ICONIX

Notas del método
con ampliaciones y mejoras

Juan Manuel Fernández Peña y
María de los Ángeles Sumano López
Colaboración de Josué Andrade Mirós
Octubre de 2011

Método ICONIX
Referencia

- El método original se encuentra en:

  Rosenberg, Doug, with Kendall Scott

  "Use case driven object modeling with UML. A practical approach"

  Addison Wesley, 1999
- Más información en la página:

  http://www.iconix.com

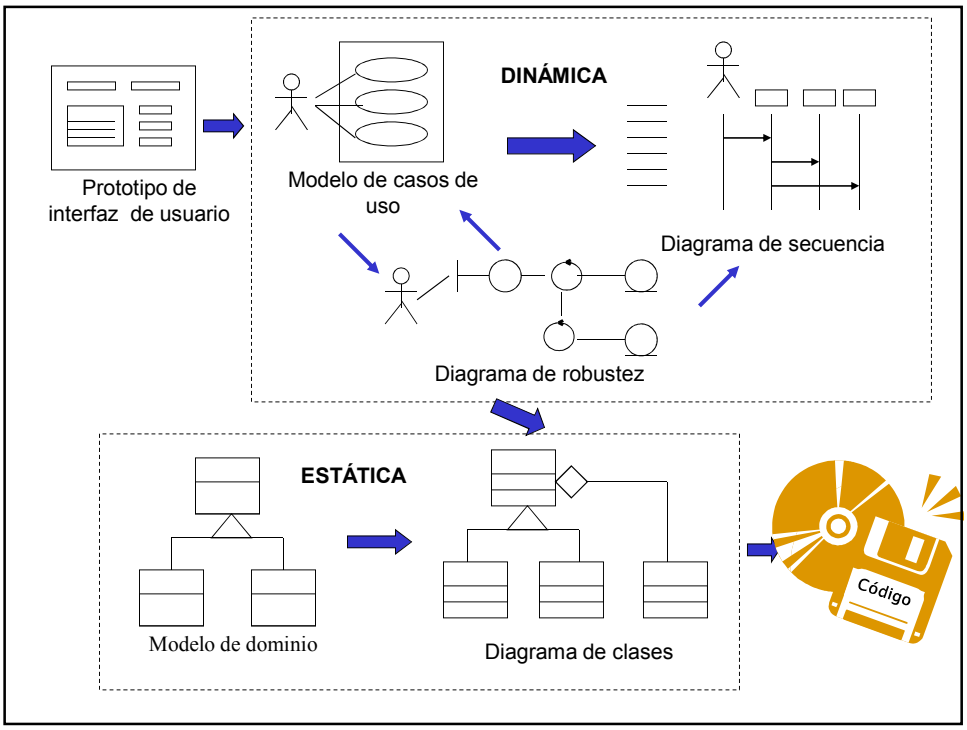
Método ICONIX
¿Por qué esta versión?

- El texto original incluye muchas disgresiones, generalmente obsoletas
- El texto supone ciertos conocimientos, que no siempre tienen los alumnos
- El tratamiento de algunos temas es insuficiente para los usos modernos

- Por ello se realizó esta versión, que sirva para un primer curso de desarrollo orientado a objetos y usando UML.

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.

Pasos principales
I Análisis de requerimientos

- 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.

- META: revisión de requerimientos

Pasos principales
II Análisis y diseño preliminar

- 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

- META: revisión del diseño preliminar

  De usuarios hacia sistema

  De datos hacia sistema

  Detallar a partir de modelos de alto nivel

Pasos principales
III Diseño

- 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

- META: revisión crítica del diseño

Pasos principales
IV Implementación

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

- META: entrega del sistema

Capítulo 2
Modelando el dominio

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

Clases y objetos

- Objeto:
  - Algo tangible o visible
  - Algo que puede aprenderse intelectualmente
  - Algo hacia lo cual se dirigen pensamientos o acciones
- Un objeto tiene:
  - Estado
  - Comportamiento
  - Identidad

