Padrões de Projeto

Prof. Adilson Vahldick

Departamento de Engenharia de Software

Udesc Ibirama

Objetivos da aula

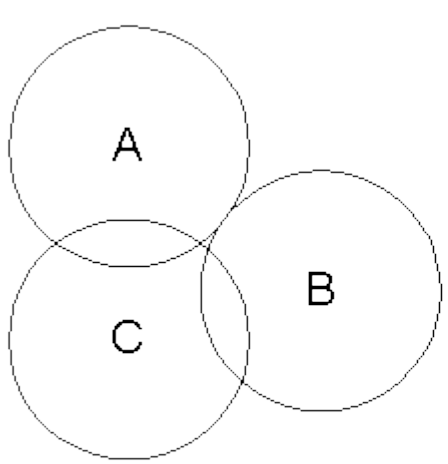
- Conhecer e entender o que é um framework
- Entender o processo de desenvolvimento de um framework

Framework

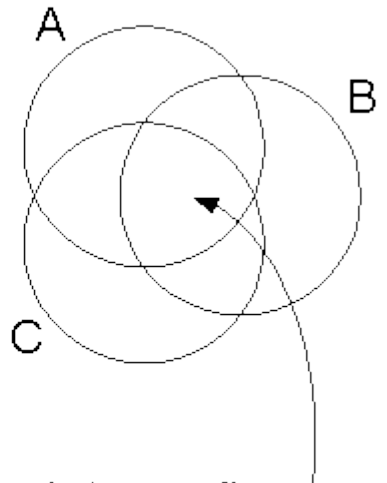
- Motivação
  - Temos problemas que são comuns em quase todas as aplicações de nossa empresa
  - Temos funcionalidades que são comuns em quase todas as aplicações de nossa empresa
- Solução
  - Desenvolver software reusável

Framework

- Captura a funcionalidade comum a várias aplicações
- Um framework é um conjunto de classes se cooperando para realizar um projeto reusável para uma categoria específica de software (Gamma et al)



Impossível criar Framework



Interseção grande
Possível criar Framework

Framework

- Características principais
  1. Solução para uma família de problemas
  2. Conjunto de classes e interfaces que decompõe esse problema
  3. Objetos colaboram para cumprir suas responsabilidades
  4. Flexível e extensível para permitir a construção de várias aplicações com pouco esforço

Framework

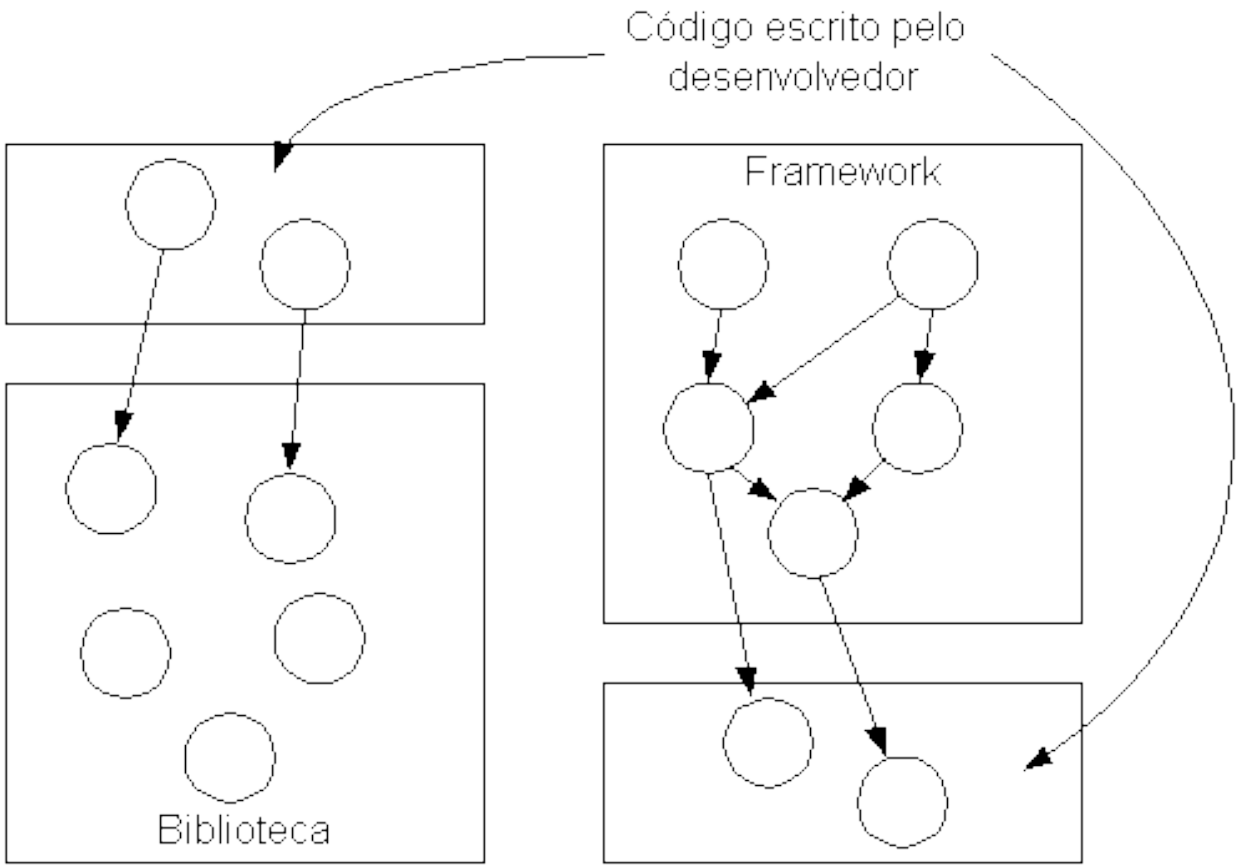
- É uma aplicação quase completa, com pedaços faltando

- Ao receber um framework, o nosso trabalho consiste em juntar os pedaços para nossa aplicação

Framework

- Diferenças entre framework, biblioteca, API e componente
  - API: é uma especificação, ou conjunto de regras, de como realizar uma tarefa.
  - Biblioteca: é a implementação da API, composta por classes não extensíveis. Todas as classes estão prontas para serem usadas.
  - Framework: pode ser um conjunto de bibliotecas, com pontos que as unem, mas essencialmente diz respeito a capacidade de extensibilidade das classes
  - Componente: é um artefato de software que tem um conjunto de entradas e saídas bem definidas que pode ser acoplado às aplicações

