Instituto de Computação

Frameworks

Viviane Torres da Silva
viviane.silva@ic.uff.br

http://www.ic.uff.br/~viviane.silva/es1

Frameworks

- Motivação
- Definição
- Classificação
- Características
- Propriedades
- Técnicas de Customização
- Frameworks X Outras Abordagens

Motivação

- “Reusar” software não é simples
- Maioria dos esforços resultam apenas reutilização de pequenos componentes
- Principais vantagens do uso de frameworks
  - Aumento do reuso
  - Diminuição do tempo para produção de família de aplicações
- Framework provê reutilização de
  - Design
  - Código
- Reuso em larga escala

Definição

- Provê **solução para uma família de problemas** semelhantes
- Usa conjunto de **classes e interfaces que mostra como decompor** a família de problemas
- Como objetos dessas classes **colaboram** para cumprir suas responsabilidades
- Conjunto de classes deve ser **flexível** e **extensível**
  - Permite a construção de várias aplicações com pouco esforço
  - Especifica-se apenas as particularidades de cada aplicação

Outras Definições

- “Um framework é um conjunto de classes que constituem um design abstrato para soluções de uma família de problemas.”

  Johnson e Foote (1988)

- “Um framework é um conjunto de objetos que colaboram com o objetivo de cumprir um conjunto de responsabilidades para uma aplicação ou um domínio de um subsistema.”

  Johnson (1991)

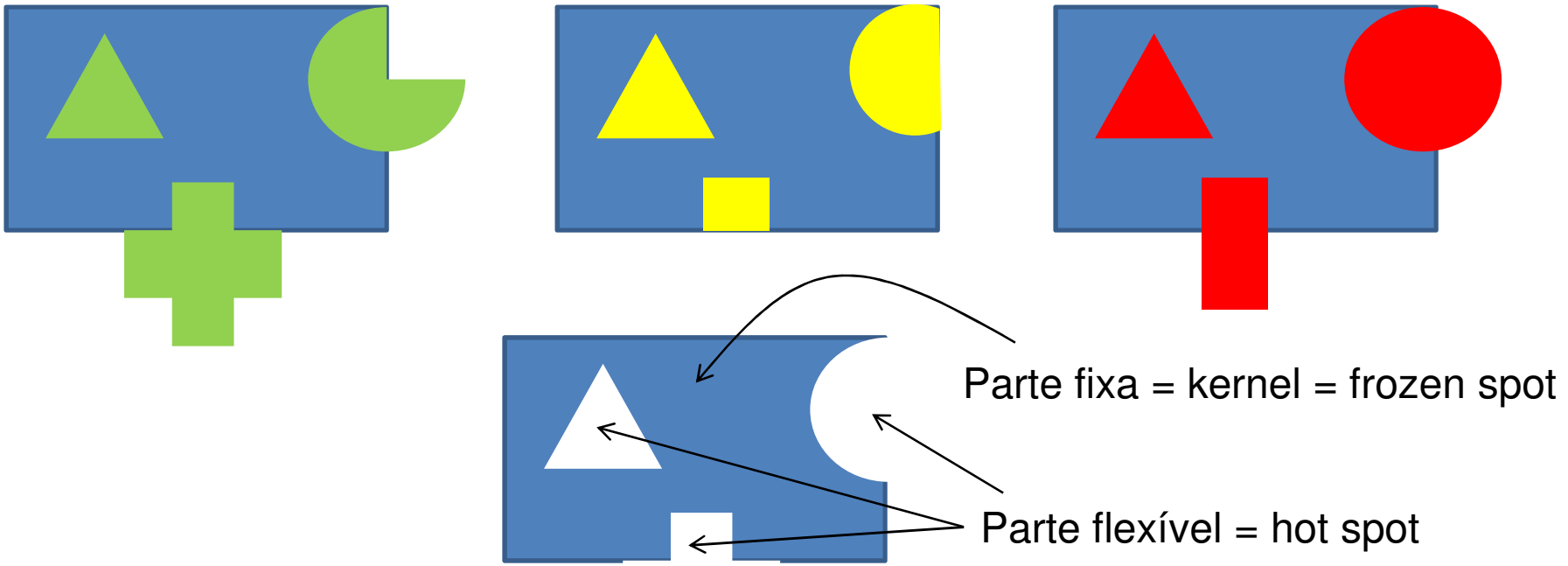
- “Uma arquitetura desenvolvida com o objetivo de se obter a máxima reutilização, representada como um conjunto de classes abstratas e concretas, com grande potencial de especialização.”

