



Excepciones UML Cuestiones conceptuales

Carlos Fontela
cfontela@fi.uba.ar



Temario

Excepciones

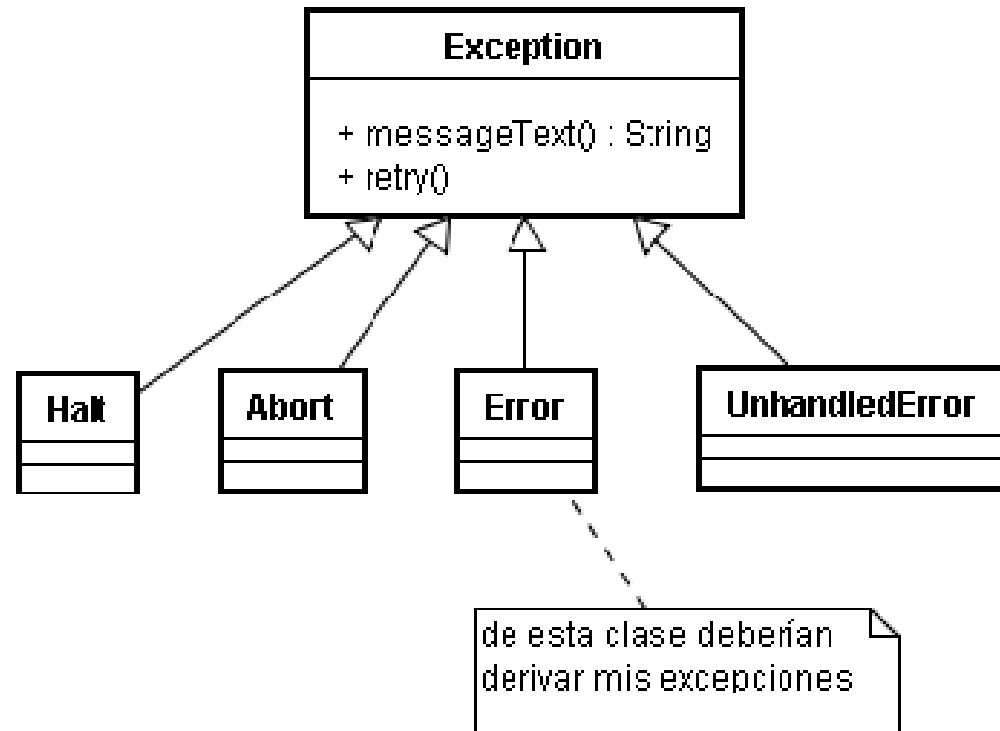
Diagrama de estados y cierre de UML

Polimorfismo con y sin herencia

Herencia con excepciones: el círculo y la elipse



Smalltalk: jerarquía de excepciones



La jerarquía influye en la captura

Cuando decimos capturar un tipo de excepción, capturamos cualquier instancia de esa clase o una descendiente

Cuándo lanzar excepciones

Si en el contexto en el que estamos no hay suficiente información para resolver el potencial problema

Hablamos de excepciones cuando el problema no se puede resolver en un determinado contexto

Y la lanzamos a un contexto de nivel superior para que resuelva qué hacer

Si se pudiera resolver, lo trataríamos allí y no lanzaríamos una excepción

También para aislar el código que se usa para tratar problemas del código básico (camino feliz)

Para crear software más robusto

En el modelo contractual, una excepción se provoca cuando no se cumple una precondición



Qué hacer al capturar

Resolverla mediante

Finalización súbita

Continuación ignorando las fallas

Avance y recuperación

Nuevo intento

No resolverla y enviarla al contexto invocante

La misma

Otra excepción



Jerarquías de excepciones propias

Sirven para dar mayor información sobre el
tipo de problema

Podrían agregar atributos y métodos

Pero no es lo más habitual

Cuidar bien la jerarquía



Estados, eventos, transiciones

Estado

representado por el conjunto de valores adoptados por los atributos de un objeto en un momento dado

situación de un objeto durante la cual satisface una condición, realiza una actividad o espera un evento

