

ICONIX

Dra. Ángeles Sumano López
Dr. Juan Manuel Fernández Peña

Enero 2012



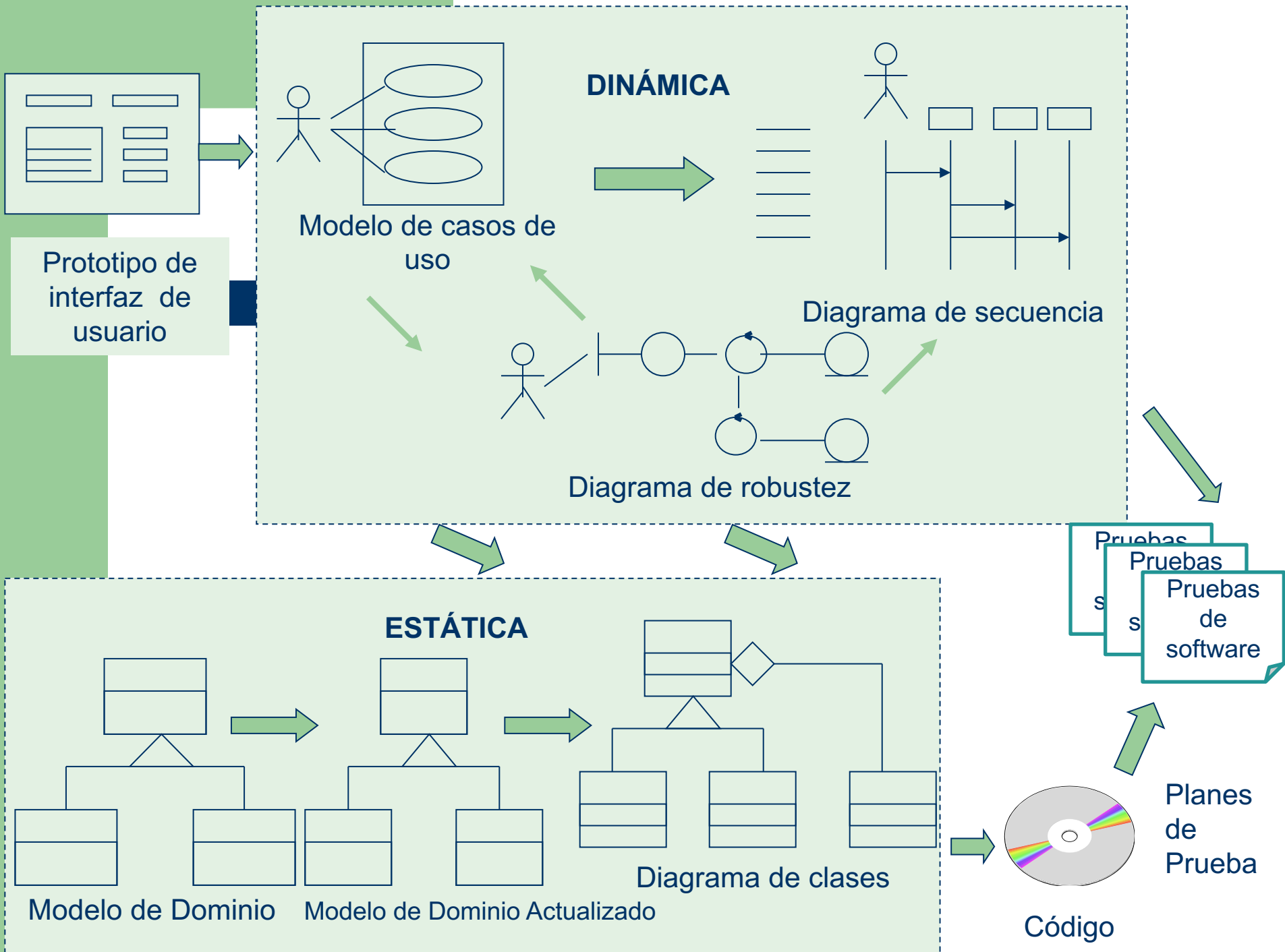
Metodología ICONIX

Referencias

- El método original se encuentra en:
 - Doug Rosenberg with Kendall Scott
 - “Use case driven object modeling with UML. A practical approach”
 - Addison Wesley, 1999
- Nueva Edición
 - Doug Rosenberg with Matt Stephens
 - “Use case driven object modeling with UML. Theory and Practice”
 - Apress, 2007
- Más información en la página:
 - <http://www.iconixsw.com>

Enfoque ICONIX

- Modelado de objetos conducido por casos de uso.
- Centrado en datos: se descompone en fronteras de datos.
- Basado en escenarios que descomponen los casos de uso.
- Enfoque iterativo e incremental.
- Ofrece trazabilidad.
- Uso directo de UML (estándar del Object Management Group).



Preguntas iniciales

- ¿Quiénes son los usuarios (actores) del sistema y qué tratan de hacer?
- ¿Cuáles son los objetos del mundo real (dominio del problema) y las asociaciones entre ellos?
- ¿Qué objetos son necesarios para cada caso de uso?
- ¿Cómo colaboran los objetos en cada caso de uso?
- ¿Cómo se manejan aspectos de tiempo real?
- ¿Cómo se construirá realmente el sistema a nivel de piezas?

Características

- Flexible para diferentes estilos y clases de problemas.
- Apoyo a la manera de trabajo de la gente.
- Guía para los menos experimentados.
- Expone los productos anteriores al código de manera estándar y comprensible.

I Análisis de requerimientos

META: revisión de requerimientos

- Pasos principales

- Identificar objetos del dominio y relaciones de agregación y generalización.
- Prototipo rápido.
- Identificar casos de uso.
- Organizar casos de uso en grupos (paquetes).
- Asignar requerimientos no funcionales a casos de uso y objetos del dominio.

II Análisis y diseño preliminar

META: revisión del diseño preliminar

- Pasos principales
 - Escribir descripciones de casos de uso
 - cursos básico y alternos
 - Análisis de robustez
 - Identificar grupos de objetos que realizan escenario
 - Actualizar diagramas de clases del dominio
 - Finalizar diagramas de clases

III Diseño

META: revisión crítica del diseño

- Pasos principales
 - Asignar comportamiento
 - Para cada caso de uso
 - Identificar mensajes y métodos
 - Dibujar diagramas de secuencia
 - Actualizar clases
 - (opcional) diagramas de colaboración
 - (opcional) Diagramas de estados
 - Terminar modelo estático
 - Verificar cumplimiento de requerimientos

IV Implementación

META: entregar el sistema

- Pasos principales
 - Producir diagramas necesarios
 - Despliegue
 - Componentes
 - Escribir el código
 - Pruebas de unidad e integración
 - Pruebas de sistema y aceptación basadas en casos de uso

Modelado del Dominio

- Dominio del problema: área que cubre las cosas y conceptos relacionados con el problema que el sistema deberá resolver
- Modelando el dominio: tarea de descubrir “objetos” (en realidad clases) que representan esas cosas y conceptos
- A partir de los datos asociados con requerimientos se llegará a construir modelo estático del dominio

Modelando el dominio

- Fuentes de información:
 - Descripción de alto nivel del problema
 - Requerimientos de bajo nivel
 - Conocimiento de expertos
 - Literatura

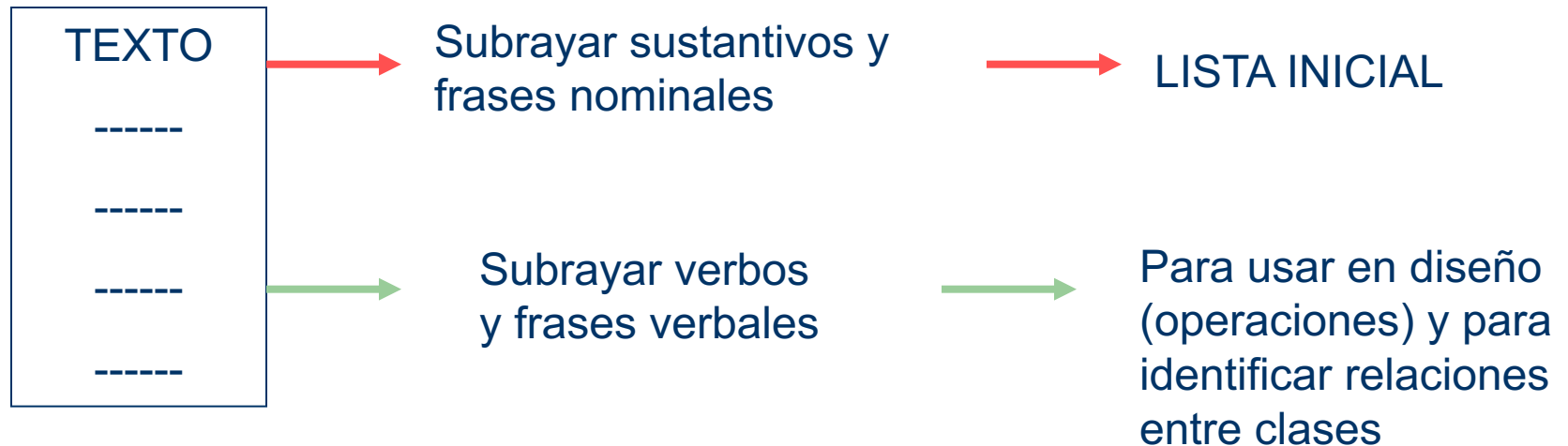
Modelando el dominio

- Procedimiento (I):
 - Tomar documentos disponibles y hacer una lectura rápida, subrayando los sustantivos y notando frases posesivas y verbos (uso posterior)
 - Los sustantivos y frases nominales se convertirán en objetos o atributos
 - Los verbos y frases verbales se convertirán en operaciones y relaciones
 - Las frases posesivas indican los sustantivos que son atributos y no objetos

