MODELANDO LA ESTRUCTURA Y EL COMPORTAMIENTO DEL SOFTWARE

FAVA - Formación en Ambientes Virtuales de Aprendizaje

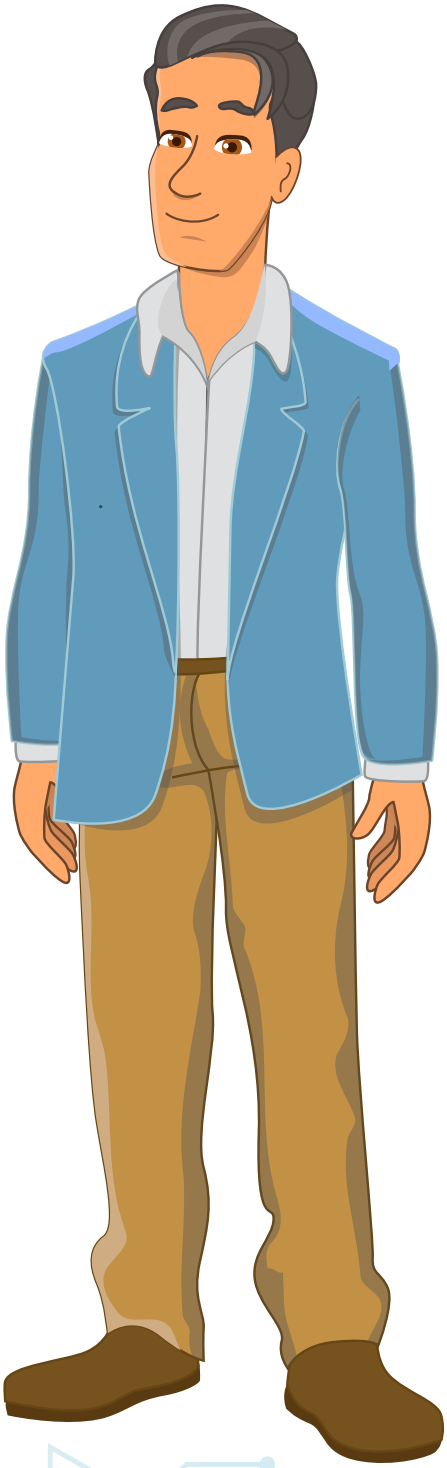
SENA - Servicio Nacional de Aprendizaje.

