

# Separação de Interesses

Eduardo Figueiredo

<http://www.dcc.ufmg.br/~figueiredo>

# [ Motivação ]

- Queremos desenvolver software de qualidade
  - Reusável
  - Flexível
  - Confiável, etc.
- A complexidade de software é sempre crescente
  - Traz novos desafios para o desenvolvimento de software

# [ Um Pouco de História ]

*Para contornar a complexidade de um problema, deve-se resolver uma questão importante por vez*

**Dijkstra, 1976**

- Separação de interesses
  - Paradigmas de desenvolvimento de software evoluem para apoiar uma melhor separação de interesses

# [ Programação Estruturada ]

- Ações
  - Funções e procedimentos
- Dados
  - Informação manipulada pelas ações
- Ações e dados são projetados separadamente
  - Inaugurou a era da programação em alto nível

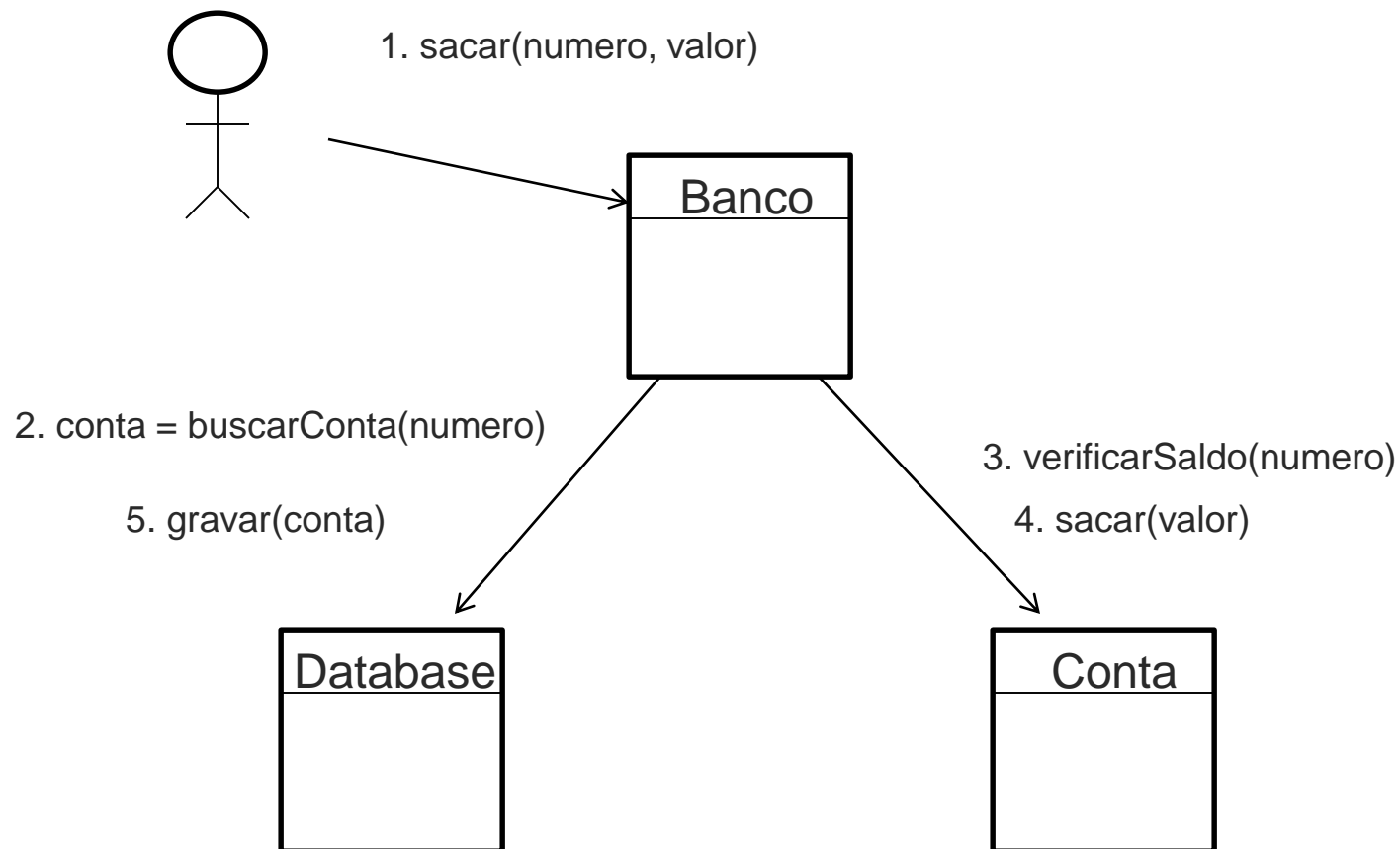
# [ Programação OO ]

