



INGENIERÍA DEL SOFTWARE I

Práctica 5, Sesión 2

Análisis de la Arquitectura

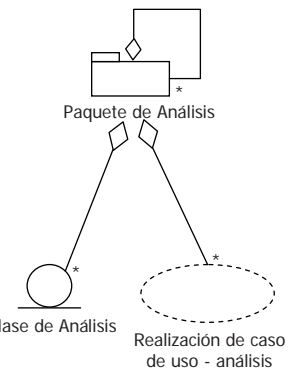
Univ. Cantabria – Fac. de Ciencias

María Sierra



Diagramas de Paquetes

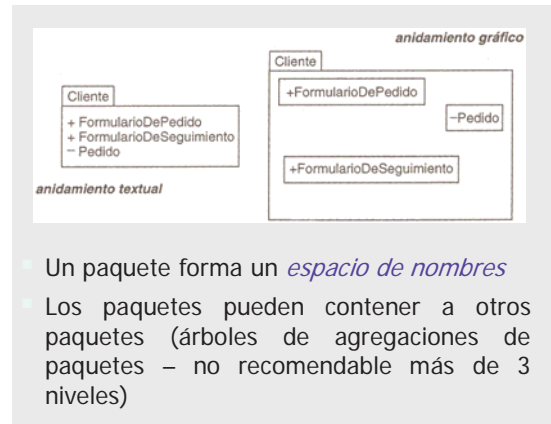
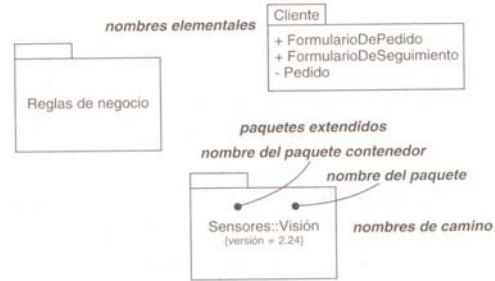
- **Paquetes de Análisis:** Artefacto que proporciona un medio para organizar los artefactos del modelo de análisis en piezas manejables (organiza elementos en grupos)
 - Organizar los elementos en los modelos para comprenderlos más fácilmente
 - Control de acceso a sus contenidos para controlar las líneas de separación de la arquitectura del sistema
 - Mecanismo importante para enfrentarse al crecimiento
- Los paquetes bien diseñados agrupan elementos cercanos semánticamente fuertemente **cohesionados** y débilmente **acoplados**
- Parecidos a las clases, pero completamente diferentes:
 - **Clases:** abstracciones de aspectos del problema o la solución
 - **Paquetes:** mecanismos para organizar, pero no tienen identidad (no puede haber instancias de paquetes).
 - Deben crearse basados en requisitos funcionales y en el dominio del problema (aplicación del negocio), y ser reconocibles por las personas con conocimiento del dominio.
 - No deberían basarse en requisitos no funcionales o en el dominio de la solución.





Diagramas de Paquetes: Elementos

- **Nombre:**
 - Unívoco
 - Simple o Cualificado
 - Se pueden dibujar paquetes adornados con valores etiquetados o con apartados adicionales para mostrar sus detalles
- **Contenidos:**
 - Clases, interfaces, componentes, nodos, colaboraciones, casos de uso e incluso otros paquetes (el título pasa a la pestaña).
 - Se puede mostrar el contenido de un paquete
 - Cuando se muestran los elementos del paquete, el nombre se coloca en la pestaña de la carpeta

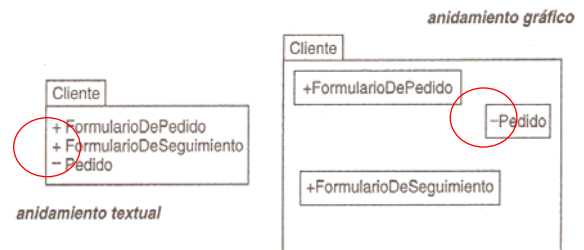


- Un paquete forma un *espacio de nombres*
- Los paquetes pueden contener a otros paquetes (árboles de agregaciones de paquetes – no recomendable más de 3 niveles)