ESTRUCTURA DE CONTENIDOS

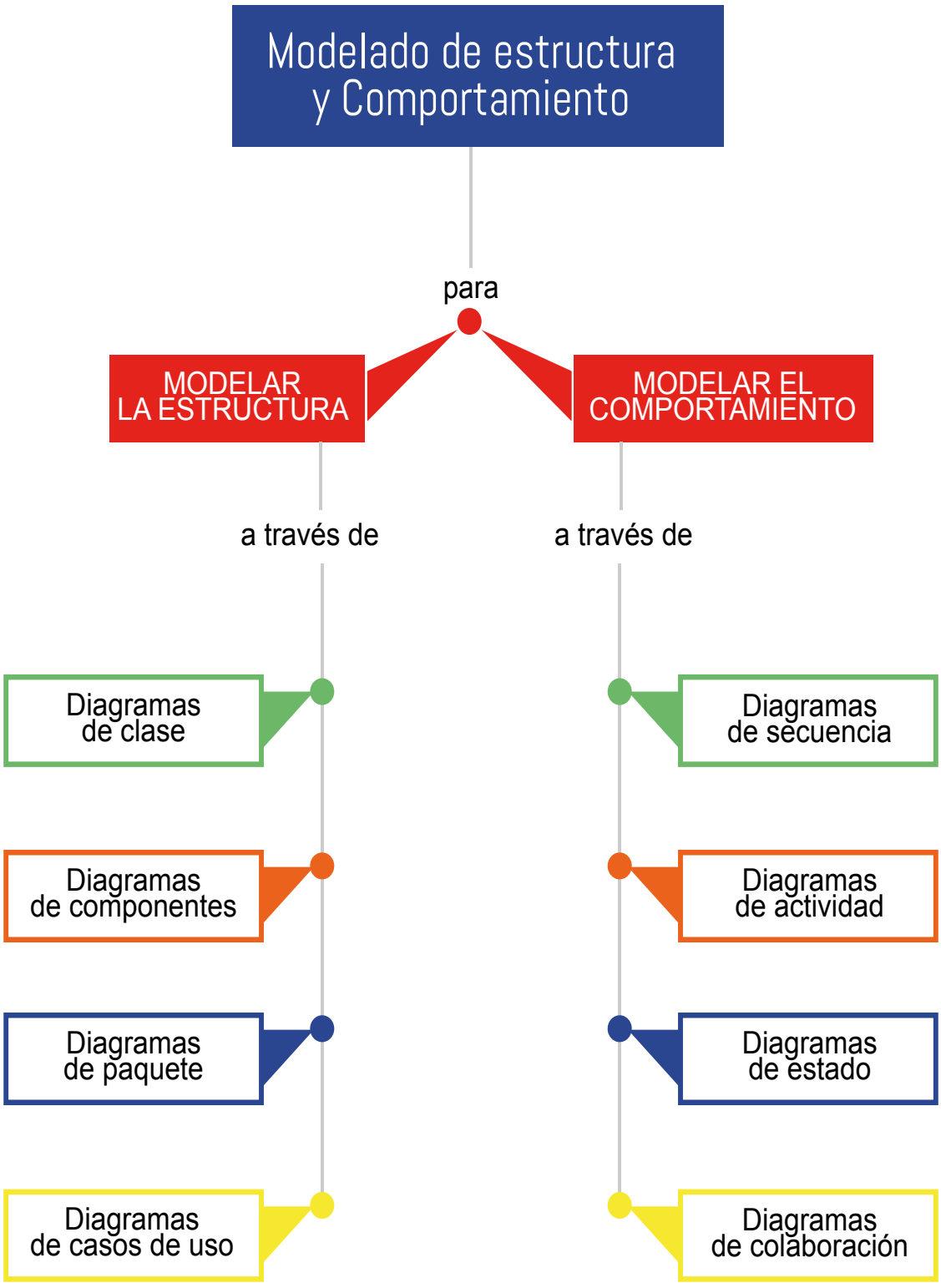
MODELANDO LA ESTRUCTURA Y EL COMPORTAMIENTO DEL SOFTWARE

INTRODUCCIÓN

Dentro de actividades para los analistas se encuentra el modelado conceptual del software. UML permite realizar el modelado conceptual a través de la diagramación de su estructura y su comportamiento, para comprender funcionalmente que es lo que va a realizar el software. Al comprender a través de los diagramas se puede construir un modelado conceptual del software a desarrollar permitiendo clarificar los requerimientos y estimar costos y esfuerzo en el desarrollo del proyecto.



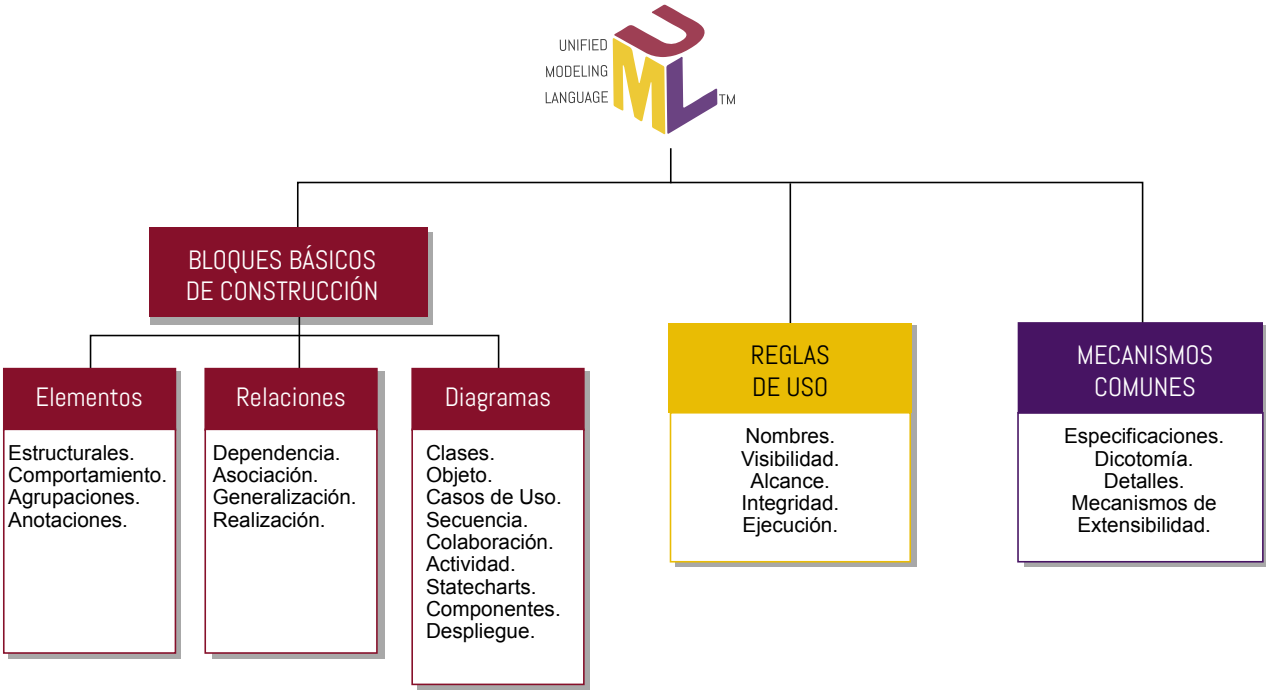
MAPA DE CONTENIDO



DESARROLLO DE CONTENIDOS

1. Generalidades.

De acuerdo con Martin Fowler (2006) en su libro “UML gota a gota”, el lenguaje unificado de modelado tiene tres elementos básicos, los bloques de construcción, las reglas y algunos mecanismos comunes.



1.1. Bloques de construcción.

Los bloques de construcción se dividen en tres partes:

- Elementos, que son las abstracciones de primer nivel.
- Relaciones, que unen a los elementos entre sí.
- Diagramas, que son agrupaciones de elementos.

Existen cuatro tipos de **Elementos** en UML, dependiendo del uso que se haga de ellos:

- Elementos estructurales.
- Elementos de comportamiento.
- Elementos de agrupación
- Elementos de anotación.

Las relaciones, a su vez se dividen para abarcar las posibles interacciones entre elementos que se nos pueden presentar a la hora de modelar usando UML, estas son:

- relaciones de dependencia.
- Relaciones de asociación.
- Relaciones de generalización.
- Relaciones de realización.

Diagramas de Componentes.

Muestra la organización y las dependencias entre un conjunto de componentes. Cubren la vista de la implementación estática y se relacionan con los diagramas de clases ya que en un componente suele tener una o más clases, interfaces o colaboraciones.

Diagramas de Despliegue.

Representan la configuración de los nodos de procesamiento en tiempo de ejecución y los componentes que residen en ellos. Muestran la vista de despliegue estática de una arquitectura y se relacionan con los componentes ya que, por lo común, los nodos contienen uno o más componentes.

Reglas de uso.

UML proporciona un conjunto de reglas que dictan las pautas a la hora de realizar asociaciones entre objetos para poder obtener modelos bien formados, estas son reglas semánticas que afectan a los nombres, al alcance de dichos nombres, a la visibilidad de estos nombres por otros, a la integridad de unos elementos con otros y a la ejecución, o sea la vista dinámica del sistema.

Mecanismos comunes.

UML proporciona una serie de mecanismos comunes que sirven para que cada persona o entidad adapte el lenguaje a sus necesidades, pero dentro de un marco ordenado y siguiendo unas ciertas reglas para que en el trasfondo de la adaptación no se pierda la semántica propia de UML.