  Mattsson (1996)

Classificação

- Frameworks da Infra-estrutura de Sistema
  - Dão suporte ao desenvolvimento das infra-estruturas de sistemas como
    - Comunicações
    - Interfaces com usuário
    - Compiladores
- Frameworks de Integração com Middleware
  - Padrões e classes que dão suporte à comunicação de componentes e a troca de informação
- Frameworks de Aplicações Corporativas
  - Suportam o desenvolvimento de tipos específicos de aplicação como telecomunicações ou sistemas financeiros
  - Relacionados com Família de Programas

Abstração



- Frameworks não são executáveis
- Para produzir uma aplicação completa e executável
  - Instanciar o framework implementando código específico à aplicação para cada hot spot

Características

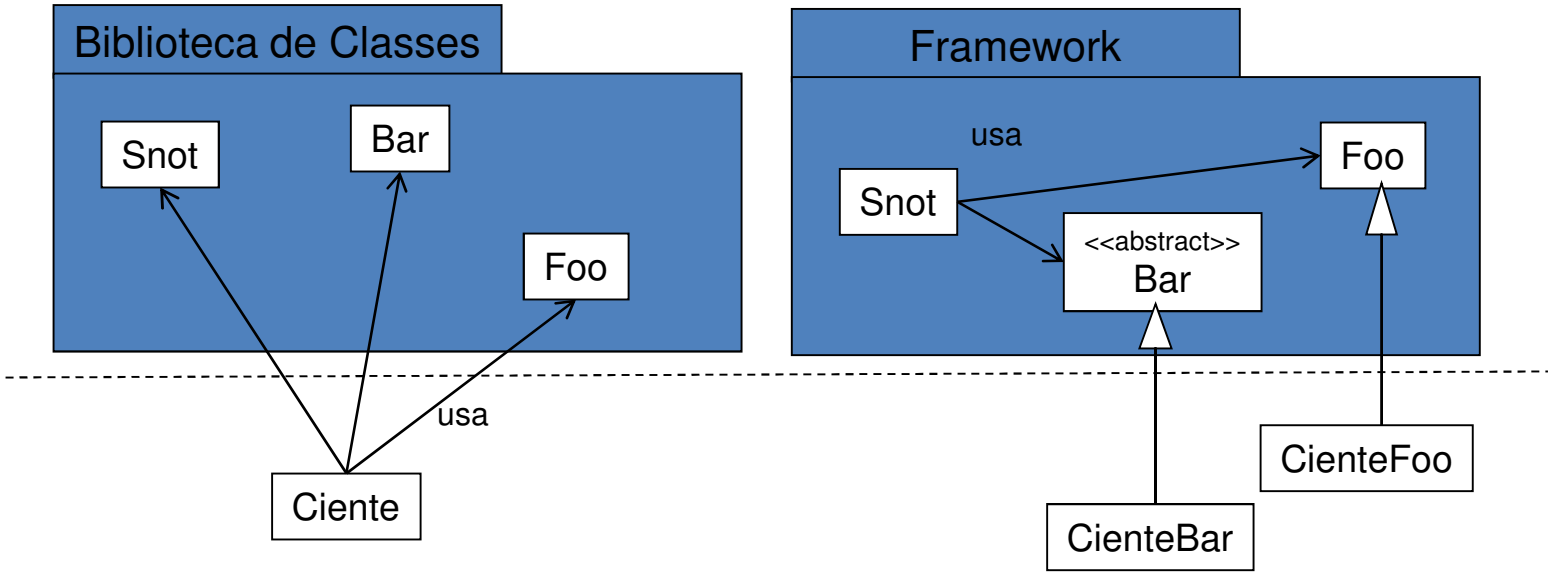
- Reusável
  - Propósito final
  - Para ser reusável, deve primeiro ser usável
    - Bem documentado
    - Fácil de usar
- Extensível
  - Framework contém funcionalidade abstrata (sem implementação) que deve ser completada
- Seguro
  - Desenvolvedor de aplicações não pode destruir o framework
- Eficiente
  - Devido a seu uso em muitas situações, algumas das quais poderão necessitar de eficiência

