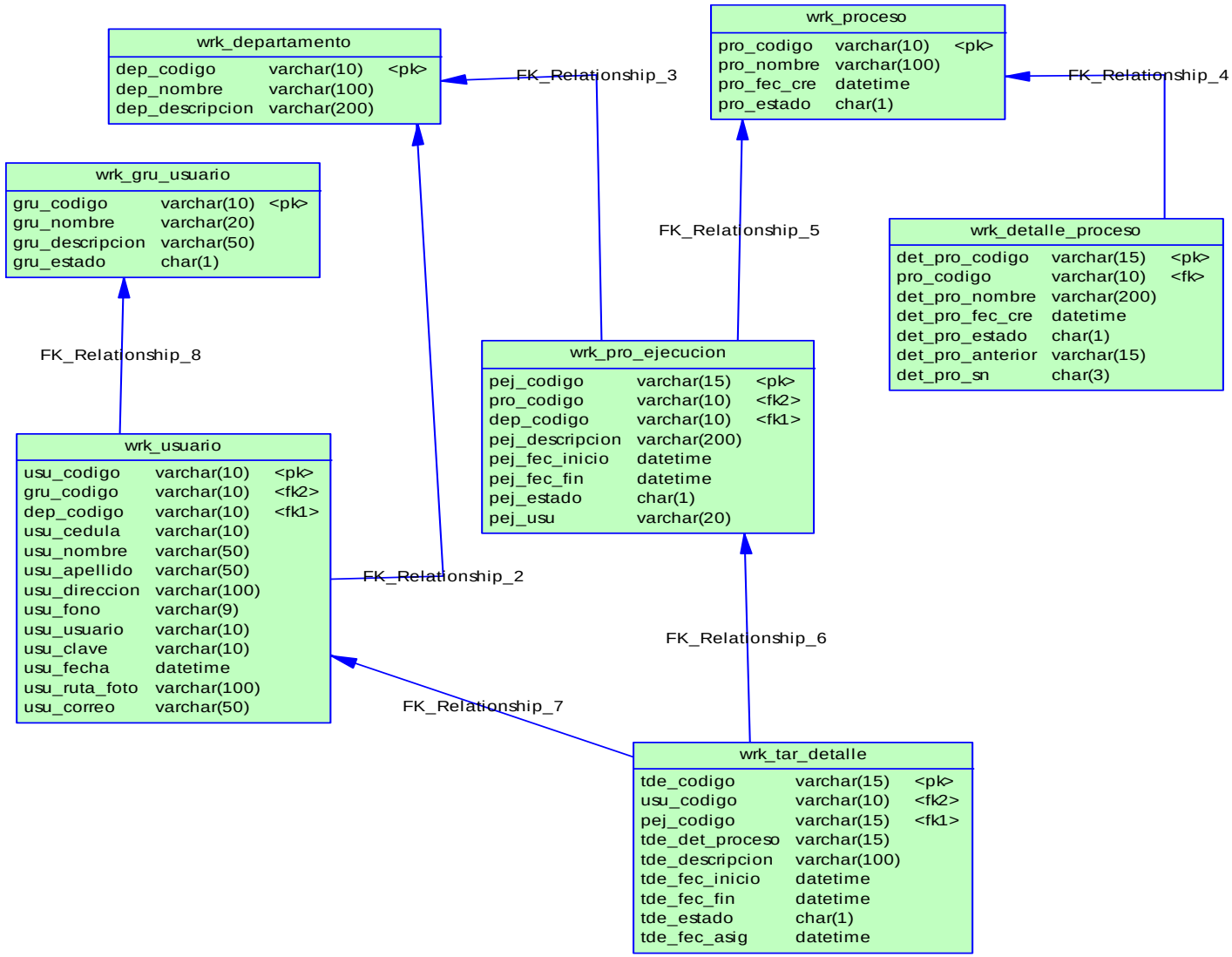
DIAGRAMA DE CLASES



MODELO CONCEPTUAL



MODELO FISICO



Generación del Script de la base de datos “workflow”