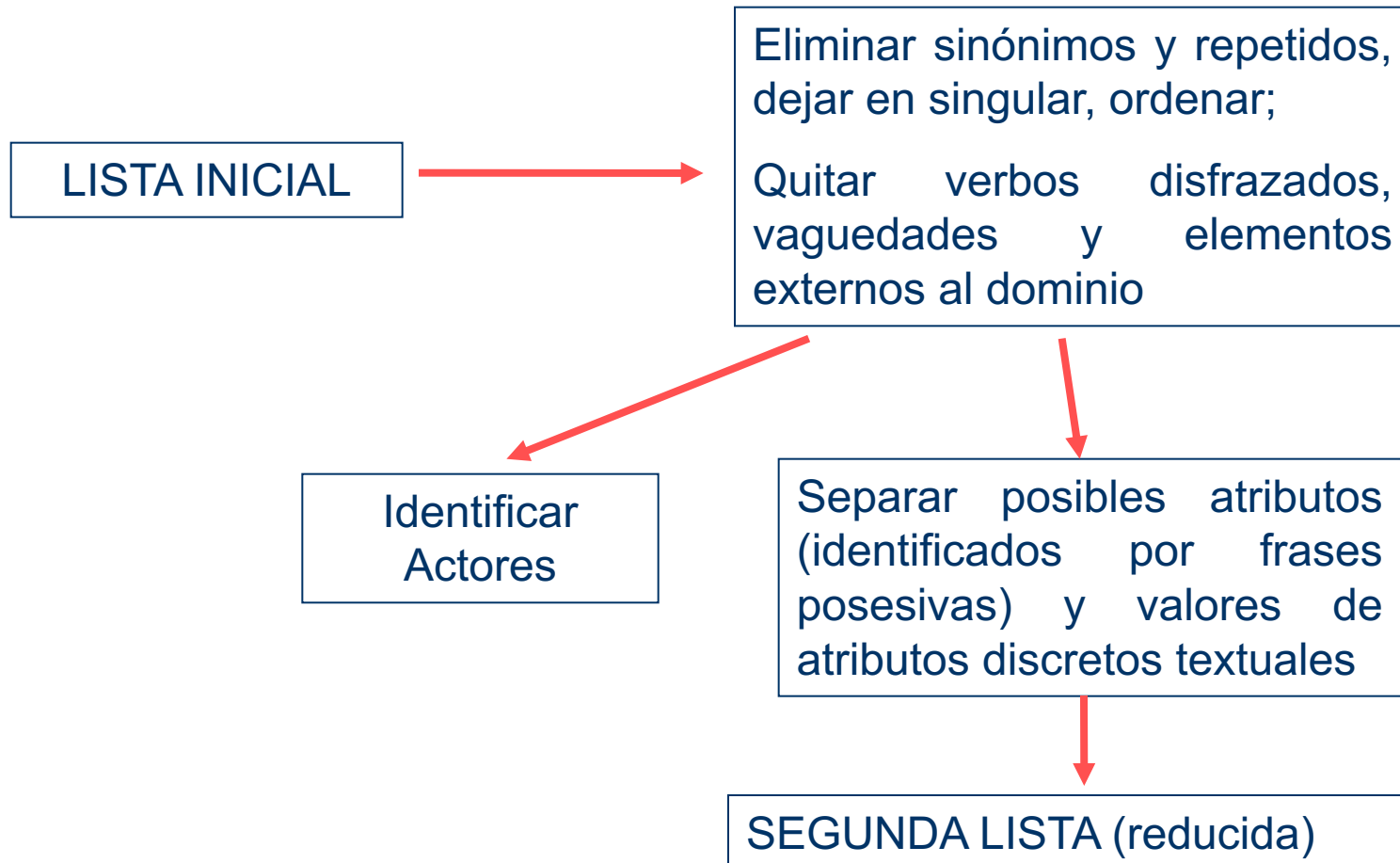
Modelando el dominio

- Procedimiento (II)
 - Formar una lista con los sustantivos y frases nominales identificados, evitando los plurales y las repeticiones y ordenándola alfabéticamente
 - Revisar la lista eliminando los elementos innecesarios (irrelevantes o redundantes) o incorrectos (vagos o conceptos fuera del alcance del modelo o representan acciones aún cuando parezcan sustantivos)
 - Volver a revisar textos, leyendo entre líneas

Modelado del dominio procedimiento



Modelado del dominio procedimiento



Modelando el dominio

- Procedimiento (III)
- Construir relaciones de generalización
 - Una generalización es una relación en la cual una clase es una generalización de otra. También se le llama “tipo-de” o “es-una”. La clase más general se llama Antecesor o Superclase y la otra (refinamiento de la primera) Descendiente o Subclase.
 - La subclase hereda los atributos y métodos de la superclase y las asociaciones en que participa. Las puede modificar.

Modelando el dominio

- Procedimiento (III)
- Establecer asociaciones entre clases
 - Una asociación es una relación estática entre dos clases; indican dependencia, pero no acción (aunque se las nombre con un verbo)
 - Deben ser persistentes (es modelo estático)

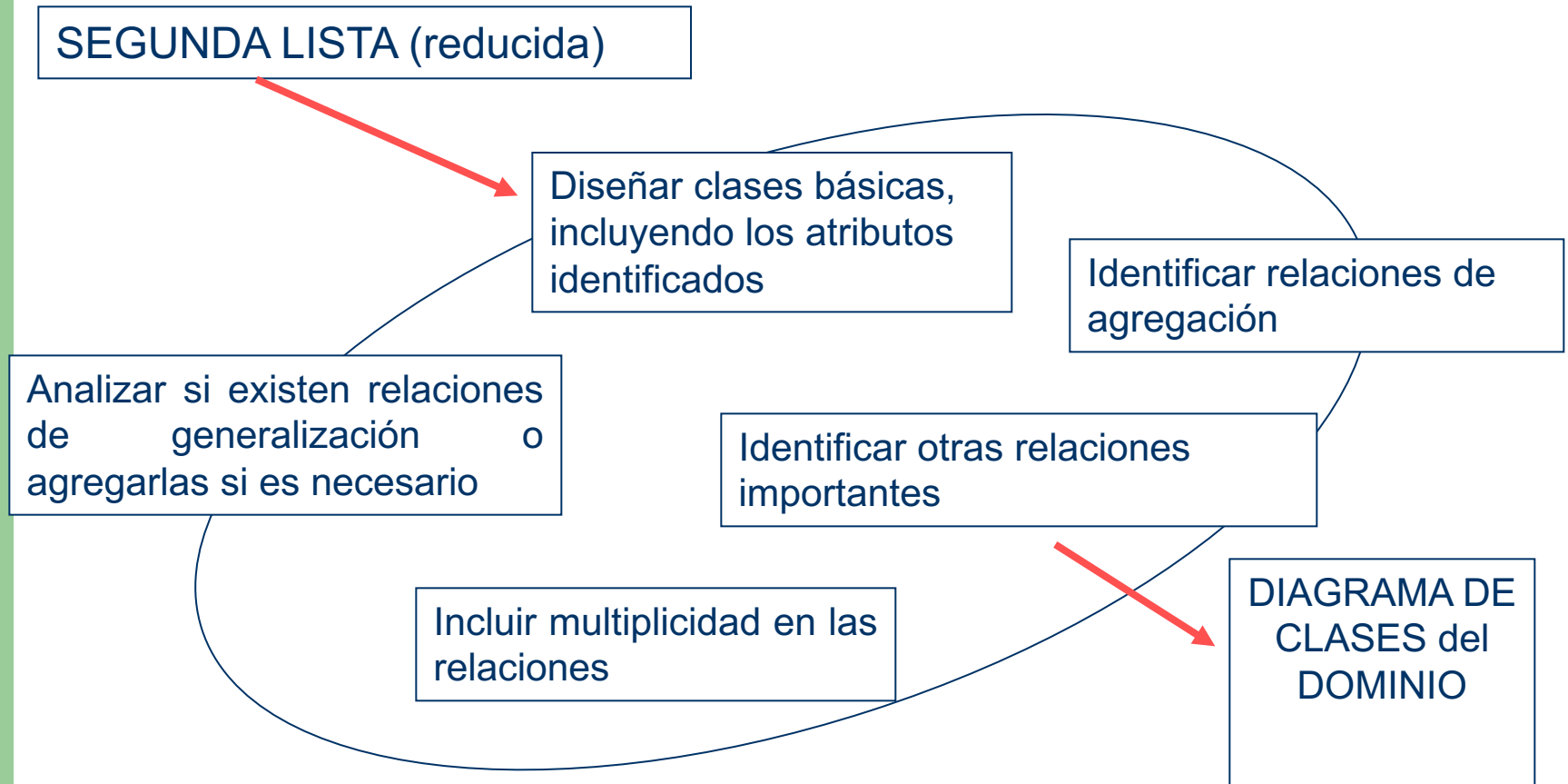
Modelando el dominio

- Procedimiento (III)
- Establecer relaciones de agregación
 - Una agregación es una relación en la cual una clase está formada por otras (sus partes)
 - A veces se le llama “parte-de”
 - En UML se distingue una forma más fuerte llamada Composición, pero para este método no se hará diferencia

Modelando el dominio

- Procedimiento (IV)
- Clases de asociación
 - Una clase de asociación es una variante de las asociaciones muy útil cuando hay relaciones muchas-a-muchas entre clases
- Pueden conseguirse clase del dominio a partir de entidades en bases de datos preexistentes
- Cuando una clase tiene demasiados atributos, conviene dividirla en clases auxiliares y usar agregación para reunir las

Modelado del dominio procedimiento



Advertencia

- No se tarde demasiado en preparar la lista; más adelante la refinará y completará

Casos de uso

- Buscan capturar los requerimientos del usuario para sistema nuevo
- Puede ser desde cero o a partir de un sistema anterior
- Especifica escenarios detallados de lo que hace el usuario para lograr sus fines
- Es la base de todo lo que sigue en este método y otros semejantes