Framework



Framework

- Deve ser usável ➔ bem documentado
- Deve ser simples ➔fácil de aprender
- Dever ser reusável ➔ funcionalidade abstrata que deve ser completada
- Deve ser seguro ➔ impossibilitar o desenvolvedor de destruir o framework

Framework

- Classificação quanto **ao como usar**

- Foco na herança
  - Modifica funcionalidades de classes com override de métodos

- Foco na composição
  - Usam-se as interfaces fornecidas e adicionam-se os objetos (composite)

- Híbrido
  - A maioria é focada na herança com alguma funcionalidade pronta

Framework

- Classificação quanto **ao onde usar**

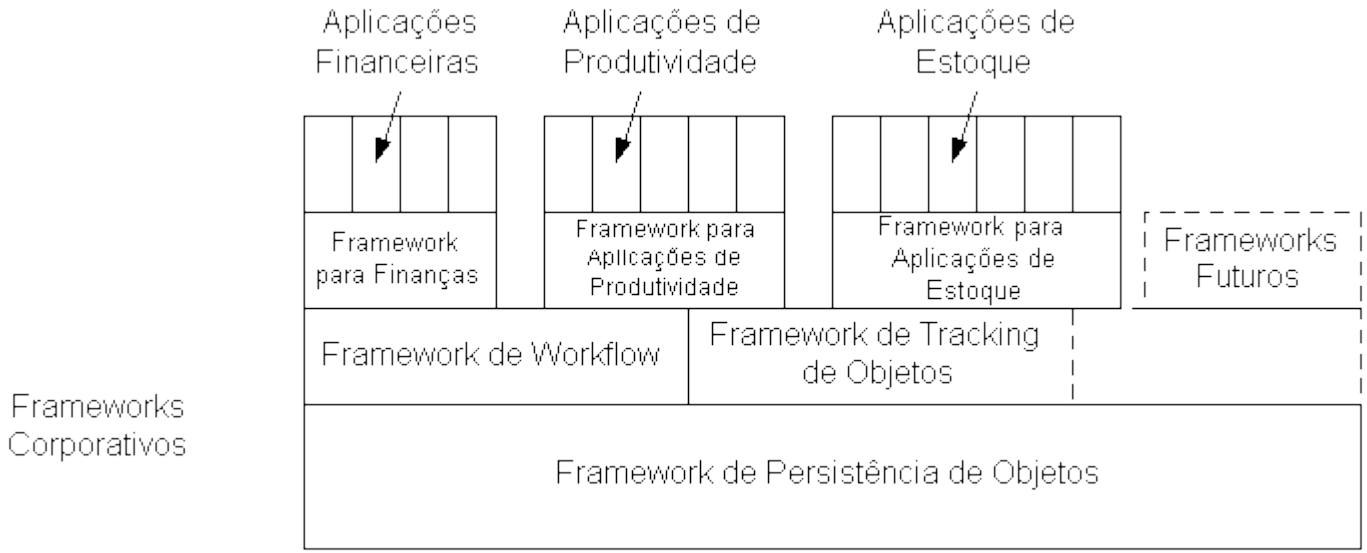
- Suporte
  - Serviços de sistemas operacionais: acessos a arquivos, computação distribuída, etc.

- Aplicação
  - Resolve uma fatia do problema da aplicação: GUI, JUnit, fluxo de aplicações web, etc.

- Domínio
  - Resolve boa parte da aplicação: controle de manufatura, gestão de projetos, etc.

