Proyecto Certificador de Desarrollo de SW 1

Tema Nº9:

Análisis y Diseño - Definir la Arquitectura

Indicador de logro Nº9:Conoce el modelo de análisis y diseño de un sistema logrando construir la arquitectura de casos de uso según análisis.

**TEMA 01 Teoría de los**

Imagen que contiene Icono

Descripción generada automáticamente

**TEMA Nº9:**

ANÁLISIS Y DISEÑO - DEFINIR LA ARQUITECTURA

Objeto De La Experiencia

* Conoce la disciplina, documenta y explica la actividad del análisis y diseño respecto a la metodología RUP, como también define, los artefactos y los roles más importantes.
* Elabora la arquitectura del sistema
* Elabora el diagrama de Análisis

Marco Teórico

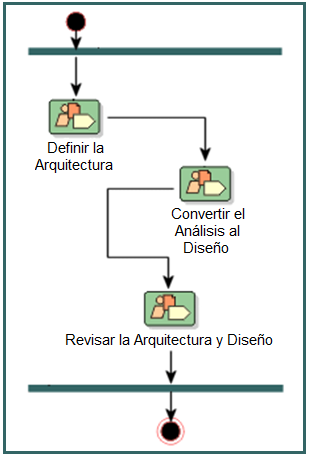
En la disciplina de análisis y diseño, se transforman los requerimientos en un diseño de sistema, se desarrolla una arquitectura robusta y se adapta el diseño para concordar con el ambiente de implementación.

Esta disciplina se caracteriza por su esfuerzo en el modelado, ya que se transforman los requerimientos en clases, subsistemas, paquetes y relaciones, se diseña de la base de datos y se hace la identificación de componentes:

Los principales artefactos que se relacionan con esta disciplina son:

* Modelo de diseño: Es un modelo de objetos describiendo la realización de casos de uso y sirve como una abstracción del modelo de implementación.
* Modelo de datos: Es un subconjunto del modelo de implementación que describe la representación lógica y física de los datos persistentes en el sistema.
* Documento de la arquitectura de software: Es un resumen arquitectónico del sistema, usa el conjunto de vistas arquitectónicas.

Flujo de Trabajo



Artefactos del Análisis

|  |  |
| --- | --- |
| **Artefacto** | **Descripción** |
| Modelo de Análisis | Representa la vista interna del sistema. Define |
| un modelo de objetos que describe la |
| realización de casos de uso, y que sirve como |
| una abstracción del modelo de diseño. |
| Paquete de Análisis | Los paquetes del análisis proporcionan un |
| medio para organizar los artefactos del modelo |
| de análisis en piezas manejables. |
| Un paquete de análisis contiene clases y |
| realizaciones de casos de uso a nivel de |
| análisis. |
| Clase de Interfaz | Es una clase utilizada para modelar la |
| interacción entre el entorno del sistema y su |
| funcionamiento interno. Modela las partes del |
| sistema que dependen de su entorno. |
| Clase Control | Representa la lógica de negocio de la |
| aplicación, es decir, el control, la coordinación y |
| la secuencia entre objetos. Encapsula el |
| comportamiento de uno o más casos de uso. |
| Clase Entidad | La entidad es una clase utilizada para modelar |
| la información y comportamiento asociado que |
| deben ser almacenados. |
| Modela las partes del sistema que son |
| independientes de su entorno. |
| Realización de Caso de Uso | La realización de análisis de un caso de uso es |
| una colaboración que describe cómo se realiza |
| el caso de uso en términos de clases de análisis |
| y sus interacciones. |
| Diagrama de Clases | El diagrama de clases describe la estructura de |
| un caso de uso. |

|  |  |
| --- | --- |
| Diagrama de Comunicación | Diagrama de interacción que describe el comportamiento del caso de uso centrado en la responsabilidades y colaboraciones entre los objetos. |

1. Definición de la Arquitectura

* 1. **Analizar la Arquitectura (Propósito)**
* Definir una arquitectura candidata para el sistema, basada en la experiencia obtenida de sistemas similares o en dominios de problemas similares.
* Definir los patrones arquitectónicos, mecanismos clave y convenciones de modelado del sistema.
* Para definir la estrategia de reutilización
* Para aportar información al proceso de planificación de las partes interesadas

* 1. **Analizar Casos de Uso (Propósito)**
* Identificar las clases que realizan el flujo de eventos de un caso de uso.
* Distribuir el comportamiento de casos de uso a esas clases, utilizando realizaciones de casos de uso.
* Identificar las responsabilidades, atributos y asociaciones de las clases.
* Tomar nota del uso de mecanismos arquitectónicos.

1. Conversión del Análisis al Diseño
   1. **Diseñar los Casos de Uso (Propósito)**

* Refinar la realización de casos de uso en términos de interacciones.
* Refinar los requisitos sobre las operaciones de las clases de diseño.
* Refinar los requisitos sobre las operaciones de los subsistemas y / o sus interfaces.
* Refinar los requisitos sobre el funcionamiento de las cápsulas.

**Pasos**

Describir interacciones entre objetos de diseño

* Para la realización de cada caso de uso, debe ilustrar las interacciones entre sus objetos de diseño participantes creando uno o más diagramas de secuencia.