- Abstrações mais próximas do mundo real (problema do cliente)
  - Objetos, não funções
- Objetos reúnem dados e ações em uma mesma entidade
- Um programa OO é um conjunto de objetos dizendo aos outros o que fazer
  - Troca de mensagens

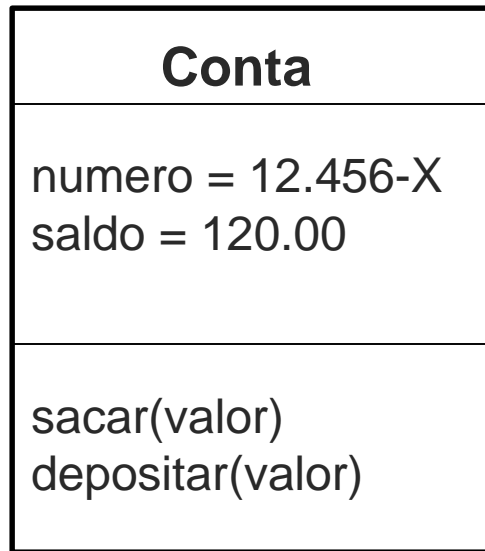
# [ Exemplos de Objetos ]

- Banco
  - Entidade financeira
- Conta
  - Uma conta bancária de um cliente
- Banco de Dados (de Contas)
  - Arquivo que armazena as contas dos clientes

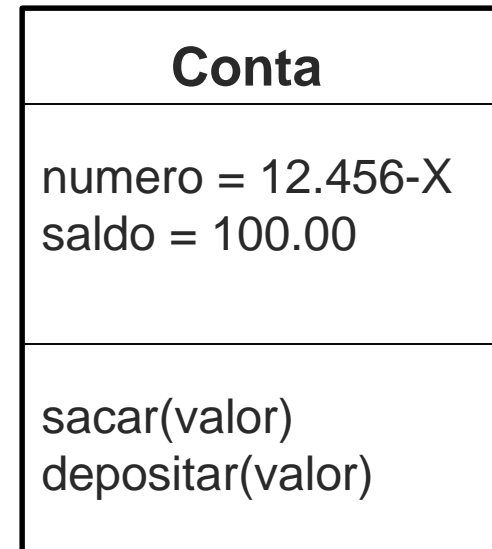
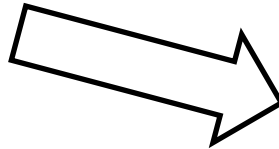
# [ A Operação Sacar ]



# [ Objeto Conta ]



sacar(20.00)



Estados do Objeto

# Definição da Classe (Java)

```
public class Conta {
    private String numero;
    private double saldo;

    ...
    public void debitar (double valor) {
        if (this.getSaldo() >= valor)
            this.setSaldo(this.getSaldo() -valor);
        else
            //erro!
    }
    public void creditar (double valor) {
        this.setSaldo(this.getSaldo()+valor);
    }
}
```

# [ Problemas de OO ]

- Complexidade de software sempre aumenta
- Fatores de qualidade não são adequadamente tratados
- Separação de interesses em OO
  - Oferece suporte a implementação de requisitos funcionais
  - Não se preocupa com os requisitos não funcionais