Base de datos ***workflow*** ejecutándose en localhost

```
# phpMyAdmin MySQL-Dump
# version 2.3.2
# http://www.phpmyadmin.net/ (download page)
#
# servidor: localhost
# Tiempo de generación: 09-11-2005 a las 13:32:06
# Versión del servidor: 3.23.47
# Versión de PHP: 4.1.1
# Base de datos : `workflow`
# --------------------------------------------------------

#
# Estructura de tabla para la tabla `wrk_departamento`
#

CREATE TABLE wrk_departamento (
  DEP_CODIGO varchar(10) NOT NULL default '',
  DEP_NOMBRE varchar(100) default NULL,
  DEP_DESCRIPCION varchar(200) default NULL,
  PRIMARY KEY  (DEP_CODIGO)
) TYPE=MyISAM;
# --------------------------------------------------------

#
# Estructura de tabla para la tabla `wrk_detalle_proceso`
#

CREATE TABLE wrk_detalle_proceso (
  DET_PRO_CODIGO varchar(15) NOT NULL default '',
```

```
  PRO_CODIGO varchar(10) default NULL,
  DET_PRO_NOMBRE varchar(200) default NULL,
  DET_PRO_FEC_CRE datetime default NULL,
  DET_PRO_FEC_MOD datetime default NULL,
  DET_PRO_ESTADO char(1) default NULL,
  DET_PRO_ANTERIOR varchar(15) default NULL,
  DET_PRO_SN char(3) default NULL,
  PRIMARY KEY  (DET_PRO_CODIGO),
  KEY PROCESO_DETALLE_PRO_FK (PRO_CODIGO)
) TYPE=MyISAM;
# --------------------------------------------------------

#
# Estructura de tabla para la tabla `wrk_gru_usuario`
#

CREATE TABLE wrk_gru_usuario (
  GRU_CODIGO varchar(10) NOT NULL default '',
  GRU_NOMBRE varchar(20) default NULL,
  GRU_DESCRIPCION varchar(50) default NULL,
  GRU_ESTADO char(1) default NULL,
  PRIMARY KEY  (GRU_CODIGO)
) TYPE=MyISAM;
# --------------------------------------------------------

#
# Estructura de tabla para la tabla `wrk_pro_ejecucion`
#

CREATE TABLE wrk_pro_ejecucion (
  PEJ_CODIGO varchar(15) NOT NULL default '',
  PRO_CODIGO varchar(10) default NULL,
```

```
  DEP_CODIGO varchar(10) default NULL,
  PEJ_DESCRIPCION varchar(200) default NULL,
  PEJ_FEC_INICIO datetime default NULL,
  PEJ_FEC_FIN datetime default NULL,
  PEJ_ESTADO char(1) default NULL,
  PEJ_USU varchar(20) default NULL,
  PRIMARY KEY  (PEJ_CODIGO),
  KEY PROCESO_PRO_EJECUCION_FK (PRO_CODIGO),
  KEY DEPARTAMENTO_PRO_EJECUCION_FK (DEP_CODIGO)
) TYPE=MyISAM;
# --------------------------------------------------------

#
# Estructura de tabla para la tabla `wrk_proceso`
#

CREATE TABLE wrk_proceso (
  PRO_CODIGO varchar(10) NOT NULL default '',
  PRO_NOMBRE varchar(100) default NULL,
  PRO_FEC_CRE datetime default NULL,
  PRO_FEC_MOD datetime default NULL,
  PRO_ESTADO char(1) default NULL,
  PRIMARY KEY  (PRO_CODIGO)
) TYPE=MyISAM;
# --------------------------------------------------------

#
# Estructura de tabla para la tabla `wrk_tar_detalle`
#

CREATE TABLE wrk_tar_detalle (
  TDE_CODIGO varchar(15) NOT NULL default '',
```

```
  PEJ_CODIGO varchar(15) default NULL,
  USU_CODIGO varchar(10) default NULL,
  TDE_DET_PROCESO varchar(15) default NULL,
  TDE_DESCRIPCION varchar(100) default NULL,
  TDE_FEC_INICIO datetime default NULL,
  TDE_FEC_FIN datetime default NULL,
  TDE_ESTADO char(1) default NULL,
  TDE_FEC_ASIG datetime default NULL,
  PRIMARY KEY  (TDE_CODIGO),
  KEY PRO_EJECUCION_TAREAS_PROCESO_FK (PEJ_CODIGO),
  KEY USUARIO_TAREAS_PROCESO_FK (USU_CODIGO)
) TYPE=MyISAM;
# --------------------------------------------------------

#
# Estructura de tabla para la tabla `wrk_usuario`
#

CREATE TABLE wrk_usuario (
  USU_CODIGO varchar(10) NOT NULL default '',
  DEP_CODIGO varchar(10) default NULL,
  GRU_CODIGO varchar(10) default NULL,
  USU_CEDULA varchar(10) default NULL,
  USU_NOMBRE varchar(50) default NULL,
  USU_APELLIDO varchar(50) default NULL,
  USU_DIRECCION varchar(100) default NULL,
  USU_FONO varchar(9) default NULL,
  USU_USUARIO varchar(10) default NULL,
  USU_CLAVE varchar(10) default NULL,
  USU_FECHA datetime default NULL,
  USU_RUTA_FOTO varchar(100) default NULL,
  USU_CORREO varchar(50) default NULL,
```

```
  PRIMARY KEY  (USU_CODIGO),
  KEY GRU_USUARIO_USUARIO_FK (GRU_CODIGO),
  KEY DEPARTAMENTO_USURIO_FK (DEP_CODIGO)
) TYPE=MyISAM;
```