La actualización de los diagramas de secuencia implica los siguientes pasos:

* Identifique cada objeto que participa en el flujo del caso de uso.
* Represente cada objeto participante en un diagrama de secuencia. Haga una línea de vida para cada objeto participante en el diagrama de secuencia:
* Puede utilizar las interfaces realizadas por el subsistema. Esto es preferible en los casos en los que desee mostrar que cualquier elemento del modelo que realice la misma interfaz puede usarse en lugar de la interfaz. Si opta por mostrar interfaces en el diagrama de secuencia, tenga en cuenta que querrá asegurarse de que no se envíen mensajes desde la interfaz a otros objetos. La razón de esto es que las interfaces encapsulan completamente la realización interna de sus operaciones. Por lo tanto, no podemos estar seguros de que todos los elementos del modelo que realizan la interfaz estén realmente diseñados de la misma manera. Por lo tanto, en los diagramas de secuencia no se deben mostrar mensajes enviados desde interfaces.
* Representar la interacción que tiene lugar con los actores. Represente cada instancia de actor y objeto externo con el que interactúan los objetos participantes mediante una línea de vida en el diagrama de secuencia.
* Ilustra el envío de mensajes entre los objetos participantes. El flujo de eventos comienza en la parte superior del diagrama y continúa hacia abajo, indicando un eje cronológico vertical. Ilustre el envío de mensajes entre objetos creando mensajes (flechas) entre las líneas de vida. El nombre de un mensaje debe ser el nombre de la operación invocada por el mensaje. En las primeras etapas del diseño, no se asignarán muchas operaciones a los objetos, por lo que es posible que deba omitir esta información y darle al mensaje un nombre temporal; se dice que estos mensajes están "sin asignar". Más tarde, cuando haya encontrado más operaciones de los objetos participantes, debe actualizar el diagrama de secuencia "asignando" los mensajes con estas operaciones.
* Describe lo que hace un objeto cuando recibe un mensaje. Esto se hace adjuntando un script al mensaje correspondiente. Coloque estos guiones en el margen del diagrama. Utilice texto estructurado o pseudocódigo. Si usa pseudocódigo, asegúrese de usar construcciones en el lenguaje de implementación para que la implementación de las operaciones correspondientes sea más fácil. Cuando la persona responsable de la clase de un objeto asigna y define sus operaciones, los scripts del objeto proporcionarán una base para ese trabajo.
* Simplifique los diagramas de secuencia usando subsistemas (opcional)

Cuando se realiza un caso de uso, el flujo de eventos generalmente se describe en términos de los objetos en ejecución, es decir, como interacción entre los objetos de diseño. Para simplificar los diagramas e identificar el comportamiento reutilizable, puede ser necesario encapsular un subflujo de eventos dentro de un subsistema. Cuando se hace esto, las subsecciones grandes del diagrama de secuencia se reemplazan con un solo mensaje al subsistema. Dentro del subsistema, un diagrama de secuencia separado puede ilustrar las interacciones internas dentro del subsistema que proporcionan el comportamiento requerido.

* Describir el comportamiento relacionado con la persistencia

Todo el objetivo del paradigma orientado a objetos es encapsular los detalles de implementación. Por lo tanto, con respecto a la persistencia, nos gustaría que un objeto persistente se pareciera a un objeto transitorio. No deberíamos tener que ser conscientes de que el objeto es persistente, o tratarlo de manera diferente a como lo haríamos con cualquier otro objeto. Al menos ese es el objetivo.

* Escribir objetos persistentes
* Leer objetos persistentes
* Eliminar objetos persistentes
* Modelado de transacciones
* Manejo de condiciones de error
* Manejo del control de simultaneidad
* Refinar la descripción del flujo de eventos.

En el flujo de eventos de la realización del caso de uso, es posible que deba agregar una descripción adicional a los diagramas de secuencia, en los casos en que el flujo de eventos no esté completamente claro con solo examinar los mensajes enviados entre los objetos participantes. Algunos ejemplos de estos casos incluyen casos en los que se necesitan anotaciones de tiempo, notas sobre el comportamiento condicional o aclaración del comportamiento de la operación para facilitar la lectura de los diagramas a los observadores externos.

* Unificar clases y subsistemas.

A medida que se realizan los casos de uso, es necesario unificar las clases y subsistemas identificados para garantizar la homogeneidad y la coherencia en el modelo.

Puntos a considerar:

* Los nombres de los elementos del modelo deben describir su función.
* Evite nombres y sinónimos similares porque dificultan la distinción entre elementos del modelo.
* Fusionar elementos del modelo que definan un comportamiento similar o que representen el mismo fenómeno.
* Fusionar clases de entidad que representan el mismo concepto o tienen los mismos atributos, incluso si su comportamiento definido es diferente.
* Utilice la herencia para abstraer elementos del modelo, lo que tiende a hacer que el modelo sea más robusto.
* Al actualizar un elemento de modelo, actualice también la descripción del flujo de eventos correspondiente de las realizaciones de casos de uso.
* Evalúe sus resultados

Debe verificar el modelo de diseño en esta etapa para verificar que su trabajo va en la dirección correcta. No es necesario revisar el modelo en detalle, pero debe considerar los puntos de control del modelo de diseño mientras trabaja en él.

**Artefactos de entrada:**

* Especificaciones complementarias (Definido anteriormente)
* Realización de casos de uso (Definido anteriormente)

**Artefactos resultantes:**

* Realización de casos de uso (Definido anteriormente)

**Rol:**

* Diseñador (Definido anteriormente)

1. Revisión de la Arquitectura del Sistema

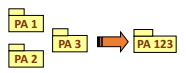
Organizar los artefactos del Modelo de Análisis en piezas más manejables.