Clases y objetos

- Estado: propiedades y sus valores particulares
- Comportamiento: cómo actúa y responde (a cambios de estado y paso de mensajes)

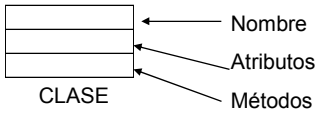
Clases y objetos

- Clase:
  - Descripción de un conjunto de objetos que comparten una estructura, un comportamiento, relaciones y semántica comunes
- Interfaz:

  Vista exterior de una clase; permite contrato acerca de las responsabilidades que ofrece y exige; aísla el interior. Es el QUÉ hace
- Implementación:

  Vista interior, particular; CÓMO lo hace

Clase y objeto en notación UML



Para objetos, el nombre:

Nomobj:nomclase

:nomclase

cuenta
saldo clave
dimesaldo() deposita(cant) retira(cant)

Ejemplo de clase

Micuenta:cuenta
saldo clave
dimesaldo() deposita(cant) retira(cant)

Ejemplo de objeto

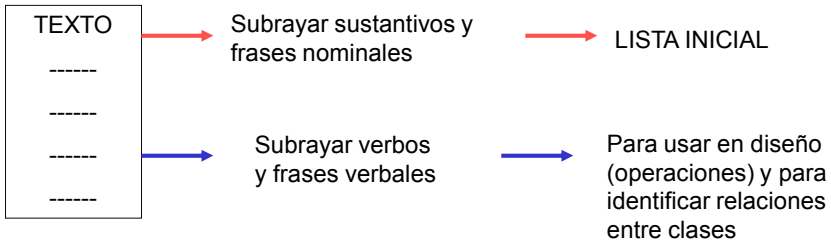
Modelando el dominio

- Procedimiento:
  - 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

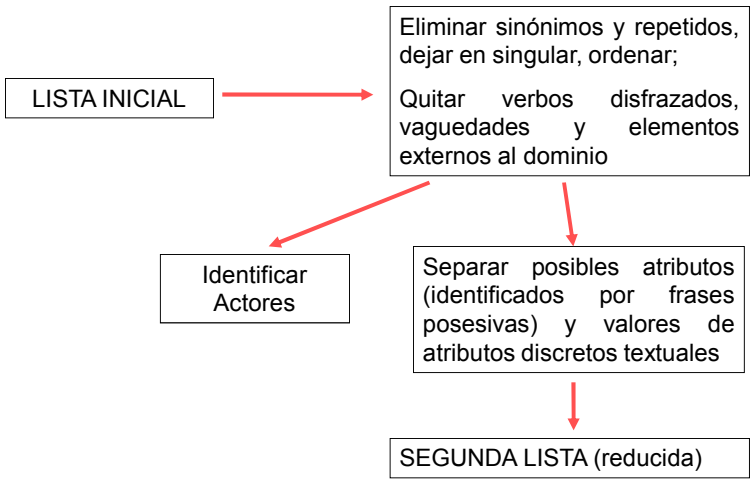
Modelando el dominio

- 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



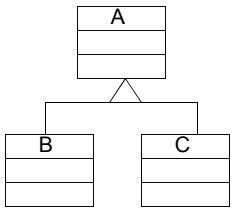
Modelando el dominio

- 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.

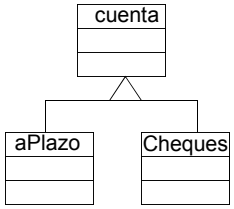
Relación de agregación en UML



La clase A es una generalización de las clases B y C

Las clases B y C son particularizaciones de la clase A

Las clases B y C heredan de la clase A



Ejemplo

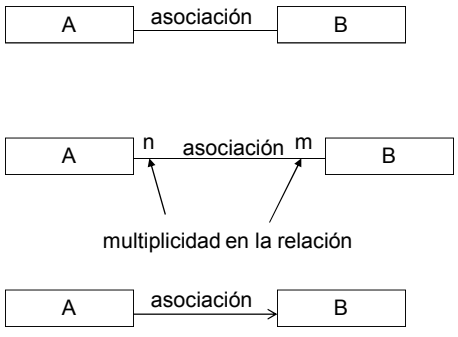
Modelando el dominio

- 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)

