Estructural Creacional

Composite & Builder

OO2 - 2025

Federico Balaguer

1 Talco
Código Producto

Array de 4 Talcos
Cód. Prod. == Talco
Precio * 4

1 Bundle
Cód. Prod. != Talco
Precio promo

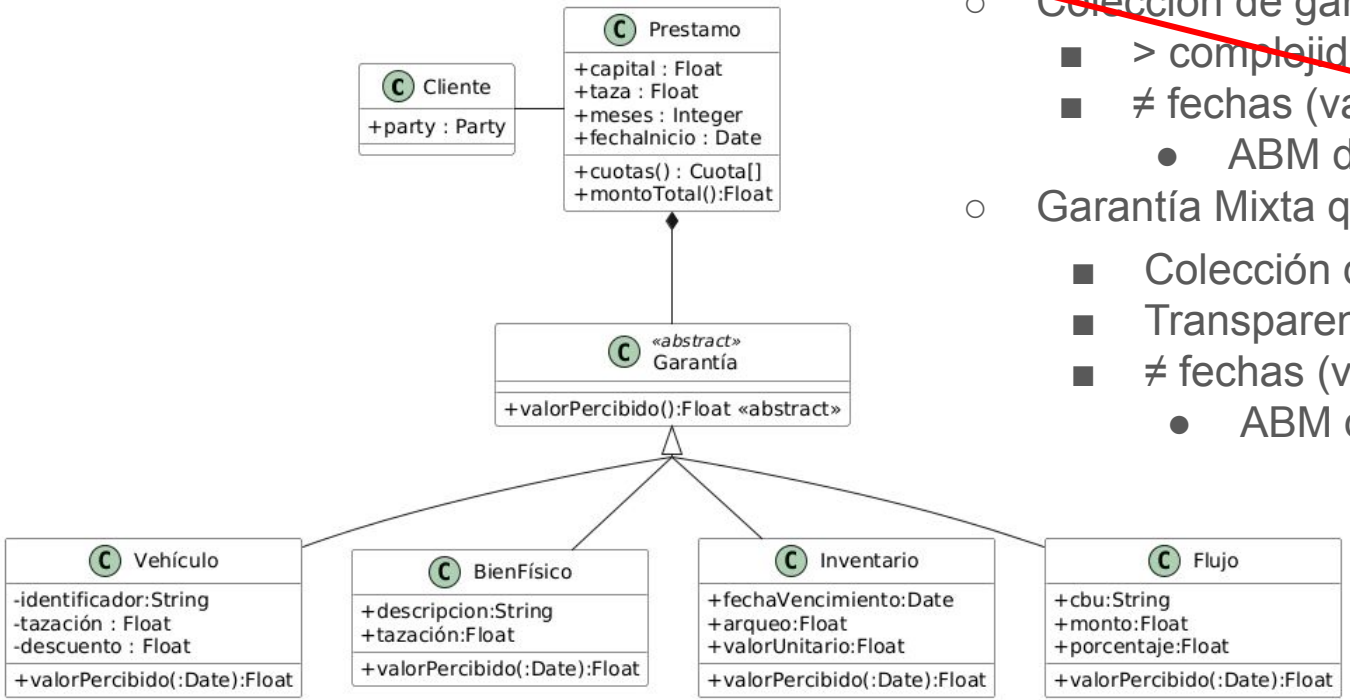
1 Bundle
Cód. Prod. != Talco != Bundle
Precio Kit

Prestamos con Garantías Prendarias

- Prestamo
  - Adelanto de dinéro (capital)
  - Se devuelve: capital + intereses
- Garantía: un bien (mueble) que se ejecutará si el deudor no paga
  - Vehículos (autos, motos, camionetas), maquinarias, joyas
  - Flujos: sueldos, alquileres, facturas futuras
- Garantía vs Capital
  - ¿Cuál es el valor de un auto usado?
    - El máximo valor de mercado puede demorar una venta.
    - El valor de recupero es el valor de venta - gastos (publicidad, impuestos, trámites)

