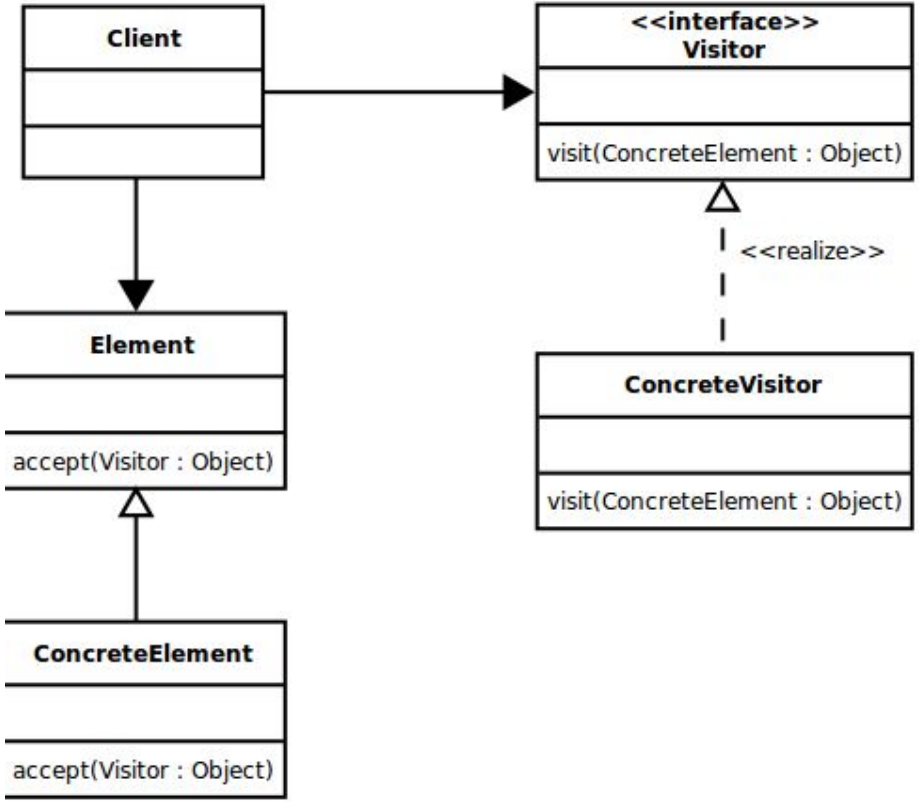
Padrão Visitor

Padrão comportamental

Visitor

- *""Representar uma operação a ser realizada sobre os elementos de uma estrutura de objetos. " (GoF)*
- Separação de um algoritmos da estrutura dos objetos em que atua
  - Percorre elementos e realiza operações
  - Nova funcionalidade pode ser incluida
- Exemplos
  - Impressão, transformação de dados, análise sintática e semântica
- Princípio
  - Interface elemento com os dados
  - Interface visitor com as operações



Fonte: wikipedia

Definição e uso

```
interface IVisitor {
    void visit(Depto depto);
    void visit(Aluno aluno);
}
interface IEntidade {
    void accept( IVisitor visitor);
}
class Aluno implements IEntidade {
    private String name;
    public void accept( IVisitor visitor) {
        visitor.visit(this);
    }
}
class Depto implements IEntidade {
    ArrayList pessoas = new ArrayList();
    public void accept(IVisitor visitor) {
        for(IEntidade pessoa : pessoas) {
            pessoa.accept(visitor);
        }
        visitor.visit(this);
    }
}
class AlunoPrint implements IVisitor {
    public void visit(IEntidade entidade) {
        if (entidade instanceof Aluno)
           System.out.println("Visiting aluno);
    }
}
```

```
public class VisitorDemo {
    static public void main(String[] args){
        Depto depto= new Depto();
        depto.accept(new AlunoPrint());
    }
}
```

Visitor

- Considerações
  - o objeto visitor pode ter um estado (atributo) – ex.: indentação
  - implementar o percurso no visit ou no accept ?
- Vantagens
  - Facilita adição de operações
  - Agrupa relações relacionadas
  - Clareza no percurso
- Desvantagens
  - **Accept** no elemento -> encapsulamento rompido
  - Inclusão de novos elementos é trabalhosa

Relacionamento entre os padrões

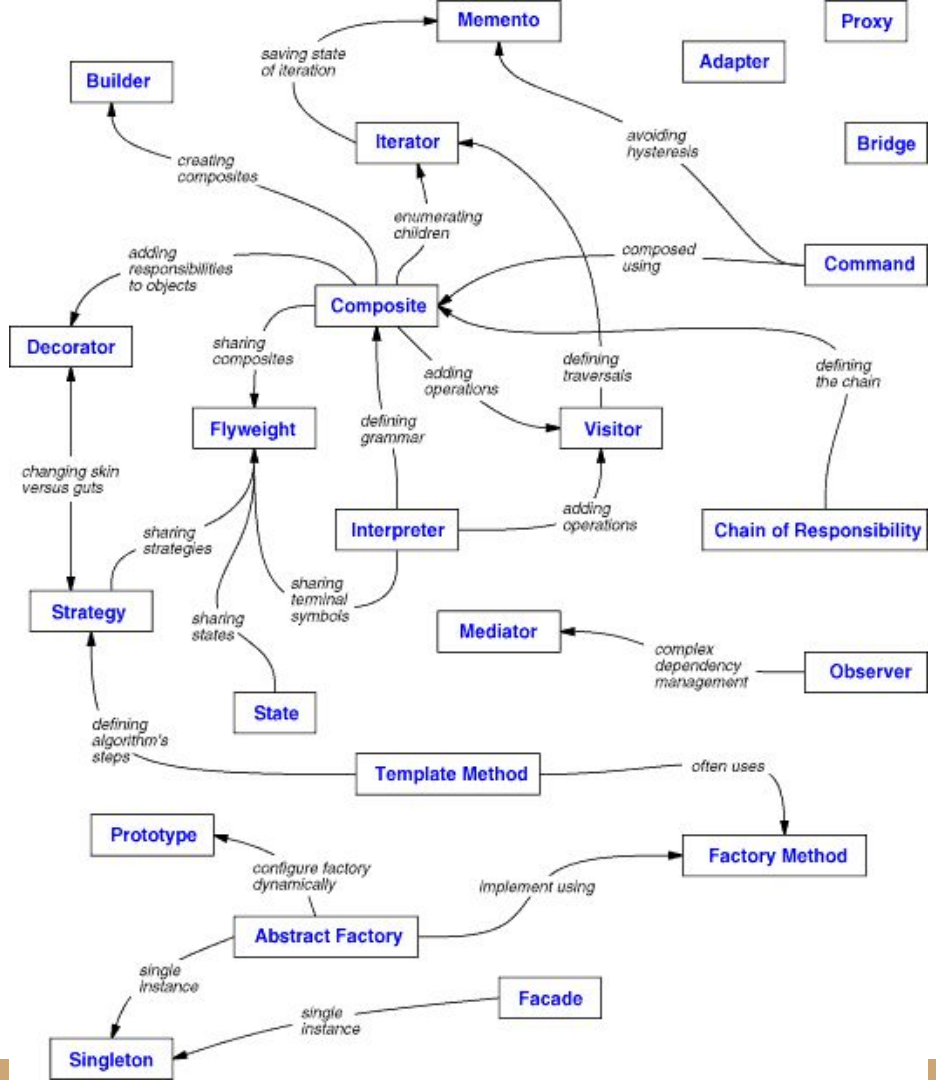


Figura 1.1 do livro