Asociaciones con UML

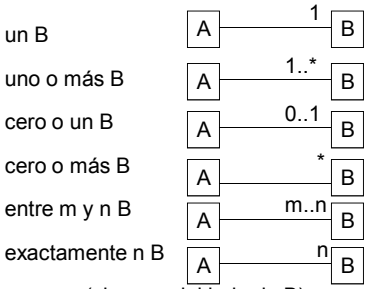


Navegabilidad: la flecha indica que podemos hallar a B a partir de A.

Sin flecha puede indicar doble sentido o indefinido

Multiplicidad

Se lee así: A se relaciona con



(siempre del lado de B)

Lo mismo en sentido de B a A

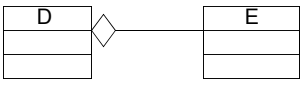
Modelando el dominio

- Procedimiento (III)

Establecer relaciones de agregación

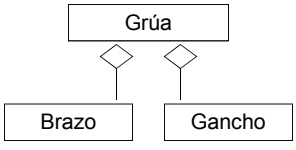
  - Una agregación es una relación en la cual una clase está formada por otras (sus partes)
  - A veces se le llama “parte-de”
  - En UML se distingue una forma más fuerte llamada **Composición**, pero para este método no se hará diferencia

Agregación con UML



Relación de Agregación o Contención: la clase D contiene a la clase E, es decir, la clase E se agregó a la clase D.

También llamada **parte-de**: E es parte de D



Ejemplo

Modelando el dominio

- Procedimiento (IV)

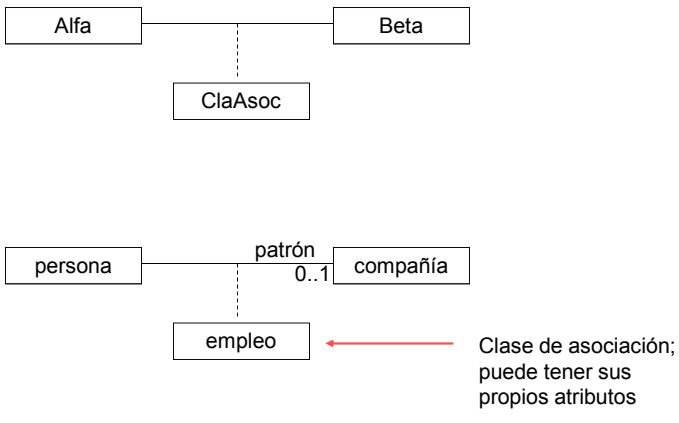
Clases de asociación

  - Una clase de asociación es una variante de las asociaciones muy útil cuando hay relaciones muchas-a-muchas entre clases

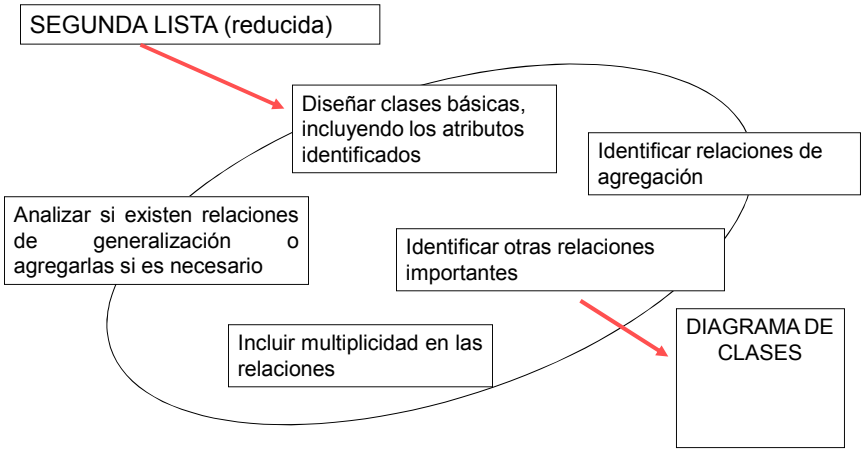
Pueden conseguirse clase del dominio a partir de entidades en bases de datos preexistentes