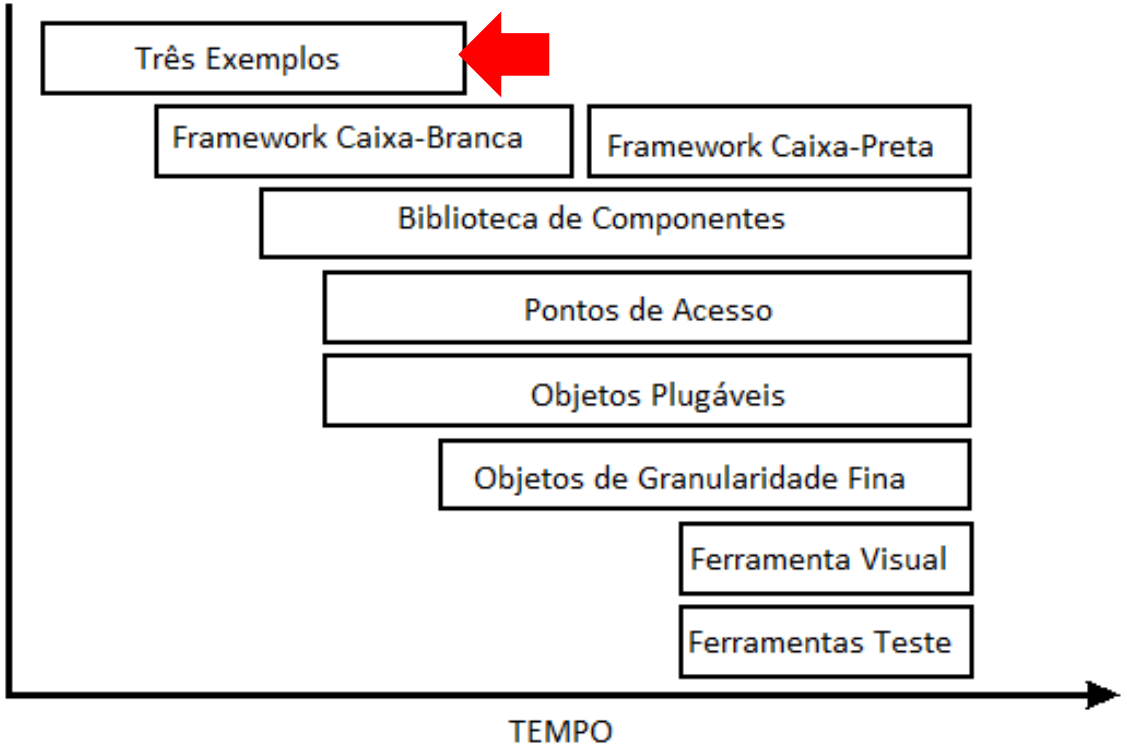
Framework

- Estrutura



Framework

- Padrões para Desenvolvimento/Evolução de um Framework



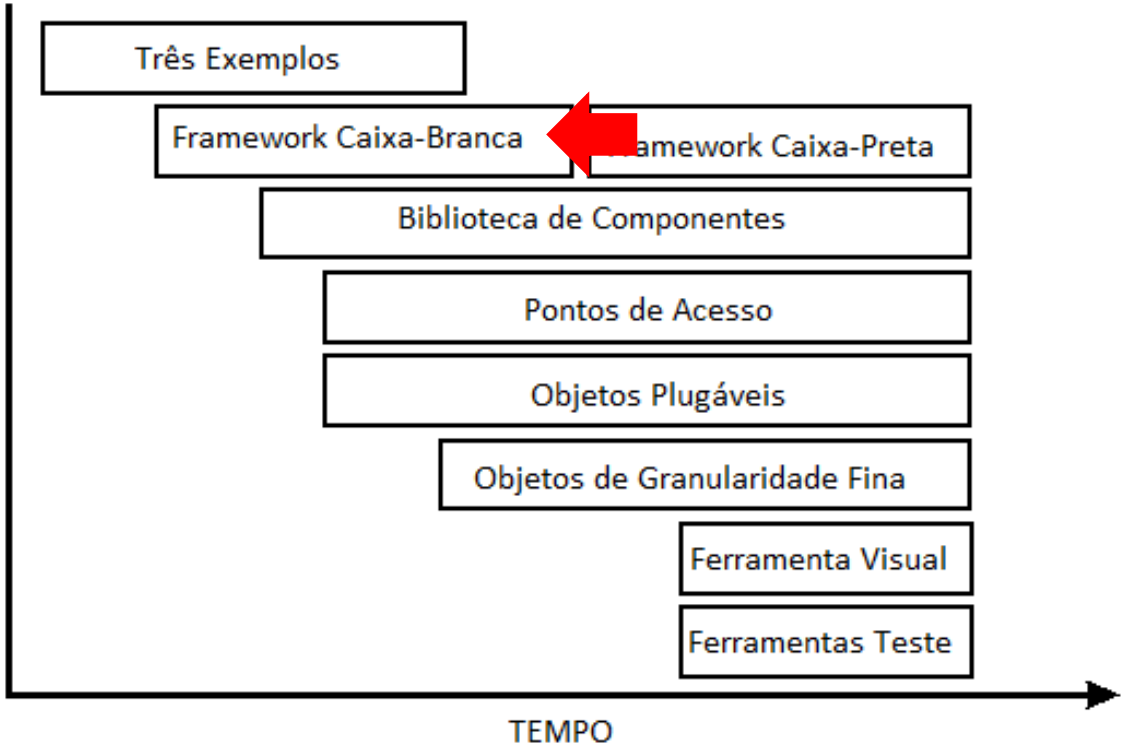
Framework

- **<u>Três Exemplos</u>: vai precisar desenvolver um framework para um problema em particular**

- **Problema:** como iniciar o desenvolvimento de um framework?
- **Solução:**
  - Implementar três aplicações em que o framework deve ajudar no desenvolvimento
- **Implementação:**
  - Podem ser protótipos, ou até certo grau de funcionalidade. Identificar os códigos em comum que podem ser separados.

Framework

- Padrões para Desenvolvimento/Evolução de um Framework



Framework

- **<u>Framework Caixa-Branca</u>: preciso construir minha segunda aplicação**

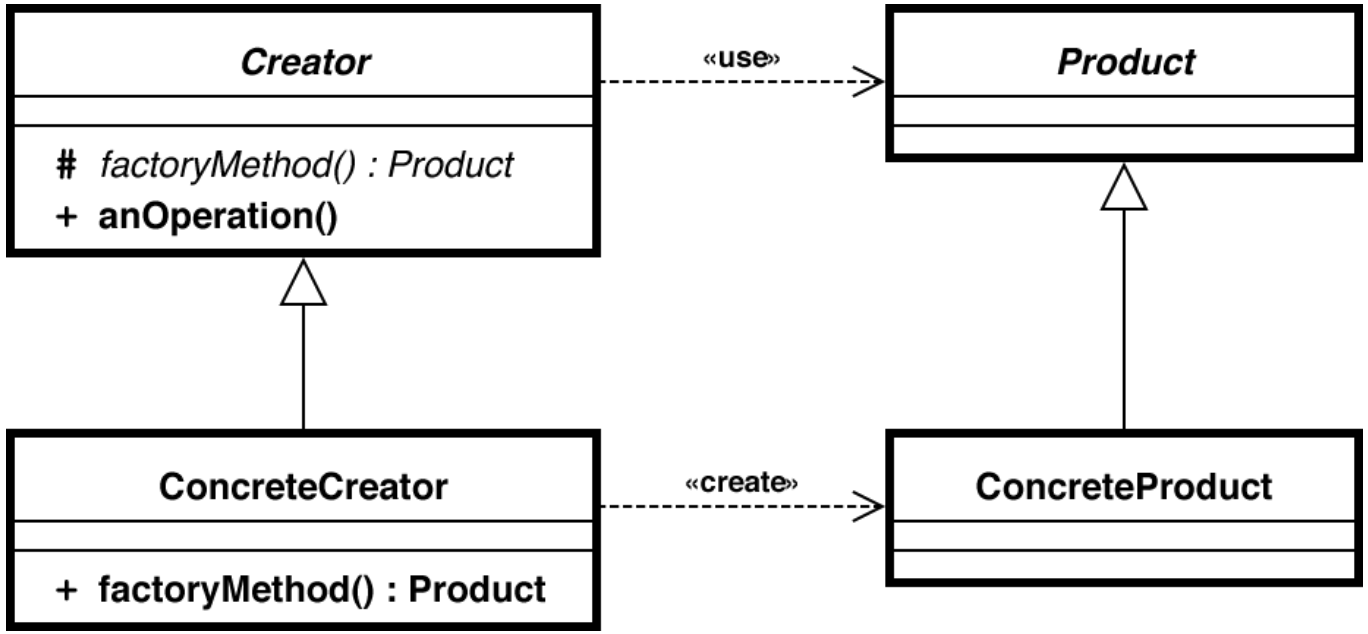
- **Problema:** Alguns frameworks recaem sobre o uso de herança e outros de composição. Qual devo usar?