Paquetes del sistema

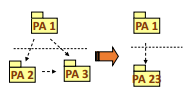
* Un paquete es una colección de artefactos (casos de uso, actores, relaciones, diagramas y otros paquetes) que se utiliza para dividir un modelo en partes de menor tamaño.
* Representa un subsistema o módulo de la aplicación.
* Hace más fácil la definición de la arquitectura.
* Facilita la asignación de responsabilidades y tareas a los miembros del equipo de proyecto.

Tipos de Paquetes

1. **Altamente Cohesivos:** sus contenidos deberían estar fuertemente relacionados.



1. **Débilmente Acoplados:** dependencias entre paquetes deberían minimizarse



Los paquetes identificados se organizarán en la Capa de Aplicación, la cual se subdivide en dos capas internas:

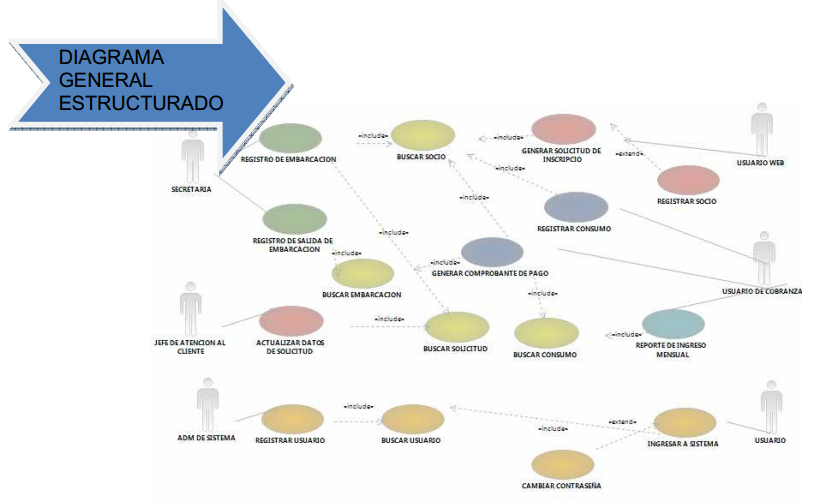
1. **Capa Específica:** Aquí se ubican los paquetes correspondientes al proceso del negocio (Core), de la empresa identificados (Nivel Superior).
2. **Capa General:** Aquí se ubican los paquetes de maestros de información, paquetes de servicio, seguridad y casos de apoyo del sistema (Nivel Inferior).

Para identificar las dependencias entre paquetes es conveniente revisar el diagrama de casos de uso estructurado según análisis, esto con el fin de ubicar las relaciones que existen entre los casos de uso. Las dependencias se crean a partir de los paquetes de análisis que contienen los casos de uso base.

A continuación, se muestra un ejemplo de distribución de paquetes en las capas de la aplicación y sus dependencias para definir la arquitectura de análisis.



**Ejemplo:** Construya el diagrama de paquetes a partir del siguiente diagrama estructurado.



**SOLUCIÓN**:

1. Diagrama de Análisis

El Análisis Orientado a Objetos se traduce en el modelo de análisis, el cual es usado para representar la estructura global del sistema, describe la realización de casos de uso y sirve como una abstracción del modelo de diseño.

Durante el análisis, se identifica, de manera continua, nuevos paquetes del análisis, clases y requisitos comunes a medida que el modelo de análisis evoluciona, y los paquetes de análisis concretos continuamente se refinan y mantienen.

Las actividades que se realizan para elaborar el modelo de análisis son los siguientes:

* Análisis de la Arquitectura
* Análisis de Casos de Uso
* Análisis de Clases
* Análisis de Paquetes.

Según Ivar Jacobson, “El modelo de análisis es un nivel de diseño intermedio entre las etapas de captura de requisitos y la de diseño.”

Este modelo de análisis no es un diagrama final que describe todos los posibles conceptos y sus relaciones, es un primer intento por definir los conceptos claves que describen el sistema. Este artefacto es opcional, pero también tiene a su vez la propiedad de ser temporal en el caso en que se planea su desarrollo. Su utilidad radica en que permite una apreciación global conceptual del sistema.

El modelo de análisis puede contener las clases y paquetes de análisis, las realizaciones de los casos de uso, las relaciones y los diagramas. Es opcional detallar aquí las realizaciones de los casos de uso, ya que estas pueden estar en el modelo de diseño donde se recomienda que se encuentre.

A diferencia del modelo de casos de uso que captura la funcionalidad del sistema, el modelo de análisis da forma a la arquitectura para soportar las funcionalidades que en el anterior modelo se expresan.

Análisis de los Casos de Uso

El Análisis de Caso de Uso es el proceso de examinar los casos de uso para descubrir los objetos y clases de análisis del sistema a desarrollar. Las clases identificadas deben agruparse en los paquetes según criterios de Arquitectura de Software.

El rol responsable de esta tarea es el Diseñador. Esta tarea describe cómo desarrollar las Realizaciones de los Casos de Uso del nivel de análisis de un caso de uso particular. Tiene los siguientes propósitos:

* Identificar las clases que llevan a cabo el flujo de eventos de un caso de uso.
* Distribuir el comportamiento de casos de uso a las clases identificadas, usando realizaciones de casos de uso a nivel de análisis.
* Identificar atributos, responsabilidades y relaciones entre las clases.
* Observar los mecanismos arquitectónicos.