Cuando una clase tiene demasiados atributos, conviene dividirla en clases auxiliares y usar agregación para reunirlas

Clase de asociación con UML



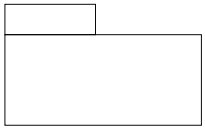
Modelado del dominio procedimiento



Advertencia

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

Capítulo 3
Modelado de casos de uso



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

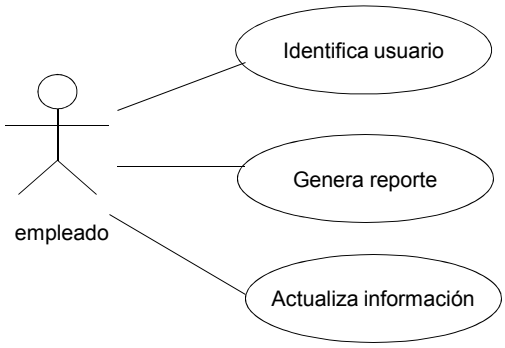
- Se describe mejor como una frase verbal en presente y en voz activa.
- Ejemplos:
  - Admite paciente,
  - Realiza transacción o
  - Genera reporte
- Especifica de manera precisa, no ambigua, un aspecto del uso del sistema sin suponer un diseño o implementación particulares.
- Toda la funcionalidad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

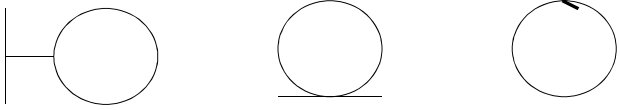
Casos de uso
cómo escribirlos

- 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

Capítulo 4
Análisis de Robustez

Análisis de Robustez
Identificación de Objetos

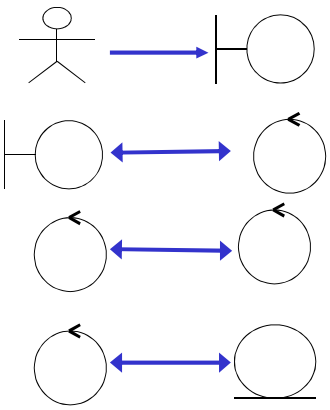
- Objetos que participan en cada caso de uso
- Clasificación de objetos:
  - **Objetos Fronterizos** (de limite, boundary): objetos con los cuales puede interactuar el usuario – interfaz de usuario -.
  - **De Entidad (Entity)**: generalmente objetos del modelo de dominio
  - **De control** (controles,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



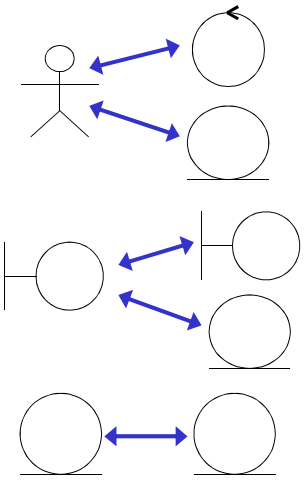
Objeto fronterizo Entidad Control

Análisis de Robustez
Relaciones entre objetos

PERMITIDO

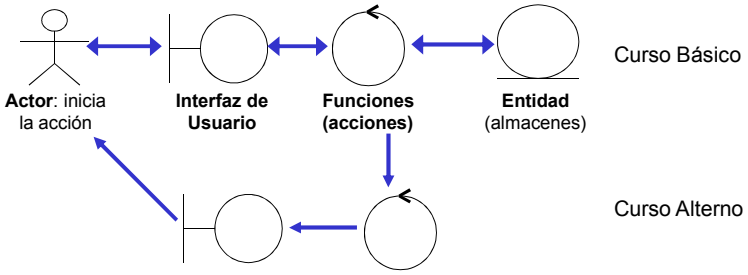


NO PERMITIDO



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.