Diagramas de Paquetes: Elementos

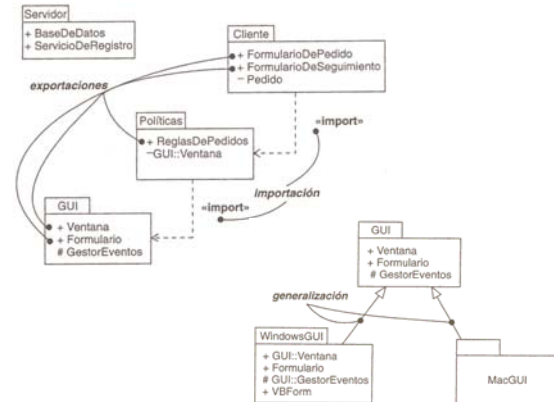
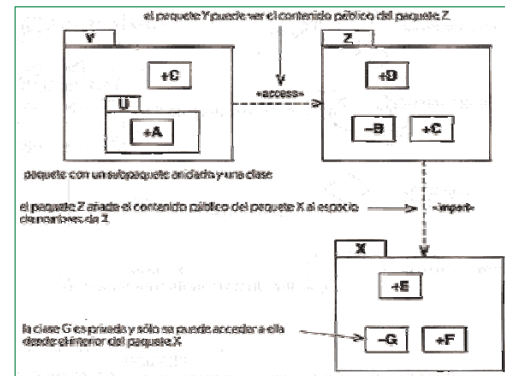
- **Visibilidad elementos:**
 - - *Privado*: No visibles por otro paquete
 - # *Protegido*: Visible por los paquetes que heredan del paquete contenedor
 - + *Público*: Visible a los contenidos de cualquier paquete que importe al paquete contenedor
- **Relaciones:** Dos clases en paquetes diferentes, sin relación son dos clases aisladas.
 - **Importación:** <<Import>>
 - No Transitivas
 - Indican que el paquete origen tiene acceso al de destino
 - **Acceso:** <<Access>>
 - **Exportaciones:** modeladas indicando la visibilidad pública en los elementos de un paquete
 - No se exporta explícitamente a algún paquete
 - Público, para que cualquier otro paquete pueda importarlo
 - **Generalizaciones**





Diagramas de Paquetes: Relaciones

- **Importación:** <<Import>> añade el contenido del destino al espacio de nombres público del origen (paquete importador)
 - No hay que calificar los nombres
 - Posibilidad de colisión de nombres
- **Acceso:** <<Access>> añade el contenido del destino al espacio de nombres privado del origen
 - No se pueden reexportar los elementos importados si un tercer paquete importa el origen
- **Generalizaciones:** parecida a la generalización entre clases
 - Un paquete especializado puede usarse donde quiera usarse un paquete más general.



María Sierra - IS1

P5.2.5



Diagramas de Paquetes: Estereotipos

- UML incluye varios estereotipos predefinidos para algunas categorías especiales de paquetes:
 - **System** - Paquete que representa el sistema completo que se está modelando
 - **Subsystem** - Expresa que el paquete es independiente del sistema completo que se modela
 - **Framework** - Es un paquete de patrones
 - **Facade** (fachada) - Proporciona una vista simplificada del paquete
 - **Stub** - Un paquete sirve de sustituto para el contenido público de otro paquete

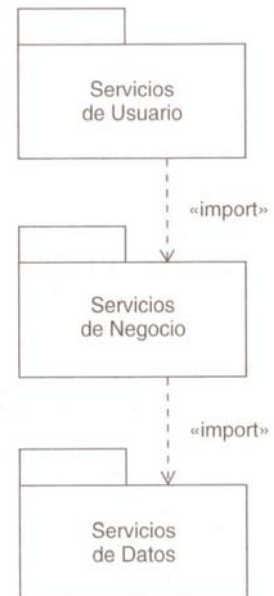
María Sierra - IS1

P5.2.6



Diagramas de Paquetes: **Modelado**

- El uso más habitual será la *agrupación de los elementos de modelado*
- En aplicaciones sencillas no será necesario
- **Modelado de grupos de elementos:**
 1. Examinar los modelos para identificar grupos de elementos cercanos semántica o conceptualmente
 2. Englobar cada grupo en un paquete
 3. Para cada paquete diferenciar los elementos a los que se podrá acceder desde fuera (públicos frente a protegidos o privados)
 4. Hay que conectar los paquetes que dependan de otros por importaciones
 5. Si hay familias de paquetes es necesario utilizar la generalización