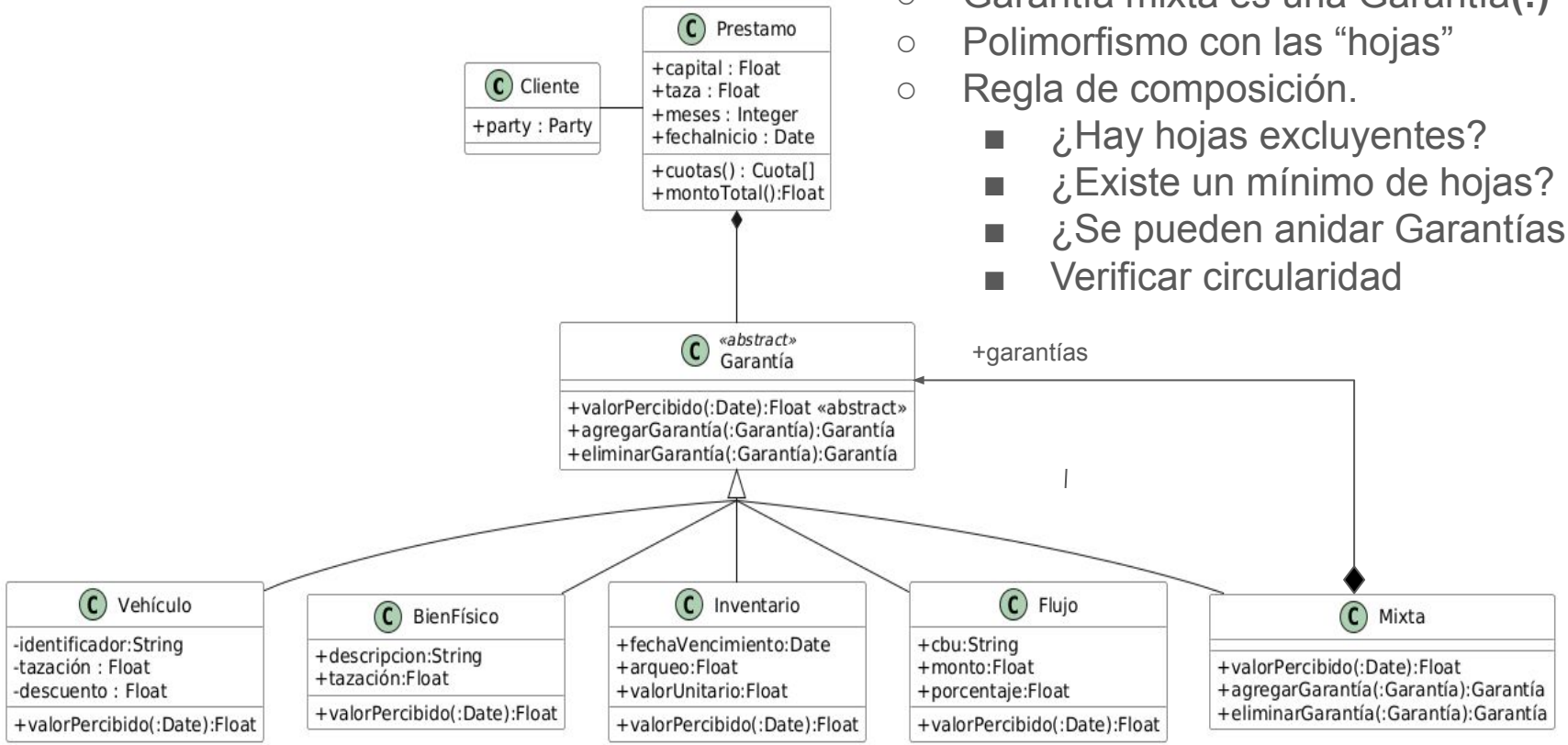
- Garantía inmueble == Prestamo Hipotecario
  - “Subprime” son prestamos de alto riesgo

- Garantía.valorPercibido >= Prestamo.capital
- 1 Prestamo ⇒ 1 Garantía
- Cómo podríamos tener multi-garantías?
  - Colección de garantías en Prestamo
    - > complejidad en Prestamo
    - ≠ fechas (validez y maturity)
      - ABM de (sub) Garantías
  - Garantía Mixta que es una Garantía **(!)**
    - Colección de Garantías
    - Transparente para Prestamo
    - ≠ fechas (validez y maturity)
      - ABM de (sub) Garantías



Descargo:foco en comportamiento relevante al problema. Variables publicas pueden ser implementadas como setters/getters o ocultadas en la implementacion

