Ingeniería de Software I (semana 6-7)

Semana 6:

Análisis de la Arquitectura del Sistema:

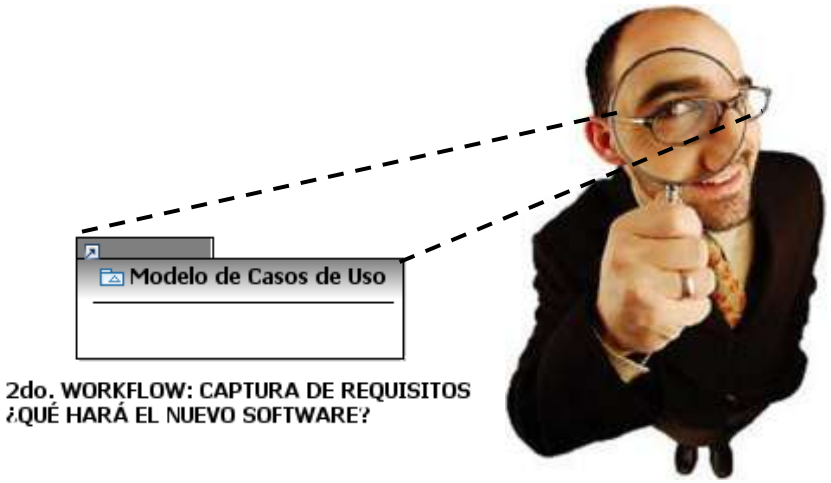
Paquetes de Análisis (PA)

Identificación de Paquetes de Análisis

1.1. Análisis Orientado a Objetos (AOO)

Objetivo

Analizar los requisitos que fueron descritos en el Modelo de Casos de Uso del Sistema (MCUS), logrando **comprender de manera más precisa el sistema**, logrando una descripción refinada que sea fácil de mantener y que ayude a estructurar el sistema.

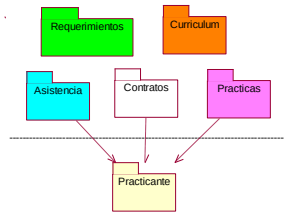


1.2. Actividades del Análisis OO (AOO)

Durante el análisis …
Se identifica continuamente nuevos paquetes, clases y requisitos comunes.
Se refinan y mantienen continuamente los paquetes de análisis concretos.

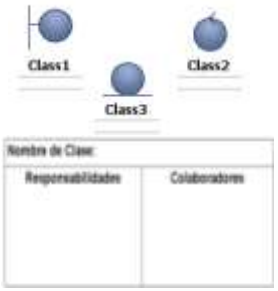
Se realizan las siguientes actividades

Análisis de la Arquitectura

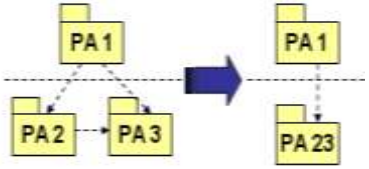


Análisis de los Casos de Uso de Sistema

Análisis de Clases



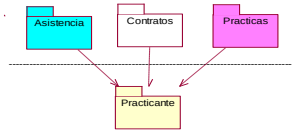
Análisis de paquetes



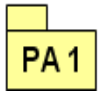
1.3. Artefactos del Análisis OO (AOO)

- Modelo de análisis

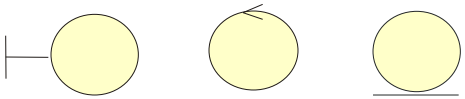
- Arquitectura de análisis



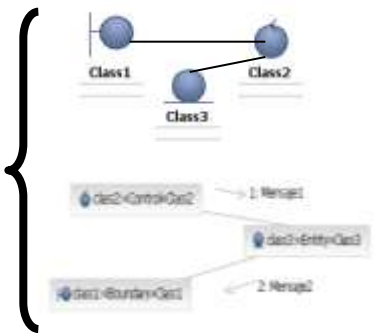
- Paquetes de análisis



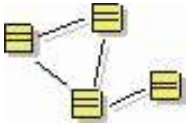
- Clases de análisis



- Realizaciones de análisis



- Modelo Conceptual



1.4. Modelo de Análisis

es

Una abstracción del Modelo de Diseño

Su utilidad radica en que

Permite una ***apreciación global conceptual del sistema*** porque es un primer intento por definir los conceptos claves que describen el sistema.