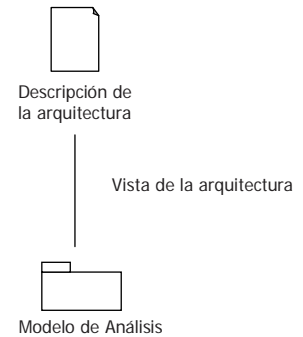
Diagramas de Paquetes: **Modelado**

- **Consejos y Sugerencias**
 - **Un paquete bien estructurado:**
 - Es Cohesivo
 - Está poco acoplado, exportando sólo los elementos que otros paquetes necesitan ver realmente, e importando sólo aquellos elementos necesarios y suficientes para que los elementos del paquete hagan su trabajo
 - No está profundamente anidado
 - Posee un conjunto equilibrado de elementos



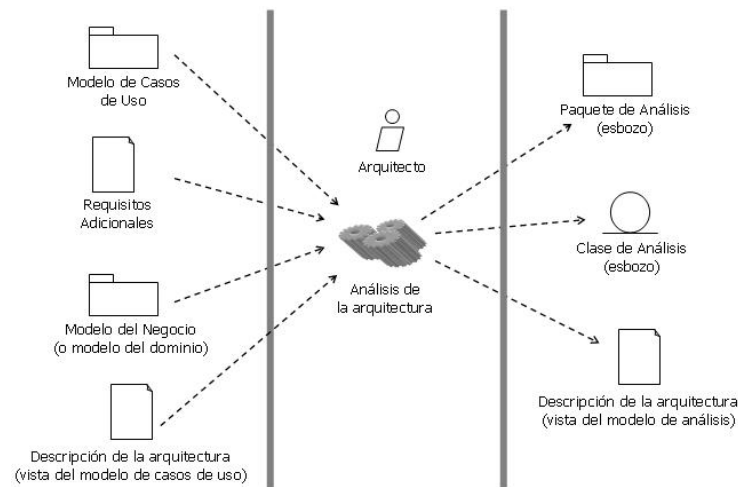
Arquitectura (Vista Modelo de Análisis)

- **Descripción de la arquitectura**
 - Contiene una *Vista de la Arquitectura del Modelo de Análisis*, que muestra los artefactos significativos para la arquitectura: *paquetes de análisis y sus dependencias, clases de análisis y realizaciones de casos de uso*
- **Análisis de la Arquitectura**
 - **Propósito:** esbozar el modelo de análisis y la arquitectura mediante la identificación de paquetes del análisis, clases del análisis evidentes, y requisitos especiales comunes
 - **Paquetes de Análisis:** permiten organizar el modelo de análisis en piezas más pequeñas y manejables. Se identifican inicialmente como una forma de dividir el trabajo de análisis, o a medida que el modelo de análisis evoluciona y “crece” convirtiéndose en una gran estructura que debe descomponerse.



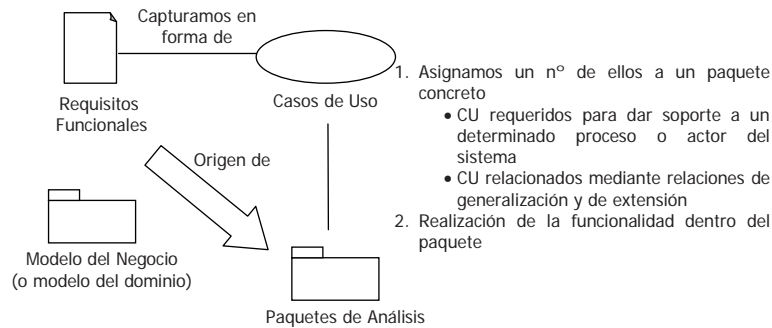
Análisis de la Arquitectura

- **Actividad 1: Identificación de paquetes de análisis**





Análisis de la Arquitectura



Los paquetes localizan los cambios en un proceso del negocio, en el comportamiento de un actor, y en un conjunto de casos de uso estrechamente relacionados



Análisis de la Arquitectura

• Ejemplo Identificación de paquetes de análisis

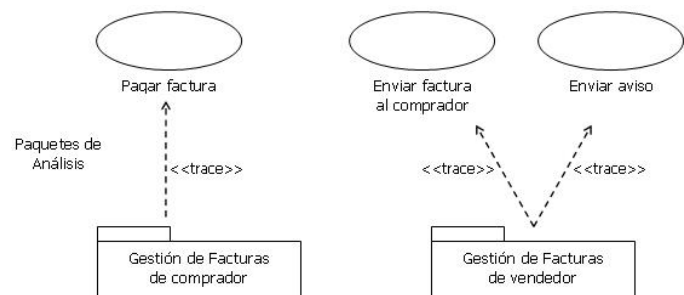
▪ Empresa venta de Software por Internet