- Cómo podríamos tener multi-garantías?
  - Garantía mixta es una Garantía**(!)**
  - Polimorfismo con las “hojas”
  - Regla de composición.
    - ¿Hay hojas excluyentes?
    - ¿Existe un mínimo de hojas?
    - ¿Se pueden anidar Garantías Mixtas?
    - Verificar circularidad



Descargo:foco en comportamiento relevante al problema. Variables publicas pueden ser implementadas como setters/getters o ocultadas en la implementacion

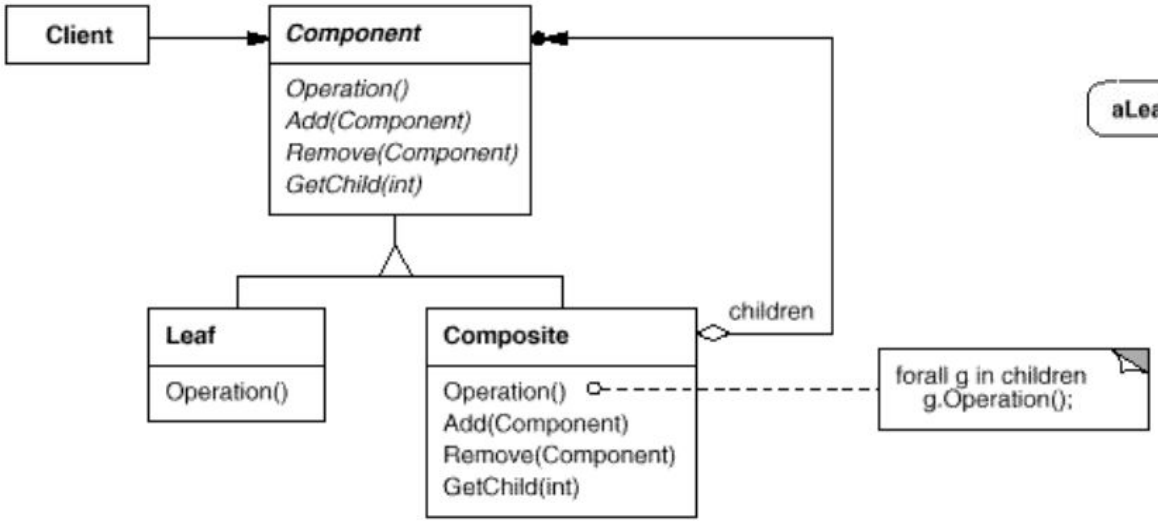
Composite Design Pattern

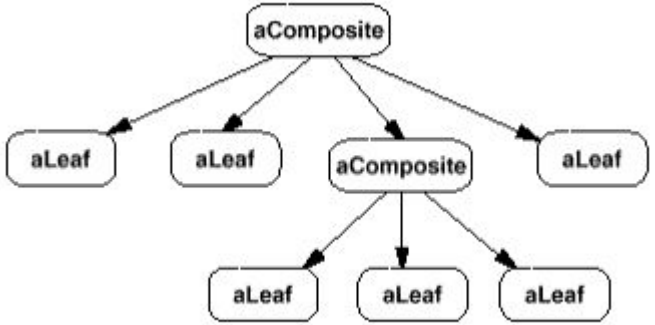
Propósito: Componer objetos en estructuras de árbol para representar jerarquías parte-todo. El Composite permite que los clientes traten a los objetos atómicos y a sus composiciones uniformemente.

Aplicabilidad: Use el patrón Composite cuando

- quiere representar jerarquías parte-todo de objetos.
- quiere que los objetos “clientes” puedan ignorar las diferencias entre composiciones y objetos individuales. Los clientes trataran a los objetos atómicos y compuestos uniformemente.

Composite.Estructura





Composite. Participantes

Component:

- Declara la interfaz para los objetos de la composición.
- Implementa comportamientos default para la interfaz común a todas las clases.
- Declara la interfaz para definir y acceder “partes de la composición”.

Leaf:

- Las hojas no tienen sub-árboles.
- Define el comportamiento de objetos primitivos en la composición.