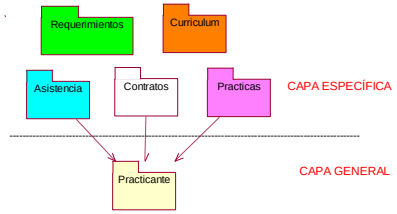
2.1. Análisis de la Arquitectura

Propósito

Esbozar el modelo de análisis y su arquitectura

mediante la identificación

Paquetes de análisis y dependencias



Clases de entidad evidentes

Requisitos especiales comunes

SRS

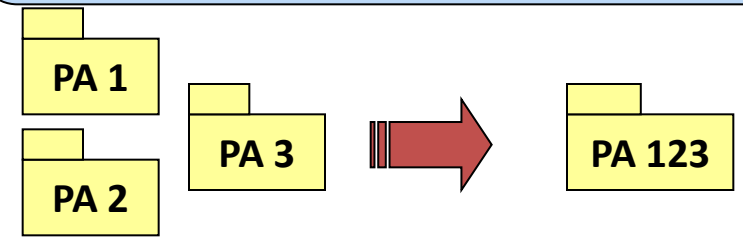
2.2. Paquetes de Análisis

permiten

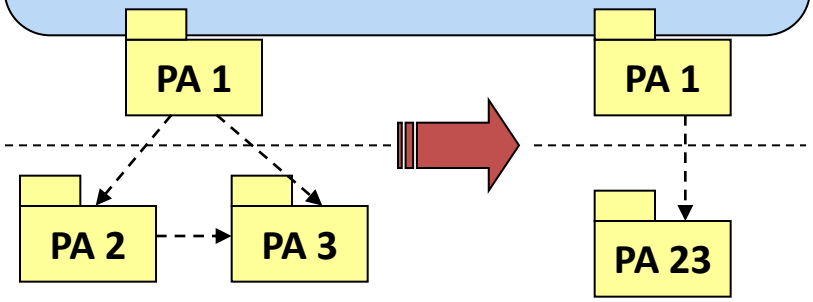
Organizar los artefactos del modelo de análisis (MA) en piezas más manejables.

Características

ALTAMENTE COHESIVOS
(sus contenidos deberían estar fuertemente relacionados)