- Casos de uso "Pagar Factura", "Enviar Aviso" y "Enviar Factura al Comprador" están implicados en el mismo proceso de negocio, "Ventas: del pedido a la entrega" → pueden incluirse en el mismo paquete de análisis
- La empresa además ha de ser capaz de poner su sistema a disposición de diferentes cliente con diferentes necesidades: algunos clientes utilizan el sistema sólo como compradores, otros sólo como vendedores, y algunos como compradores y vendedores

Separar la realización de los casos de uso necesarios para el vendedor de la realización de los casos de uso necesarios para el comprador → dos paquetes de análisis que pueden distribuirse separadamente a los clientes dependiendo de sus necesidades: "Gestión Facturas Comprador" y "Gestión Facturas Vendedor"



Solución



Observación: hay otros casos de uso que soportan el proceso de negocio "Ventas: del Pedido a la Entrega", se han ignorado para simplificar el ejemplo



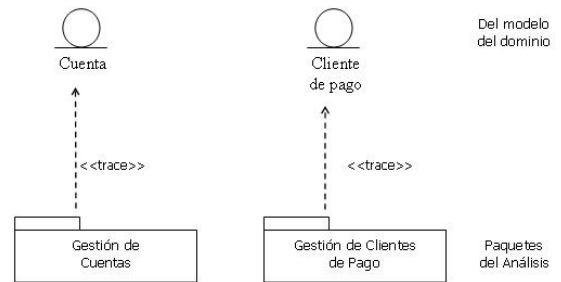
Análisis de la Arquitectura

• *Gestión de aspectos comunes entre paquetes de análisis*

- Ejemplo: cuando dos o más necesitan compartir la misma clase de análisis
- Extraer la clase compartida, colocarla dentro de su propio paquete o simplemente fuera de cualquier otro paquete, y hacer después que los paquetes sean dependientes de este paquete o clase más general
 - Las clases compartidas de este tipo son probablemente clases de entidad de las cuales se puede hacer una traza hasta clases del dominio o clases de entidad del negocio si están compartidas, y si se van a identificar inicialmente los paquetes en el análisis

• *Ejemplo: Identificación de paquetes de análisis generales* (a partir del modelo de dominio)

- En el sistema del ejemplo anterior las clases de dominio "Cliente de Banco" y "Cuenta" representan entidades importantes y complejas del mundo real, estando además compartidas por otros paquetes del análisis más específicos



Crear para cada una de ellas un paquete aparte: "Gestión de Cuentas" y "Gestión de Clientes de Banco"

María Sierra - IS1

P5.2.13

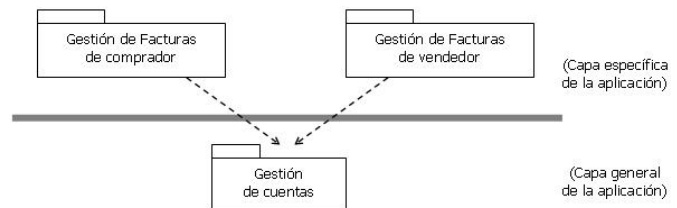


Análisis de la Arquitectura

• *Dependencias entre paquetes de Análisis*

- Cuando sus contenidos están relacionados (la dirección de la dependencia debe ser la misma que la de la relación)
- **Objetivo:** conseguir paquetes independientes y débilmente acoplados pero que posean cohesión interna alta.
 - Son más fáciles de mantener (el cambio en algunas clases del paquete afectará fundamentalmente a clases dentro del propio paquete) → Reducir el numero de relaciones entre clases en paquetes diferentes reduce las dependencias entre paquetes.
 - Para clarificar las dependencias puede ser útil estratificar el modelo de análisis haciendo que los paquetes específicos de la aplicación queden en una capa de nivel superior y los paquetes generales en una capa inferior. Esto aclara la diferencia entre funcionalidad específica y funcional

En el sistema del ejemplo anterior, el Paquete "Gestión de Cuentas" contiene varias clases utilizadas por clases de otros paquetes. Ej.: la clase "cuenta" se utiliza por clases en los paquetes "Gestión de Facturas de Comprador" y "Gestión de Facturas de Vendedor", por tanto estos paquete dependerán del paquete "Gestión de Cuentas"