Dentro de estos mecanismos están las **especificaciones**, que proporcionan la explicación textual de la sintaxis y semántica de los bloques de construcción.

Otro mecanismo es el de los **adornos** que sirven para conferir a los modelos de más semántica, los adornos son elementos secundarios ya que proporcionan más nivel de detalle, que quizá en un primer momento no sea conveniente descubrir. Las divisiones comunes permiten que los modelos se dividan al menos en un par de formas diferentes

para facilitar la comprensión desde distintos puntos de vista, en primer lugar tenemos la división entre clase y objeto (clase es una abstracción y objeto es una manifestación de esa abstracción), en segundo lugar tenemos la división interfaz / implementación donde la interfaz presenta un contrato (algo que se va a cumplir de una determinada manera) mientras que la implementación es la manera en que se cumple dicho contrato.

Por último, los mecanismos de **extensibilidad** que UML proporciona sirven para evitar posibles problemas que puedan surgir debido a la necesidad de poder representar ciertos matices, por esta razón UML incluye los estereotipos, para poder extender el vocabulario con nuevos bloques de construcción, los valores etiquetados, para extender las propiedades un bloque, y las restricciones, para extender la semántica. De esta manera UML es un lenguaje estándar "abierto-cerrado" siendo posible extender el lenguaje de manera controlada.

2. Modelado de la Estructura.

Como en una obra de ingeniería civil, es importante obtener los planos de la estructura de la obra, que en este caso es un sistema software. UML permite modelar los elementos estructurales, que en su mayoría, son las partes estáticas del modelo y representan cosas que son conceptuales o materiales. Estos elementos corresponden análogamente a los planos del sistema de información.

De acuerdo con Martin Fowler (2006) en su libro "UML gota a gota", dentro de los elementos estructurales se tienen:

Clases.

Una clase es una descripción de un conjunto de objetos que comparten los mismos atributos, operaciones, relaciones y semántica. Una clase implementa una o más interfaces. Gráficamente se representa como un rectángulo que incluye su nombre, sus atributos y sus operaciones.

Figura 1. Clases

Interfaz.

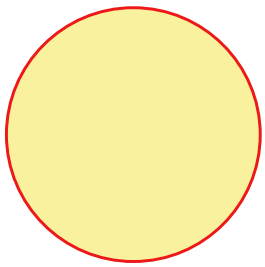


Figura 2. Interfaz

Una interfaz es una colección de operaciones que especifican un servicio de una determinada clase o componente. Una interfaz describe el comportamiento visible externamente de ese elemento, puede mostrar el comportamiento completo o sólo una parte del mismo. Una interfaz describe un conjunto de especificaciones de operaciones (o sea su signatura) pero nunca su implementación. Se representa con un círculo, y rara vez se encuentra aislada sino que más bien conectada a la clase o componente que realiza.

Colaboración.

Define una interacción y es una sociedad de roles y otros elementos que colaboran para proporcionar un comportamiento cooperativo mayor que la suma de los comportamientos de sus elementos. Las colaboraciones tienen una dimensión tanto estructural como de comportamiento. Una misma clase puede participar en diferentes colaboraciones. Las colaboraciones representan la implementación de patrones que forman un sistema. Se representa mediante una elipse con borde discontinuo.

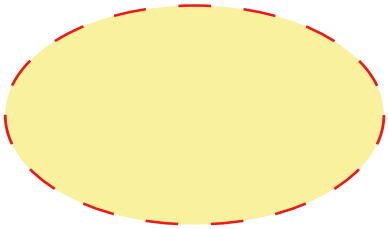


Figura 3. Colaboración

Casos de Uso.

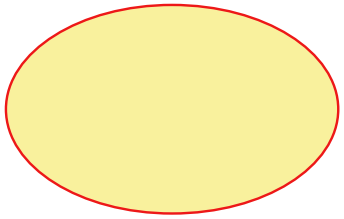


Figura 4. Casos de uso.

Un caso de uso es la descripción de un conjunto de acciones que un sistema ejecuta y que produce un determinado resultado que es de interés para un actor particular. Un caso de uso se utiliza para organizar los aspectos del comportamiento en un modelo. Un caso de uso es realizado por una colaboración. Se representa como en la figura 6, una elipse con borde continuo.

Clase Activa.

Es una clase cuyos objetos tienen uno o más procesos o hilos de ejecución por lo tanto pueden dar lugar a actividades de control. Una clase activa es igual que una clase, excepto que sus objetos representan elementos cuyo comportamiento es concurrente con otros elementos. Se representa igual que una clase, pero con líneas más gruesas.

Figura 5. Clase activa.

Componentes.

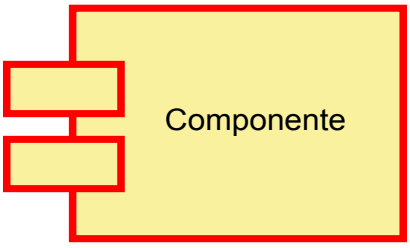


Figura 6. Componentes.

Un componente es una parte física y reemplazable de un sistema que conforma un conjunto de interfaces y proporciona la implementación de dicho conjunto. Un componente representa típicamente el empaquetamiento físico de diferentes elementos lógicos, como clases, interfaces y colaboraciones.

Nodos.

Un nodo es un elemento físico que existe en tiempo de ejecución y representa un recurso computacional, que por lo general, dispone de algo de memoria y con frecuencia de capacidad de procesamiento. Un conjunto de componentes puede residir en un nodo.

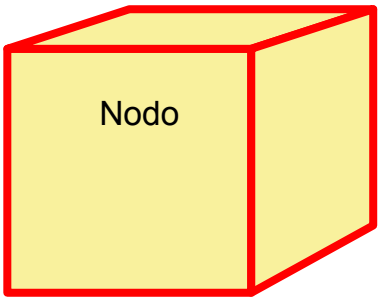


Figura 7. Nodos.