4.3.- FASE DE DESARROLLO E IMPLEMENTACION DEL SISTEMA

4.3.1.- INTRODUCCIÓN

El sistema WORKFLOW 1.0 permitirá gestionar de forma automatizada, los procesos y flujo de actividades que componen una empresa, esta conformado por dos elementos principales:

- El Graficador, es la herramienta base que permitirá diseñar y crear los procesos que se realizan dentro de la Institución.

- EL sitio Web, a través del cual se realizará la ejecución y el monitoreo.

Mediante este sistema se pretende simplificar y armonizar los procesos existentes en la empresa. También proveerá de información para la toma oportuna de decisiones sobre estructura de procesos y organizativa de la entidad.

4.3.1.1.- Selección de las herramientas de Implementación

4.3.1.1.1.- Lenguaje Programación

El Sistema WORKFLOW 1.0 se desarrollo en los siguientes leguajes de Programación:

Diagramador: Para el Diagramador se utilizo el **LENGUAJE JAVA** gracias a que brinda una programación orientada a objetos más flexible que otros lenguajes con C++, también proporciona flexibilidad, modularidad y rehusabilidad. Para el Sistema WORKLOW 1.0 se ha trabajado basándose en el AWT (Abstract Window ToolKit) que es un paquete en el que se encuentran clases capaces de crear componentes de GUI es decir clases desarrolladas para crear objetos visuales sobre los cuales pueden actuar los usuarios, mediante el empleo del ratón y el teclado para comunicarse con el programa.

Monitoreo: Para el desarrollo del Monitoreo y ejecución web se trabajo con el **Lenguaje de programación PHP** ya que permite la construcción de Webs con independencia de la Base de Datos y del servidor Web, válida para cualquier plataforma. El objetivo final es conseguir la integración de las paginas HTML con aplicaciones que corran en el servidor como procesos integrados en el mismo, y no como un proceso separado. Igualmente interesa que dichas aplicaciones sean totalmente independientes del navegador (lo que no ocurría con otros lenguajes basados en scripts, como JavaScript o VisualBasic Script), independientes de la plataforma y de la Base de Datos.