Durante el diseño y la implementación, se refinan estas capas, añadiendo más capas de bajo nivel a medida que consideramos el entorno de implementación y los requisitos no funcionales

María Sierra - IS1

P5.2.14



Análisis de la Arquitectura

• **Actividad 2: Identificación de clases de entidad obvias**

- Propuesta preliminar de entidades más importantes (10 ó 20) basadas en las de dominio o de negocio, identificadas en la captura de requisitos y en la realización de los casos de uso
 - Esbozo de las más significativas para la arquitectura, no es necesario llegar a un gran detalle
- Ejemplo:** Una clase entidad identificada a partir de una clase del dominio. "Factura" era una clase del dominio, podemos proponerla como clase entidad inicial, manteniendo sus atributos y asociaciones

• **Actividad 3: Identificación de Requisitos especiales comunes**

- Req. Especial**, aparece en análisis (al explorar la realización de casos de uso y las clases de análisis) y debe ser tratado adecuadamente en diseño e implementación. Ej.: limitaciones o restricciones sobre persistencia, distribución y concurrencia, características de seguridad, tolerancia a fallos, gestión de transacciones
- Deben indicarse las **características de cada requisito especial** común
 - Ej., un requisito de persistencia tiene las siguientes características: Rango de tamaño, Volumen, Periodo de persistencia, Frecuencia de actualización, Fiabilidad
 - Se calificarán para cada clase o realización de caso de uso que haga referencia al requisito especial

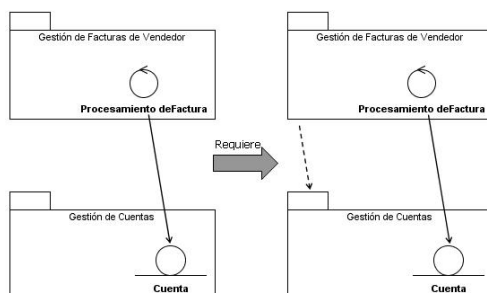


Análisis de la Arquitectura

• **Análisis de un Paquete**

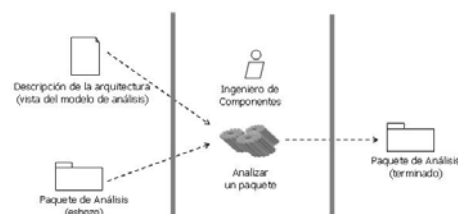
■ **Objetivos**

- Garantizar que el paquete de análisis es tan independiente de otros paquetes como sea posible
- Garantizar que el paquete de análisis cumple su objetivo de realizar algunas clases de dominio o casos de uso
- Describir las dependencias de forma que pueda estimarse efecto de los cambios futuros



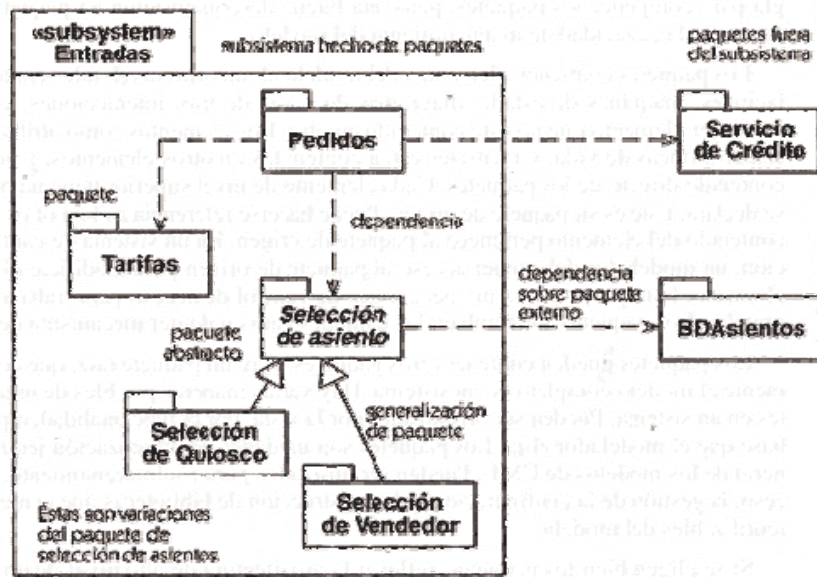
■ **Normas Generales**

- Definir y mantener las dependencias del paquete con otros paquetes cuyas clases contenidas estén asociadas con él
- Asegurarnos de que el paquete contiene las clases correctas. Intentar hacer cohesivo el paquete incluyendo sólo objetos relacionados funcionalmente
- Limitar las dependencias con otros paquetes. Considerar la reubicación de aquellas clases contenidas en paquetes que son demasiado dependientes de otros paquetes