Características

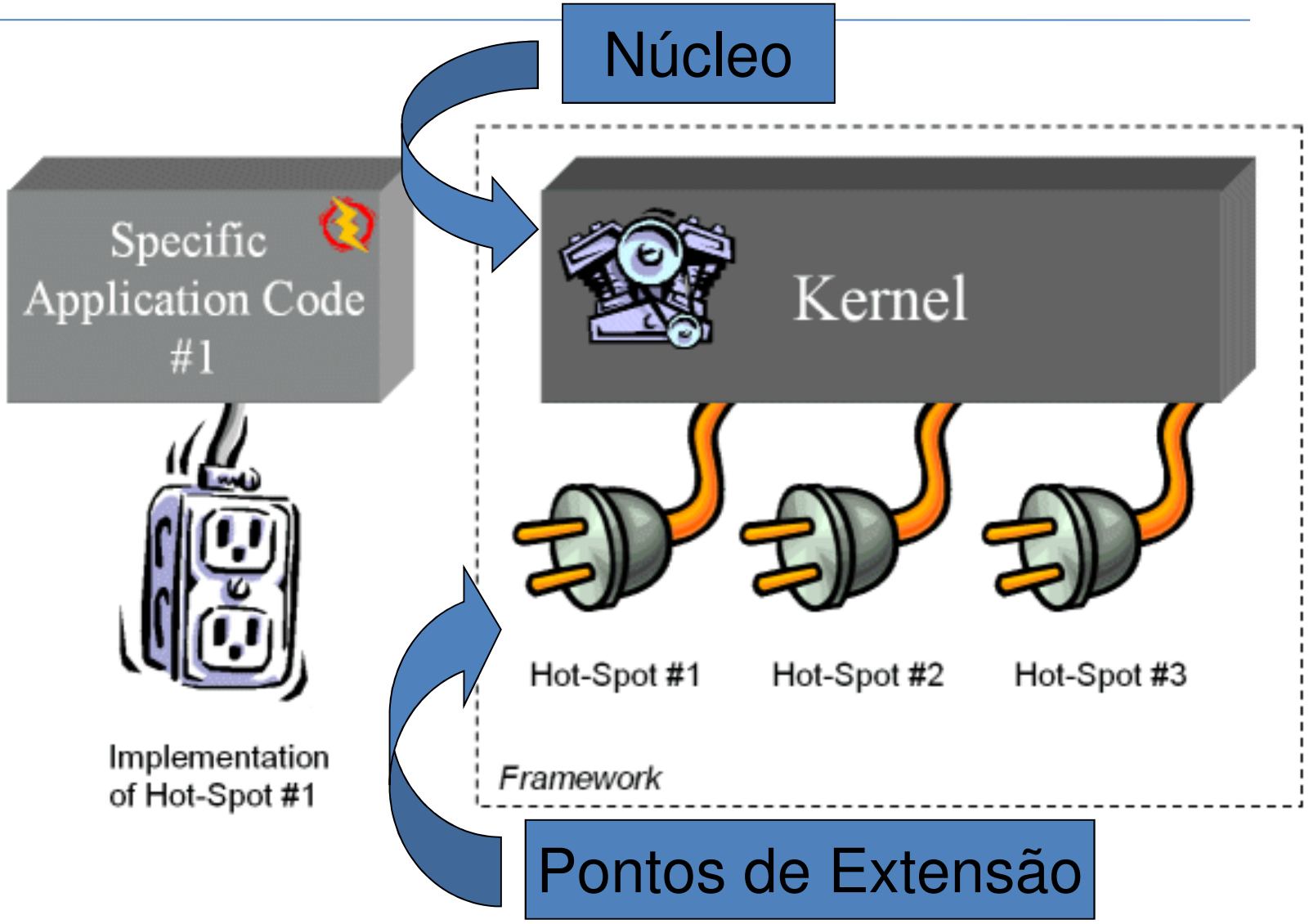
- Completo
  - Para endereçar o domínio do problema pretendido
- Inversion-of-Control (IoC)
  - Quem é responsável por chamar os objetos de uma instância do framework?
  - Princípio de Hollywood: “Don’t call us, we call you”
    - Framework define o controle de fluxo
      - Inversão do controle de fluxo, comparativamente às bibliotecas de classes
    - Framework comanda a resposta do sistema aos eventos externos
      - Invocando operações definidas pelo programador

