## Patrones de Diseño: Patrones de Comportamiento. Tema 5-12: Visitor

#### Descripción del patrón

#### Nombre:

Visitante

#### Propiedades:

- Tipo: comportamiento
- Nivel: objeto, componente

#### Objetivo o Propósito:

 Proporcionar una forma fácil y sostenible de ejecutar acciones en una familia de clases. Este patrón centraliza los comportamientos y permite que sean modificados o ampliados sin cambiar las clases sobre las que actúan.





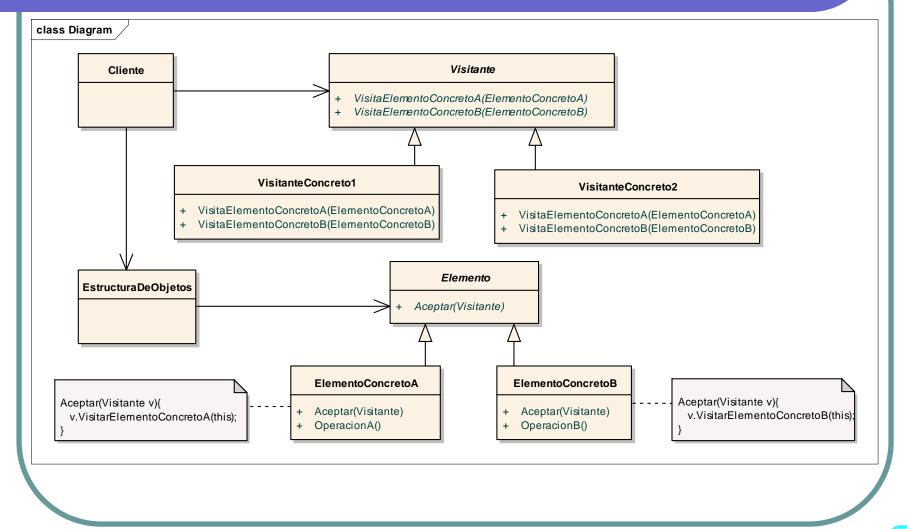
#### Aplicabilidad

- Use el patrón Visitor cuando:
  - Un sistema contenga un grupo de clases relacionadas.
  - Haya que realizar muchas operaciones distintas y no relacionadas sobre algunos o todos los objetos de una estructura de objetos y no queremos contaminar sus clases con dichas operaciones.
  - Las operaciones deban ejecutarse de forma diferente en cada una de las distintas clases.





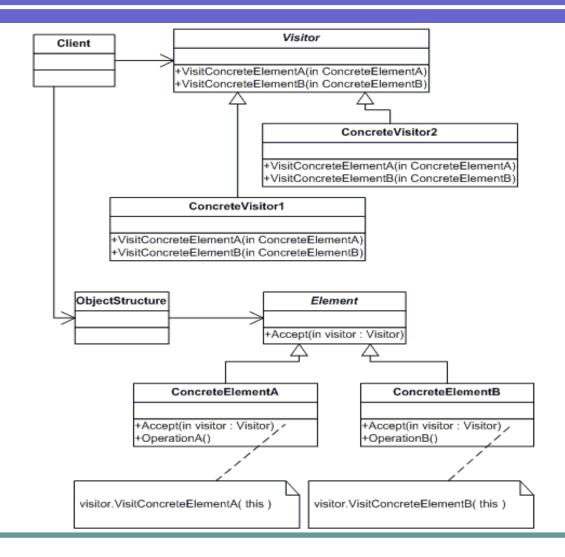
#### Estructura







#### Estructura





#### Estructura. Participantes

- Visitante: Interfaz o clase abstracta que define el método visitar para cada una de las clases Elemento concretas.
- VisitanteConcreto: Clase que representa una operación específica del sistema. Implementa la interfaz Visitante para una operación o algoritmo específico.
- Elemento: Clase abstracta o interfaz que representa los objetos sobre los que actúa Visitante. Define el método aceptar que recibe un visitante como argumento.
- **ElementoConcreto:** Implementa la interfaz Elemento. Implementa el método *aceptar* invocando el método *visitar* apropiado definido en Visitante.
- EstructuraDeObjetos: Para poder enumerar todos los elementos.





#### Consecuencias

- Facilita en gran medida la introducción de un nuevo comportamiento en un sistema. Para añadir nuevas funciones basta con crear simplemente una nueva clase que implemente la interfaz Visitante y escribir el nuevo código para realizar la función.
- Visitante es útil porque permite centralizar código funcional para una operación. Hace que el código sea más fácil de ampliar y modificar y el mantenimiento es más sencillo.
- Un visitante agrupa operaciones relacionadas y separa el comportamiento no relacionado en las subclases del visitante.
- El inconveniente de este patrón es que ofrece muy poca flexibilidad en las clases Elemento. Cualquier nueva clase Elemento hace necesario definir un nuevo método en la interfaz Visitante y en cada visitante concreto su implementación.





#### Patrones relacionados

- Interpreter: Se puede utilizar Visitor para centralizar la interpretación de las operaciones
- Composite: Los visitantes pueden usarse para aplicar una operación sobre una estructura de objetos definida por Composite.



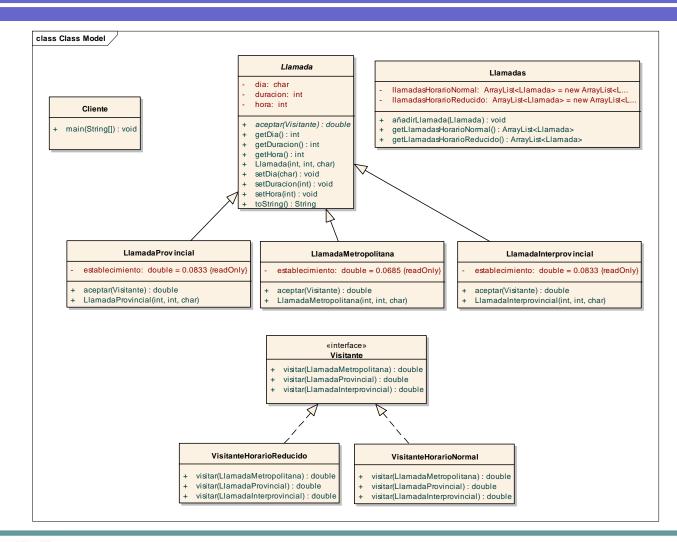
## Código de ejemplo

# Factura Telefono





### Código de ejemplo







## Código de ejemplo

- Identificamos a continuación los elementos del patrón:
  - Visitante: Visitante
  - VisitanteConcreto: VisitanteHorarioReducido, VisitanteHorarioNormal.
  - Elemento: Llamada.
  - ElementoConcreto: LlamadaProvincial, LlamadaMetropolitana, LlamadaInterprovincial.
  - EstructuraDeObjetos: Llamadas.