Composite:

- Define el comportamiento para componentes complejos.
- Implementa operaciones para manejar el sub-árboles

Composite. Consecuencias

- **(+)**Define “jerarquías” de objetos primitivos y compuestos.
- **(+)**Los objetos primitivos pueden componerse en objetos complejos, los que a su vez pueden componerse recursivamente.
- **(+)**Simplifica los objetos cliente. Los clientes usualmente no saben (y no deberían preocuparse) acerca de si están manejando un compuesto o un simple.
- **(+)**Hace más fácil el agregado de nuevos tipos de componentes porque los clientes no tienen que cambiar cuando aparecen nuevas clases componentes.
- **(-)**Debe ser “customizado” con reglas de composición (si fuera necesario)

Composite. Cuestiones de Implementación

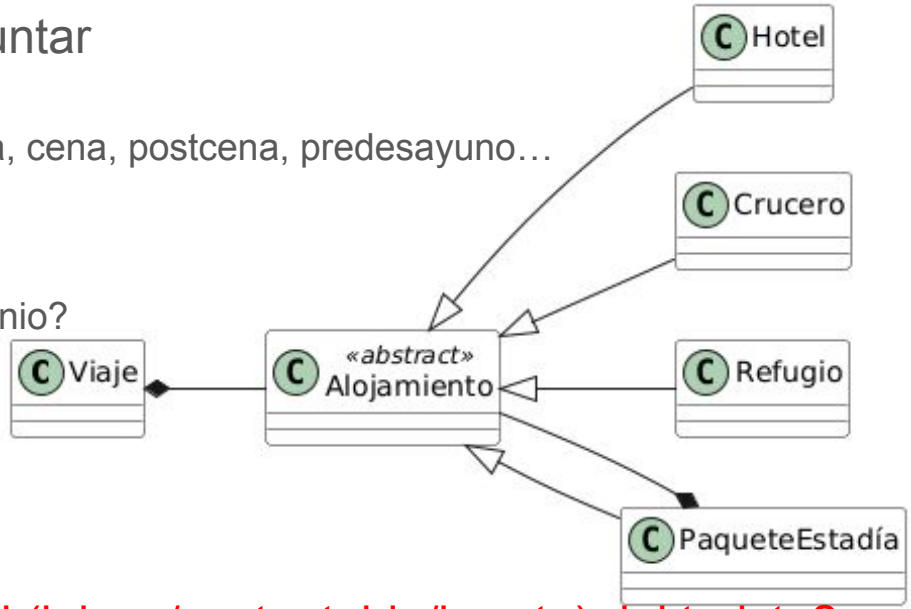
- Referencias explícitas a la raíz de una hoja
- Maximizar el protocolo de la clase/interfaz Component
- Orden de las hojas
- Borrado de componentes
- Búsqueda de componentes(fetch) por critérios
- Diferentes estructuras de datos para guardar componentes

Algunos usos y cosas a tener en cuenta

- Ejemplos comunes:
  - group/ungroup de elementos gráficos
  - Carpetas, archivos y link simbolicos (sistemas de archivos)
- Garantías en operaciones
  - Alquileres de inmuebles (garantía: Sueldo, Inmueble)
  - Prod. Financieros (collaterals: acciones, bonos, plazos fijos)
  - Prestamos prendarios (lo visto en el ejemplo)
- Agrupamiento
  - Combos (cajita feliz, talcos, útiles escolares)
  - Productos financieros (prestamos hipotecarios)
  - Pixels/Celdas en sensado remoto (imagenes satelitales)
  - Group/Ungroup de elementos gráficos
  - Paquetes de Alojamiento…
- **Es fundamental entender**
  - **el comportamiento de las hojas y los sub-árboles**
  - **Cuáles son las reglas de composición (configuración)**
    - Solapamiento, continuidad, circularidad, mutua exclusión, etc
  - **¿Qué objetos/métodos se encargan de crear las configuraciones?**

Paquete Turístico… alojamiento

- Durante un viaje se puede pernoctar en
  - Hoteles / Hostels
  - Cruceros
  - Refugios (montaña, selva, Antartida)
- En todos los casos de puede preguntar
  - Si una noche está cubierta
  - Si cubre desayuno, refrigerio, merienda, cena, postcena, predesayuno…
  - Valor Final y Costo
- Existe el “paquete alojamiento”?
  - Tiene sentido para un experto del dominio?
- Configuraciones
  - Verificar que tengan continuidad
  - Evitar solapamientos
  - Evitar Circularidad

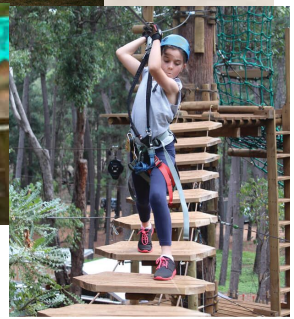


¿Qué pasa si el mix depende de el perfil (lujoso/sustentable/barato) del turista?

