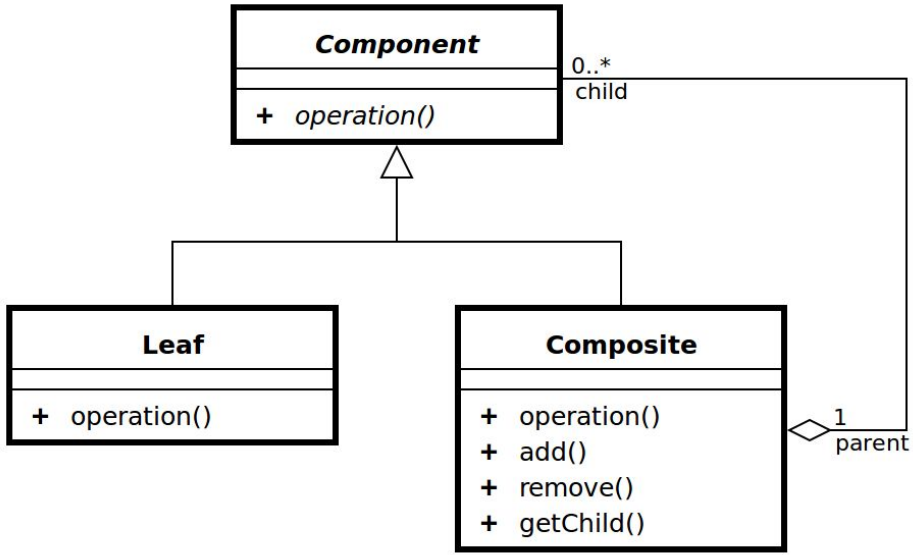
Padrão Composite

Padrão estrutural

Composite

- *""Compor objetos em estruturas de árvore para representar hierarquias todo-parte" (GoF)*
- Tratamento uniforme entre objetos compostos e individuais
  - Sistemas de arquivos, arquivos PDFs, sistemas de janelas (interfaces), toolkits de desenvolvimento
- Princípio
  - Interface *Component* com operações
  - Classes para folha e “obj. compostos”



Fonte: wikipedia

Definição

```
interface Graphic {
    public void print();
}

class CompositeGraphic implements Graphic {
    private List<Graphic> child = new ArrayList<Graphic>();
    public void print() {
        for (Graphic graphic : child) {
            graphic.print();
        }
    }
    public void add(Graphic graphic) {
        child.add(graphic);
    }
    public void remove(Graphic graphic) {
        child.remove(graphic);
    }
}
class Square implements Graphic {
    public void print() {
        System.out.println("Square");
    }
}
```

Uso

```
/** Cliente */
public class Program {
    public static void main(String[] args) {
        //Instancia os quadrados
        Square s1 = new Square();
        Square s2 = new Square();
        //Instancia os gráficos
        CompositeGraphic graphic = new CompositeGraphic();
        CompositeGraphic graphic1 = new CompositeGraphic();
        CompositeGraphic graphic2 = new CompositeGraphic();
        //Cria a composição
        graphic1.add(s1);
        graphic1.add(s2);
        graphic.add(g2);
        //Imprime todos os objetos gráficos
        graphic.print();
    }
}
```

Composite

- Vantagens
  - Adequado quando relacionamento entre objeto define árvores
- Desvantagens
  - Dificuldade quando há ciclos
  - Relação de composição precisa ser implementada manualmente
- Questões
  - Que estrutura usar para a lista ?
  - Incluir referências inversas ?
  - Compartilhar componentes ? (associação)
  - Maximizar (muitos métodos) a interface *Component ?*

Relacionamento entre os padrões

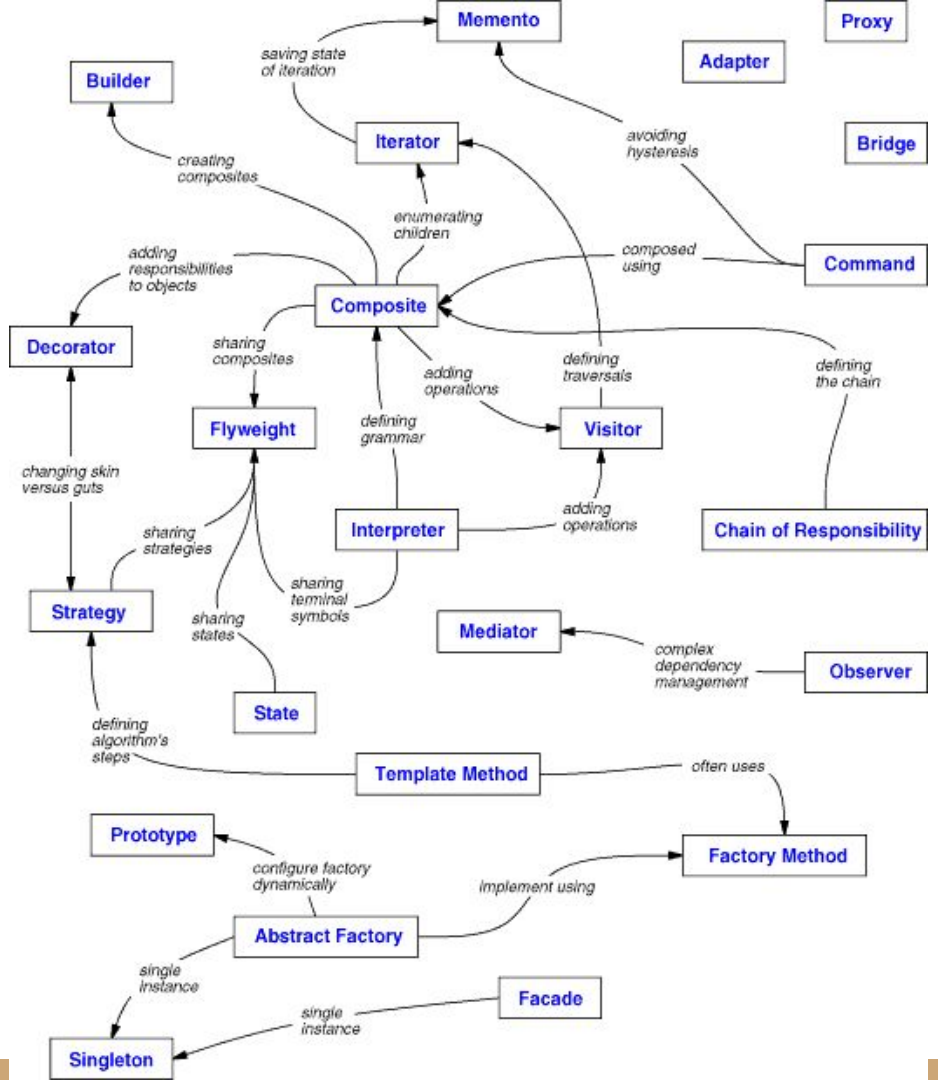


Figura 1.1 do livro