- **Solução:**
  - Use herança. Construa um framework de caixa-branca pela generalização das classes a partir das aplicações de exemplo. Use os padrões Template Method e Factory Method.

- **Implementação:**
  - Cada nova aplicação se baseia em especializar classes do framework existente. Com o passar do tempo, serão identificadas coisas em comum nessas subclasses, e então criar uma classe abstrata com esse código em comum.

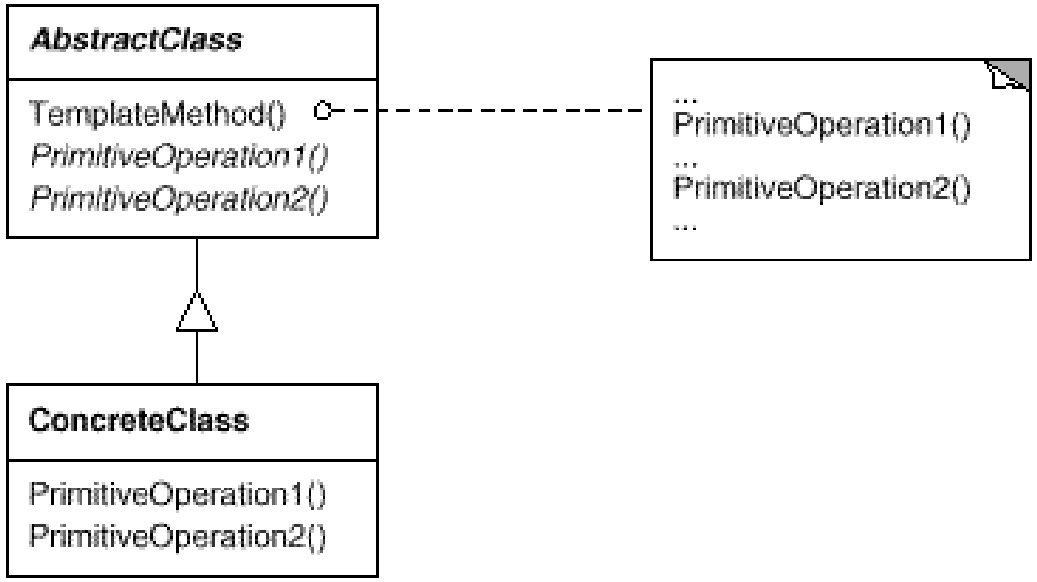
GoF

- Factory Method
  - Definir uma interface para criar um objeto, mas deixar as subclasses decidirem que classe instanciar



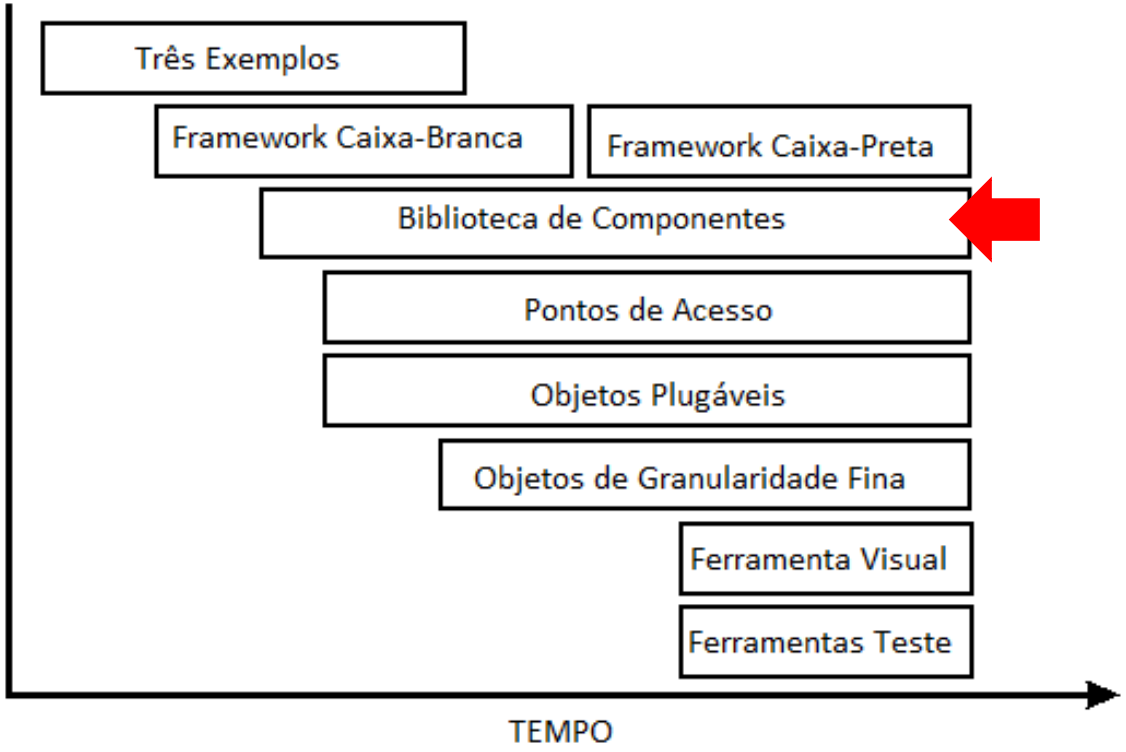
GoF

- Template Method
  - Definir o esqueleto de um algoritmo em uma operação, postergando alguns passos para as subclasses.