Evento

Estímulo que puede disparar una transición de estados

Especificación de un acontecimiento significativo

Señal recibida, cambio de estado o paso de tiempo

Síncrono o asíncrono



Diagrama de estados UML: ajedrez

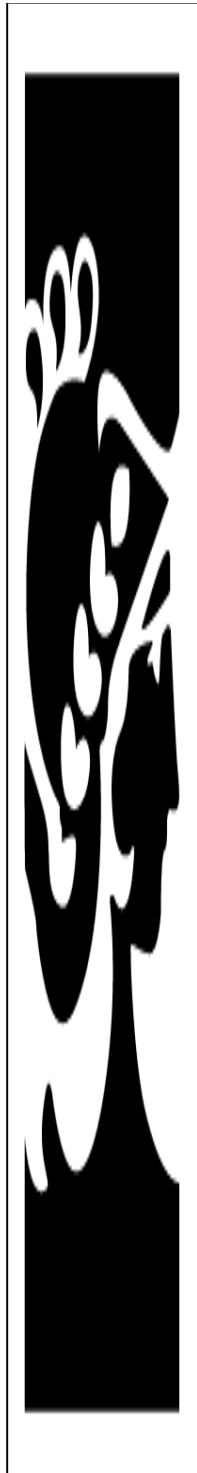
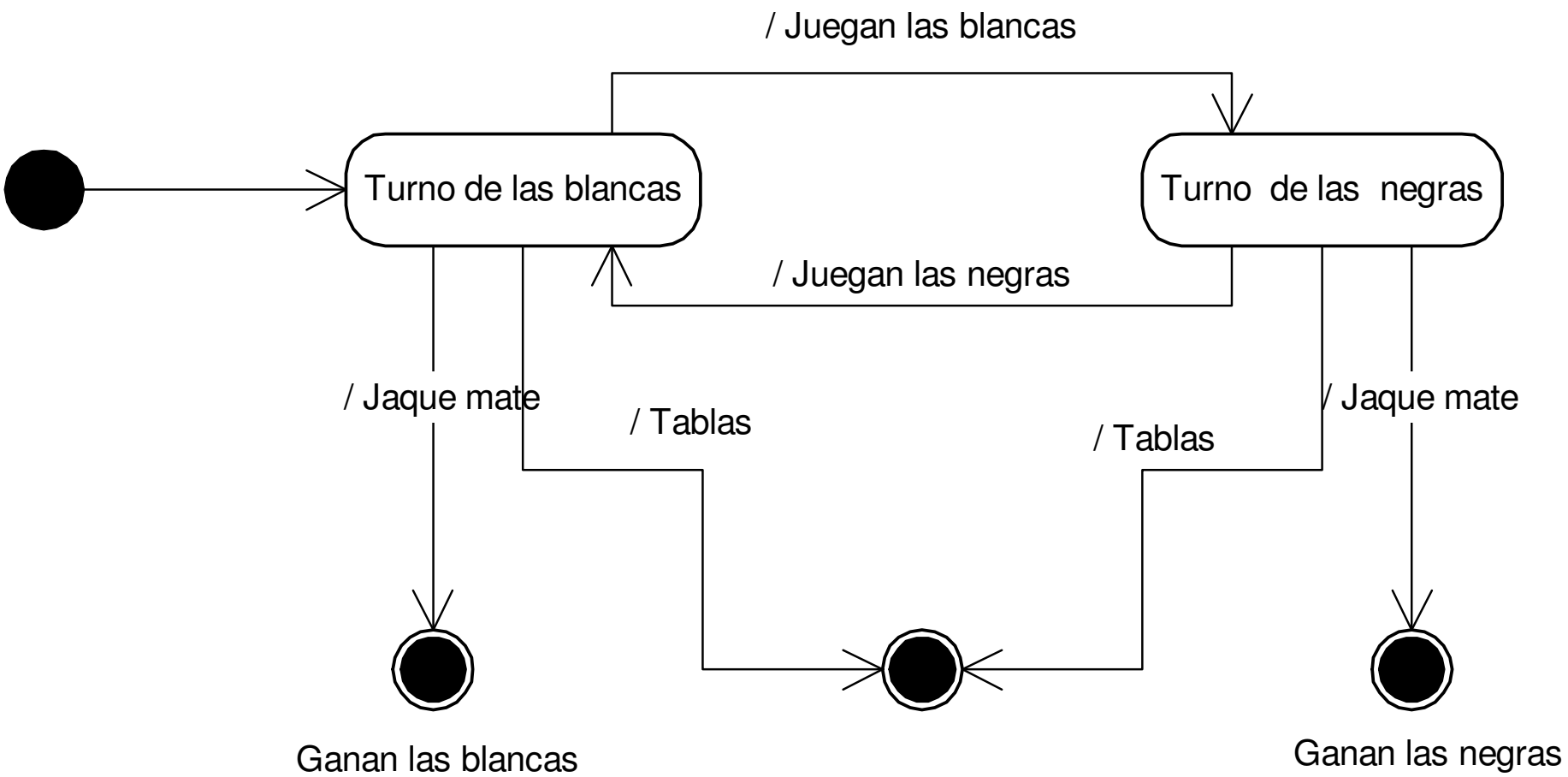