Estos siete elementos vistos son los elementos estructurales básico que se pueden incluir en un modelo UML. Existen variaciones sobre estos elementos básicos, tales como actores, señales, utilidades (tipos de clases), procesos e hilos (tipos de clases activas) y aplicaciones, documentos, archivos, bibliotecas, páginas y tablas (tipos de componentes).

Elementos de agrupación.

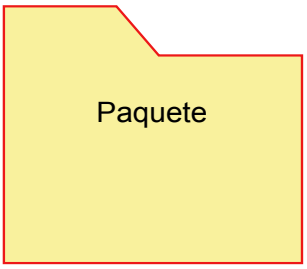


Figura 8. Paquetes.

Forman la parte organizativa de los modelos UML. El principal elemento de agrupación es el paquete, que es un mecanismo de propósito general para organizar elementos en grupos. Los elementos estructurales, los elementos de comportamiento, incluso los propios elementos de agrupación se pueden incluir en un paquete. Un paquete es puramente conceptual (sólo existe en tiempo de desarrollo).

Gráficamente se representa como una carpeta conteniendo normalmente su nombre y, a veces, su contenido.

Siguiendo con el ejemplo del vendedor viajero descrito en el OA de requerimientos como diagramas de casos de uso.

"Se requería que el vendedor genera la factura de venta y la envíe a través del sistema externo. El vendedor puede realizar búsquedas de las facturas, adicionalmente puede consultar cuánto ha vendido y cuánto hace falta para completar la meta por último el vendedor listar los clientes de su ruta correspondiente en el viaje…"

Se identifican los elementos de la estructura entre ellos se tienen:

- vendedores.
- facturas.
- clientes.

El diagrama de la estructura sería así:

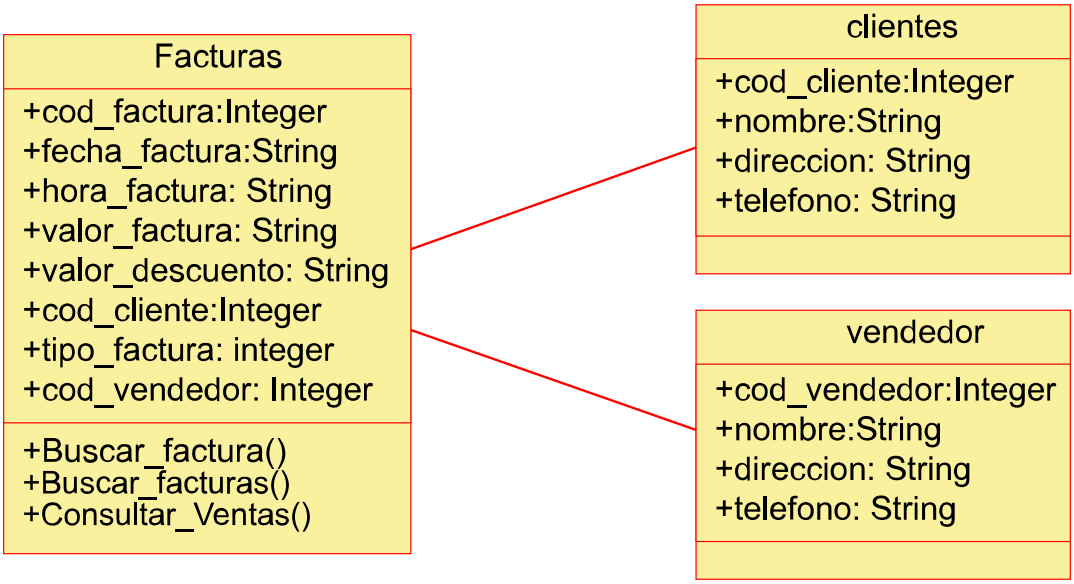


Figura 9. Estructura de clases.

Donde los campos cod_factura, fecha_Factura, hora_factura, valor_factura, valor_descuento, valor_total, cod_cliente, tipo_factura son los atributos y los posibles métodos serán Buscar_factura(), Buscar_facturas(), Consultar_Ventas().

La relación entre clases es de simple asociación.

Es posible que esta clase posteriormente se incorpore en un paquete y componente, los diagramas serían los siguientes:

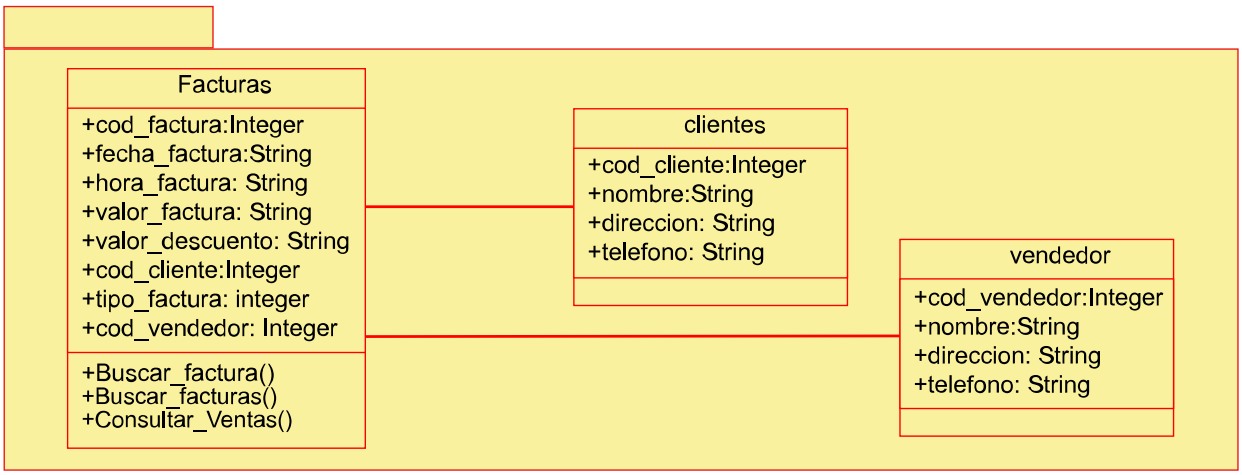


Figura 10. Estructura en diagrama de paquetes.