Framework

- Padrões para Desenvolvimento/Evolução de um Framework



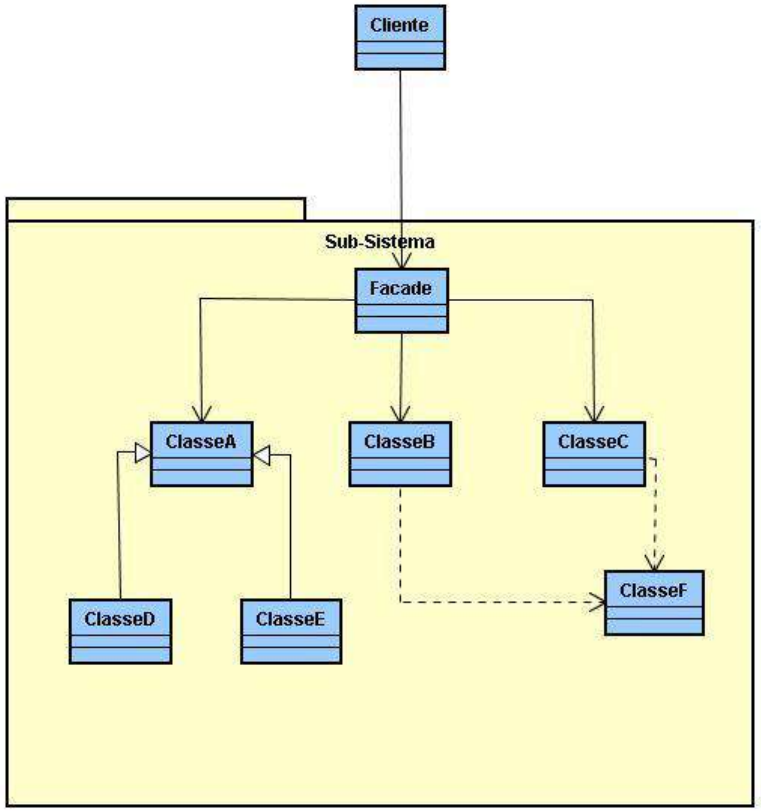
Framework

- **Biblioteca de componentes: desenvolvendo as aplicações com framework caixa-branca**

- **Problema:** como evitar de escrever objetos similares a cada vez que se instancia um framework?
- **Solução:**
  - Comece com uma biblioteca simples de objetos óbvios e adicione objetos à medida que vai precisando deles.
- **Implementação:**
  - Existem algumas classes concretas do framework são sempre usadas em cada aplicação. Usar o padrão Façade ou Mediator.

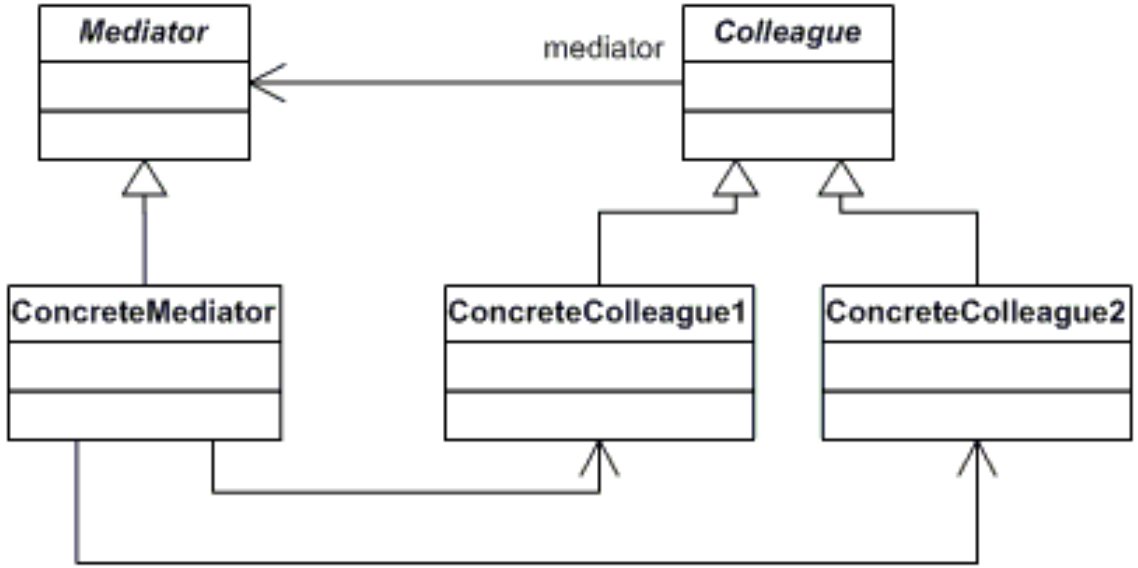
GoF

- Façade
  - Fornecer uma interface unificada para um conjunto de interfaces em um subsistema



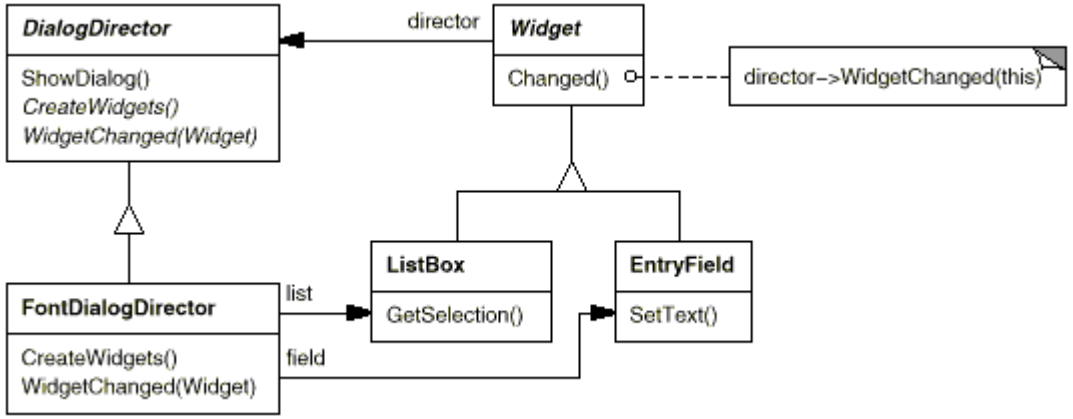
GoF

- Mediator
  - Definir um objeto que encapsula a forma como um conjunto de objetos interage.