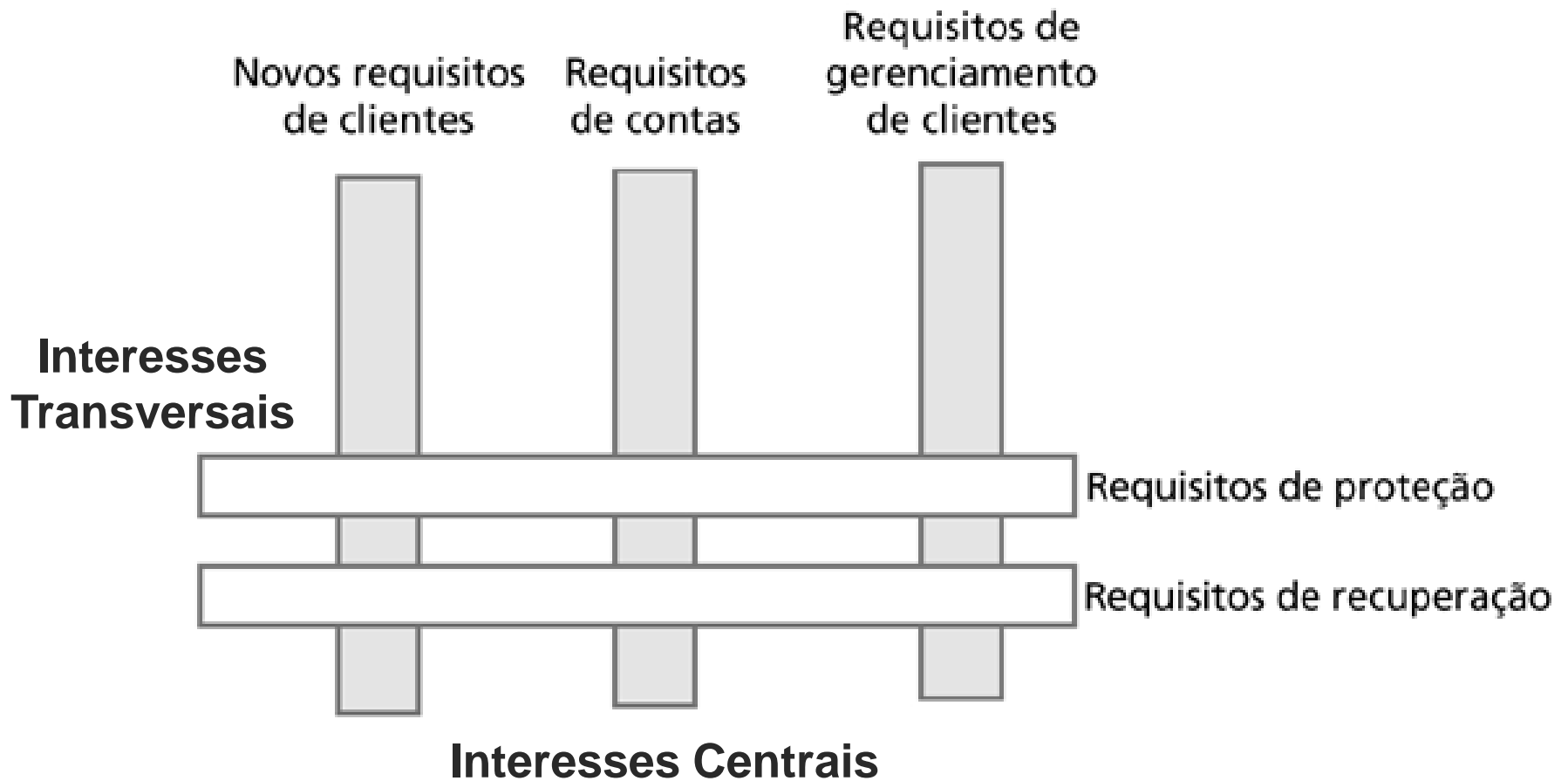
# [ Alguns Tipos de Interesses ]

- Interesse funcional
  - Relacionado à uma funcionalidade específica a ser incluída em um sistema
- Interesse de qualidade de serviço
  - Relacionado ao comportamento não funcional
- Interesse organizacional
  - Relacionado aos objetivos e às prioridades da organização
- Interesse de sistema
  - Relacionado à atributos do sistema como um todo