En diagrama de componentes:

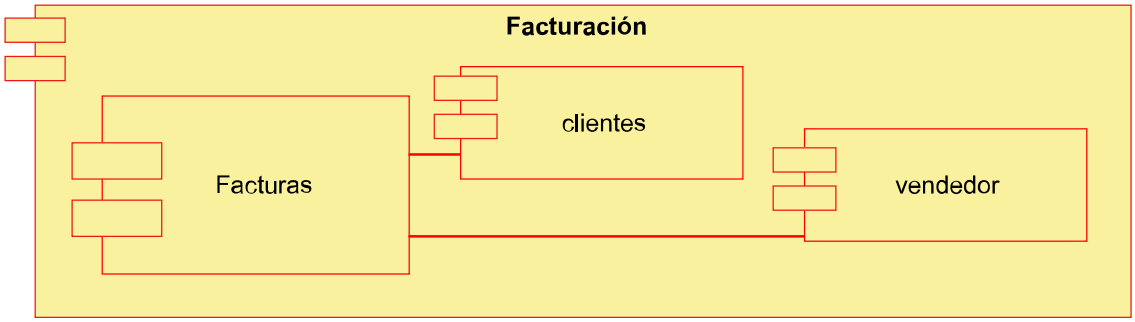


Figura 11. Estructura en diagrama de facturación.

Este sería una forma muy sencilla de modelar la estructura del programa a desarrollar, teniendo en cuenta los diferentes elementos que nos ofrece el UML para plasmar en un plano (modelo UML) las partes principales de la estructura del software.

3. Modelando el comportamiento.

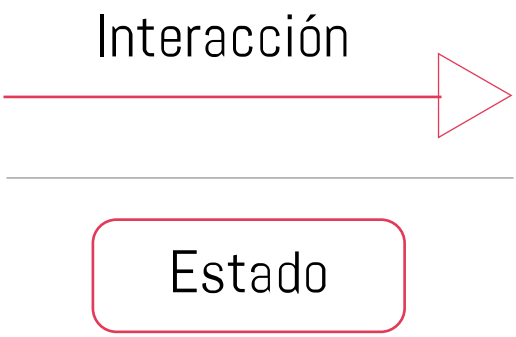
El comportamiento dentro de un software corresponde a las acciones que ejecutan los objetos dentro del mismo, es decir la funcionalidad o la acción (verbo) específica. Por ejemplo copiar, cortar, pegar son funciones o acciones que se ejecutan en un software que tiene un comportamiento definido y están controladas todas sus entradas y salidas.

El objetivo de modelar el comportamiento es determinar el rango de acción de la función para poder controlar el objeto y que dentro del programa este no cause dificultades o errores (bugs).

De acuerdo con Martin Fowler(2006) en su libro “UML gota a gota”, los siguientes elementos son los que hacen para el modelo del comportamiento.

3.1. Interacción.

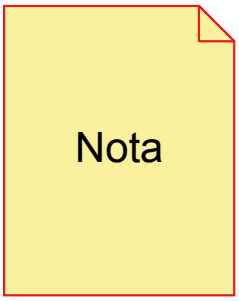
Es un comportamiento que comprende un conjunto de mensajes intercambiados entre un conjunto de objetos, dentro de un contexto particular para conseguir un propósito específico. Una interacción involucra otros muchos elementos, incluyendo mensajes, secuencias de acción (comportamiento invocado por un objeto) y enlaces (conexiones entre objetos). La representación de un mensaje es una flecha dirigida que normalmente con el nombre de la operación.



3.2. Máquinas de estados.

Es un comportamiento que especifica las secuencias de estados por las que van pasando los objetos o las interacciones durante su vida en respuesta a eventos, junto con las respuestas a esos eventos. Una máquina de estados involucra otros elementos como son estados, transiciones (flujo de un estado a otro), eventos (que disparan una transición) y actividades (respuesta de una transición)

Elementos de anotación.



Los elementos de anotación son las partes explicativas de los modelos UML. Son comentarios que se pueden aplicar para describir, clasificar y hacer observaciones sobre cualquier elemento de un modelo. El tipo principal de anotación es la nota que simplemente es un símbolo para mostrar restricciones y comentarios junto a un elemento o un conjunto de elementos.

Relaciones.

Existen cuatro tipos de relaciones entre los elementos de un modelo UML. Dependencia, asociación, generalización y realización, estas se describen a continuación:

Dependencia.

Es una relación semántica entre dos elementos en la cual un cambio a un elemento (el elemento independiente) puede afectar a la semántica del otro elemento (elemento dependiente). Se representa como una línea discontinua, posiblemente dirigida, que a veces incluye una etiqueta.

Asociación.

Es una relación estructural que describe un conjunto de enlaces, los cuales son conexiones entre objetos. La agregación es un tipo especial de asociación y representa una relación estructural entre un todo y sus partes. La asociación se representa con una línea continua, posiblemente dirigida, que a veces incluye una etiqueta. A menudo se incluyen otros adornos para indicar la multiplicidad y roles de los objetos involucrados.

Generalización.