4.3.1.2.- Sistema Operativo

4.3.1.2.1.- Windows XP

Microsoft Windows XP ha sido seleccionado como el Sistema Operativo del Servidor ya que posee avanzadas utilidades de administración que lo hacen un poderoso Sistema Operativo de Red.

En grandes entornos de redes, Windows XP da soporte a los varios entornos como:

1. Servidor de bases de datos
2. Servidor de mensajería.
3. Servidor de archivos y de impresión.
4. Servidor de comunicaciones.
5. Servidor WEB.
6. Soporte a múltiples plataformas.
7. Administración Centralizada

Además de las características señaladas cabe recalcar que Windows XP es uno de los sistemas operativos para red más utilizados.

4.3.1.2.2.- Windows 9x en adelante

La familia de Microsoft Windows 9x o superior puede ser utilizado como alternativa para el Sistema Operativo del cliente, todo depende de su elección y familiarización del mismo.

4.3.1.3.- Servidor Web

4.3.1.3.1.- Apache 1.3.23

Apache está diseñado para ser un servidor web potente y flexible que pueda funcionar en la más amplia variedad de plataformas y entornos.

Las diferentes plataformas y los diferentes entornos, hacen que a menudo sean necesarias diferentes características o funcionalidades, o que una misma característica o funcionalidad sea implementada de diferente manera para obtener una mayor eficiencia. Apache se ha adaptado siempre a una gran variedad de entornos a través de su diseño modular. Este diseño permite a los administradores de sitios web elegir que funcionalidades van a ser incluidas en el servidor seleccionando que módulos se van a usar, ya sea al compilar o al ejecutar el servidor. Por sus potentes funcionalidades se ha seleccionado a APACHE como el Servidor Web de la presente Tesis.

4.3.1.4.- Servidor de Base de Datos

4.3.1.4.1.- MySQL 3.23.21-beta

MySQL es un servidor de base de datos que posee una arquitectura cliente servidor multi-hebra, con un robusto soporte de SQL (Structured Query Language). Este administrador de bases de datos relacionales es un software de código abierto que posee licencia GPL a partir de la versión 3.23.12

Fue escogida para el desarrollo de la Aplicación Web de esta tesis por las siguientes razones.

- Consume pocos recursos tanto de memoria como Procesador.
- Incluye API (clientes) para C, C++, PHP, Pitón, Java, Perl, ODBC

- Es muti-hebra para consultas paralelas, es decir cada conexión tiene su propio hilo de modo que ningún hilo tiene que esperar por otro a menos que un hilo este modificando la tabla y otro hilo quiera acceder a la misma.
- No existe límite para el tamaño de los registro
- Trabaja en distintos Sistemas Operativos y plataformas
- Los password viajan encriptados en la red.

4.3.1.5.- Herramientas de Programación

4.3.1.5.1.- Jbuilder X Enterprise

Jbuilder es una herramienta para el desarrollo impulsada por la corporación de Software Borland, la misma que ha sido probado en Solaris, Linux, y Windows XP, NT, y 2000. Jbuilder posee la versión 1.3 de JDK que tiene ventajas de depuración para mejorar la ejecución en el lado del cliente. Esta herramienta de cuarta generación ha facilitado la realización del Diagramador de la presente Tesis ya que posee la característica de recompilar y reutilizar clases existentes que permitan rapidez en el desarrollo.

4.3.1.5.2.- Dreamwaver MX

Se ha elegido como Herramienta web a DreamWeaver MX porque combina facilidad y potencia en un entorno de desarrollo integrado para los sitios web, HTML, XHTML, **PHP** ASP, ASP.NET, o JSP. El producto permite un control

completo sobre el código y el diseño con la precisión de las herramientas de presentación y las potentes características de codificación como sugerencias de código, editor de etiquetas, codificación del color ampliable, selector de etiquetas, fragmentos y validación de código.