- Transporte
- Alojamiento
- Asistencia Médica
- Coordinación (profes y Guardavidas)
- Fiesta
- Excursiones

El esquema general es el mismo

- Transporte
- Alojamiento c/comidas
- Asistencia Médica
- Coordinación
  - Profes
  - Guardavidas
- Fiesta
- Excursiones



¿Cuál es el problema? (pregunta de final)

- Cambios de estado de un objeto en tiempo de ejecución?
- Una definición única que puede ser “customizable” a casos específicos?
- Adaptar protocolos entre objetos?
- Manejar la configuración de objetos para que tengan diferente estructura?
  - Pero existe una definición reusable que se aplica a todos los casos
  - La "estructura" se repite en la estructura del producto pero las partes pueden ser diferentes en cada caso.

Builder

- Intención: separa la construcción de un objeto complejo de su representación (implementación) de tal manera que el mismo proceso puede construir diferente representaciones (implementaciones)
- Estructura (**Roles**)

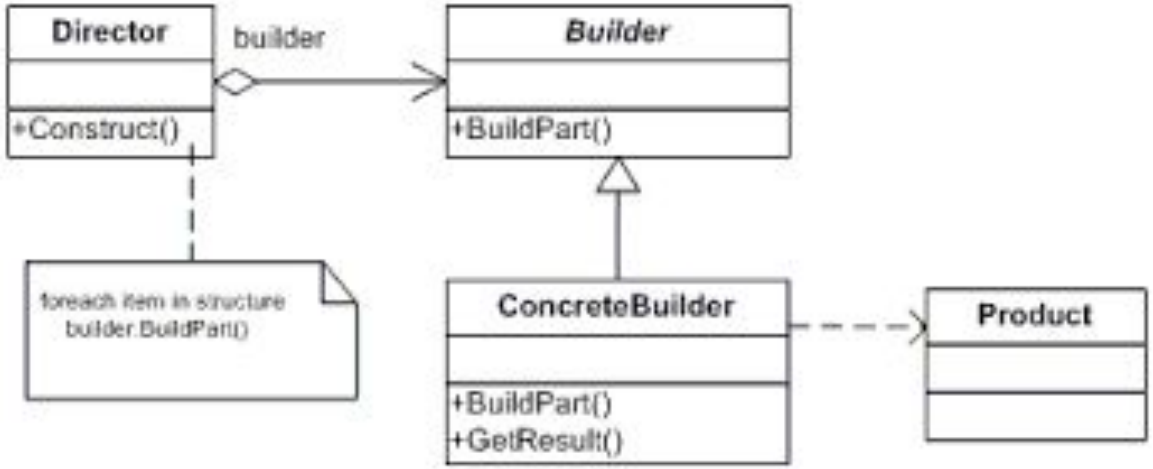
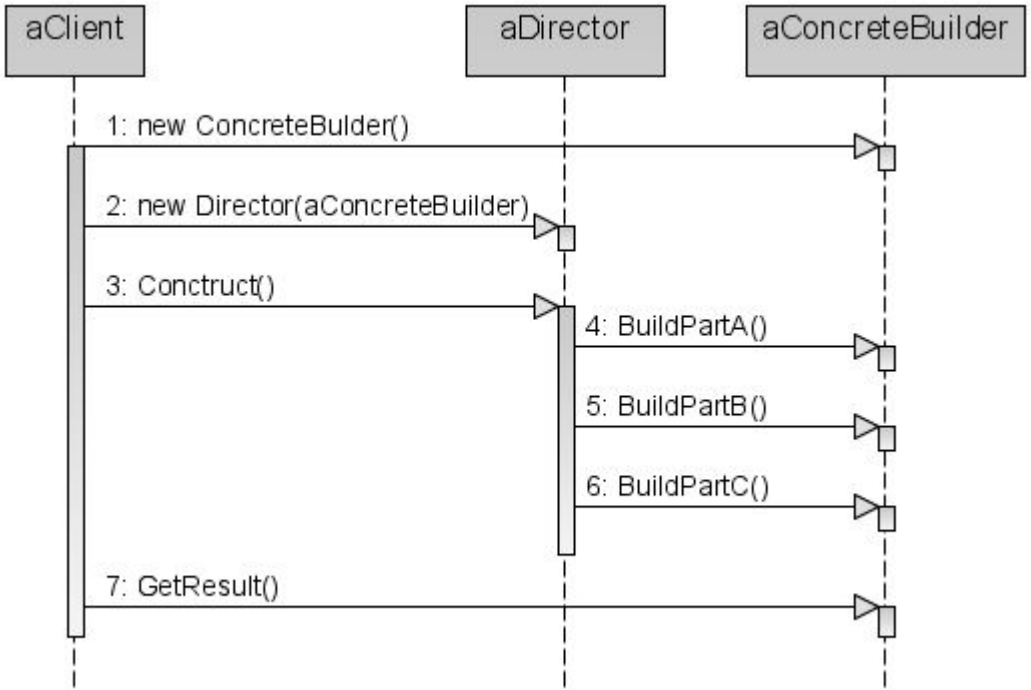