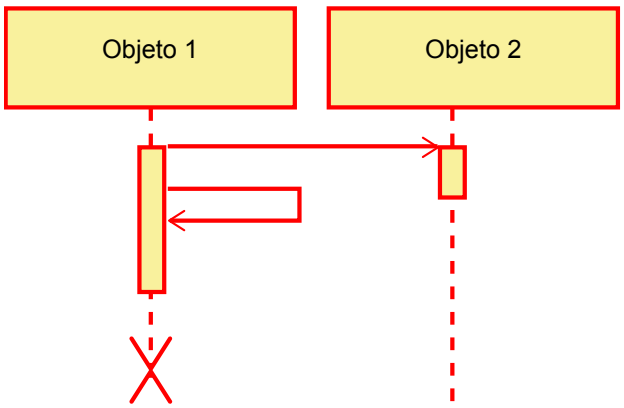
Es una relación de especialización / generalización en la cual los objetos del elemento especializado (el hijo) pueden sustituir a los objetos del elemento general (el padre). De esta forma, el hijo comparte la estructura y el comportamiento del padre. Gráficamente, la generalización se representa con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

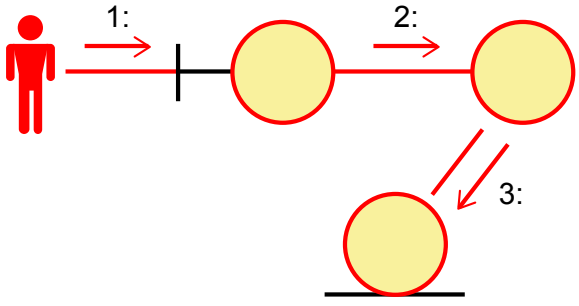
Realización.

Es una relación semántica entre clasificadores, donde un clasificador especifica un contrato que otro clasificador garantiza que cumplirá. Se pueden encontrar relaciones de realización en dos sitios: entre interfaces y las clases y componentes que las realizan, y entre los casos de uso y las colaboraciones que los realizan. La realización se representa como una mezcla entre la generalización y la dependencia, esto es, una línea discontinua con una punta de flecha vacía.

Diagramas de Secuencia y de Colaboración.

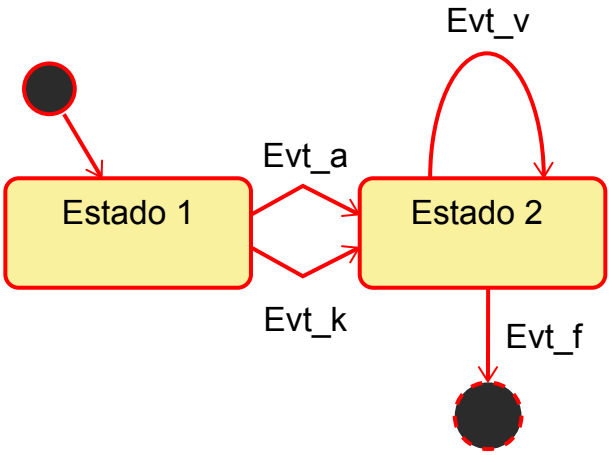
Tanto los diagramas de secuencia como los diagramas de colaboración son un tipo de diagramas de interacción. Constan de un conjunto de objetos y sus relaciones, incluyendo los mensajes que se pueden enviar unos objetos a otros. Cubren la vista dinámica del sistema. Los diagramas de secuencia enfatizan el ordenamiento temporal de los mensajes, mientras que los diagramas de colaboración muestran la organización estructural de los objetos que envían y reciben mensajes. Los diagramas de secuencia se pueden convertir en diagramas de colaboración sin perdida de información, lo mismo ocurre en sentido opuesto.



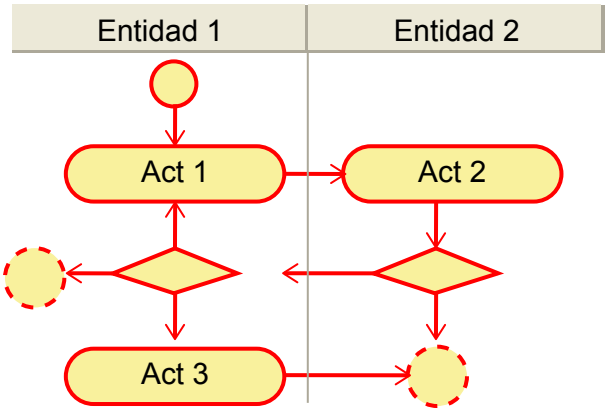


Diagramas de Estados.

Muestran una máquina de estados compuesta por estados, transiciones, eventos y actividades. Estos diagramas cubren la vista dinámica de un sistema y son muy importantes a la hora de modelar el comportamiento de una interfaz, clase o colaboración.



Diagramas de Actividades.

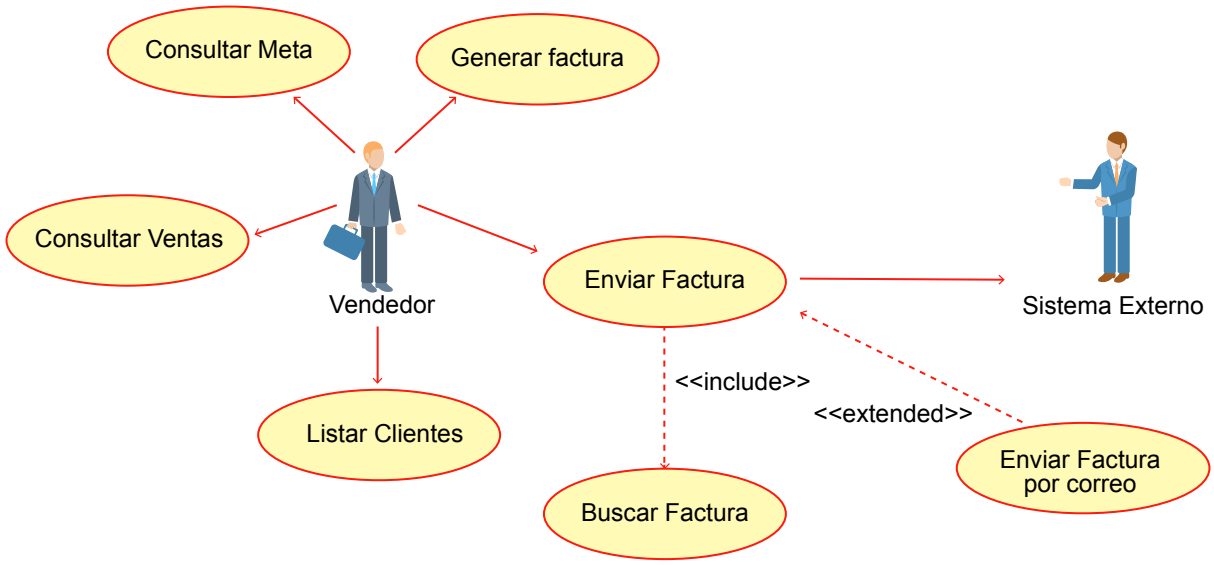


Son un tipo especial de diagramas de estados que se centra en mostrar el flujo de actividades dentro de un sistema. Los diagramas de actividades cubren la parte dinámica de un sistema y se utilizan para modelar el funcionamiento de un sistema resaltando el flujo de control entre objetos.