CAPITULO 5
Modelado de la Interacción

Modelado de la Interacción
Objetivos

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

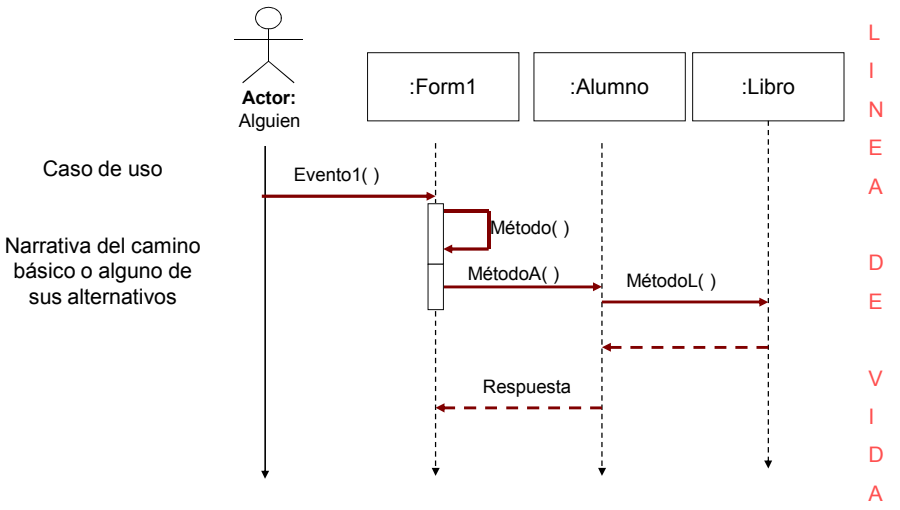
Modelado de la Interacción
Diagramas de Secuencia

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

Modelado de la Interacción
Cómo crear un diagrama de Secuencia

1. Copiar texto del caso de uso (parte izquierda).
2. Agregar objetos entidad del diagrama de robustez (parte superior – derecha).
3. Agregar objetos fronterizos y actores (parte superior – izquierda).
4. 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
     - Pero, Si un objeto de control se necesita, se agrega

Modelado de la Interacción
Diagramas de Secuencia



Modelado de la Interacción
Asignación de métodos – Tarjeta CRC

- 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:
    1. Reusabilidad: considera que las clases pueden ser utilizadas en otros proyectos.
    2. Aplicabilidad: asignar los métodos realmente necesarios para la clase y el proyecto.
    3. Complejidad: métodos fáciles de construir y de entender.
    4. Conocimiento de la implementación

CAPITULO 6
Modelado de la Colaboración y Estados

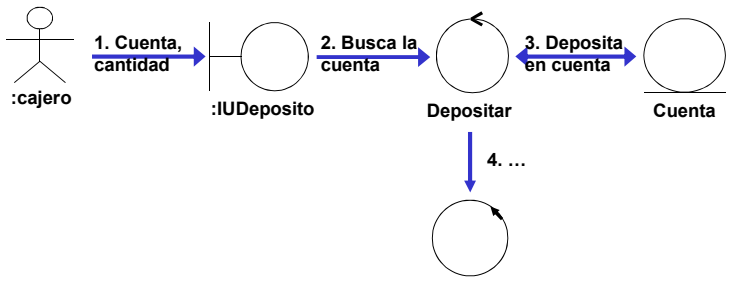
Modelado de la Colaboración y Estados

- 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 = Máquinas 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.

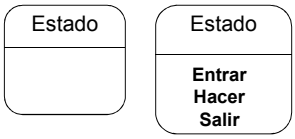
  b) Hacer (Do): acciones mientras esta en el estado.

  c) Salida (Exit): acciones cuando sale del estado.
- Transiciones: cambio de estados.
- Estado final.

Diagramas de Estados Representación

- Estado inicial.