Casos de uso

- Definición:
 - Un caso de uso es una secuencia de acciones que un actor (usualmente una persona, pero que puede ser una entidad externa, como otro sistema o un elemento de hardware) realiza dentro del sistema para lograr una meta

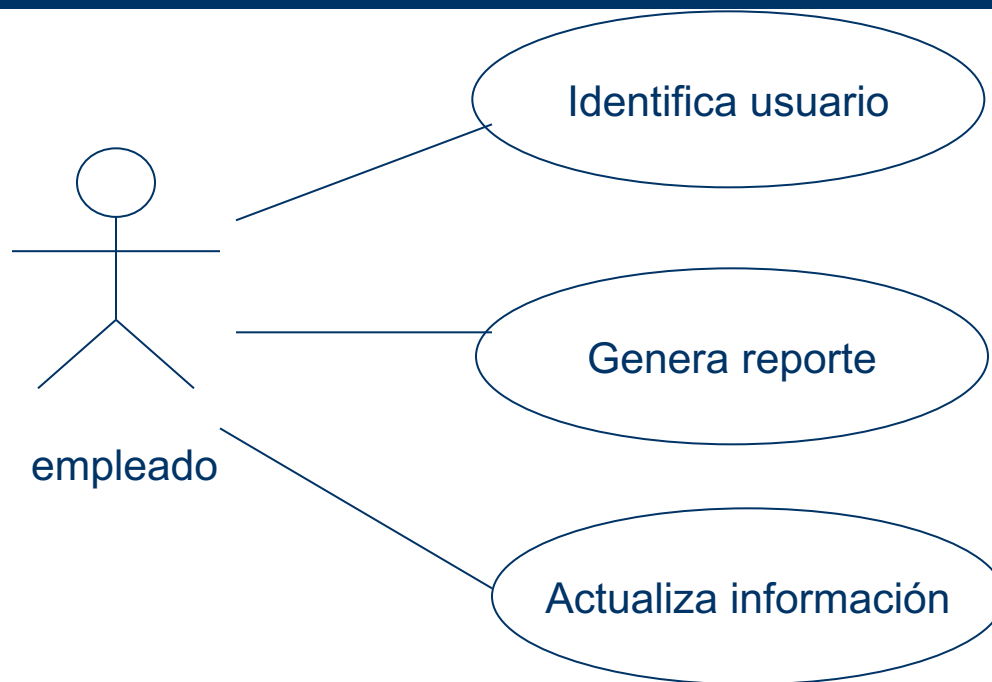
Casos de uso

- Se describe mejor como una frase verbal en presente y en voz activa.
- Ejemplos:
 - Admite paciente,
 - Realiza transacción o
 - Genera reporte
- Especifica de manera precisa, no ambigua, un aspecto del uso del sistema sin suponer un diseño o implementación particulares.
- Toda la funcionalidad del sistema debe estar expresada en casos de uso

Casos de uso

- Actor: es un papel realizado por: una persona, un sistema externo, un hardware.
- Los actores reflejan todas las entidades que deben intercambiar información con el sistema.
- Varias personas pueden realizar un mismo papel
- Una persona puede jugar varios papeles, en momentos distintos
- Diagrama de casos de uso: reúne actores y casos de uso

Casos de uso



Usualmente, actores van a derecha e izquierda, casos de uso al centro

No cambie símbolos, son parte de un estándar internacional

Casos de uso

- Algunos autores separan los actores en dos:
 - Primarios: los que inician casos de uso
 - Secundarios: responden a una necesidad del sistema que el software no puede resolver, no inician la acción.

Casos de uso

- Existen dos tipos de caso de uso:
 - De nivel de análisis: representa comportamiento común de un grupo de casos
 - De nivel de diseño: instancias del anterior, con comportamiento específico

Casos de uso, como escribirlos

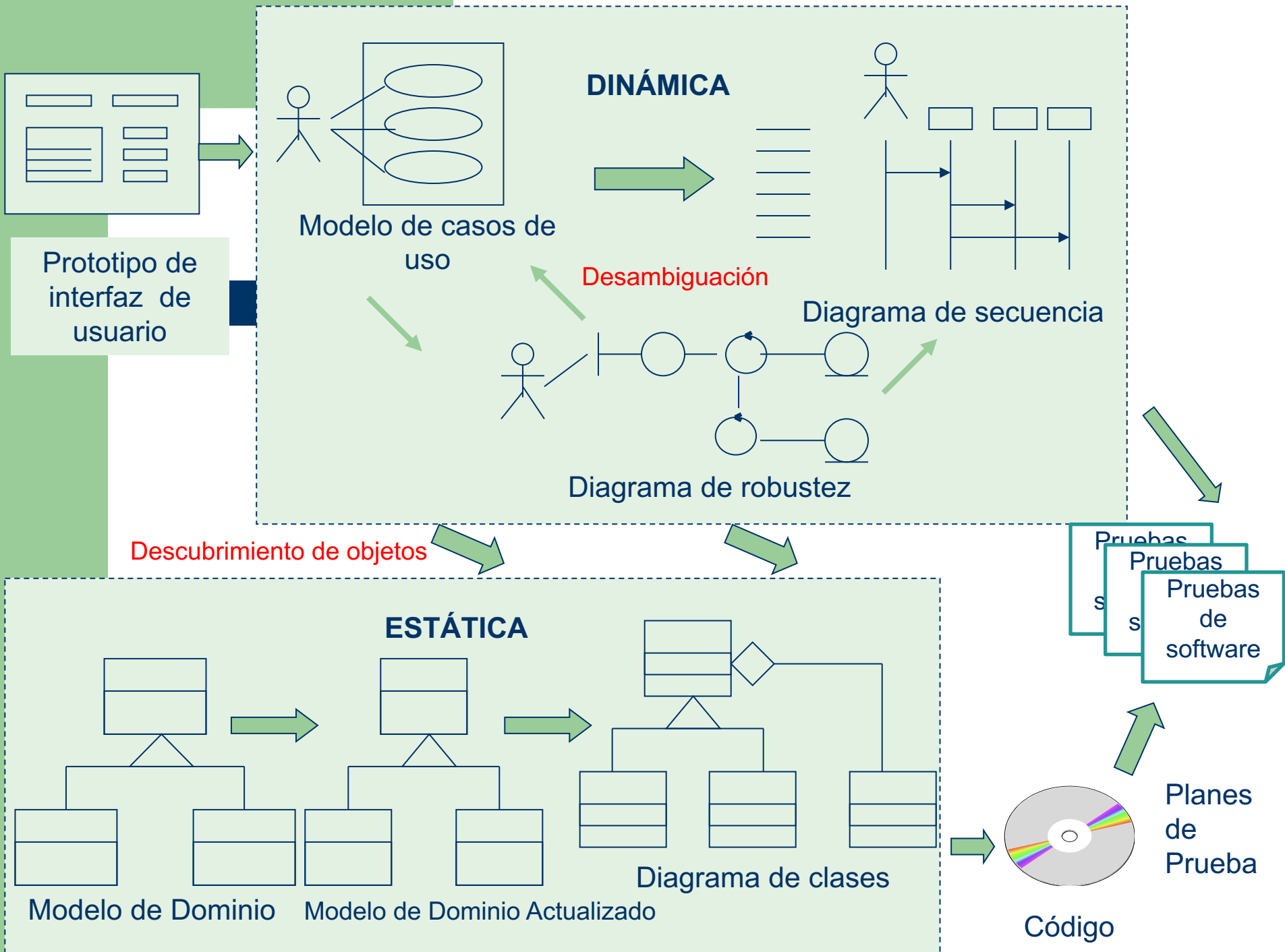
- Escriba un párrafo o más para cada caso de uso, describiendo su comportamiento
- Si sólo hay una frase, quizá dividió demasiado finamente los casos de uso y deberían reunirse varios
- Si es demasiado extenso o complicado, quizá debe subdividirlo
- Importa más identificar la mayoría que refinarlos desde el principio
- Más adelante se descubrirán otros y se refinarán

Casos de uso cómo escribirlos

- Recomendación importante:
- Deben guardar estrecha correlación con manual de usuario y la Interfaz gráfica de usuario (GUI)
- Primero se escribe el manual y luego se trabaja en el código (como sea: dibujos, texto, prototipo rápido, objetos de utilería, etc)

Casos de uso cómo escribirlos

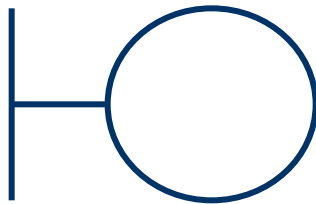
- En la descripción no detalle demasiado elementos que pueden cambiar más tarde.
 - Por ejemplo, no especifique tipo de botón si puede cambiar por un menú desplegable o una lista para seleccionar.
- Otras fuentes para casos de uso:
 - Si existe un sistema anterior, use los manuales de usuario para extraer casos de uso
- Asegúrese que los casos de uso corresponden a lo que efectivamente hacen los usuarios



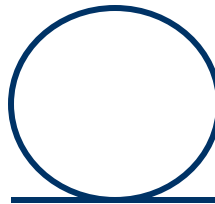
Análisis de Robustez

Identificación de Objetos

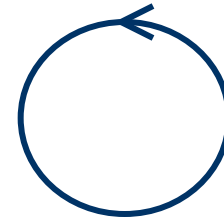
- Objetos que participan en cada caso de uso
- Clasificación de objetos:
 - Objetos Fronterizos (de limite, boundary): objetos con los cuales puede interactuar el usuario – interfaz de usuario -.
 - De Entidad (Entity): generalmente objetos del modelo de dominio
 - De control (controles, control): intermediarios entre los fronterizos y de entidad.