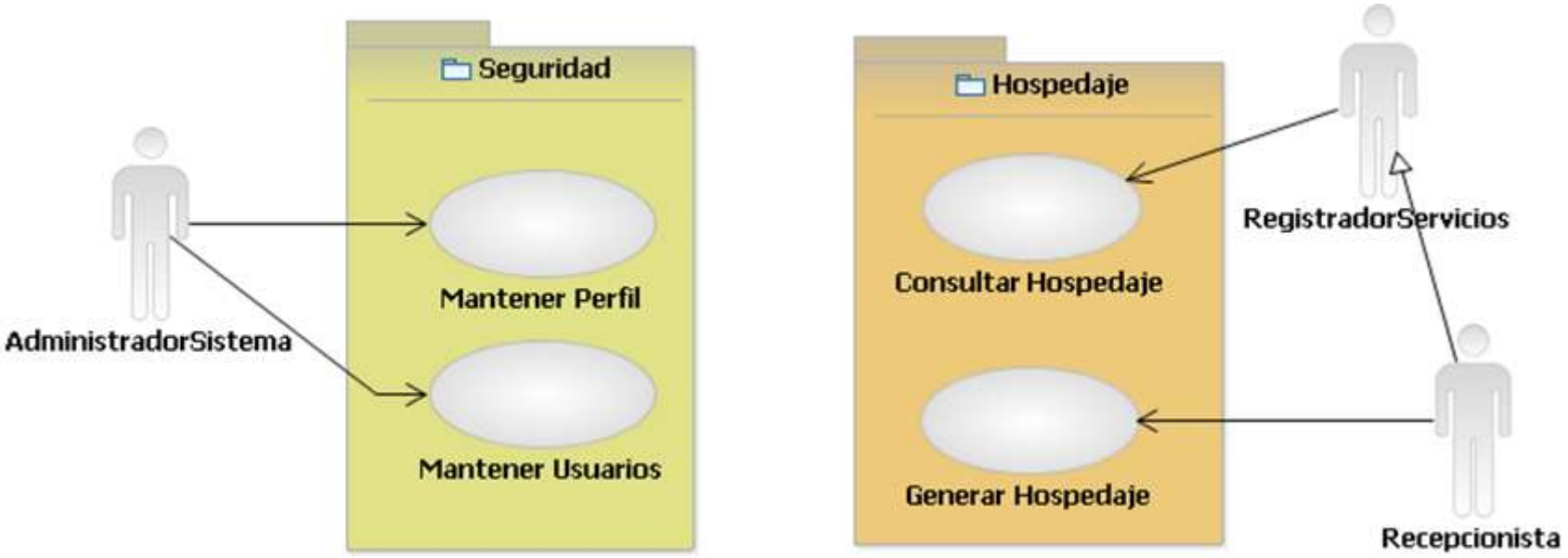


DÉBILMENTE ACOPLADOS
(dependencias entre paquetes deberían minimizarse)



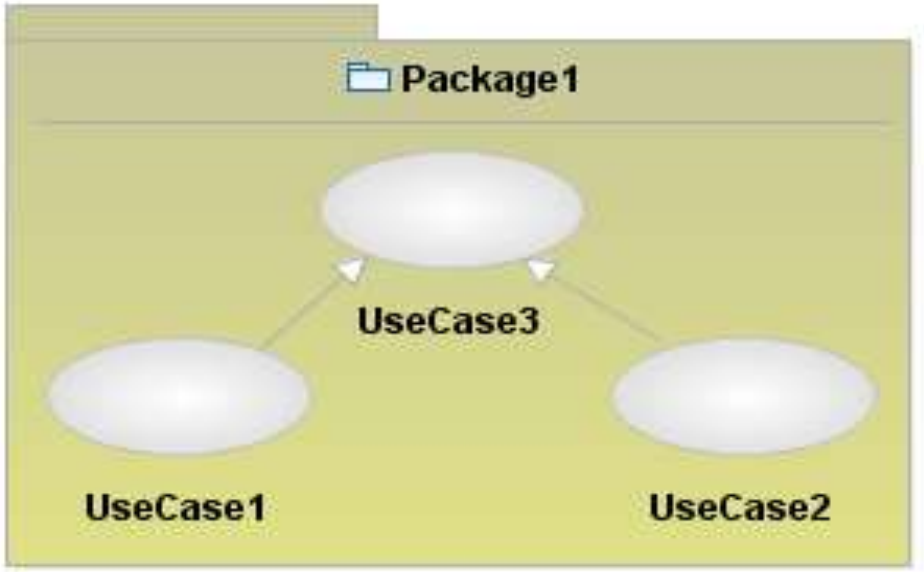
2.3. Identificación de Paquetes de Análisis

CONTENIDOS ESTRECHAMENTE RELACIONADOS
EN UN MISMO PAQUETE
(para dar soporte a un proceso de negocio o a un determinado actor)