4.3.1.5.3.- Instalación de las Herramientas de Programación

Instalación de JBuilder X. Ver ANEXO 1

Instalación de Macromedia Dreamweaver MX. Ver ANEXO 2

4.3.1.6.- Arquitectura de la Aplicación

El Sistema WORKFLOW se implementa en una arquitectura de tres capas:

CAPA 1: PRESENTACIÓN
Navegador Web
Internet Explore, Mozilla, Netscape, etc

CAPA 2: LÓGICA
Contenedor de Páginas Web
Apache 1.3.23

CAPA 3: DATOS
Servidor de Base de Datos
MySQL 3.23.21-beta

La comunicación entre capas se realiza con HTTP

Nota: Para el Diagramador de procesos se ha utilizado una arquitectura local que alimentará a la Base de Datos

4.3.2.- MANUAL DE PROGRAMACIÓN

Ver **ANEXO 3**

4.3.3.- MANUAL DE INSTALACION DEL SISTEMA WORKFLOW 1.0

Ver **ANEXO 4**

4.3.4.- MANUAL DE USUARIO

Ver **ANEXO 5**

V. CONCLUSIONES Y RECOMENDACIONES

5.1.- CONCLUSIÓN

Se ha comprobado que la metodología UML a más de ser una herramienta útil para el modelado de Sistemas de Gestión también puede ser usado para realizar sistemas que tienen como punto partida la Tecnología Workflow, esto tomando en cuenta que UML es un lenguaje de modelamiento que apareció hace unos años atrás y no es muy usada en la actualidad.

Con respecto a las herramientas que se usó para la programación del Sistema, se concluye que el lenguaje de programación JAVA es uno de los mejores lenguajes para realizar este tipo de sistemas de tecnología Workflow sobre todo para la elaboración del motor de Workflow puesto que es un lenguaje orientado a objetos; en cuanto al sistema de seguimiento que es el complemento del motor Workflow, concluimos que con el lenguaje PHP el resultado final de las interfases de usuario son mucho más amigables y de fácil uso y comprensión para los usuarios, ya que esta desarrollado en un entorno web.

En cuanto a Workflow podemos acotar lo siguiente:

El énfasis de la necesidad de un sistema de workflow, se vuelca en el incremento a todos los niveles de: **Productividad, eficiencia y servicio al cliente.**

Un sistema de workflow busca la automatización de las políticas y procedimientos de procesos básicos de negocio, por medio de asociar individuos y grupos de trabajo con tareas estructuradas y no estructuradas para manejar automáticamente los trabajos que se realizan dentro de una organización, haciendo posible la cooperación de distintas personas mejorando la eficiencia global y se eliminan los cuellos de botella.

La tecnología workflow ha ayudado a muchas empresas a simplificar y optimizar procesos complejos, mejorando la atención a sus clientes y reduciendo costes. Workflow es la automatización de un proceso de negocio durante el cual “documentos”, “información” y “tareas” son pasados de un participante a otro acorde a un conjunto de reglas procedimentales.

Un sistema de esta naturaleza provee importantes beneficios. Las tareas de los trabajadores se realizan más fácilmente y la organización conoce y controla de forma inmediata las tareas que se están llevando a cabo.

Cada vez más, las empresas identifican la necesidad creciente de automatizar, sin codificar, los flujos de trabajo con el fin de lograr:

- Ciclos más cortos
- Mejor Calidad y Servicio al Cliente
- Mayor coordinación
- Mayor cooperación
- Mejor comunicación
- Mejora continua en los procesos
- Eliminación de procesos innecesarios a través de la automatización de los flujos de información
- Coordinación, Comunicación y Cooperación independiente de la hora y situación geográfica
- Mayor agilidad y flexibilidad de la informática que soporte al negocio

Otra de las grandes ventajas de implantar una herramienta workflow es la estandarización de los procesos y un mayor control sobre los mismos, lo que permite reducir los costes de operación, elevar la eficiencia de los empleados y a la vez ofrecer un mejor servicio al cliente.