Objeto fronterizo



Entidad

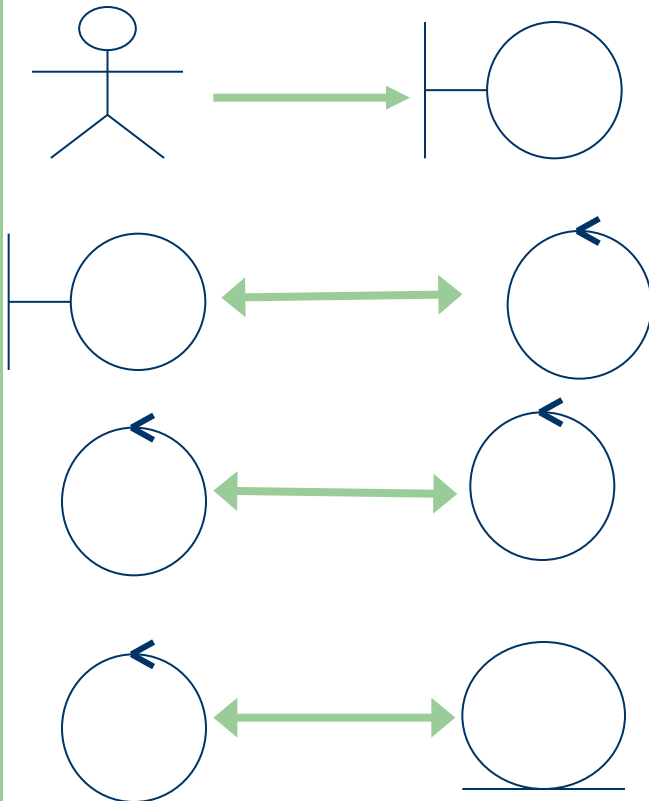


Control

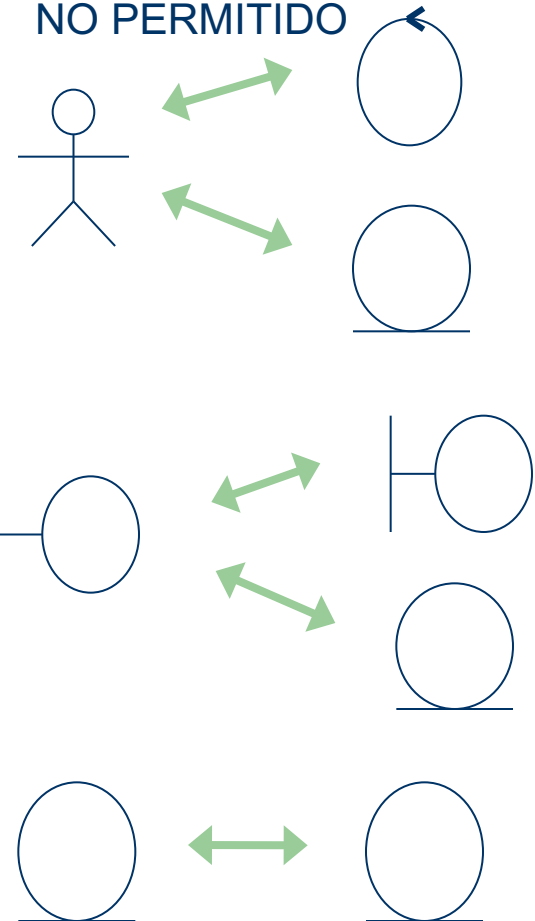
Análisis de Robustez

Relaciones entre objetos

PERMITIDO

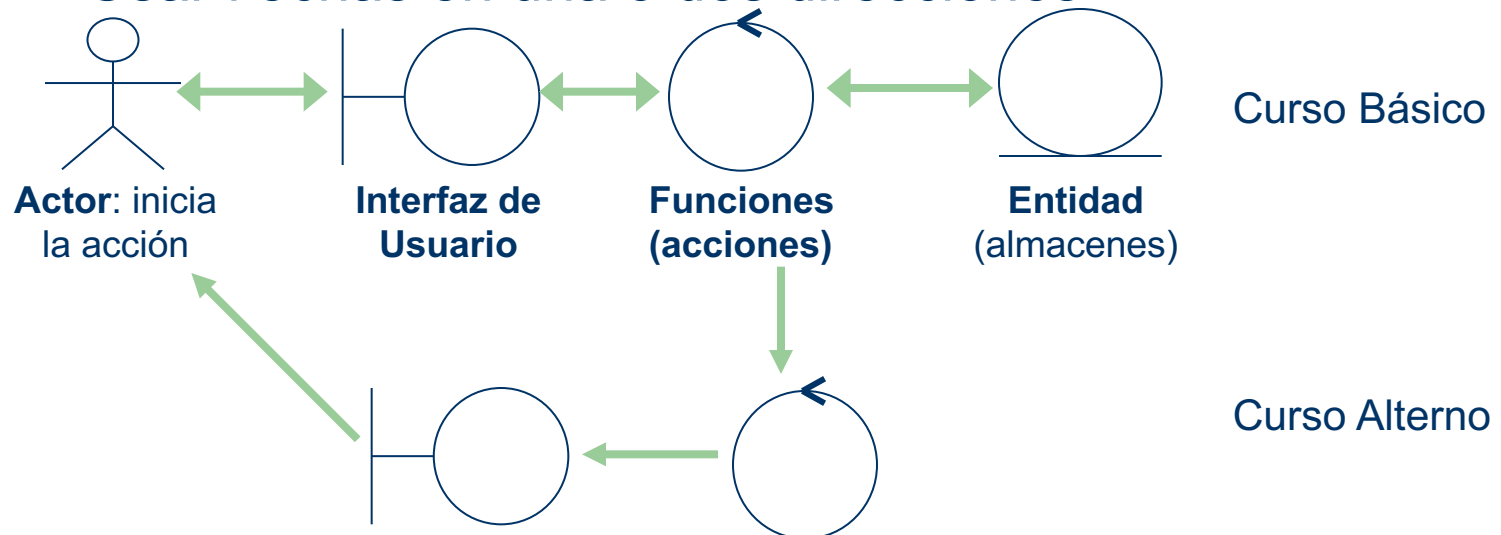


NO PERMITIDO



Análisis de Robustez (Diagramas de Robustez)

- Representa el curso básico y los alternos de cada caso de uso.
- Tener entre 2 y 5 objetos de control por caso de uso.
- Usar flechas en una o dos direcciones.



NO SON DIAGRAMAS DE FLUJO

Análisis de Robustez

Para qué sirven

- Comprobación de Sanidad:
 - revisar las ideas de los casos de uso (comportamiento razonable).
- Comprobación de entereza:
 - asegurar que en los casos de uso se cubra el camino básico y los posibles caminos alternos.
- Descubrir objetos (si son necesarios)
- Diseño preliminar:
 - los diagramas son la primera vista del nuevo sistema.

Ejemplos del libro de ICONIX, edición 2007

- Ejemplo 1. Ver ejemplo páginas 139 y 143 del archivo pdf que contiene el libro
- Ejemplo 2. Páginas 148 y 149

Arquitectura del software



Arquitectura de software

- Es el fundamento para construir el sistema
- Es un diseño de alto nivel, basado en requerimientos
- Representa la estructura del sistema:
 - Partes
 - Conexiones
- También llamada Arquitectura Técnica o Arquitectura del Sistema

Arquitectura de software

- Junto con análisis, son el puente entre requerimientos y diseño detallado
- Alcance variable:
 - Una aplicación
 - Una línea de productos
 - Toda una empresa

Arquitectura de software

Ejemplo. (No se requiere que sea formal)

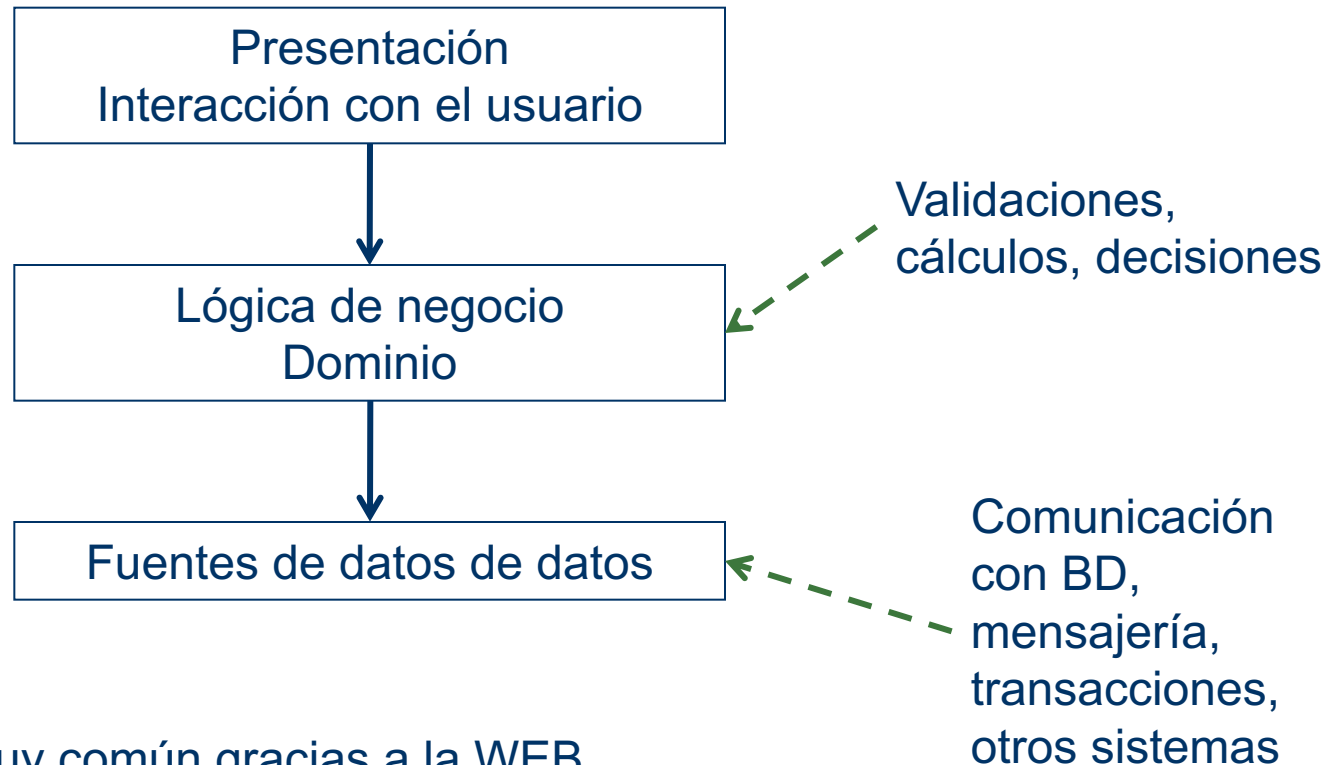


Algunos Artefactos (Ambler)