# Interesses Transversais

- São interesses cuja implementação atravessa uma série de componentes de programa
- Interesses transversais resultam em dois problemas no código
  - Espalhamento (*scattering*)
  - Entrelaçamento (*tangling*)
- Causam problemas quando mudanças devem ser feitas em um interesse

# [ Representação ]

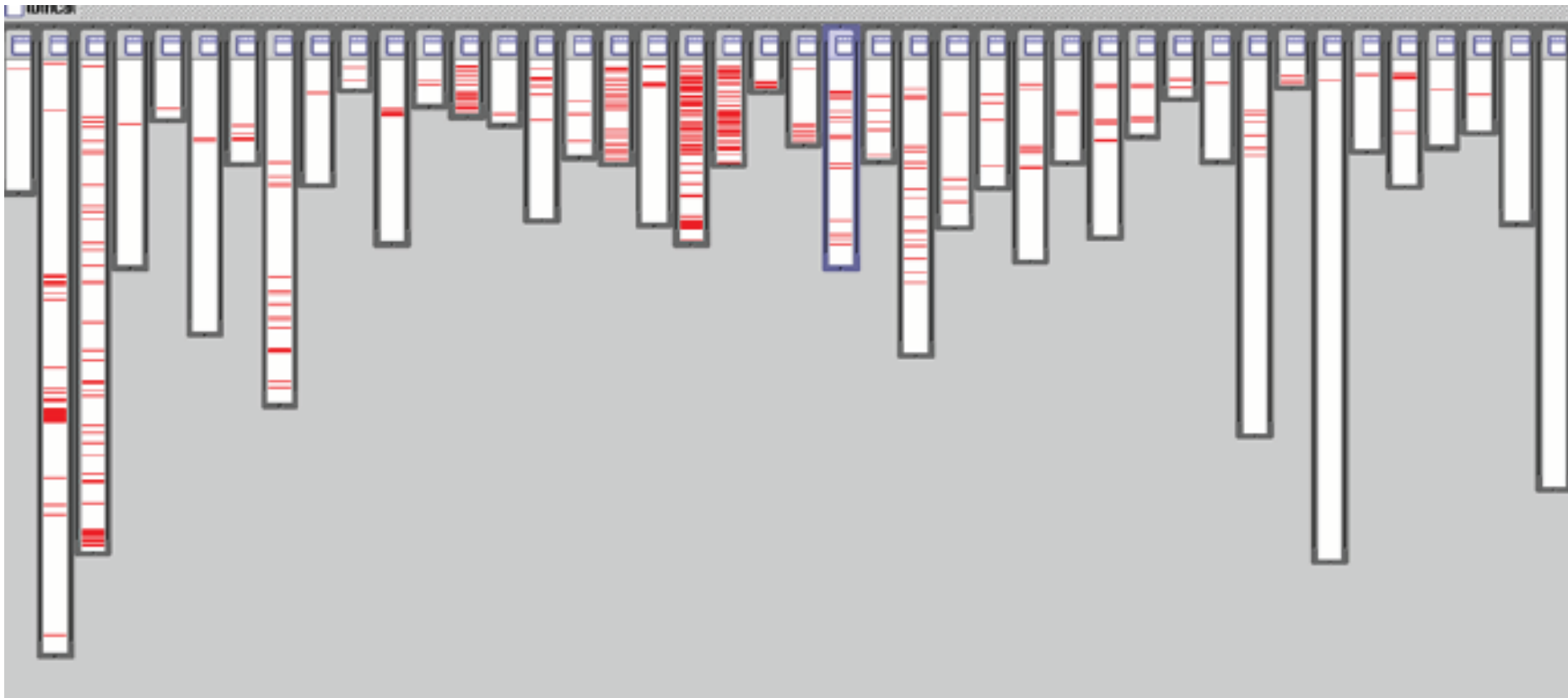


# [ Exemplo: *Logging* ]

```
public class Conta {
    private String numero;
    private double saldo;
    ...
    public void debitar (double valor) {
        if (this.getSaldo() >= valor){
            this.setSaldo(this.getSaldo()-valor);
            System.out.println("ocorreu um debito!");
        }...
    }...
}
```