Diagrama de estados UML: estados civiles (1)

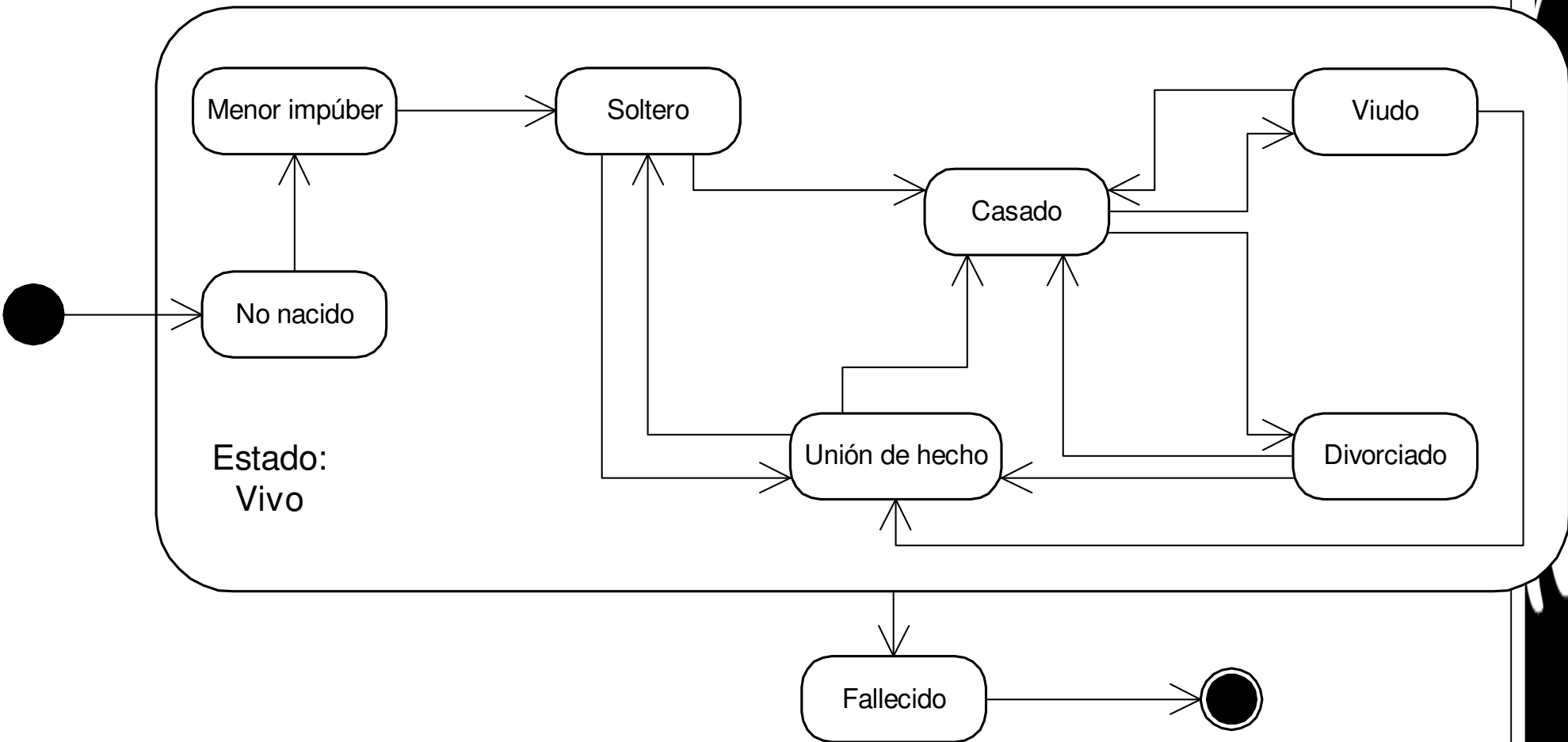