Para ahondar un poco más en el modelo del comportamiento se realizará un ejemplo con el caso del vendedor viajero.

Uno de los diagramas más utilizados para determinar el comportamiento del software es el diagrama de secuencias.

Para poder realizar el diagrama de secuencias se debe determinar el actor, los objetos que intervienen y el escenario que se quiere modelar. El escenario no es más que las condiciones iniciales y finales que puedan presentarse al ejecutar un caso de uso determinado. por ejemplo para el ejercicio se habían identificado los siguientes casos de uso:



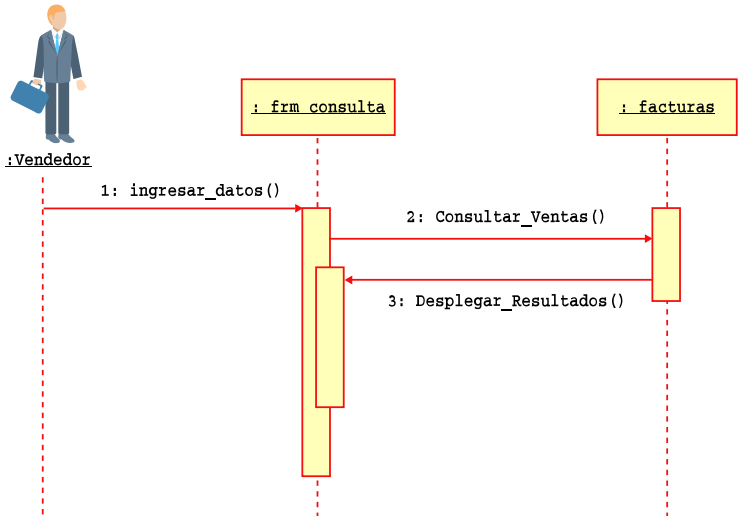
Para efectos de la práctica se seleccionará un caso de uso y se plantean los escenarios correspondientes.

Se selecciona el caso de uso *consultar ventas* se analiza los siguientes escenarios o posibilidades.

Al momento de *consultas ventas* pueden suceder los siguientes casos:

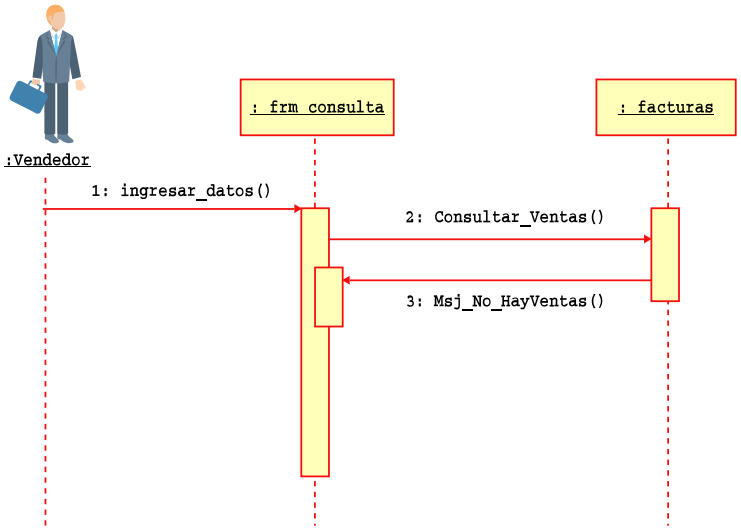
1. Al momento de consultar no hay ventas o facturas registradas.
2. Al momento de consultar hay ventas o facturas registradas.