Ejemplos

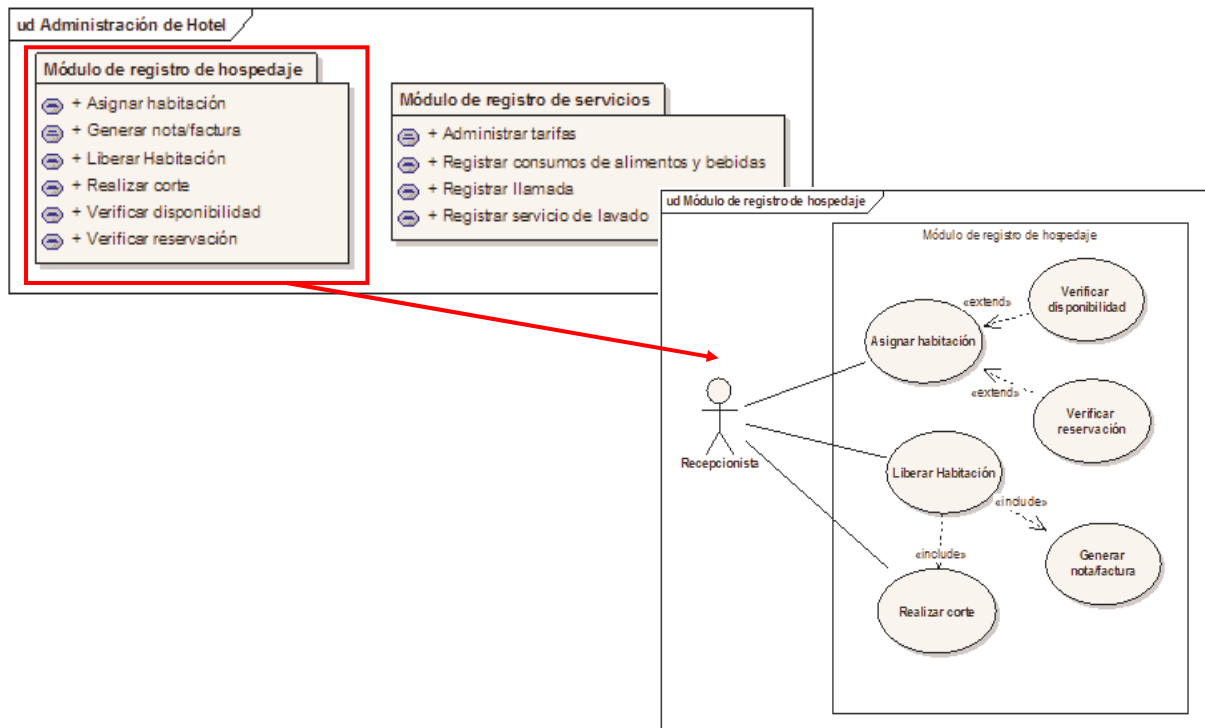


María Sierra - IS1

P5.2.17



Ejemplos



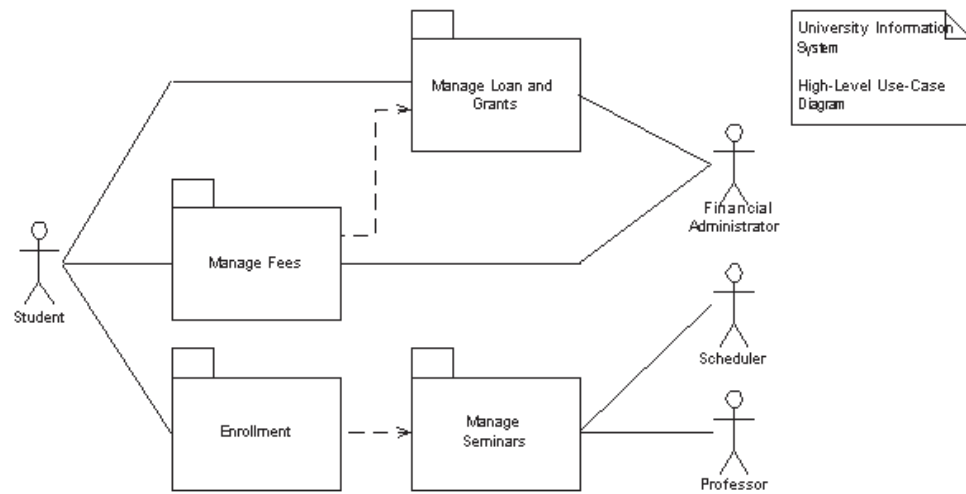
María Sierra - IS1

P5.2.18



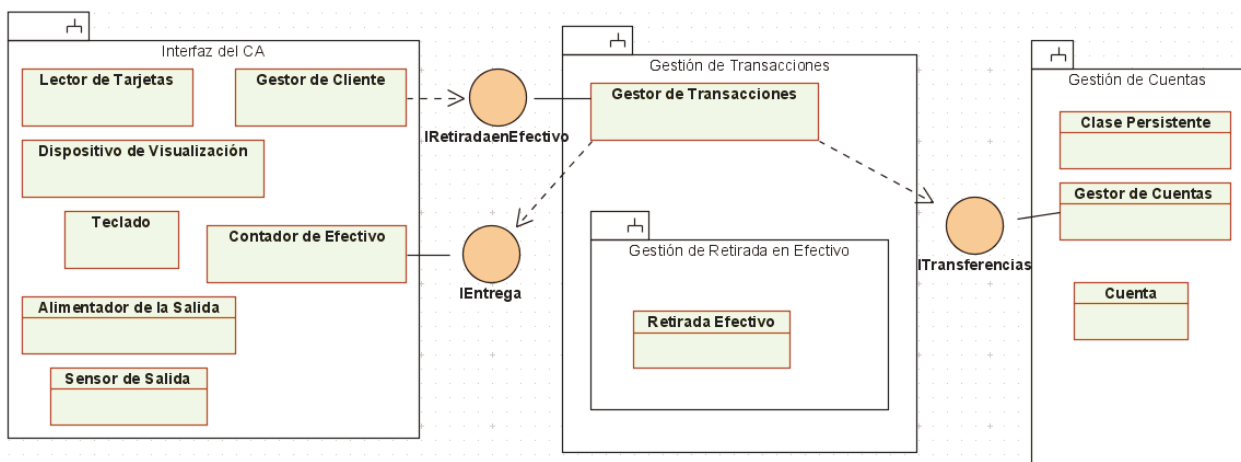
Ejemplos

DCU simplificado con el uso de Paquetes



Ejemplos

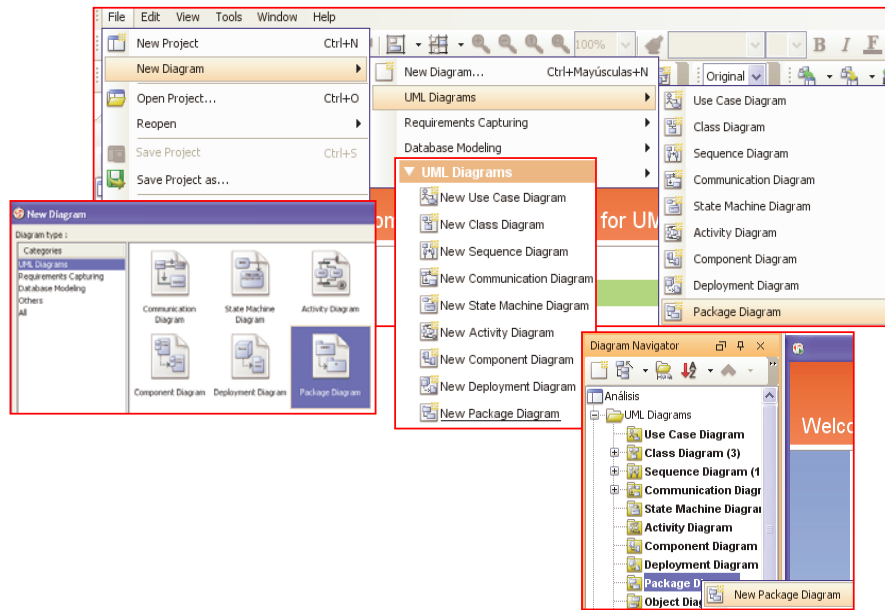
Arquitectura de 3 capas





Diagramas de Paquetes con VP

Crear Diagrama



María Sierra - IS1

P5.2.21