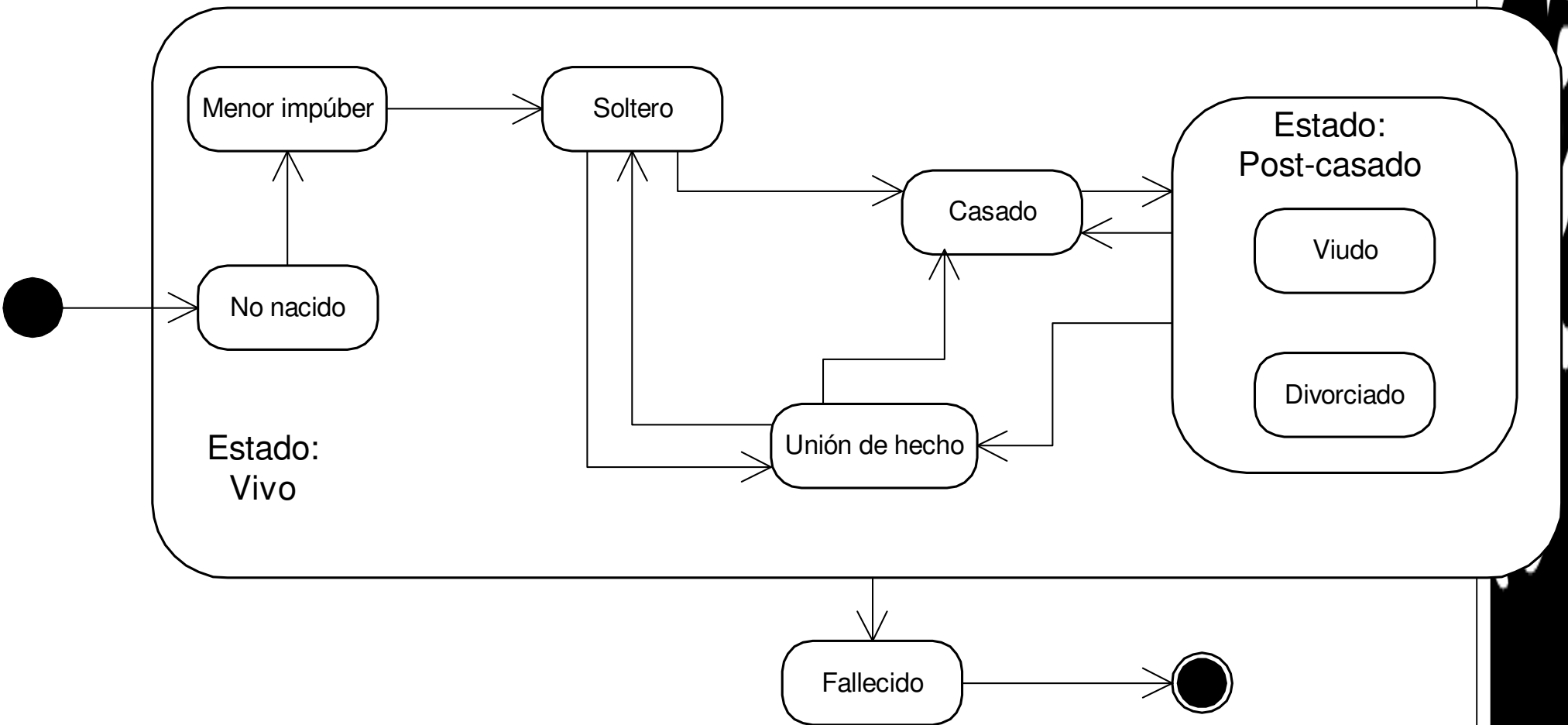