Escenario 1

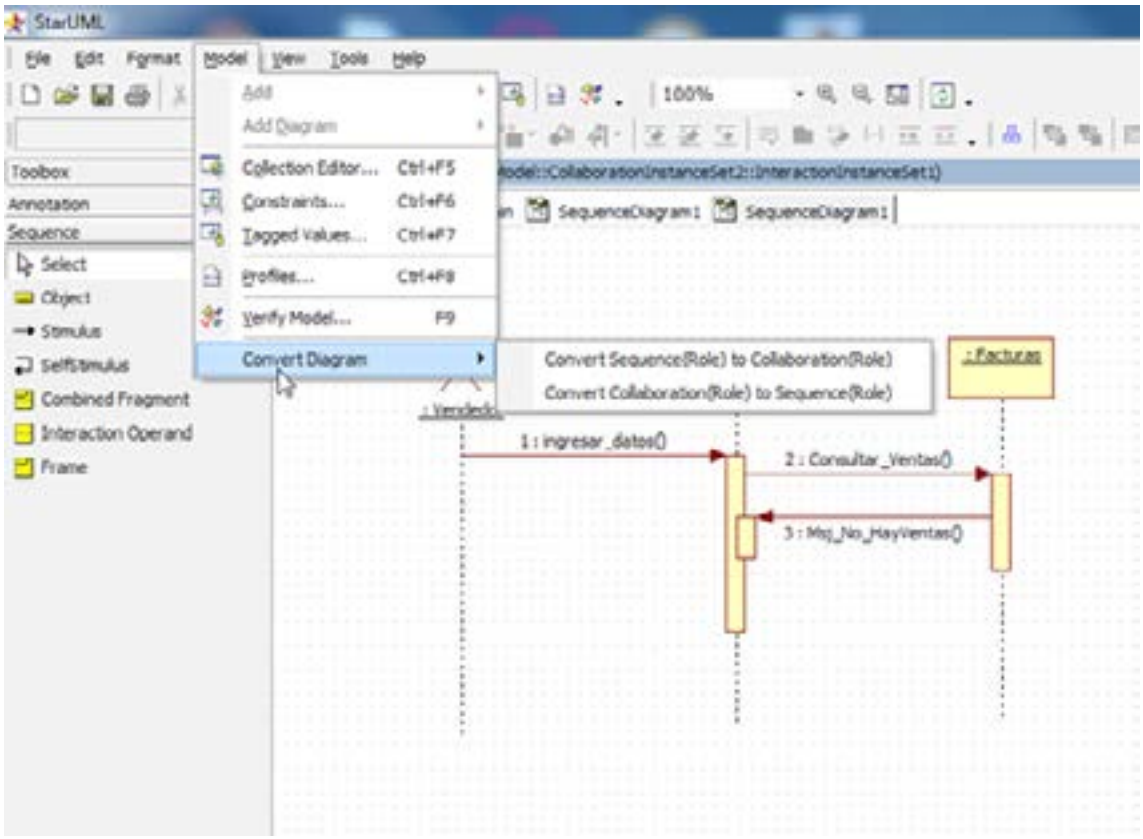


El actor que ejecutará el caso de uso es el vendedor por lo tanto este sería el diagrama de secuencias para el caso de uso consultar ventas y los objetos que intervienen en el proceso:

Escenario 2



Estos diagramas se pueden complementar con los diagramas de colaboración para ello desde la herramienta starUML desde la opción Model Convert Diagram se puede generar el diagrama de colaboración.



GLOSARIO

Agregación: corresponde a la inclusión de para cualquier sustancia o mezcla u objetos, modificando sus condiciones iniciales, pueden obtenerse distintos estados o fases, denominados estados de agregación de la materia, en relación con las fuerzas de unión de las partículas (moléculas, átomos o iones) que la constituyen.

Composición: composición es una forma fuerte de composición donde la vida de la clase contenida debe coincidir con la vida de la clase contenedor. Los componentes constituyen una parte del objeto compuesto. De esta forma, los componentes no pueden ser compartidos por varios objetos compuestos. La supresión del objeto compuesto conlleva la supresión de los componentes.

Nodo: en términos generales, un nodo es un espacio en el que confluyen parte de las conexiones de otros espacios reales o abstractos que comparten sus mismas características y que a su vez también son nodos.

Objetos: un objeto es una unidad dentro de un programa de computadora que consta de un estado y de un comportamiento, que a su vez constan respectivamente de datos almacenados y de tareas realizables durante el tiempo de ejecución. Un objeto puede ser creado instanciando una clase, como ocurre en la programación orientada a objetos, o mediante escritura directa de código y la replicación otros objetos, como ocurre en la programación basada en prototipos.

Procedimiento: forma específica de llevar a cabo una actividad. En muchos casos los procedimientos se expresan en documentos que contienen el objeto y el campo de aplicación de una actividad; que debe hacerse y quien debe hacerlo; cuando, donde y como se debe llevar a cabo; que materiales, equipos y documentos deben utilizarse; y como debe controlarse y registrarse.

Proceso: conjunto de recursos y actividades interrelacionados que transforman elementos de entrada en elementos de salida. Los recursos pueden incluir personal, finanzas, instalaciones, equipos, técnicas y métodos.

Relación: correspondencia o conexión que hay entre dos o más cosas. En el caso de UML es la conexión que existe entre uno o más elementos o diagramas UML.

Requerimiento: un requisito es una necesidad documentada sobre el contenido, forma o funcionalidad de un producto o servicio. Se usa en un sentido formal en la ingeniería de sistemas, ingeniería de software e ingeniería de requisitos.

Sinergia: comúnmente, refleja un fenómeno por el cual actúan en conjunto varios factores, o varias influencias, observándose así un efecto, además del que hubiera podido

esperarse operando independientemente, dado por la concausalidad, 1 a los efectos en cada uno. En estas situaciones, se crea un efecto extra debido a la acción conjunta o solapada, que ninguno de los sistemas hubiera podido generar en caso de accionar aisladamente.

Sistema: estructura organizativa, procedimientos, procesos y recursos necesarios para implantar una gestión determinada, como por ejemplo la gestión de la calidad, la gestión del medio ambiente o la gestión de la prevención de riesgos laborales. Normalmente están basados en una norma de reconocimiento internacional que tiene como finalidad servir de herramienta de gestión en el aseguramiento de los procesos.

Subprocesos: son partes bien definidas en un proceso. Su identificación puede resultar útil para aislar los problemas que pueden presentarse y posibilitar diferentes tratamientos dentro de un mismo proceso.

BIBLIOGRAFÍA

Bennett, S., McRobb, S., Farmer, R. (2006). *Análisis y diseño orientado a objetos de sistemas usando UML*. Madrid: McGrawHill.

Fowler, M., Kendall, S. (1999). *UML gota a gota.* México: Addison Wesley.

International Organization for Standardization. (2017). Recuperado de https://www.iso.org/home.html.

CONTROL DEL DOCUMENTO

CONSTRUCCIÓN
OBJETO DE
APRENDIZAJE

MODELANDO LA ESTRUCTURA Y EL COMPORTAMIENTO DEL SOFTWARE	
Centro Industrial de Mantenimiento Integral - CIMI Regional Santander	
Líder línea de producción:	Santiago Lozada Garcés.
Asesores pedagógicos:	Rosa Elvia Quintero Guasca. Claudia Milena Hernández Naranjo.
Líder expertos temáticos: **Experto temático:**	Rita Rubiela Rincón Badillo. Edward José Beltrán Lozano.
Diseño multimedia:	Tirso Fernán Tabares Carreño.
Programador:	Francisco José Lizcano Reyes.
Producción de audio:	Víctor Hugo Tabares Carreño.