Workflow se relaciona con la automatización de los procedimientos donde los documentos, la información o tareas son pasadas entre los participantes del

sistema de acuerdo a un conjunto de reglas previamente establecidas. El fin de lo anterior es llegar a culminar una meta común impuesta por la empresa.

Podemos ver al Workflow como un conjunto de métodos y tecnologías que nos ofrece las facilidades para modelar y gestionar los diversos procesos que ocurren dentro de una empresa. El Workflow es el último, de una gran línea de facilidades propuestas en respuesta de las exigencias de las organizaciones. Las cuales apuntan a poder reaccionar tan rápido como sea posible ante la frenética demanda de la competencia.

Por lo tanto, surge la necesidad de remplazar las actividades manuales por actividades automáticas, buscando tener un mayor control y coordinación sobre toda la información, logrando automatizar ciertas tareas, que antes se realizaban manualmente. Por esto podemos hablar de un Workflow Automatizado.

Se podría decir que la tecnología de Workflow se basa sobre la idea de que algunas cosas son realizadas más efectivamente por las computadoras que por las personas. Los humanos somos buenos para tomar decisiones, innovar, identificar hechos inesperados. Pero usualmente no somos eficientes en actividades tales como: buscar un documento entre cientos; tener presentes los vencimientos de las tareas que se tienen que realizar dentro de ciertos plazos; así como también el asegurarse de que el trabajo terminado pase de un lugar a otro respetando la secuencia definida. Es por esta razón que un sistema de Workflow debe tener lo siguiente:

1. **Diseño gráfico de procesos**: herramienta gráfica para crear los mapas de procesos que definen el flujo del trabajo y las tareas desde el comienzo hasta el final.
2. **Habilidad de asignar "roles" o "funciones de trabajo"** para que el diseño del flujo de trabajo no deba ser cambiado debido a la movilidad laboral, vacaciones, bajas, sustituciones, reestructuraciones…
3. **Manejo de excepciones:** manejo de las siempre presentes "excepciones a la regla". Por ejemplo, la característica de reasignar una tarea de un

usuario a otro si el usuario está ausente o por causa de un daño en su computador.

4. **Monitorización:** Este punto es especial importancia para los supervisores que podrán controlar los cuellos de botella y ver o prever necesidades de ampliación o reestructuración de recursos.
5. **Rendimiento:** Para la medición del tiempo de cada proceso.
6. **Comunicación:** Los usuarios son notificados con sus nuevas tareas.
7. **Anexo de documentos:** A través de un link al correo electrónico se pueden enviar documentos entre los participantes de cada proceso.

Los sistemas de gestión Workflow proporcionan un marco ideal para la automatización y apoyo al desarrollo de los procesos, ofreciendo una serie de ventajas competitivas como:

1. Modelado formal de procesos para su mejor planificación.
2. Monitorización de los procesos por parte de los gestores.
3. Automatización de la ejecución de procesos.
4. Organizar y planificar sus actividades de acuerdo a la prioridad, estado o fecha de culminación de estas, administrando adecuadamente su tiempo, es decir visualizar los puntos críticos de los procesos - cuellos de botella – para poder regresar a su proceso original, redefinirlo y mejorarlo.
5. Minimizar los tiempos de ejecución de los procesos de su empresa, agilizando el flujo entre una actividad y otra, mejorando su productividad, lo que significa, monitorear en todo momento los estados de cada actividad, realizando seguimiento de auditoria para controlar la productividad de sus empleados y de sus procesos.
6. Mejorar la imagen de la empresa, ayudándolo a resolver fácilmente consultas que antes le hubieran demandado significativo tiempo o que tal vez no habría podido responder.
7. Asignar responsables y suplentes para cada actividad, esto puede ser realizado de dos maneras, en forma manual o en forma automática

mediante un análisis de los cargos de trabajo y control de ausencias del personal.