Diagrama de Secuencia



Builder

Participantes

- Builder: especifica una interface abstracta para crear partes de un Producto
- Concrete Builder: construye y ensambla partes del producto.
  - Guarda referencia al producto en construccion
- Director: conoce los pasos para construir el objeto
  - Utiliza el Builder para construir las partes que va ensamblando
  - En lugar de pasos fijos puede seguir una “especificación” (ver knwon uses@ GOF)
- Product: es el objeto complejo a ser construido

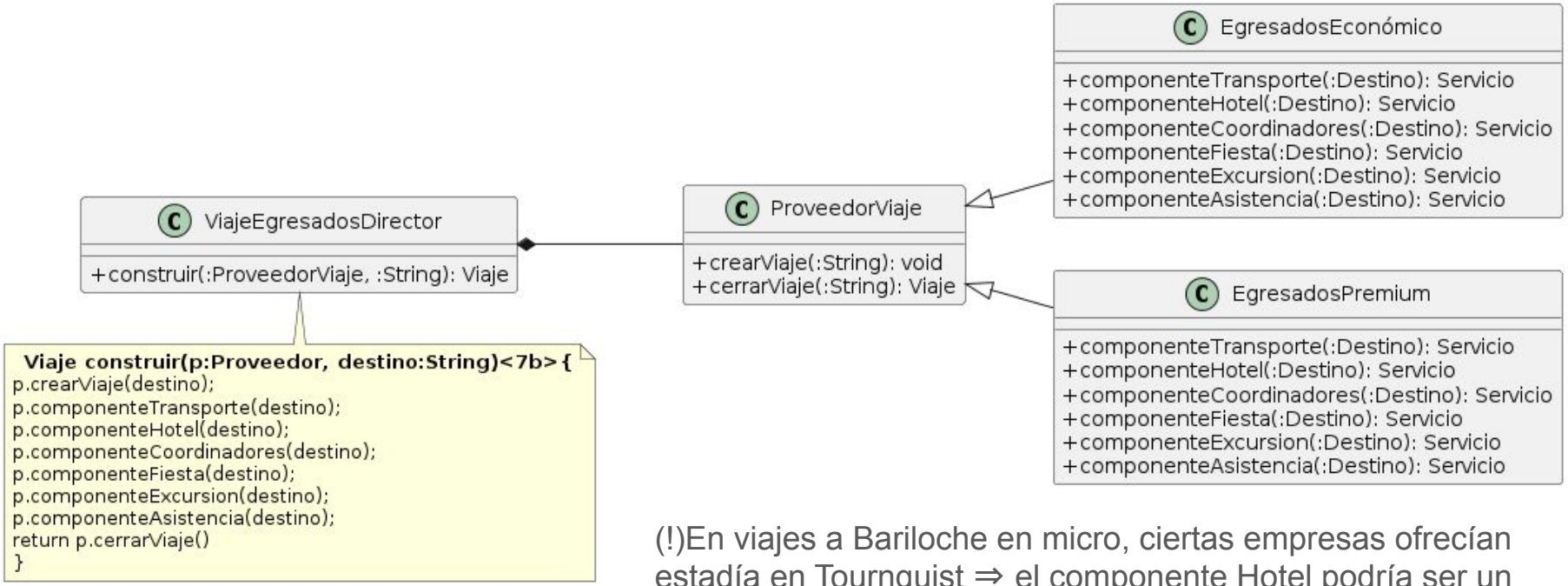
Vale la pena Builder?