Pasos en el Análisis de Casos de Uso

Para llevar a cabo el análisis de casos de uso se realiza lo siguiente:

* Crear la realización de análisis de casos de uso.
* Encontrar clases de análisis del comportamiento de casos de uso.
* Crear el diagrama de clases (estructura del caso de uso).
* Crear el diagrama de comunicación (comportamiento del caso de uso)

Crear la Realización de Análisis de Casos de Uso.

Una Realización de Caso de Uso describe cómo un Caso de Uso en particular es modelado, primero en el modelo de análisis y, después, en el modelo de diseño en términos de objetos colaboradores.

En una Realización de Caso de Uso se especifica qué clases deben ser construidas para implementar cada Caso de Uso.

En UML, las Realizaciones de Caso de Uso se representan con colaboraciones estereotipadas. El ícono para una colaboración es una elipse con líneas punteadas que se sitúa al lado izquierdo del nombre de la colaboración, tal como se ilustra en la siguiente figura.

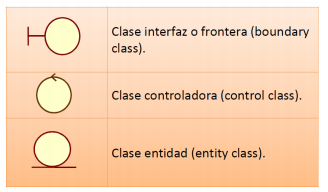


Encontrar Clases de Análisis

Este paso se realiza por cada caso de uso. Para ello, se analizan los escenarios de Caso de Uso para identificar las clases que participan en ellos

A partir de una especificación de caso de uso (ECU) se pueden obtener las clases de análisis.

Existen tres tipos de Clases de Análisis:



Boundary.

Describe una interacción entre el sistema con los usuarios y con otros sistemas. Pueden representar abstracciones de formularios, de protocolos de comunicaciones con otros sistemas o interfaces de dispositivos.

Las características importantes de este tipo de clase cuando modela un API con otro sistema son los siguientes:

a. Las funciones que provee el otro sistema

b. La información a ser pasada al otro sistema

c. El “protocolo” de comunicación usado para “hablar” con el otro sistema.

Por regla general, al menos una clase boundary sirve como medio de comunicación entre un actor y el correspondiente caso de uso. Ejemplo:

En el caso de uso “Procesar Facturación” hay información que debe ser enviada a un Sistema de Facturación externo. Se puede crear una clase de interfaz llamada CI\_SistemaFacturacion para representar la interfaz al sistema externo

Entity

Se emplean para modelar aquella información o comportamiento que posee una vida larga en el sistema.

Normalmente, están asociadas a algún fenómeno o concepto, como una persona, un objeto del mundo real o un suceso del mundo real.

El número de clases entidad variará dependiendo de los conceptos que requieren almacenamiento persistente dentro del caso de uso. Estas clases sufren un proceso de refinamiento a medida que se ubica a la misma clase entidad dentro de distintas realizaciones de caso de uso.

**Ejemplo:**

En el caso de uso “Mantener empleados” en el cual se puede registrar, modificar o desactivar empleados es evidente que la información que debe ser manipulada es del empleado. Para ello, se crea una clase entidad Empleado.

Control

Se utilizan para modelar la coordinación, secuencia, transacciones y control de otros objetos. También para representar derivaciones y cálculos complejos, cómo la lógica de negocio, que no pueden asociarse a ninguna información concreta de larga duración almacenada por el sistema.

Por regla general, se trata de encapsular la lógica de control de un caso de uso dentro de una clase Control. Suele ser un buen hábito de diseño utilizar únicamente una clase control por cada caso de uso, y así encapsular en un único elemento la lógica del caso de uso correspondiente. Por otro lado, todos los casos de uso ubicados en un mismo paquete de análisis comparten la misma clase control.

**Ejemplo:**

En un paquete de análisis denominado Evaluación se ubica los casos de uso “Evaluar empleado”, “Procesar evaluación de desempeño” y “Consultar estadísticas de Evaluaciones”. Para los tres casos de uso se crea una clase control CC\_Evaluacion que coordina el trabajo de los tres casos de uso.

Según la metodología OOSE de Ivan Jacobson, las clases de análisis son clases estereotipadas para crear modelos ideales de objetos. Esta metodología se basa en el patrón MVC (Model – View - Controller / Modelo Vista Controlador), que define clases enfocadas a la separación de responsabilidades para conseguir componentes extensibles y reutilizables.

En la siguiente figura se muestran los tipos de clases enfocados a los elementos del patrón MVC.

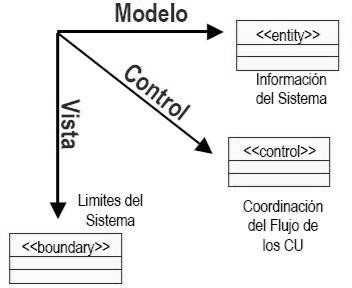


Diagrama de Clases de Análisis

Representa las clases participantes y sus relaciones para un determinado caso de uso.