- Casos de cambio
- Diagrama de componentes
- Diagrama de despliegue
- Diagrama de paquetes
- Otros diagramas

} UML

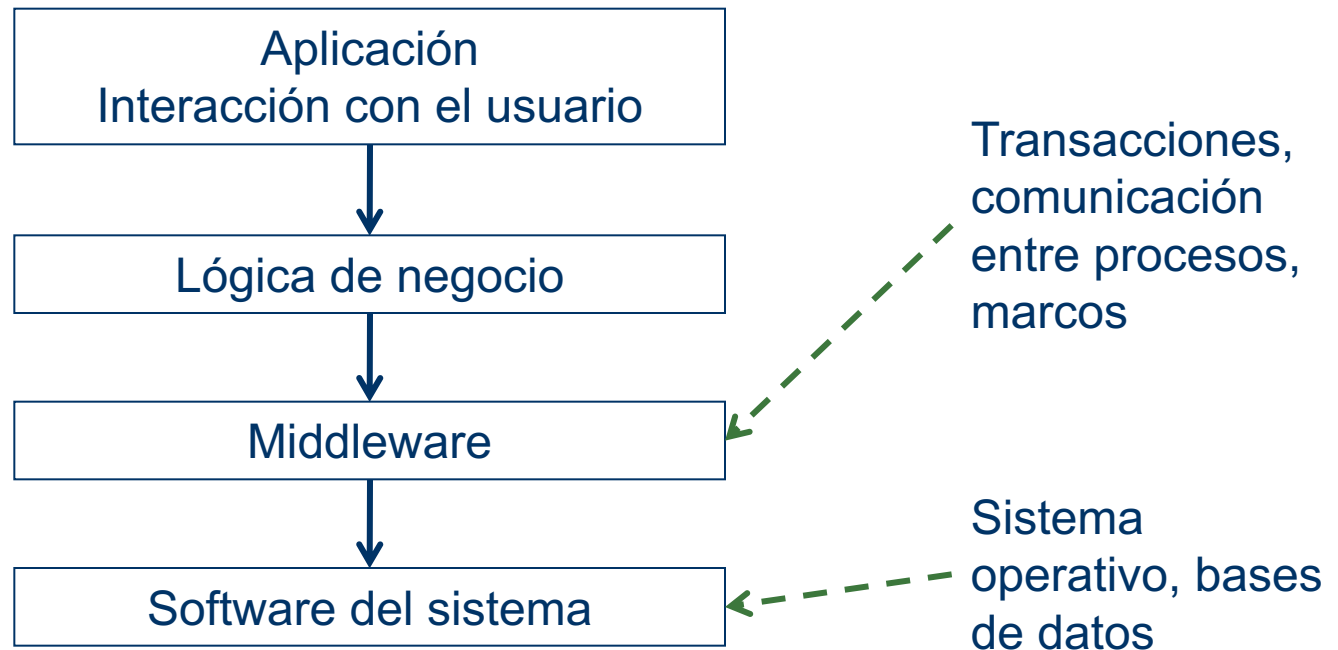
Modelo típico de tres capas



Se volvió muy común gracias a la WEB

Proceso Unificado de Desarrollo

Arquitectura de referencia



Paul Reed, "Reference Architecture: The best of best practices"
<http://www.ibm.com/developerworks/rational/library/2774.html>

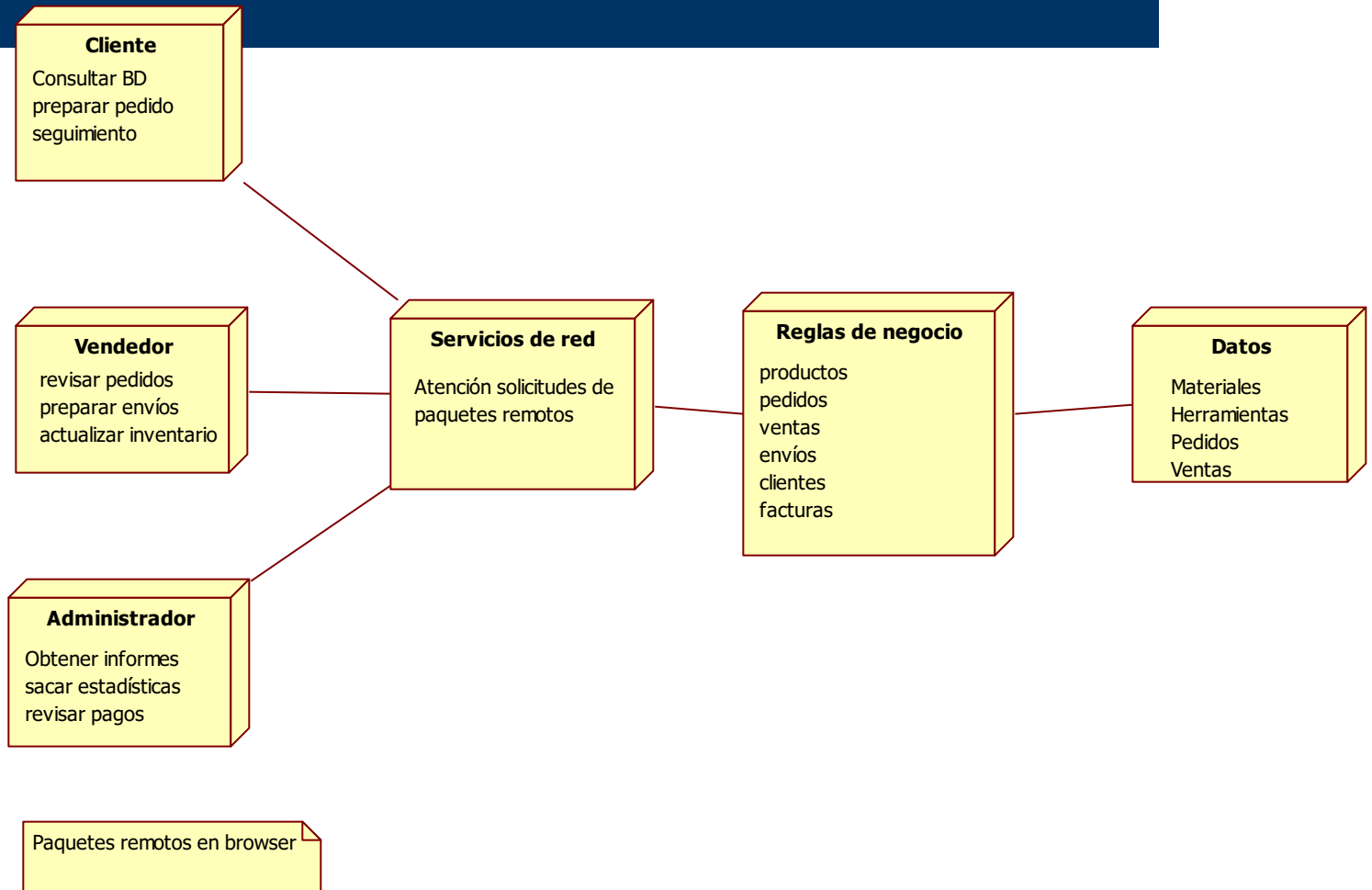
Para implementar capas

- Se implementa según complejidad de problema
- Pueden ir todas las capas en un equipo
 - Como procedimientos o paquetes diferentes
- Puede ir una capa en cada equipo
 - Como subsistemas separados; procesos separados
- Puede haber subcapas