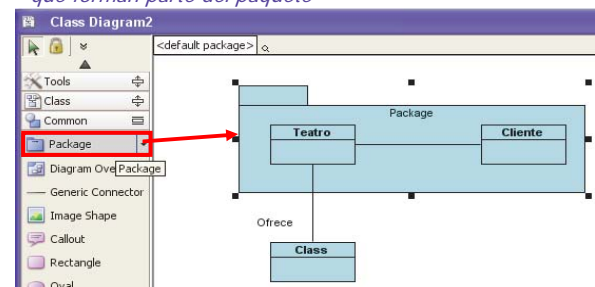
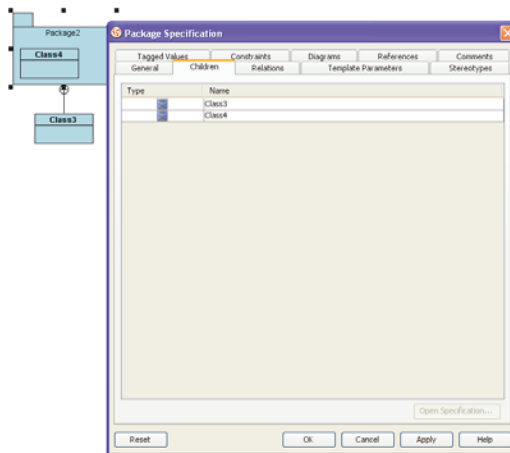


Diagramas de Paquetes con VP

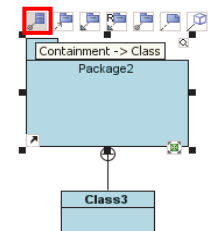
Crear Paquetes

Desde el diagrama de Clases

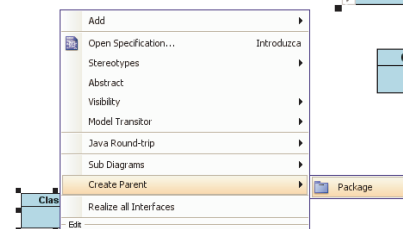
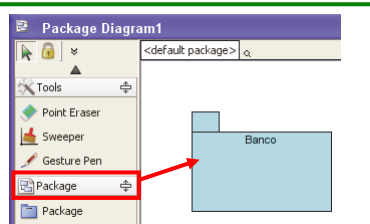
1. Crear el paquete, después la clase y arrastrarla hasta introducirla dentro del paquete
2. Pinchar en paquete y colocar encima de las clases que forman parte del paquete