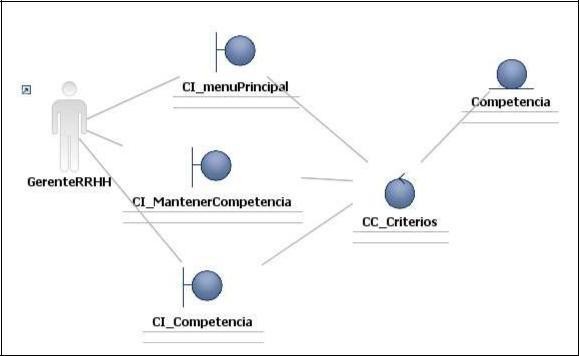


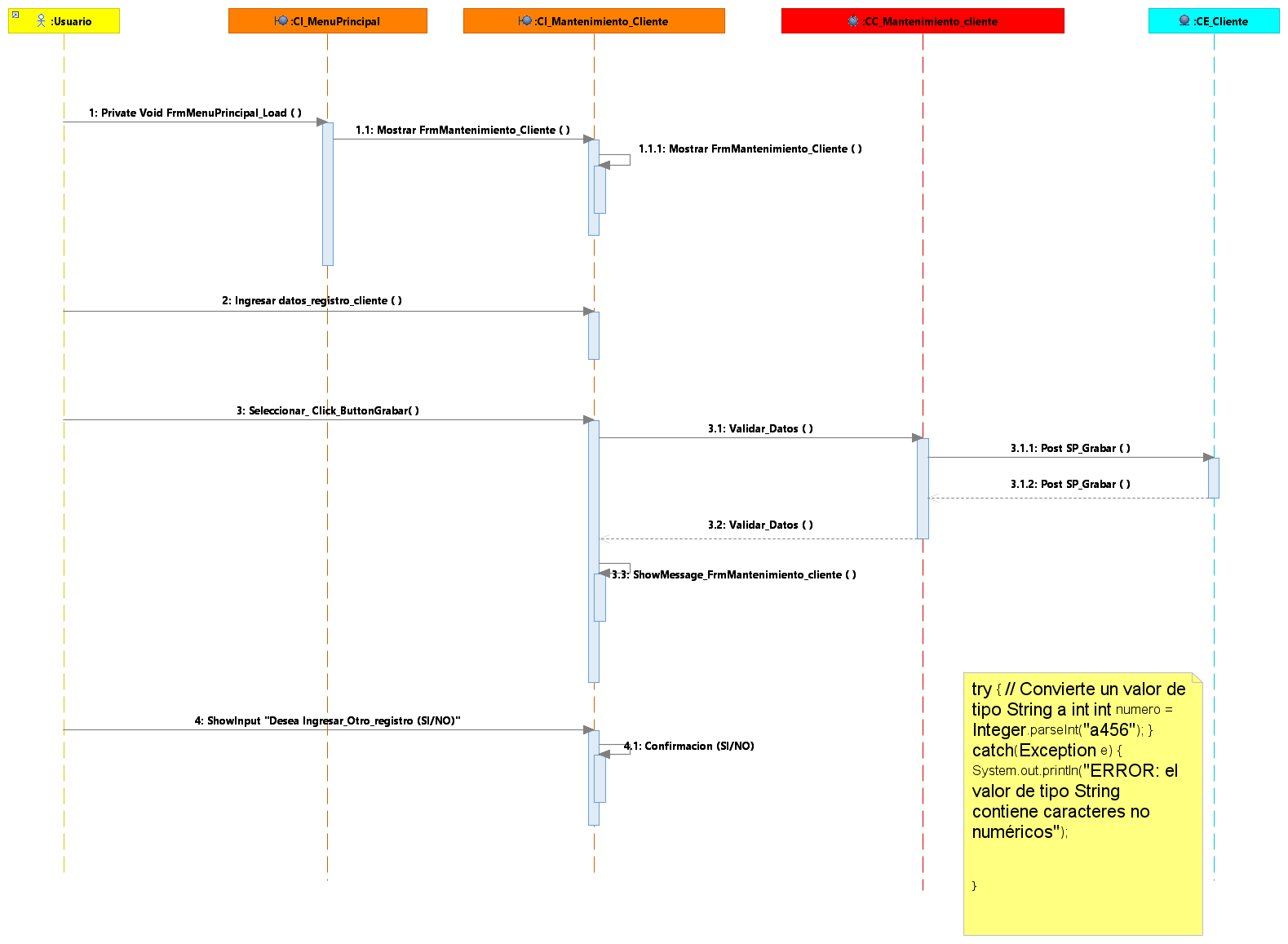
Diagrama de Comunicación

Es un tipo de diagrama de interacción. En esta etapa, no se usa diagramas de secuencia, porque no es importante la cronología de las interacciones. Un diagrama de comunicación muestra la colaboración dinámica entre los objetos, es decir, describe el comportamiento de un caso de uso mostrando explícitamente las relaciones de los objetos participantes.

La realización de un caso de uso puede tener uno o más diagramas de comunicación, esto es debido a que se representa el flujo básico, sub- flujos y flujos alternativos.

Diagrama de Secuencia de Análisis

Es un tipo de diagrama de interacción que muestra la secuencia dinámica entre los objetos, es decir, describe el comportamiento secuencial de un caso de uso.



1. Caso Práctico Nro. 2

La empresa INFOTEL BUSINESS, es una empresa que se dedica a la comercialización y distribución de productos informáticos.

Tiene 15 años en el mercado informático y sus procesos principales son Gerencia, Ventas, Compras y Almacén. Tiene una cartera de clientes muy fidelizada y sus proveedores les proporcionan los productos informáticos de las mejores marcas y modelos con tecnología moderna.