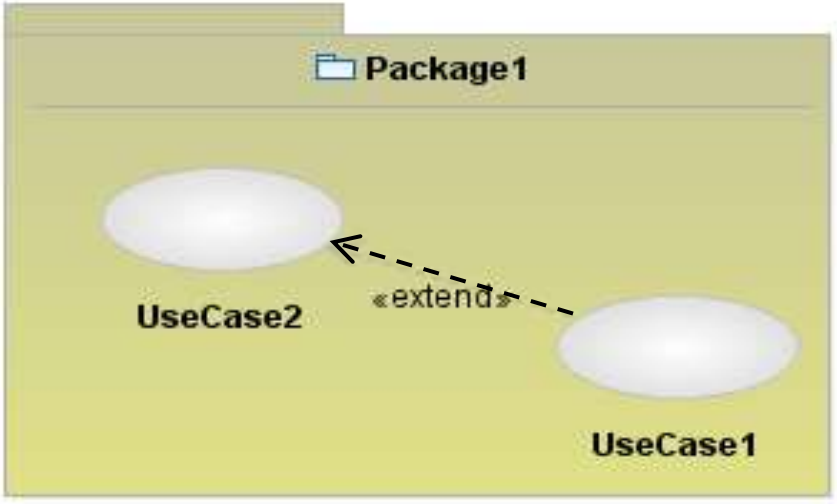
2.3. Identificación de Paquete de Análisis (Cont.)

CASOS DE USO RELACIONADOS CON GENERALIZACIÓN
EN UN MISMO PAQUETE



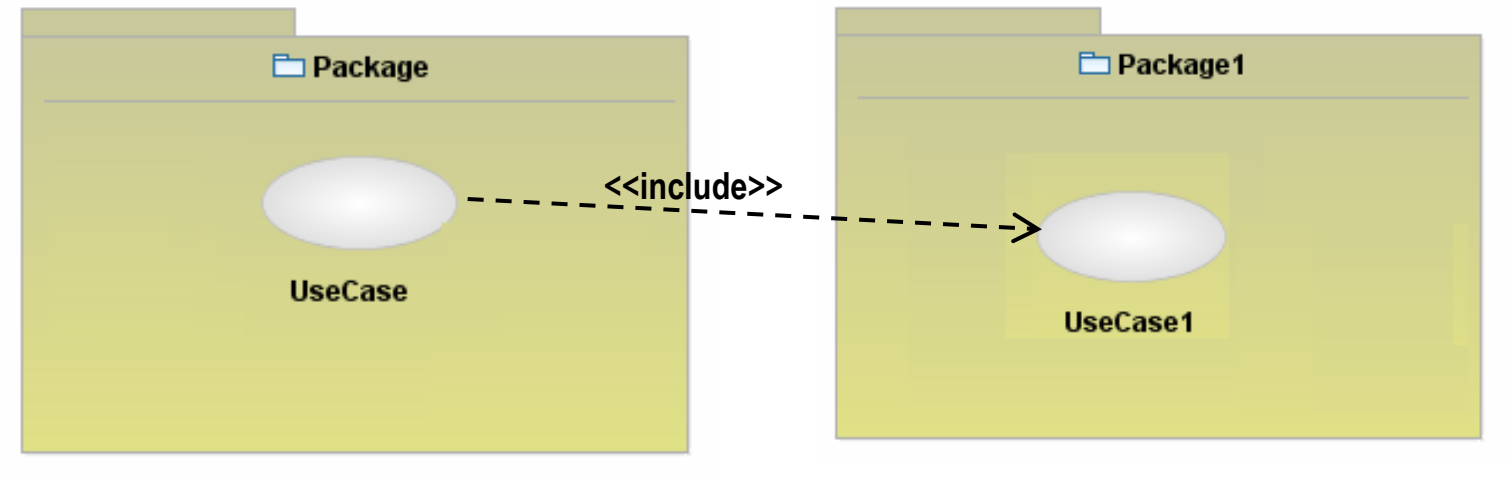
2.3. Identificación de Paquetes de Análisis (Cont.)