Características

- Inversion-of-Control (IoC)



Propriedades

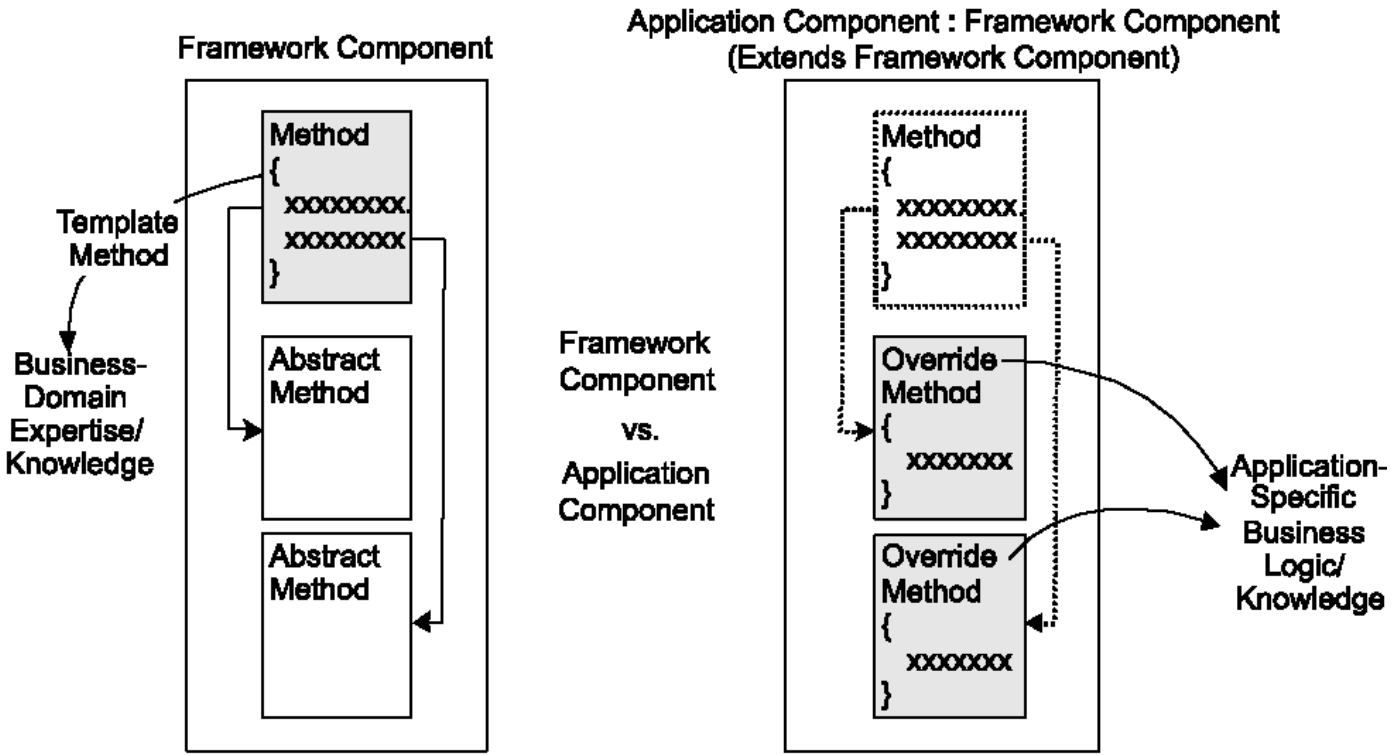


Propriedades

- Hot Spots
  - Pontos de Flexibilização
  - Partes passíveis de customização e extensão
  - Expressam aspectos do domínio do framework que não podem ser antecipados
  - Descobertos na análise do domínio
    - Posteriormente instanciados por especialistas no domínio da aplicação
- Frozen Spots
  - Partes fixas (núcleo)
  - Partes de código já implementadas
    - Chamam um ou mais hot spots definidos em cada instância