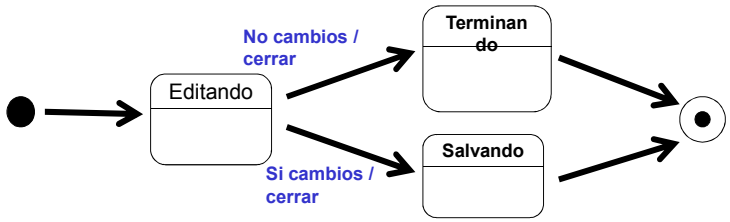
- Estados del objeto.



- 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

  Ejemplo:

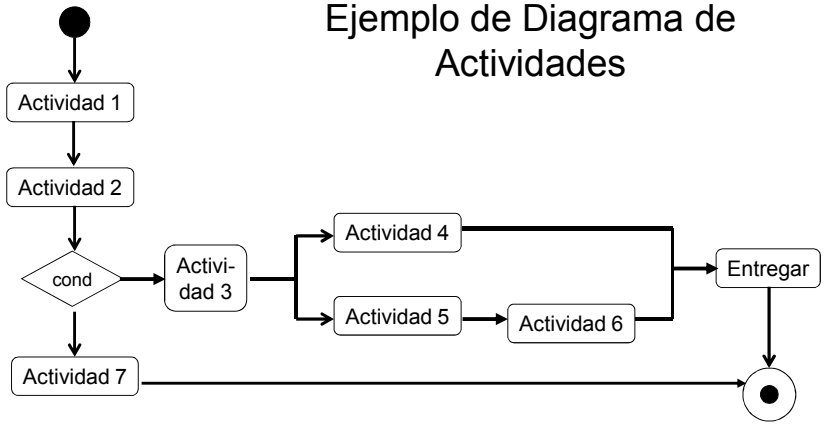
  Si cambios / cerrar

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



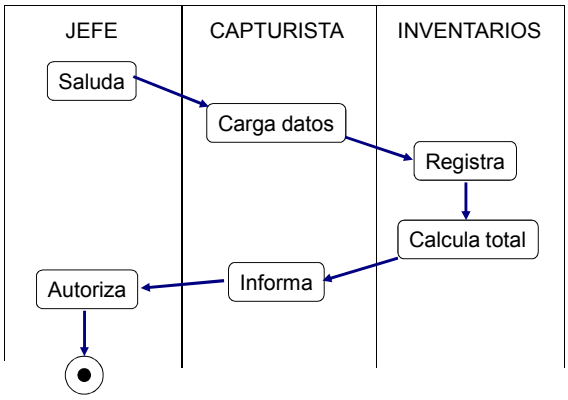
Utiliza los mismos símbolos de los diagramas de estado.

Permite representar las actividades que se pueden hacer en paralelo.

Permite colocar los diferentes caminos (decisiones).

Diagramas de Actividades

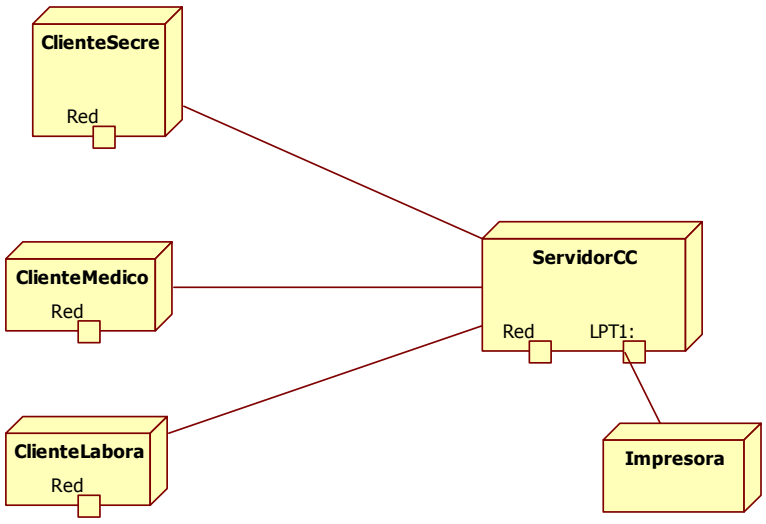
- Swimlanes (carriles) permiten agrupar las actividades dependiendo de quien las realizadas.
- Cada responsable (clase) de alguna actividad tiene un carril.