CASOS DE USO RELACIONADOS CON <<EXTEND>>
EN UN MISMO PAQUETE
(CU extendidos que sólo se extienden a partir de un CU base)



2.3. Identificación de Paquetes de Análisis (Cont.)

CASOS DE USO RELACIONADOS CON <<INCLUDE>>
EN DIFERENTES PAQUETES
(CU incluidos con contenidos diferentes al CU base)



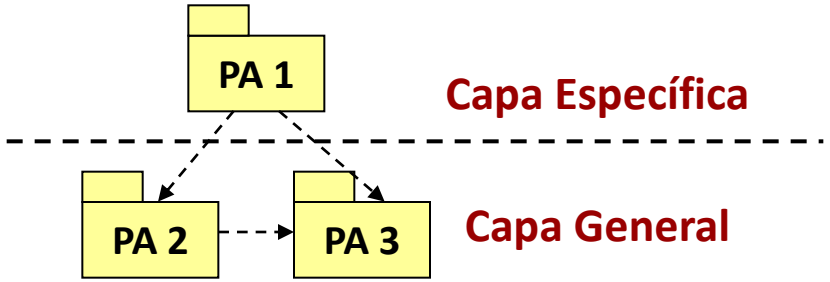
2.4. La Capa de Aplicación

Los paquetes identificados se organizarán en la **Capa de Aplicación**

Capas Internas

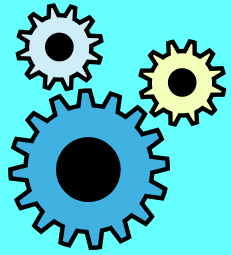
CAPA ESPECÍFICA
(Contienen CU que manejan datos transaccionales)

CAPA GENERAL
(Contienen CU que manejan datos maestros y de dominio)



CASO

A partir del Diagrama de Casos de Uso del Sistema, identifique los paquetes de análisis y ubíquelos en la capa correspondiente de la arquitectura de análisis.



Ejercicio

Semana 7

Verificación y Validación

Verificación y Validación de Software

Es un conjunto de procesos de comprobación y análisis que aseguran que el software que se desarrolla está acorde a su especificación y cumple las necesidades de los clientes.
Los objetivos de las actividades de verificación y validación son valorar y mejorar la calidad de los productos del trabajo generados durante el desarrollo y modificación del software.

Los atributos de la calidad deben ser la corrección, la perfección, la consistencia, la confiabilidad, la utilidad, la eficacia, el apego a los estándares y la eficacia de los costos totales.

Verificación y Validación de Software

Hay dos tipos de verificación: formal y del ciclo de vida. Esta última consiste en el proceso de determinar el grado de los productos de trabajo de una fase dada del ciclo de desarrollo cumplen con las especificaciones establecidas durante las fases previas. La verificación formal es una rigurosa demostración matemática de la concordancia del código fuente con sus especificaciones.

La validación es la evaluación del software al final del proceso de desarrollo del software para determinar su conformidad con los requisitos IEEE.

Verificación y Validación de Software

La verificación y validación implican la valoración de los productos de trabajo para determinar el apego a las especificaciones, incluyen las especificaciones de requisitos, la documentación del diseño, diversos principios generales de estilo, estándares del lenguaje de instrumentación, estándares del proyecto, estándares organizacionales y expectativas del usuario, al igual que las meta especificaciones para los formatos y notaciones utilizadas en la especificación de productos diversos.

Verificación y Validación de Software

¿Qué es verificación?

La verificación se enfoca más al proceso de evaluación del sistema o componentes ya que permite determinar si los productos de una determinada fase del desarrollo satisfacen las condiciones impuestas en el inicio de la etapa.

¿Qué se debe tener en la verificación?
Consistencia: vigilar que la información sea coherente.
Precisión: corrección de la sintaxis.
Completitud: lagunas en capacidad deductiva.

Lo que se hace en la verificación:
Identifica desviaciones con estándares y requerimientos.
Recolecta datos para mejorar el proceso.
Verifica que el producto:
- Cumpla con los requerimientos.
- Cumpla con los atributos de calidad.
- Se ajuste a las regulaciones, estándares y procedimientos definidos.

Verificación y Validación de Software

¿Qué es validación?

La validación también es una evaluación del sistema o de componentes, solo que es en el transcurso o al final del proceso del desarrollo, donde se determina si cumple con lo especificado.

Aspectos en la validación:
Construir el sistema correcto.
Evaluar la conformidad con la especificación de requisitos.

¿Consultas?