Técnicas de Customização

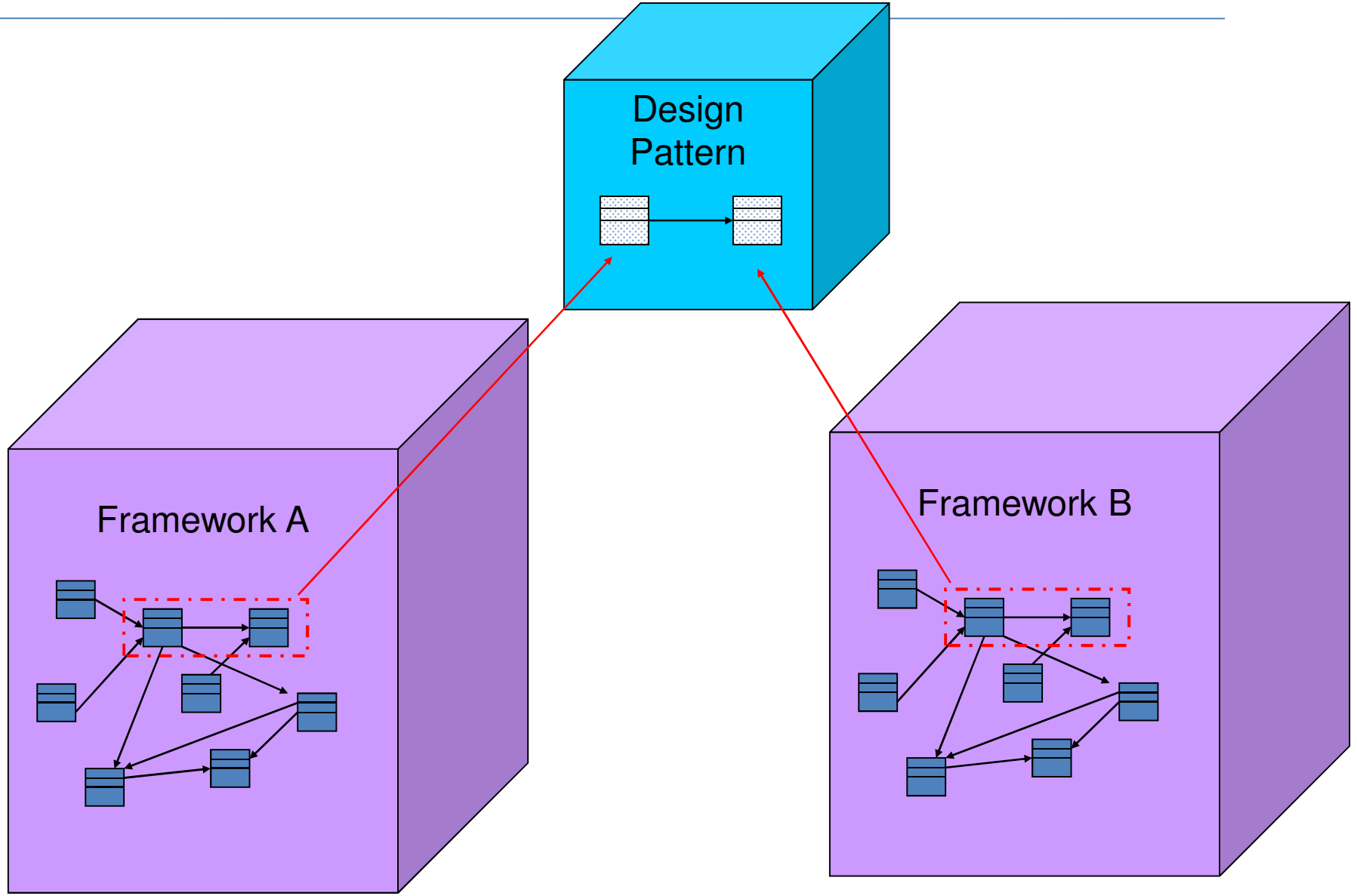
- Extensão de classes abstratas
- Implementação de interfaces
- Configuração/Composição de classes existentes
- Parametrização