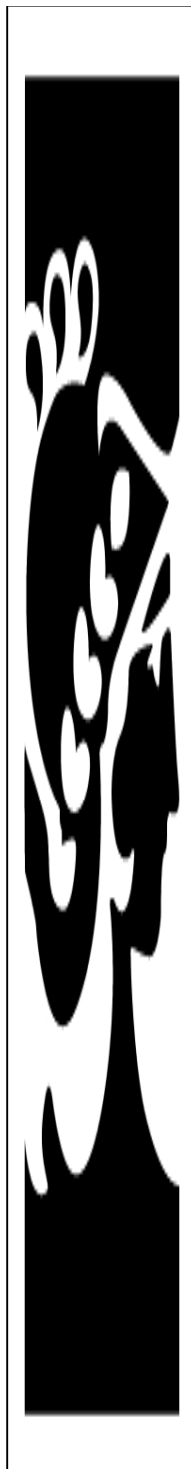
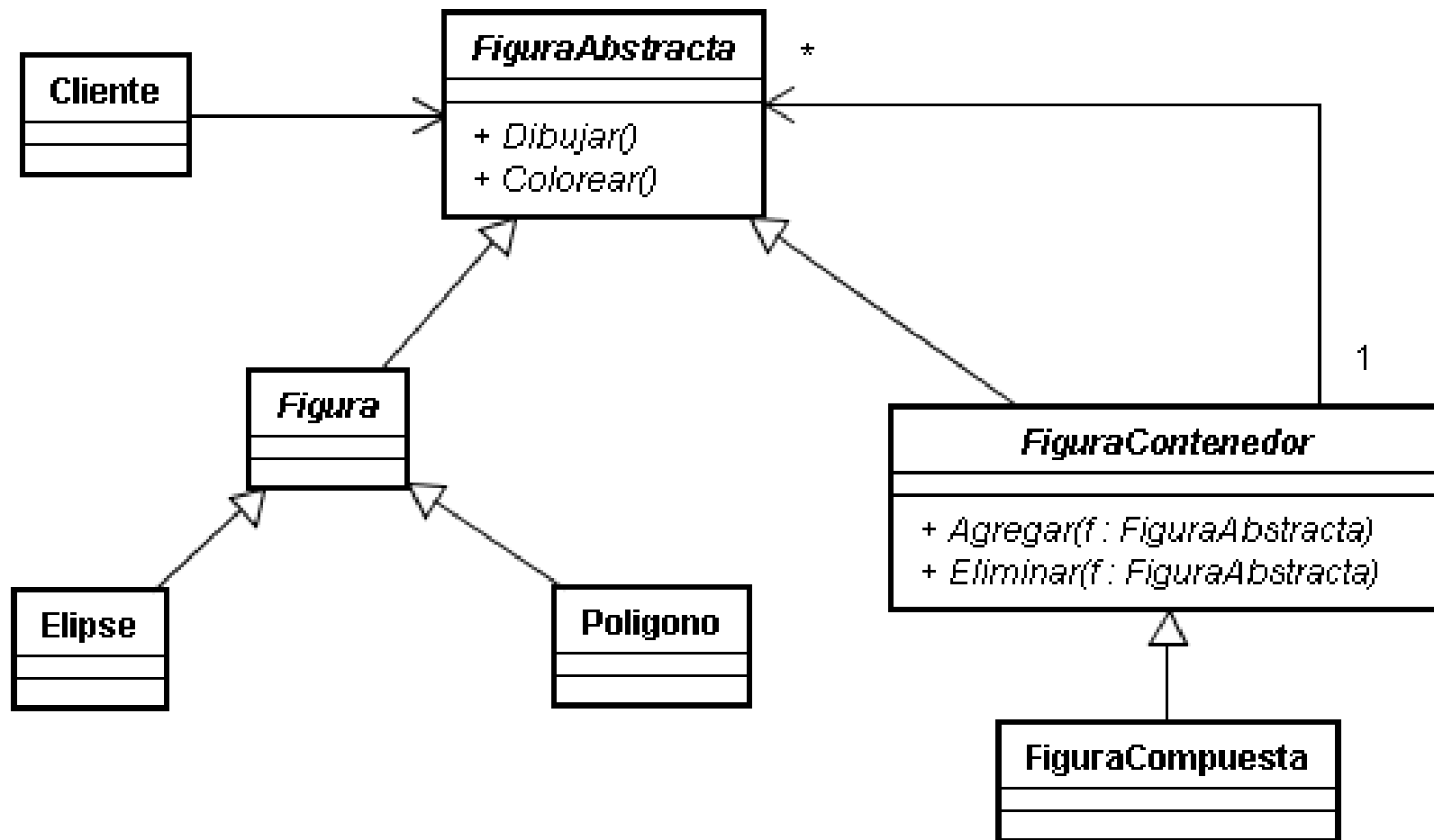