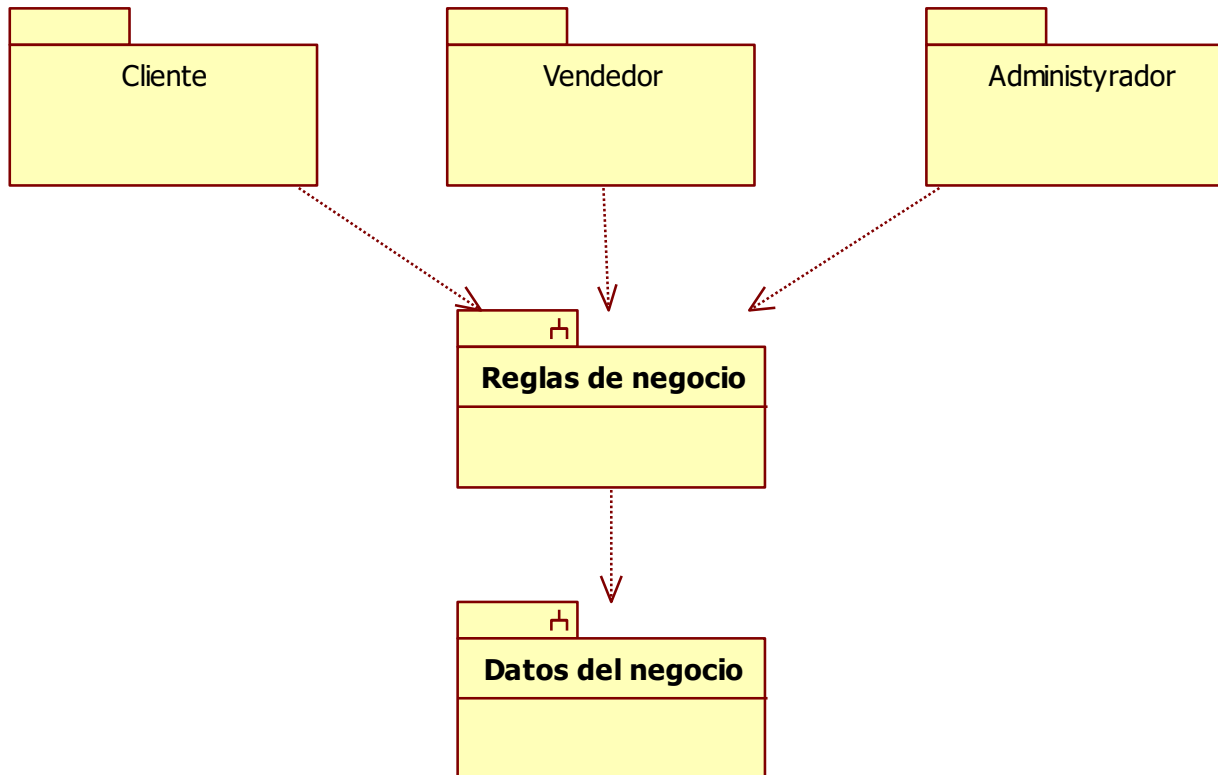
Ejemplo

- Suponga un negocio de venta por Internet de material para hacer vitrales: se ofrecen materiales (como vidrio, soldadura, tintes) y herramientas (pinzas, cortadores, etc.). El cliente puede explorar las existencias y hacer su pedido y luego rastrear en qué etapa va. Un empleado revisa los pedidos, los prepara y envía, registrando la información. Un administrador obtiene reportes y analiza la información.
- Los diagramas que siguen van de acuerdo a una separación de capas.

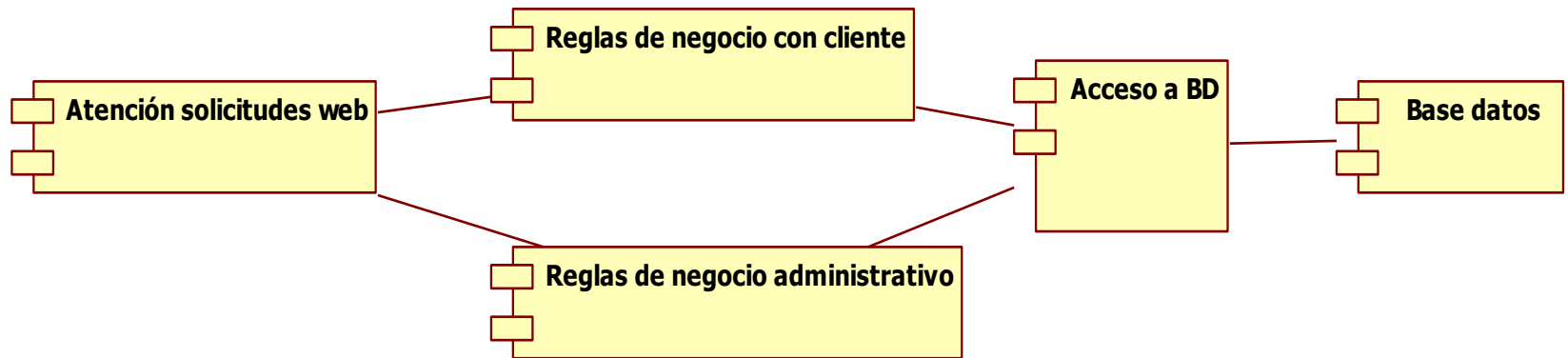
Ejemplo de diagrama de despliegue



Ejemplo de diagrama de paquetes



Ejemplo de diagrama de componentes



Ejemplo amplio

- Ver el ejemplo en el capítulo 7, página 164.

Modelado de la Interacción.

Objetivos

- Construcción de hilos sobre el comportamiento de los objetos en los casos de uso.
- Tres objetivos:
 - Asignar el comportamiento de los objetos (fronterizos, entidades y de control).
 - Detallar la interacción entre objetos (por medio de mensajes).
 - Ubicar los métodos correspondientes a cada clase (responsabilidades).

Modelado de la Interacción

Diagramas de Secuencia

- Consta de 4 elementos:
 - Texto del curso de acción (caso de uso).
 - Objetos - se representan con el nombre del objetos (opcional) y la clase.
 - Mensajes: flechas entre los objetos
 - Métodos: operaciones (objetos de control) – representados por rectángulos).

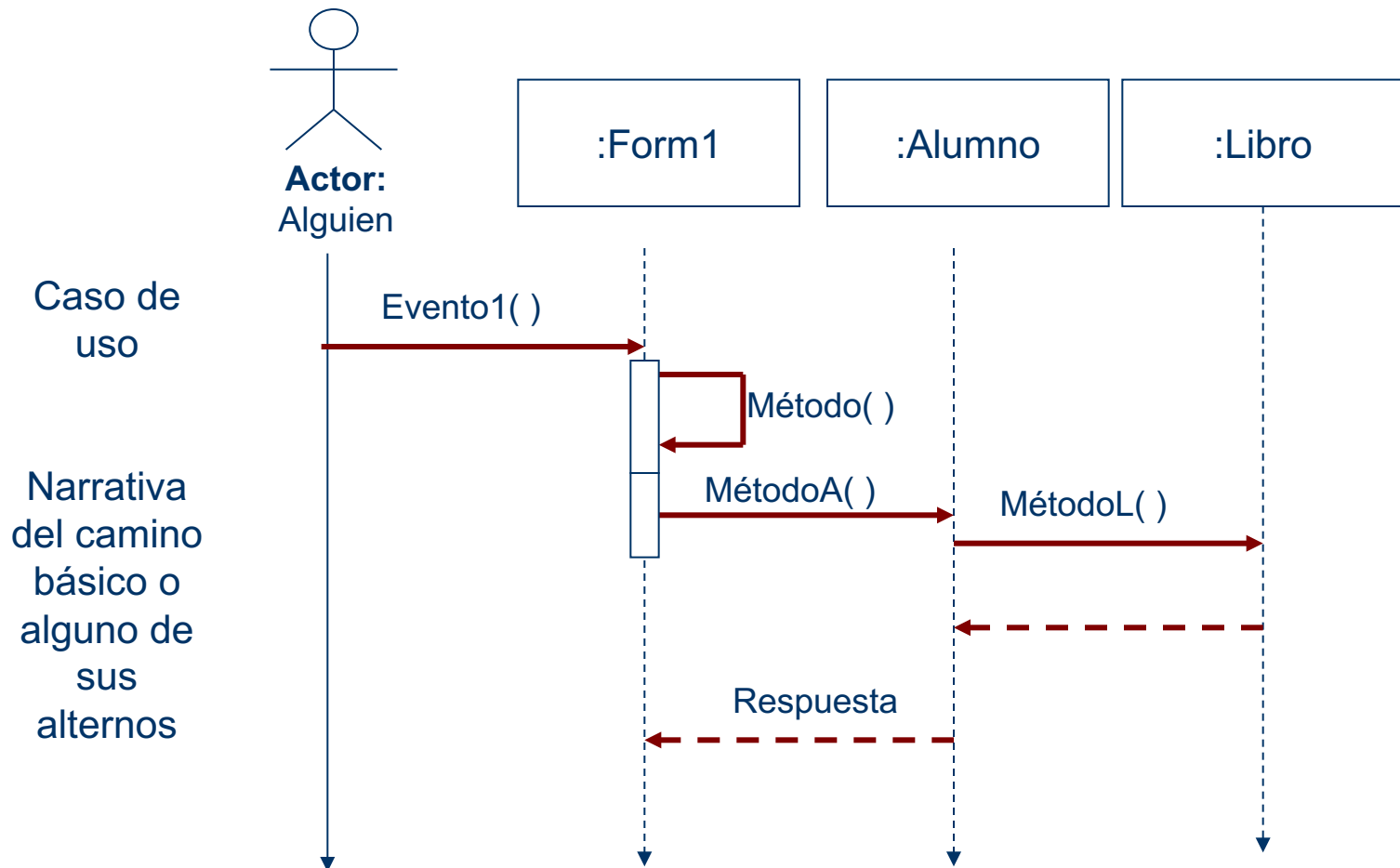
Cómo crear un diagrama de Secuencia

- Copiar texto del caso de uso (parte izquierda).
- Agregar objetos entidad del diagrama de robustez (parte superior – derecha).
- Agregar objetos fronterizos y actores (parte superior – izquierda).
- Asignar métodos y mensajes

- Posiblemente los objetos de control pasan a ser métodos de entidades o de objetos fronterizos
- Cuando sólo es intermediario sin actividad propia, se funde con fronterizo o entidad
- Si un objeto de control se necesita, se agrega

Modelado de la Interacción

Diagramas de Secuencia



Modelado de la Interacción

Asignación de métodos

- Una parte fundamental pero difícil del método es la asignación de responsabilidades para cada clase.
- Como ayuda existen las tarjetas Clase – Responsabilidad – Colaboración (CRC).
 - Estas tarjetas ayudan a decidir y aclarar cuales operaciones (métodos) corresponden a cada clase.