Proceso De Ventas

Un Cliente solicita un pedido de venta a un Vendedor. El Vendedor antes de Generar el Pedido de Venta, Consulta el Stock de Productos y el Estado del Cliente.

Si todo está correcto, el Vendedor Genera el Pedido de Venta. Luego el Cajero Consulta el Pedido de Venta para Generar el Documento de Venta previa Consulta de la Forma de Pago y del Tipo de Documento de Venta.

El Tipo de Documento de Venta puede ser Factura de Venta o Boleta de Venta (Ticket electrónico). La Forma de Pago, puede ser Contado o Crédito. Después el Cajero Genera el Documento de Venta. El Auxiliar de Almacén Consulta el Documento de Venta y Genera la Guía de Salida, para entregar los productos y los documentos de venta al Cliente.

Proceso De Compras

El Asistente de Compras generar los pedidos de compra, previa consulta del stock de productos y del proveedor. Luego envía el pedido de compra a varios proveedores para solicitarles una cotización, después el Asistente de Compra registra las cotizaciones aceptadas por el Jefe de Compras.

El Jefe de Compras verifica la cotización y genera la orden de compra para enviarla al proveedor seleccionado, indicando la forma de pago. El proveedor genera una factura de compra y guía de entrada de productos para enviarla al Jefe de Compras quien registra la factura de compra previa consulta de la guía de entrada de productos. Luego el asistente de almacén, registra la guía de entrada y actualiza el stock de productos

Proceso De Almacén

El Asistente de Almacén genera el pedido de productos faltantes previa consulta del stock de productos y del proveedor.

El Jefe de Almacén registra las guías de entrada, las guías de salida, actualiza el stock de productos y genera el inventario que puede ser diario, semanal, mensual o anual

Desarrollo del Casos Nro. 2

Modelo De Análisis

Se pide elaborar el Modelo de Análisis. Utilizar IBM Rational Software Architect Designer V9.6

5.1 Elaborar la Arquitectura de Análisis

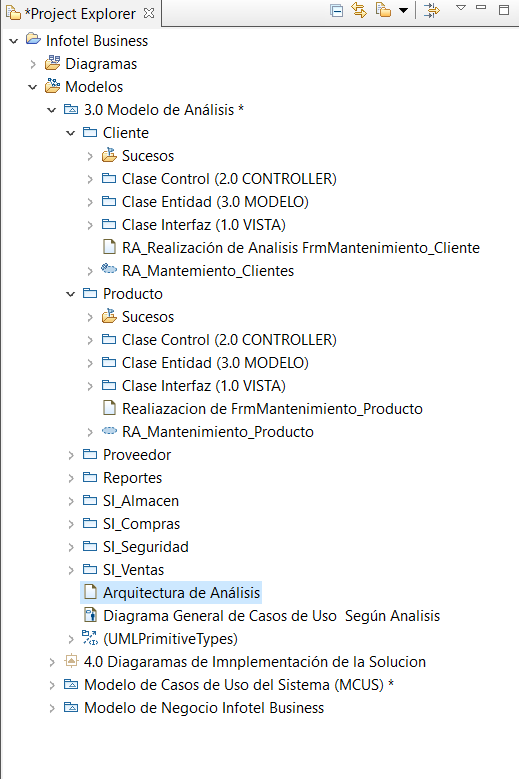
5.2 Determinar los elementos de Análisis

5.3 Determinar las Clases de Análisis

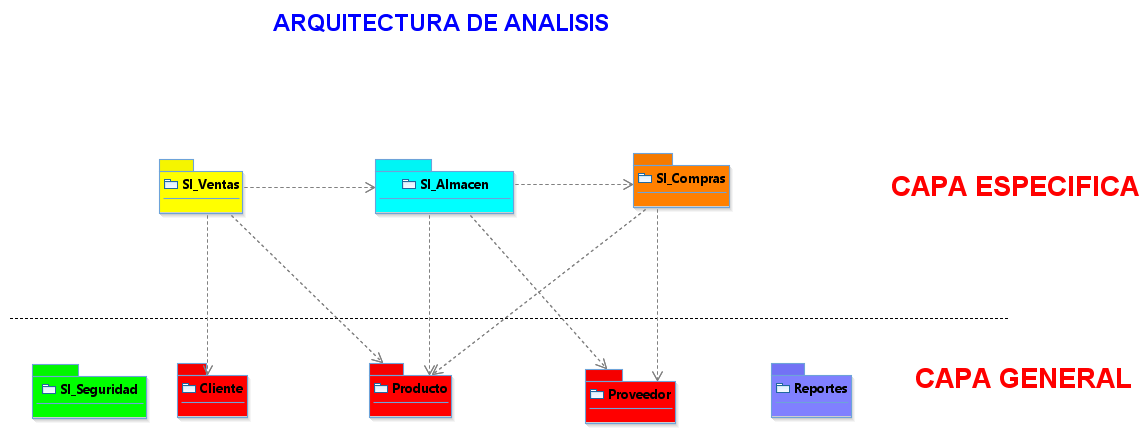
* Interfaz (Boundary)
* Control (Controller)
* Entidad (Entity)

5.4 Elaborar la Realización de Anáisis de Casos de Uso de Mantenimiento de Cliente

* Diagrama de Clases de Análisis
* Diagrama de Secuencias



5.1 Elaborar la Arquitectura de Análisis



5.2 Determinar los elementos de Análisis

Lista de Actores

|  |  |  |
| --- | --- | --- |
| **Ítem** | **Actor** | **Descripción** |
| **1** | Usuario | Es la persona que tiene un rol y su perfil está de acuerdo al acceso gestionado por el administrador |