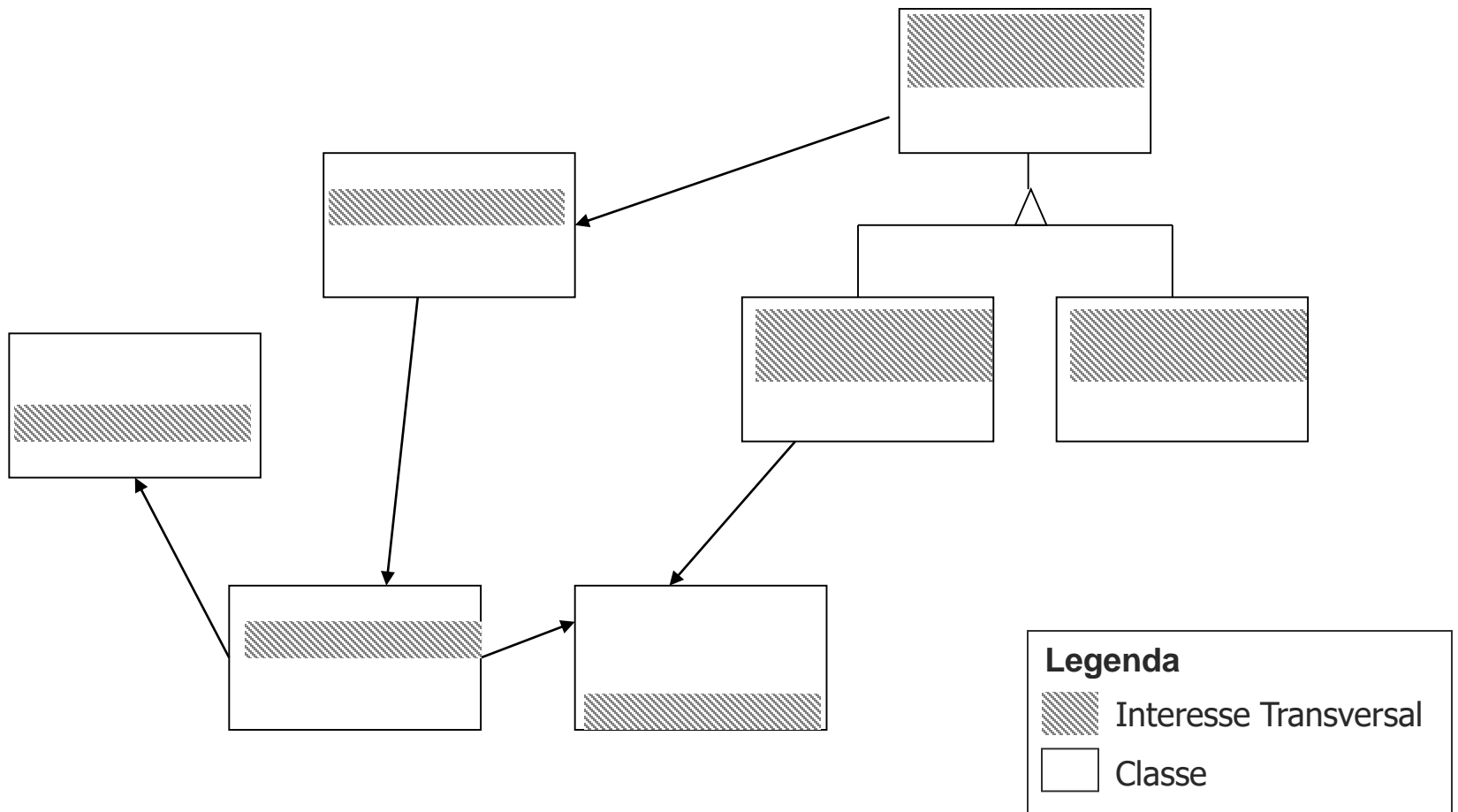
O código em vermelho implementa *Logging* e se espalha por várias partes do sistema