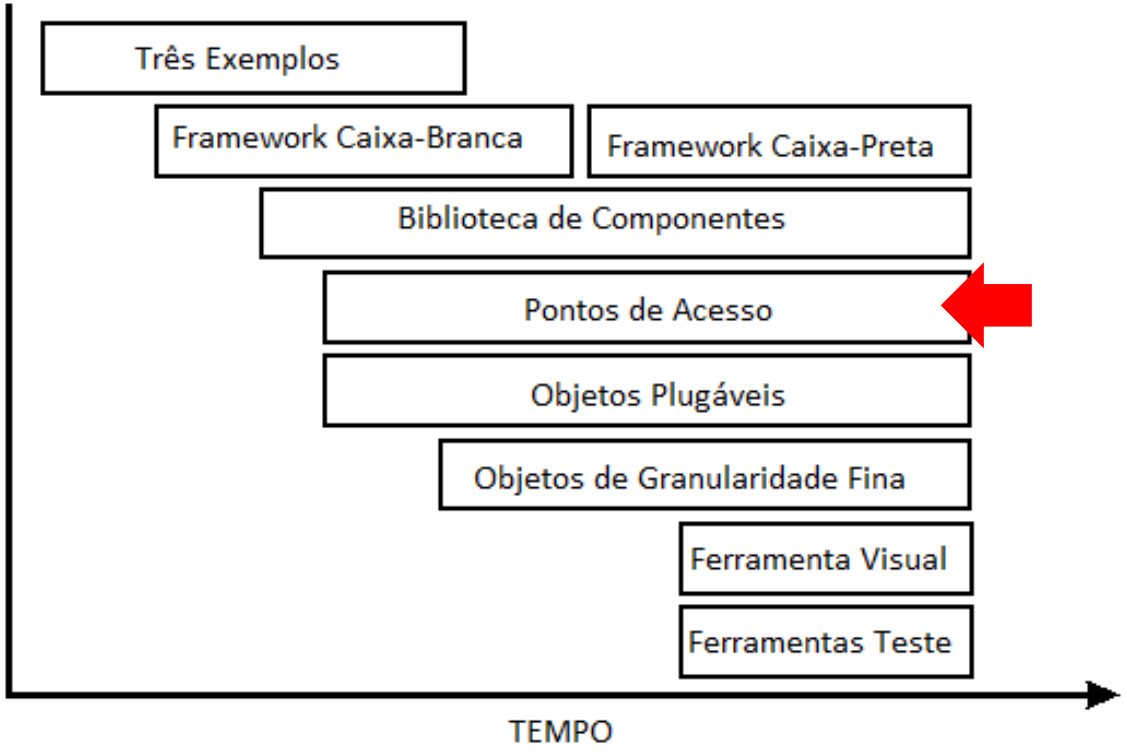
GoF

- Mediator
  - Definir um objeto que encapsula a forma como um conjunto de objetos interage.