Lista de la Clase Boundary (Interfaz, 1.0 VISTA)

|  |  |  |
| --- | --- | --- |
| **Ítem** | **Clase Boundary** | **Descripción** |
| **1** | CI\_FrmMenuPrincipal | Este formulario nos permite ingresar a las diferentes opciones del Menú del sistema integrado de Ventas, Compras y Almacén |
| **2** | CI\_FrmMantenimiento\_Cliente | Este formulario nos permite Registrar a los Clientes con opciones de Grabar (), Editar (), Listar () |
| **3** | CI\_FrmMantenimiento\_Producto | Este Formulario nos permite registrar los productos que pueden ser nacionales e importados |
| **4** | CI\_GenerarPedVenta | Este formulario nos permite generar los pedidos de venta, previa consulta del stock de productos y del estado del cliente |

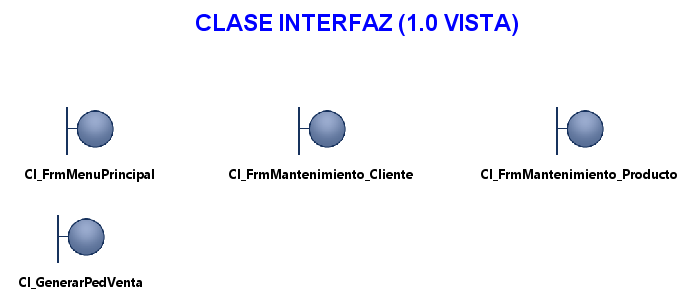
Lista de Clase Control (Control, 2.0 CONTROLLER)

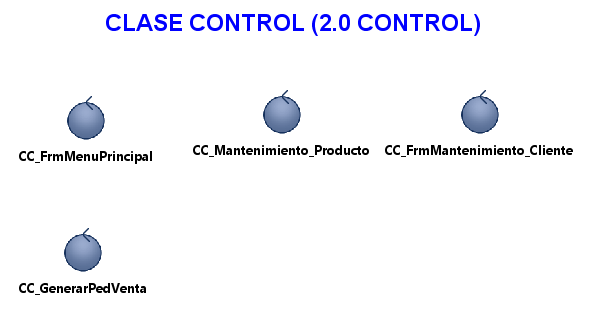
|  |  |  |
| --- | --- | --- |
| **Ítem** | **Clase Control** | **Descripción** |
| **1** | CC\_FrmMenuPrincipal | Permite gestionar la clase formulario con otras clases según las reglas de negocio |
| **2** | CC\_FrmMantenimiento\_Cliente | Esta clase permite gestionar la clase controladora con la clase Vista y clase entidad. Se encuentran los métodos, SP, Servlets, de la clase cliente |
| **3** | CC\_FrmMantenimiento\_Producto | Esta clase permite gestionar la clase controladora con la clase Vista y clase entidad. Se encuentran los métodos, SP, Servlets, de la clase producto |
| **4** | CC\_GenerarPedVenta | Esta clase permite gestionar la clase controladora con la clase Vista y clase entidad. Se encuentran los métodos, SP, Servlets, de la clase CabpedVenta y DetpedVenta |

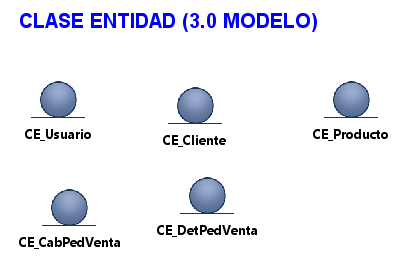
Lista de Clase Entidad (Entity, 3.0 MODELO)

|  |  |  |
| --- | --- | --- |
| **Ítem** | **Clase Entity** | **Descripción** |
| **1** | CE\_Cliente | Representa la clase Cliente |
| **2** | CE\_Producto | Representa la clase Producto |
| **3** | CE\_Usuario | Representa la clase Usuario, que tiene varios perfiles (admin, ventas, compras, almacén) |
| **4** | CE\_Proveedor | Representa la clase Proveedor |
| **5** | CE\_CabPedVenta | Representa la clase Cabecera de Pedido de Venta |
| **6** | CE\_DetPedVenta | Representa la clase Detalle de Pedido de Venta |
| **7** | CE\_CabDocVenta | Representa la clase Cabecera de Documento de Venta |
| **8** | CE\_DetDocVenta | Representa la clase Detalle de Documento de Venta |
| **9** | CE\_CabGuiaSalida | Representa la clase Cabecera de Guía de Salida de Venta |
| **10** | CE\_DetGuiaSalida | Representa la clase Detalle de Guía de Salida de Venta |