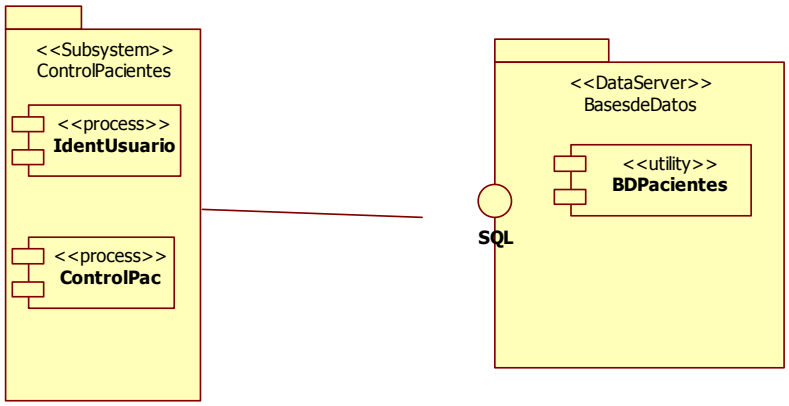
Implementación

- Producir diagramas necesarios
  - Despliegue
  - Componentes

Modelo de Despliegue



Componentes



Capitulo 7
Requerimientos

… Requerimientos

- ¿Qué es un requerimiento?
  - Criterio especifico de un usuario que un sistema tiene que satisfacer.
- Los requerimientos definen el comportamiento y funcionalidad requerida por el usuario para un sistema.
- Expresados por frases que incluyen:
  1. shall – tiene que, debe que
  2. must – debe de, haber de

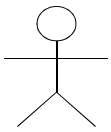
… Requerimientos

- Tipos de requerimientos:

1. **Funcionales**: “el sistema tiene que generar automaticamente …”.
2. **De Datos**:
3. **De ejecución** (desempeño): “El sistema debe de validar los datos que entran…”.
4. **De capacidad**: “El sistema tiene que mostrar información de 10,000 transacciones”.
5. **De prueba**: “El sistema tiene que permitir hacer transacciones de 50 usuarios al mismo tiempo.

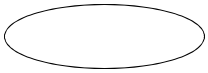
ANEXO
Resumen de símbolos empleados

Casos de uso



ACTOR

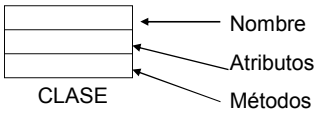
Persona, máquina o programa externo al sistema que se va a realizar, que inician una acción o responden a una solicitud del sistema



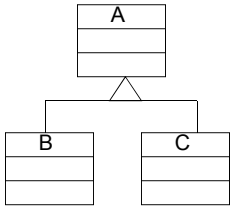
CASO DE USO

Representa una acción o función que el actor desea realizar. Se describe con un verbo o con un verbo y un complemento.

Diagramas de clases

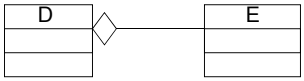


Abstracción de un conjunto de objetos con comportamiento común.



Relación de **Generalización**: A es una generalización de las clases B y C.

Inversamente: B y C heredan de la clase A

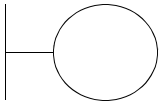


Relación de Agregación o Contención: la clase D contiene a la clase E, es decir, la clase E se agregó a la clase D.

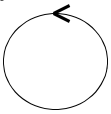
También llamada parte-de: E es parte de D

Diagramas de clases estereotipos

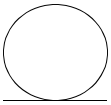
Objeto fronterizo



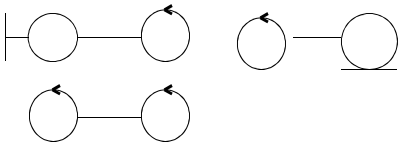
Objeto de control



Entidad



Relaciones permitidas



Relaciones prohibidas