# [ Logging no Tomcat ]

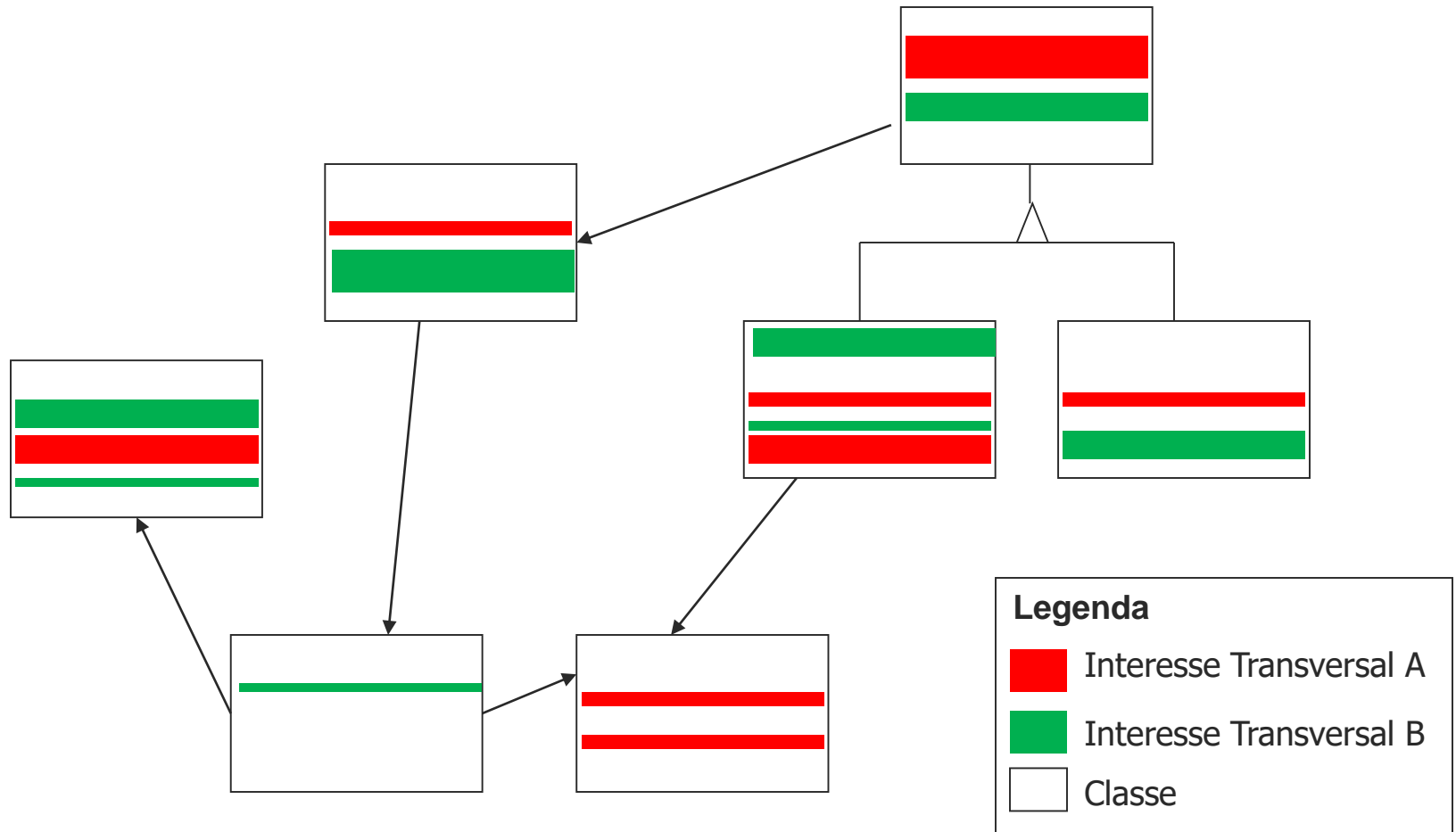


O código em vermelho implementa *Logging* e se espalha por várias partes do sistema

# [ Espalhamento ]



# [ Entrelaçamento ]



# [ Bibliografia ]

---

- Ian Sommerville. **Engenharia de Software**, 9ª Edição. Pearson Education, 2011.
  - Seção 21.1 Separação de Interesses