Frameworks X Design Patterns

- Ambos consistem de classes, interfaces e colaborações prontas
- Diferenças
  - Design patterns mais abstratos do que frameworks
    - Framework inclui código, design pattern não (apenas exemplo do uso de pattern)
    - Devido à presença de código, framework pode ser estudado a nível de código, executado, e reusado diretamente
  - Design patterns elementos arquiteturais menores do que frameworks
    - Framework típico contém vários design patterns mas o contrário nunca ocorre
    - Exemplo: Design patterns são frequentemente usados para documentar frameworks
  - Design patterns menos especializados do que frameworks
    - Frameworks sempre têm um domínio de aplicação particular enquanto design patterns não ditam uma arquitetura de aplicação particular

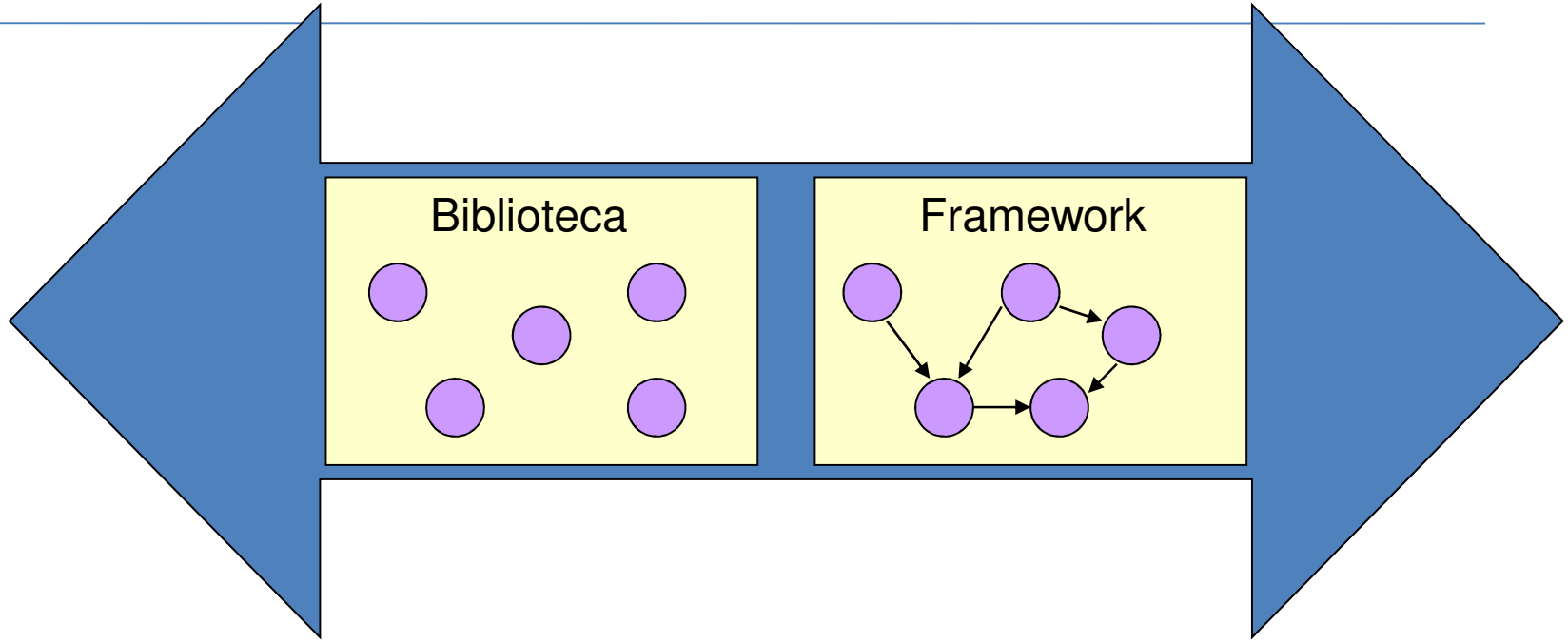
Frameworks X Design Patterns



Frameworks X Biblioteca de Classes

- Bibliotecas
  - Conjunto de classes relacionadas
    - Funcionalidades de propósito geral
  - Classes não relacionadas a um domínio de aplicação específica
    - Em contrapartida às classes de um framework
- Diferença
  - Grau de reutilização
  - Impacto na arquitetura da aplicação
- Classe de uma biblioteca
  - Reutilizada sozinha
- Classe de um framework
  - Reutilizada juntamente com as outras em uma instanciação