Nombre de clase	
Responsabilidades	Colaboración
Métodos que están a cargo de esta clase	Clases con las que va a colaborar (relacionadas)

Modelado de la Interacción

Responsabilidad (Puntos de criterio)

- Al asignar los métodos a cada una de las clases, toma en cuenta:
 - Reusabilidad: considera que las clases pueden ser utilizadas en otros proyectos.
 - Aplicabilidad: asignar los métodos realmente necesarios para la clase y el proyecto.
 - Complejidad: métodos fáciles de construir y de entender.
 - Conocimiento de la implementación

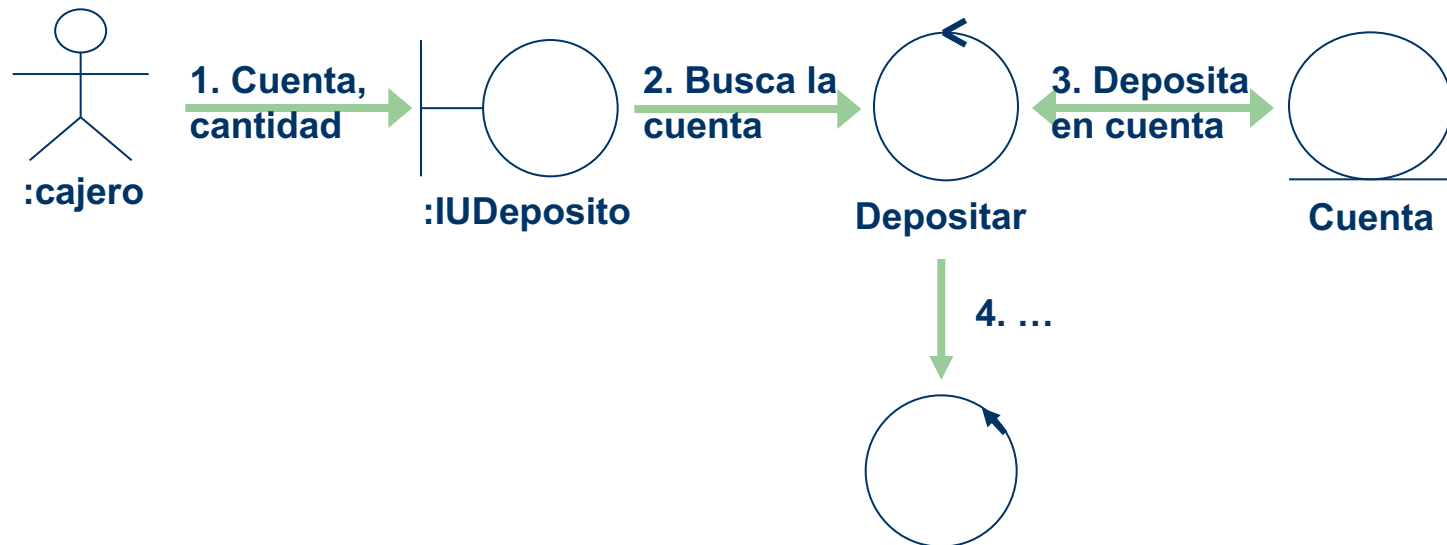
Modelado de la Colaboración y Estados

- Ayuda a agregar aspectos del comportamiento que tiene el nuevo sistema.
- Se diseñan comúnmente para sistemas de tiempo real o sistemas distribuidos.

Diagramas de Colaboración

- Especifican mas los diagramas de robustez.
- Se apegan más a la situación real.
- Énfasis en el orden de las operaciones entre los objetos del caso de uso.
- Agrega detalles extras al momento del paso de mensajes entre los objetos.

Diagramas de Colaboración



- Se representan de igual forma que los diagramas de robustez, pero llevan un números que determina o indica el orden de ejecución sobre las flechas.

Diagramas de Estados

- Diagramas de Estado = Máquina de estado finito = Autómatas
- Solucionan la representación del comportamiento dinámico de un objeto o grupo de objetos.
- Muestra el ciclo de vida de los objetos, mediante los diferentes estados que tiene o pasa un objeto.

Diagramas de Estados

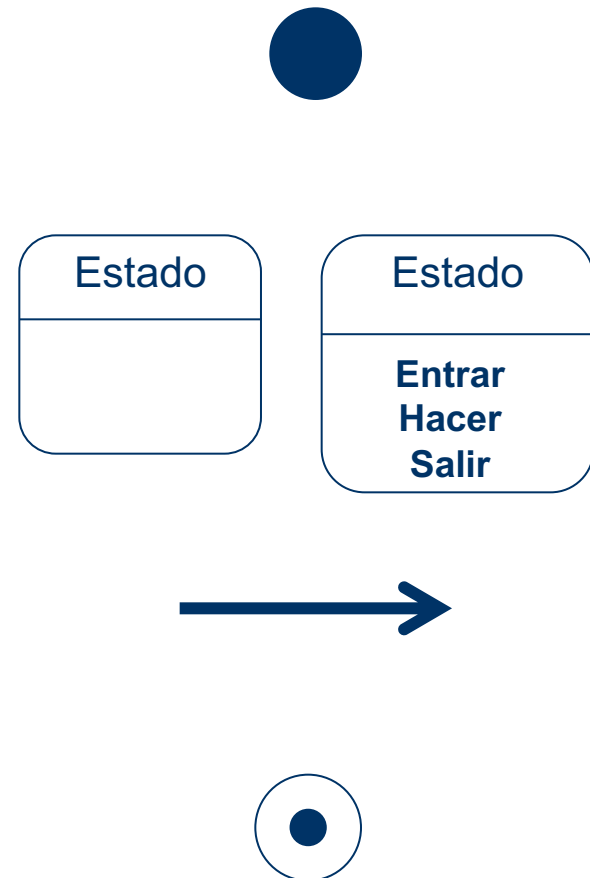
Elementos

- Estado inicial.
- Estados del objeto = rectángulo redondeado, con el nombre del estado y las actividades (opcional).
 - Tipos de actividades o eventos:
 - a) Inicio – Entrada (Enter): acciones cuando entra al estado.
 - b) Hacer (Do): acciones mientras esta en el estado.
 - c) Salida (Exit): acciones cuando sale del estado.
- Transiciones: cambio de estados.
- Estado final.

Diagramas de Estados

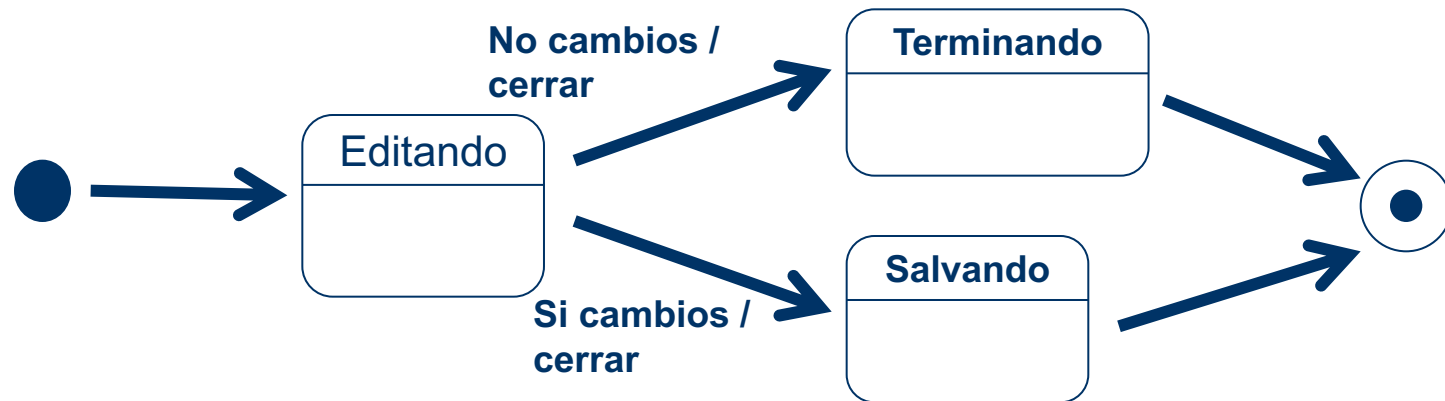
Representación

- Estado inicial.
- Estados del objeto.
- Transiciones.
- Estado final.