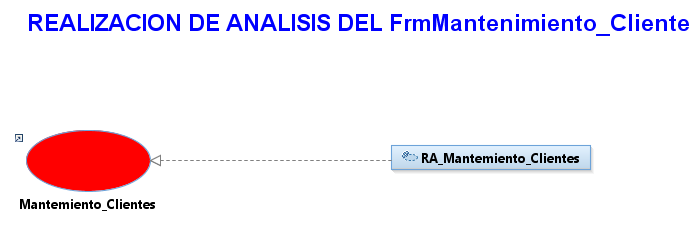
5.3 Determinar las Clases de Análisis







5.4 Elaborar la Realización de Anáisis de Casos de Uso de Mantenimiento de Clientes



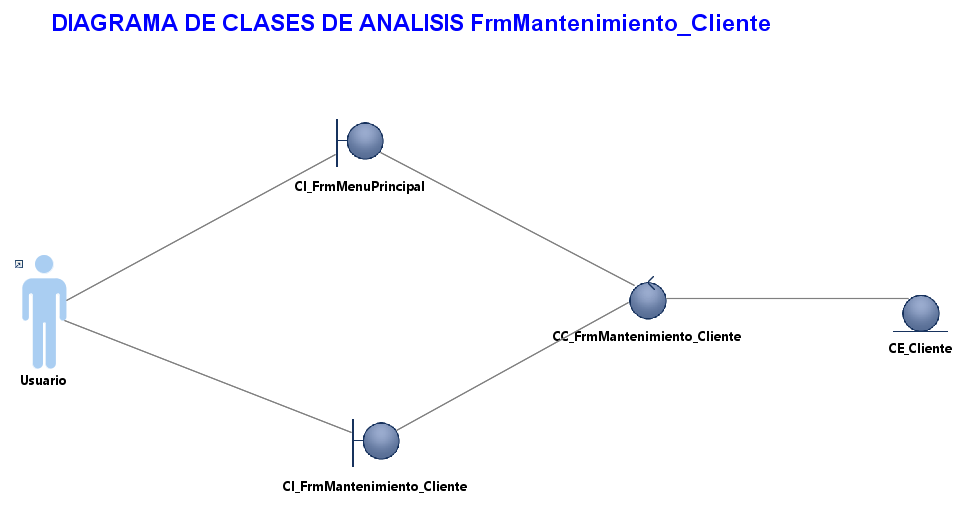
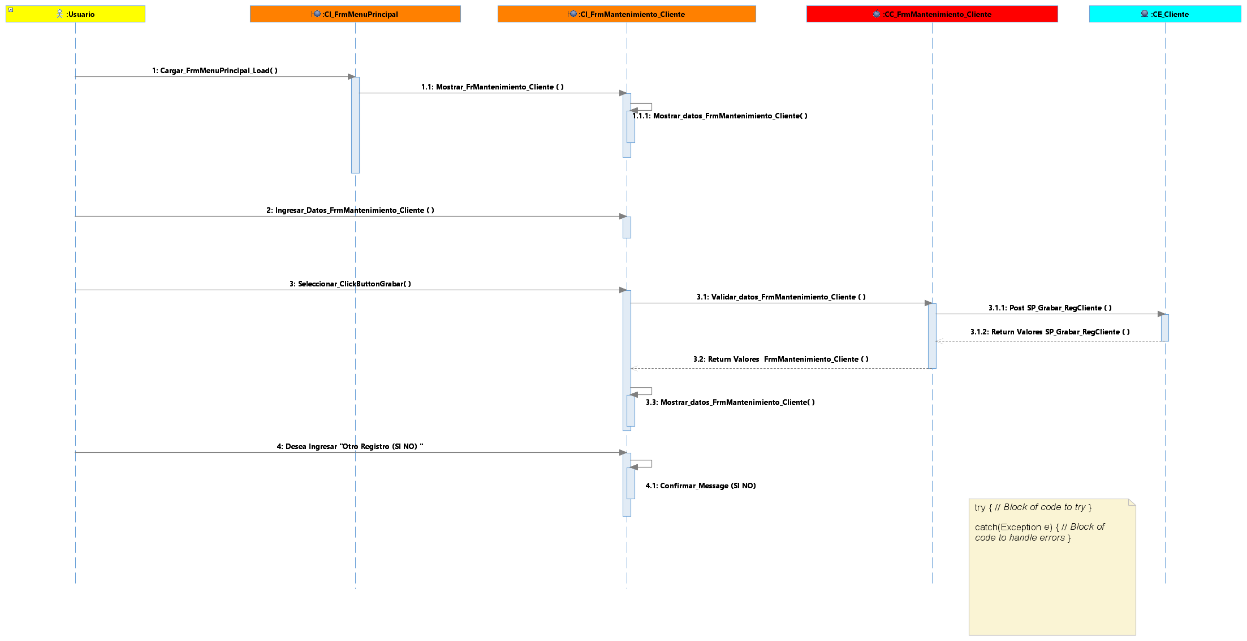


Diagrama de Secuencias: Casos de Uso de Mantenimiento de Clientes



**Actividad:**

Ingresa a la plataforma virtual, luego desarrolla la siguiente actividad propuesta:

1. **CUESTIONARIO TÉCNICO**
2. Tomando como ejemplo el Caso Nro. 02, de la presente guía Modelo de Análisis, elabore la Arquitectura de Análisis, Elementos de Análisis de su proyecto en grupo a desarrollar. Deberá utilizar metodología RUP y además IBM Rational Software Architect Designer V9.6
3. Identifique las Clases de Análisis (Interfaz, Control, Entidad) del Sistema en base a su proyecto en grupo a desarrollar
4. Elaborar la Realización de Análisis de Casos de Uso de su proyecto en grupo a desarrollar. Deberá utilizar metodología RUP y además IBM Rational Software Architect Designer V9.6
5. **CONCLUSIONES DE LA EXPERIENCIA**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_