Framework

- Padrões para Desenvolvimento/Evolução de um Framework

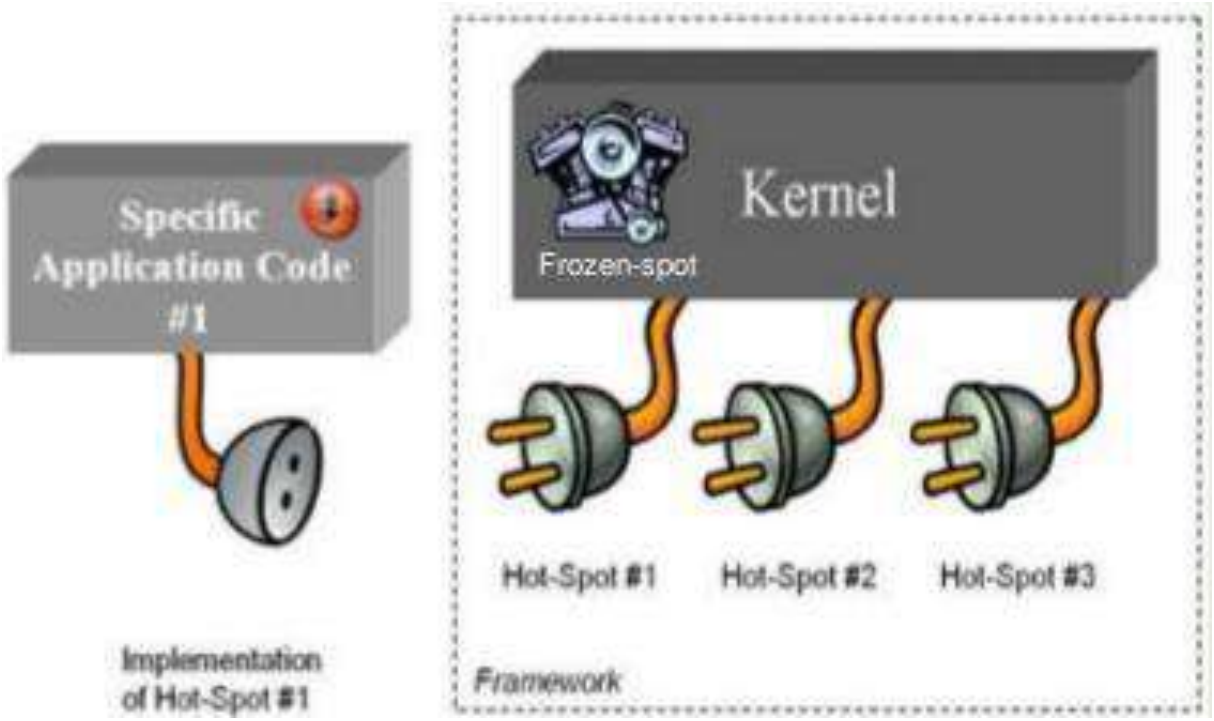


Framework

- **Pontos de acesso: estamos adicionando componentes em nossa biblioteca.**

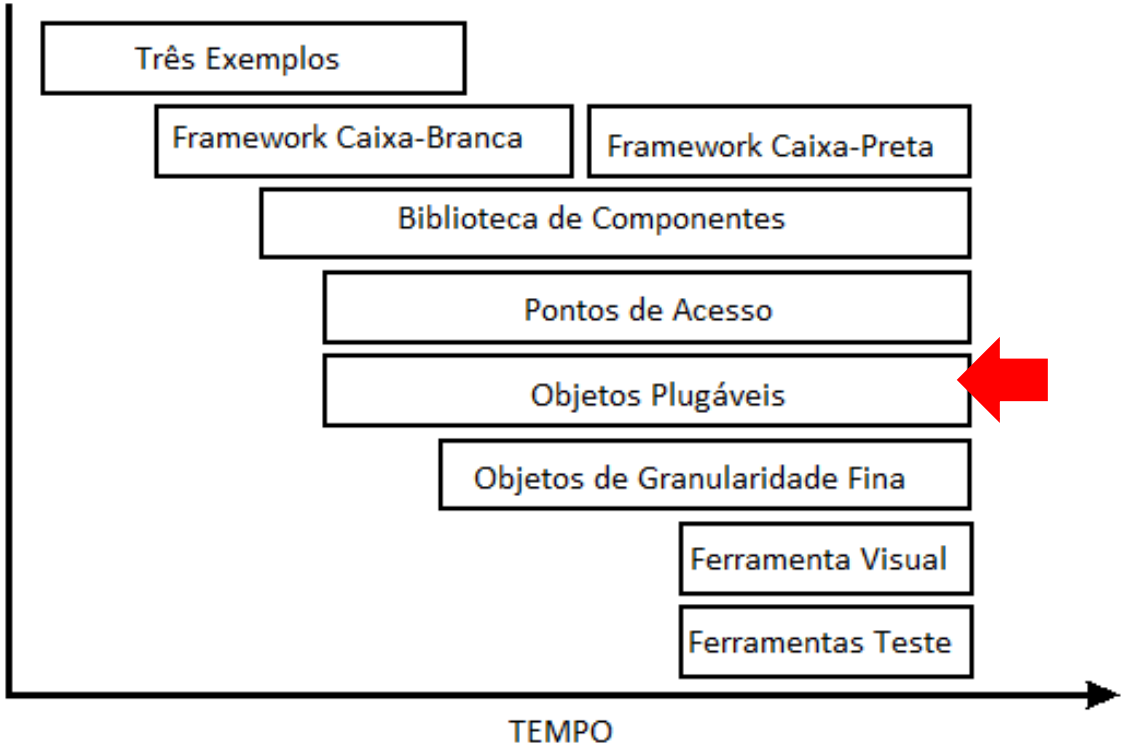
- **Problema:** como eliminar código comum encontrado em cada aplicação ?
- **Solução:**
  - Separe o código que muda daquele que não muda. Encapsule o código que muda para criar objetos.
- **Implementação:**
  - Command (encapsular ações), Observer (avisar quando as coisas mudam), Abstract Factory (criar tipos de objetos)

Framework


Specific
Application Code
#1
Implementation
of Hot-Spot #1
Kernel
Frozen-spot
Hot-Spot #1
Hot-Spot #2
Hot-Spot #3
Framework

Framework

- Padrões para Desenvolvimento/Evolução de um Framework



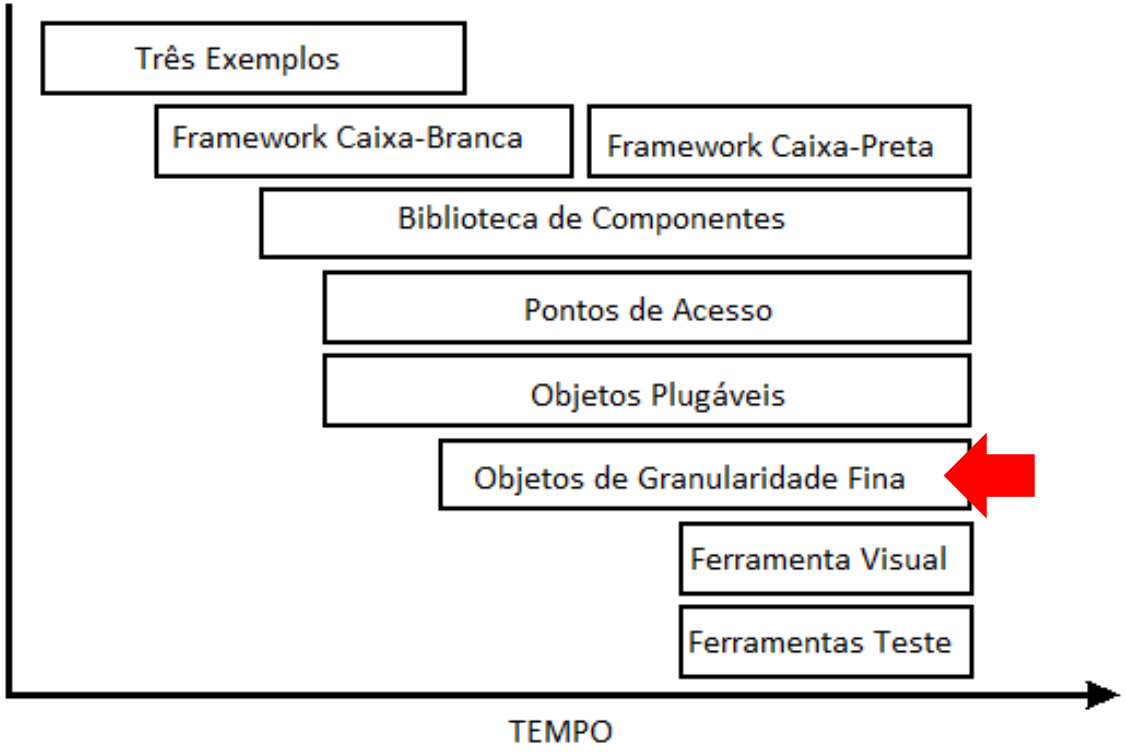
Framework

- **Objetos Plugáveis: estamos adicionando componentes em nossa biblioteca.**

- **Problema:** muitas das subclasses escritas diferem em um ponto trivial (um método). Como evitar de criar subclasses triviais toda vez que precisamos usar o framework?
- **Solução:**
  - Projete classes adaptáveis que possam ser parametrizáveis
- **Implementação:**
  - Determine o que varia em cada subclasse. Se é o valor de um variável ou a classe a ser referenciada, crie um atributo para suprir. Se é um pedaço de código, passe como parâmetro de um método.