3. Crear el paquete y luego (pinchando en Containment -> class) crear las clases que hay dentro
4. Crear el paquete desde la clase



Desde Diagrama de Paquete



María Sierra - IS1

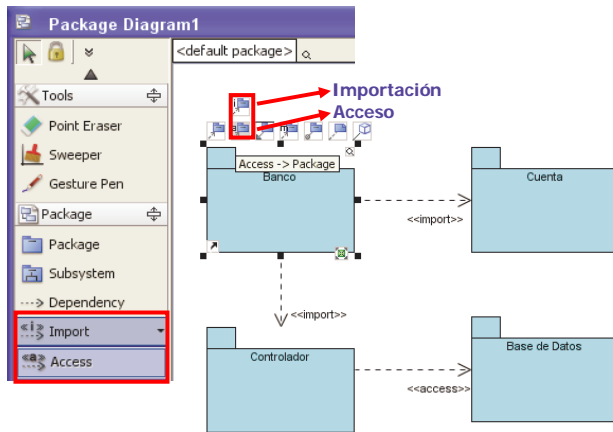
P5.2.22



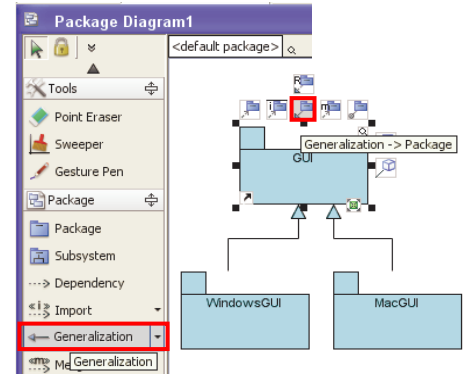
Diagramas de Paquetes con VP

Relaciones

Importación y Acceso



Generalización



Botón derecho y dibujar paquete y relación entre paquetes