5.2.- RECOMENDACIÓN

Se recomendaría que se ponga más énfasis en la enseñanza de lenguajes orientados a objetos como JAVA, pues es un lenguaje que en la actualidad es muy utilizado por la mayoría de desarrolladores de software y la demanda de sistemas orientados a objetos es alta.

Así también es recomendable incrementar en la materia el Diseño de Interfases Graficas, la instrucción de herramientas orientas al diseño de entornos web más actualizados, y de esta manera ser más competitivos frente a colegas de otras instituciones que poseen mayores conocimientos acerca de este tipo de herramientas.

Por lo que a Workflow se refiere, se recomendaría que las organizaciones consideren adoptar una solución de Workflow por las siguientes razones:

Eficiencia en los procesos y estandarización de los mismos.

Una reducción de costos dentro de una empresa (la reducción más significativa es debida a la reducción de gente empleada).

La estandarización de los procesos lleva a tener un mayor conocimiento de los mismos, lo que a su vez conduce a obtener una mejor calidad de estos.

A través de la tecnología de Workflow es posible monitorear el estado actual de las tareas así como también observar como evolucionan los planes de trabajo realizados. Permite ver cuales son los embotellamientos dentro del sistema, es decir aquellas tareas o decisiones que están requiriendo de tiempo no planificado y se tornan en tareas o decisiones críticas.

Asignación de tareas a la gente. La asignación de tareas se realiza mediante la definición de roles dentro de la empresa, eliminando la tediosa tarea de asignar los trabajos caso por caso.

Recursos disponibles. Se asegura que los recursos de información (aplicaciones y datos) van a estar disponibles para los trabajadores cuando ellos los requieran.

Hay además muchos aspectos operacionales por los cuales es recomendable contar con una tecnología de Workflow ya que permite automatizar diferentes aspectos del flujo de la información: rutear los trabajos en la secuencia correcta, proveer acceso a datos y documentos, y manejar ciertos aspectos de la ejecución de un proceso.

REFERENCIAS BIBLIOGRAFICAS

Castillo E, Cobo A, Gómez P, Solares C: Java TM; España, 2003.

Kerman P: Edición Macromedia Flash MX; Madrid, 2003.

http://www.aiim.org/wfmc/standards/docs/tc003v11.pdf

http://www.itver.edu.mx

http: www.xdata.com.uy

http://www.extremadura.com.ar

http://www.wfmc.org/

http://www.clikear.com/manuales/uml

http://www.creangel.com/uml/resumen.php

http://www.rational.com/uml

http://www.copypastes.com/tutoriales/categorias.php

http://www.lithium.com.uy(dowloads**, doc: Docflow_CT_2002.pdf+**

http://www.ignside.net/man/php/functions.php

http://www.programacion.com/java

http://www.javahispano.og/articles.articles.action?id=1

http://fcapra.ceit.es/AyudaInf/Index.htm

http://www.monografias.com/trabajos/java/java

ANEXO 1

Ver anexo en ***D:/Manuales/ Manual de Instalación de JBuilderX.doc***

ANEXO 2

Ver anexo en ***D:/Manuales/Manual de Dreamweaver.doc***

ANEXO 3

Ver anexo en ***D:/Manuales/Manual de Programación.doc***

ANEXO 3.1

Ver anexo en ***D:/Manuales/Manual de Programación.doc***

ANEXO 3.2

Ver anexo en ***D:/Manuales/Instalación de MyODBC.doc***

ANEXO 4

Ver anexo en ***D:/Manuales/Manual de Instalación del Sistema Workflow 1.0.doc***

ANEXO 4.1

Ver anexo en ***D:/Manuales/Instalación de Java Runtime Environment.doc***

ANEXO 5

Ver anexo en ***D:/Manuales/Manual de Usuario.doc***

INDICE