Diagramas de Estados

Representación - sugerencias



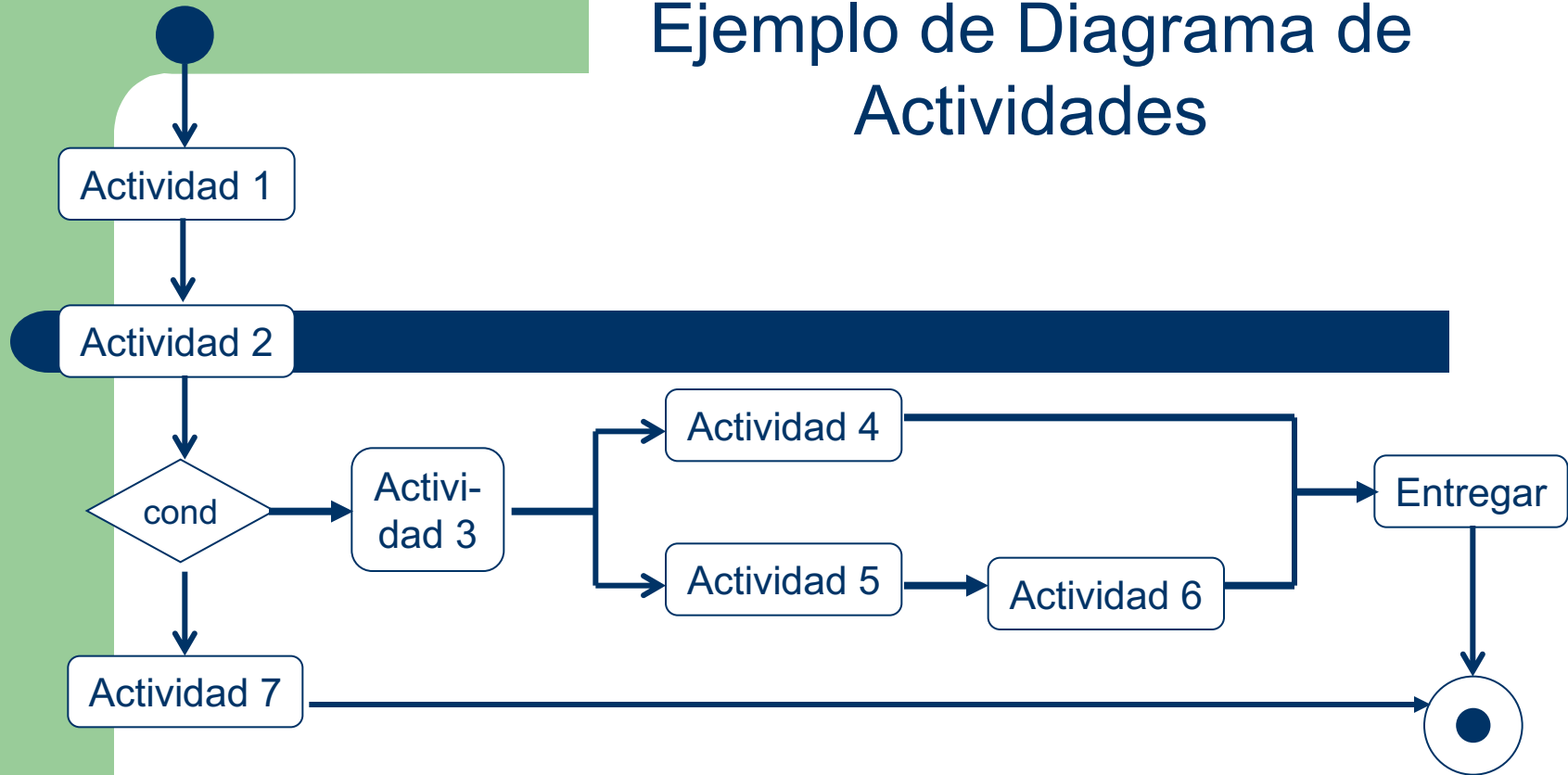
- Nombre de estados = sustantivos o verbo en participio
- Las transiciones deben llevar:
 - a) Qué la causa = {evento, mensaje, condición, tiempo, terminación natural} – OBLIGATORIO
 - b) Acción – opcional

Se permite anidar los diagramas de estado pero NO ES RECOMENDABLE

Diagramas de Actividades

- Descienden de los diagramas de flujo, redes de Petri y de las máquinas de estado.
- Capturan las acciones y los resultados de estas acciones.
- Representan la secuencia de actividades que se realizan en un caso de uso (mas detallado, como un diagrama de flujo).

Ejemplo de Diagrama de Actividades



Utiliza los mismos símbolos de los diagramas de estado.

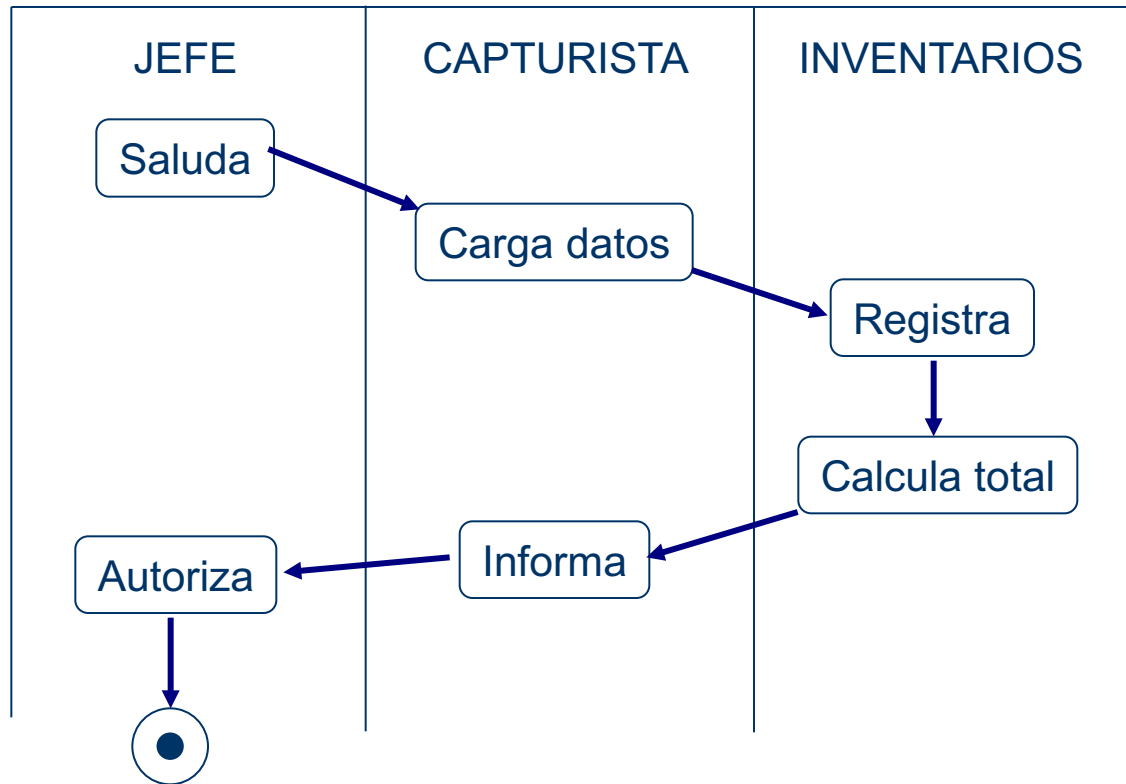
Permite representar las actividades que se pueden hacer en paralelo.

Permite colocar los diferentes caminos (decisiones).

Diagramas de Actividades

Swimlanes (carriles):

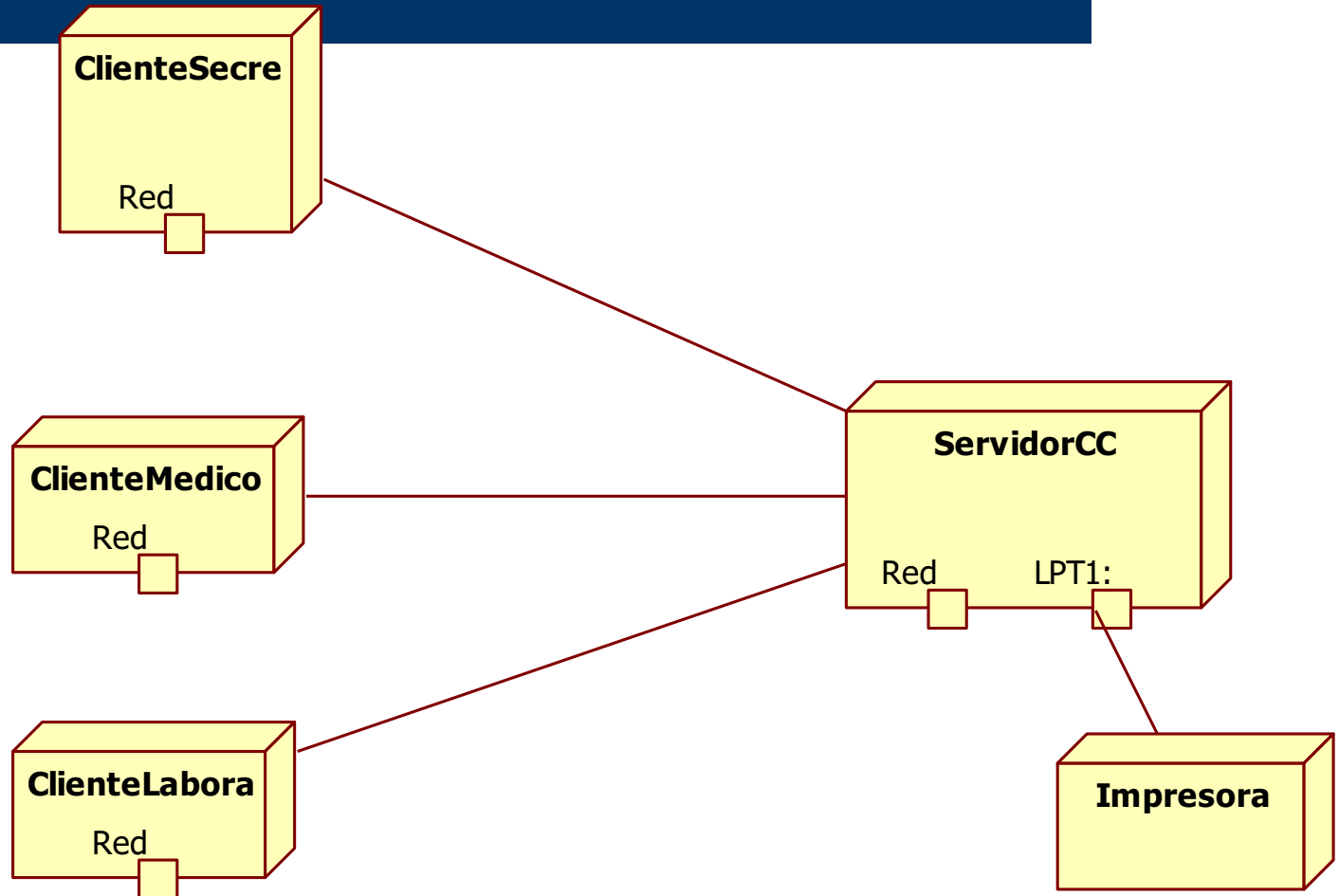
- Permiten agrupar las actividades dependiendo de quien las realiza.
- Cada responsable (clase) de alguna actividad tiene un carril.



Implementación

- Producir diagramas necesarios
 - Despliegue
 - Componentes

Modelo de Despliegue



Componentes