Si!

- Abstrae la construcción compleja de un objeto complejo
- Permite variar lo que se construye Director <-> Builder
- Da control sobre los pasos de contrucción

No!

- Requiere diseñar y implementar varios roles
- Cada tipo de producto requiere un ConcreteBuilder
- Builder suelen cambiar o son parsers de specs (> complejidad)



(!)En viajes a Bariloche en micro, ciertas empresas ofrecían estadía en Tournquist ⇒ el componente Hotel podría ser un Composite con los hoteles de Tournquist y Bariloche.

(!)Si el destino fuera Río de Janeiro podría haber 2 paradas intermedias (Sto Tomé y Curitiba).

¡Todo eso para el Director es transparente!

Consideraciones sobre Builder

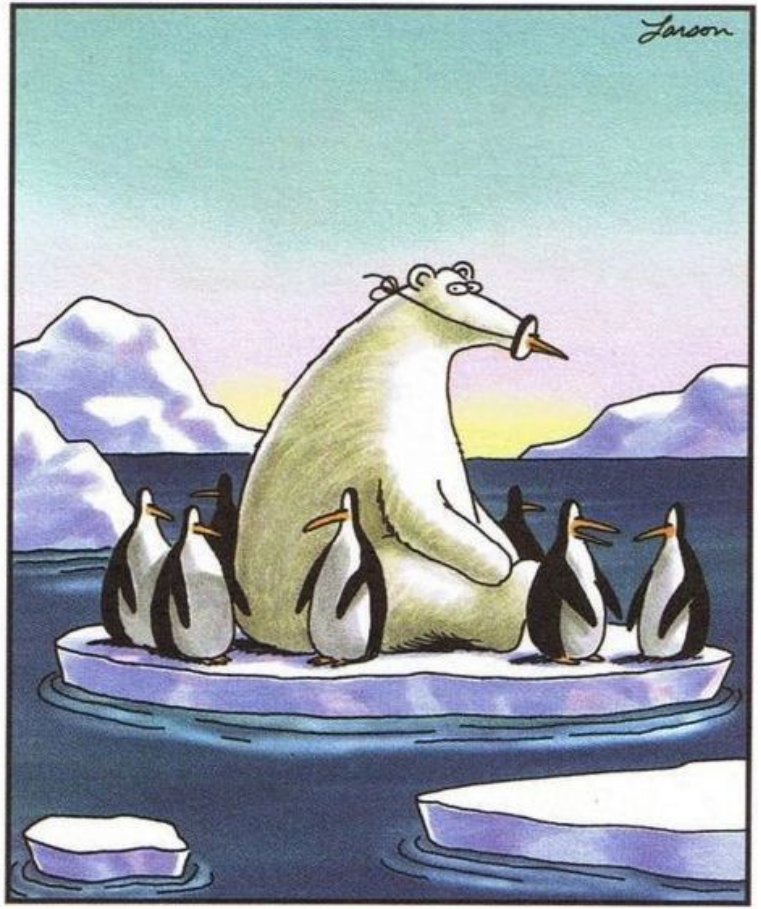
1. El Director **solo** sabe hacer **una** cosa
2. Los Builders pueden saber hacer cosas que no requiera un Director pero si otro
   a. Ej: componentePsicologico()
3. Los Builders funcionan (generalmente) "injectando dependencias" para armar configuraciones en run-time.
4. Otros Directors pueden usar los mismos Builders.
5. Nuevas definiciones de Viajes de Egresado ⇒nuevos directores
6. Nuevos servicios ⇒ nuevos Builders

Preguntas de Final…

1)¿Cómo se implementa un viaje a Punta del Este?

2) ¿Cómo se implementa un viaje a Río de Janeiro con aéreo y traductor?

3) ¿Cómo se implementa un viaje con múltiples destinos? Ej: Tandil y Miramar



And now Edgar's gone. ...
Something's going on around here.

Errores comunes

Composite	La jerarquía no es polimorfico Composite no tiene relación con otras clases de Jerarquía Composite no implementa add/remove Composite no delega en la coleccion
Builder	Director construye partes Director necesita diferentes builders al mismo tiempo Builder manda mensajes a Director Solo hay un Builder