Diagrama de estados UML: estados civiles (2)



Ejercicio: analizar diagrama de clases



UML

Lenguaje de modelado

Modelos representan la realidad

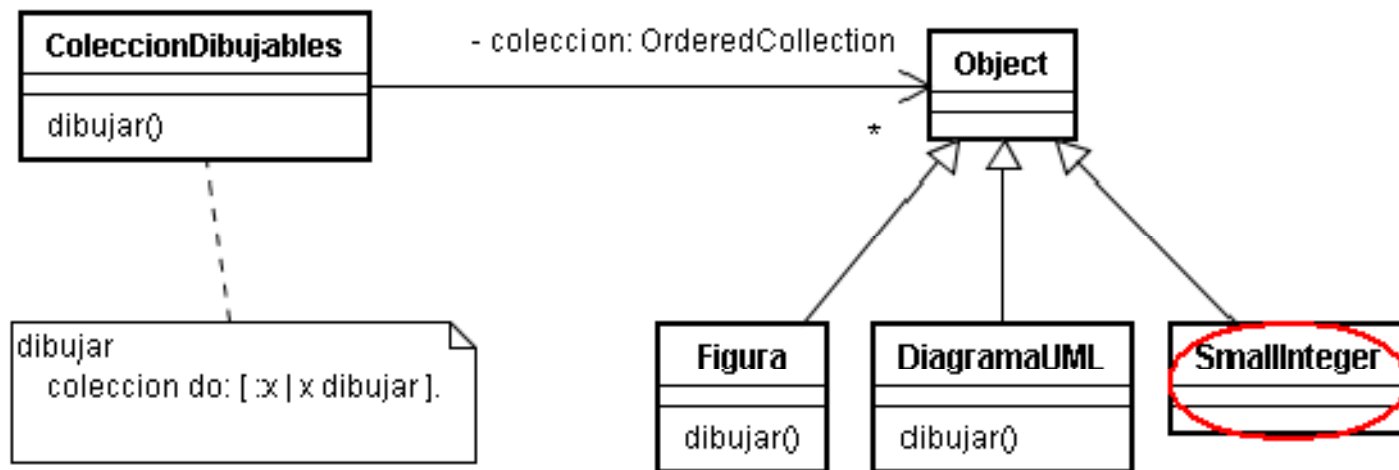
Si un programa es un modelo => un modelo de un programa es un modelo de un modelo



Polimorfismo sin herencia

Riesgo: no hay forma de asegurar que el mensaje sea comprendido por el receptor

Si no se comprende, vamos a tener un error en tiempo de ejecución



Podría solucionarse con herencia múltiple

O asegurando una interfaz mínima: solución de Java



Herencia con excepciones: el círculo y la elipse

Todo círculo es una elipse

Una elipse tiene dos radios, un círculo sólo uno

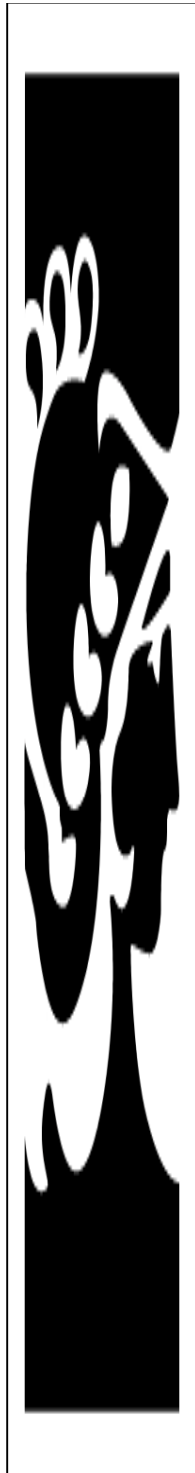
¿Quién debe ser la madre?

¿Cómo debe implementarse?

Otras cuestiones:

Girar la elipse

Achatar la elipse



Claves

Usamos excepciones cuando no hay suficiente información en el contexto del potencial problema

UML es una herramienta de modelado

- Para discutir diseños antes del código

- Para generar documentos que sirvan después de la construcción

Es preferible que el polimorfismo se presente en contextos de herencia

- Aunque no es forzoso



Lecturas obligatorias

“What’s a Model For?”, Martin Fowler

Descargable en

<http://martinfowler.com/distributedComputing/purpose.pdf>



Lecturas optativas

UML Distilled 3rd Edition, Martin Fowler,
capítulo 1 “Introduction”

Hay edición castellana de la segunda edición

Debería estar en biblioteca

UML para programadores Java, Robert Martin,
capítulo 2 “Trabajar con diagramas”

No está en la Web ni en la biblioteca

Orientación a objetos, diseño y programación,
Carlos Fontela 2008, capítulo 9:
“Excepciones”



Qué sigue

Temas de desarrollo de software

Calidad de código y buenas prácticas

Primer parcial