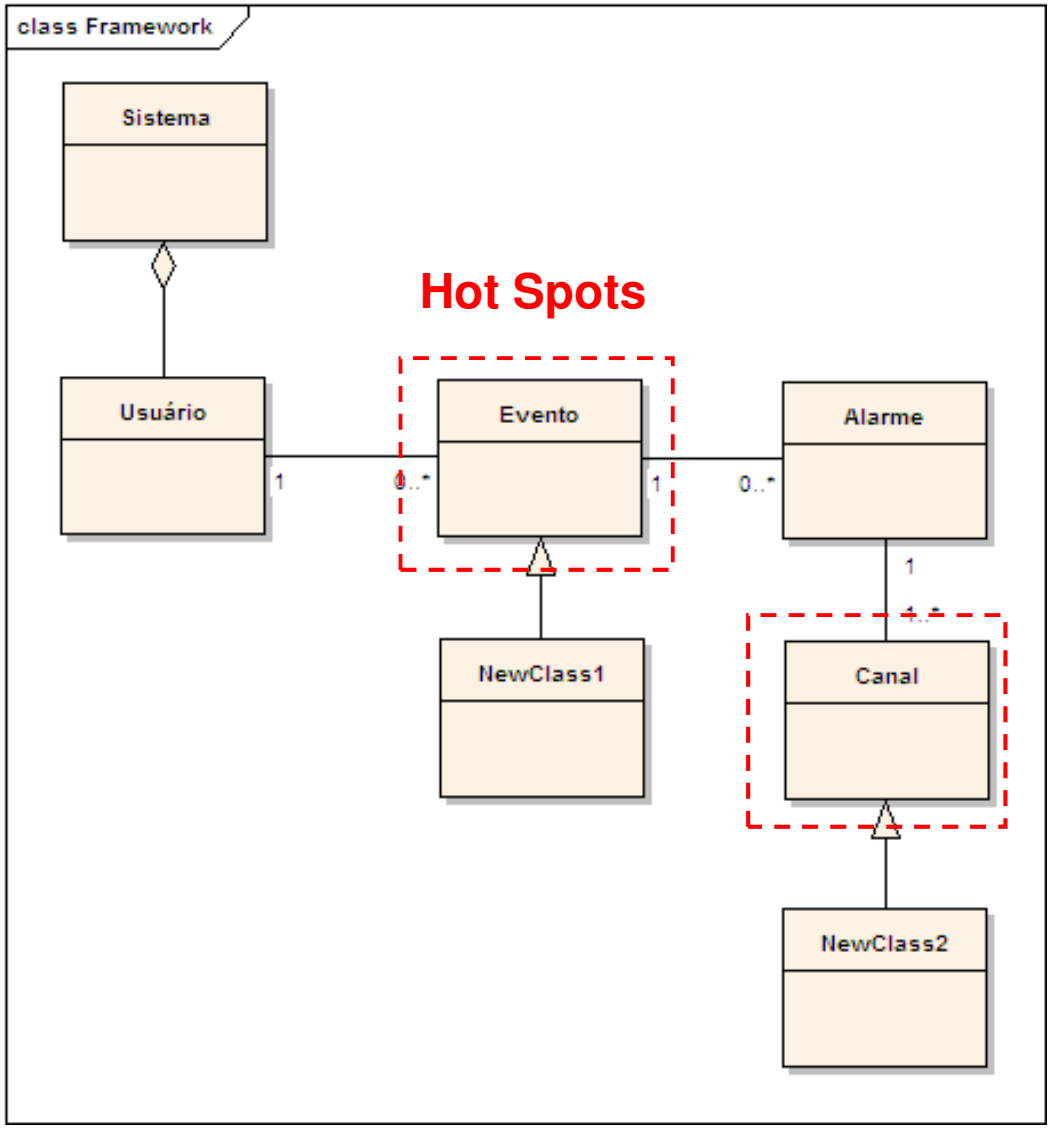
Frameworks X Biblioteca de Classes



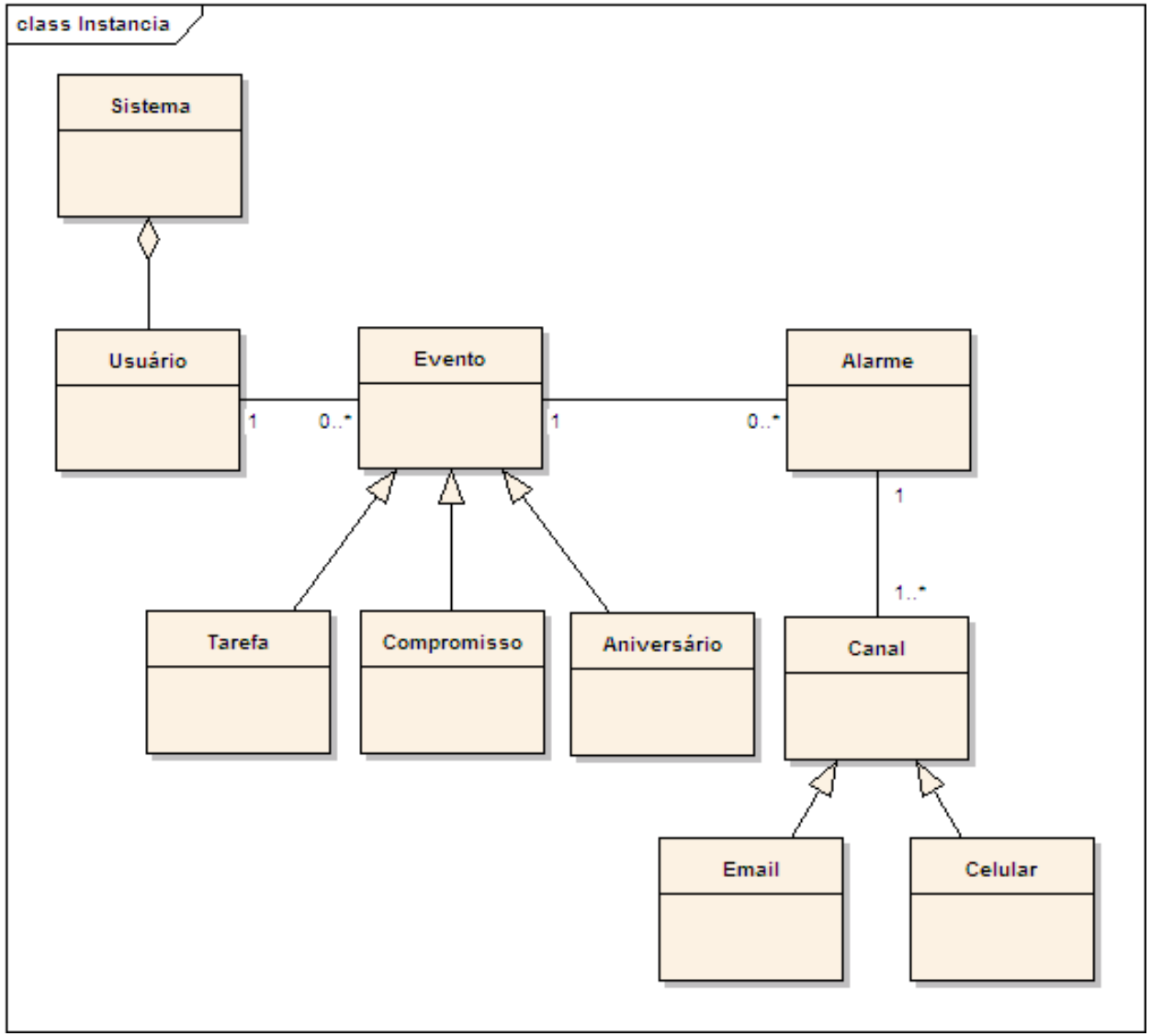
- Classes instanciadas pelo cliente
- Cliente chama funções
- Não tem fluxo de controle pré-definido
- Não tem interação pré-definida
- Não tem comportamento default

- Customização com subclasse ou composição
- Chama funções da “aplicação”
- Controla o fluxo de execução
- Define interação entre objetos
- Provê comportamento default

Exemplo - Agenda



Exemplo - Agenda



Bibliografia

- Jacques S. **Projeto de Software Orientado a Objeto**. http://www.dsc.ufcg.edu.br/~jacques/cursos/map/html/frame/oque.htm
- Markiewicz M., Lucena C. **Object Oriented Framework Development**. http://www.acm.org/crossroads/xrds7-4/frameworks.html
- Sommerville, I. **Software Engineering**. 7th edition, Chapter 18, 2000.