Framework

- Padrões para Desenvolvimento/Evolução de um Framework



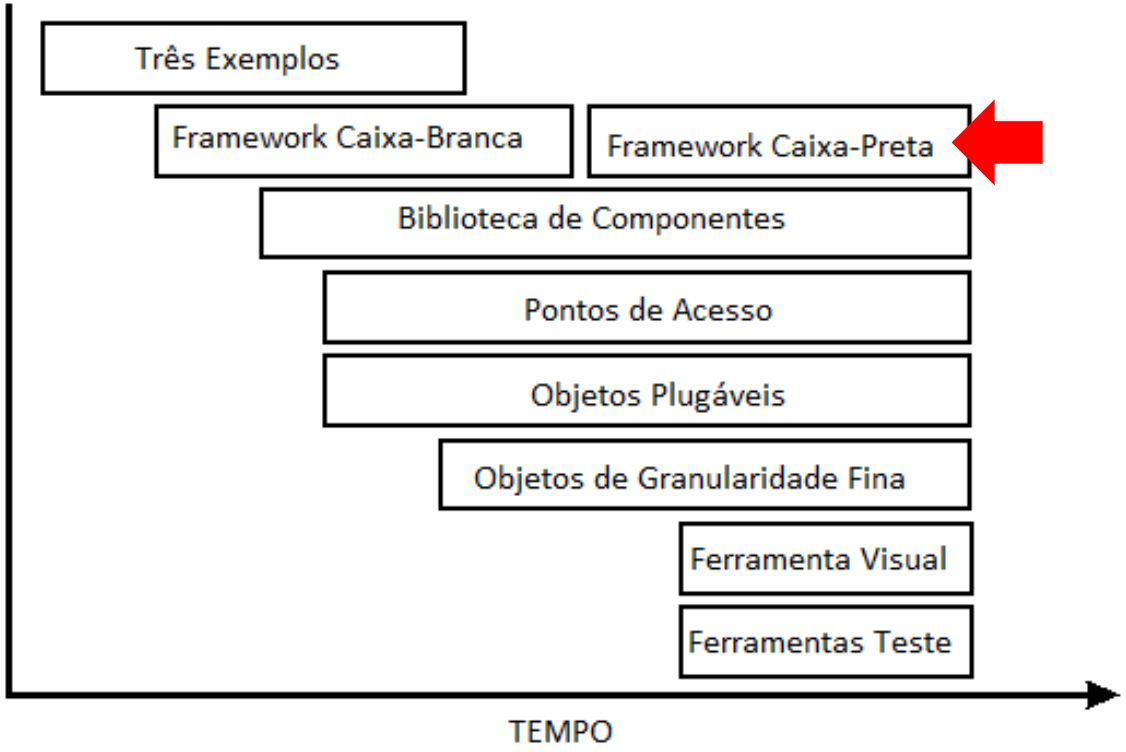
Framework

- **Objetos de granularidade fina: estamos refatorando a biblioteca para torna-la reusável**

- **Problema:** até quão pequenas podem ser as classes ?
- **Solução:**
  - Cada classe deve ter uma única responsabilidade. Repetimos o processo de dividir a classe se ainda conseguirmos identificar uma responsabilidade para cada nova classe resultado da divisão
- **Implementação:**
  - Todas as classes em nossa biblioteca que acumulam múltiplas responsabilidades em que cada comportamento possa variar, deve se transformar em uma nova classe.

Framework

- Padrões para Desenvolvimento/Evolução de um Framework



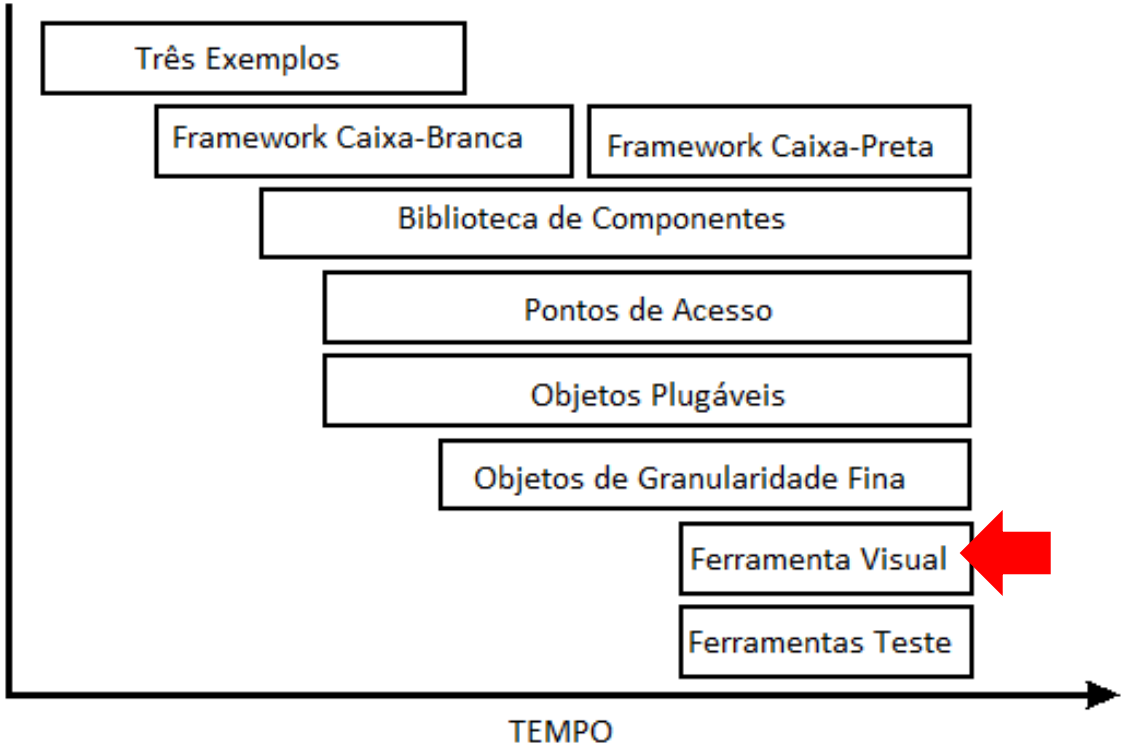
Framework

- **Framework de caixa-preta: estamos desenvolvendo objetos plugáveis encapsulando os pontos de acesso e fazendo objetos de granularidade fina**

- **Problema:** Alguns frameworks recaem sobre o uso de herança e outros de composição. Qual devo usar?
- **Solução:**
  - Usar herança para organizar a biblioteca e composição para combinar essas classes para novas aplicações
- **Implementação:**
  - Prime pela conversão das heranças em composições. Separe o código comum que estão em classes não relacionadas e encapsule para novos componentes.

Framework

- Padrões para Desenvolvimento/Evolução de um Framework



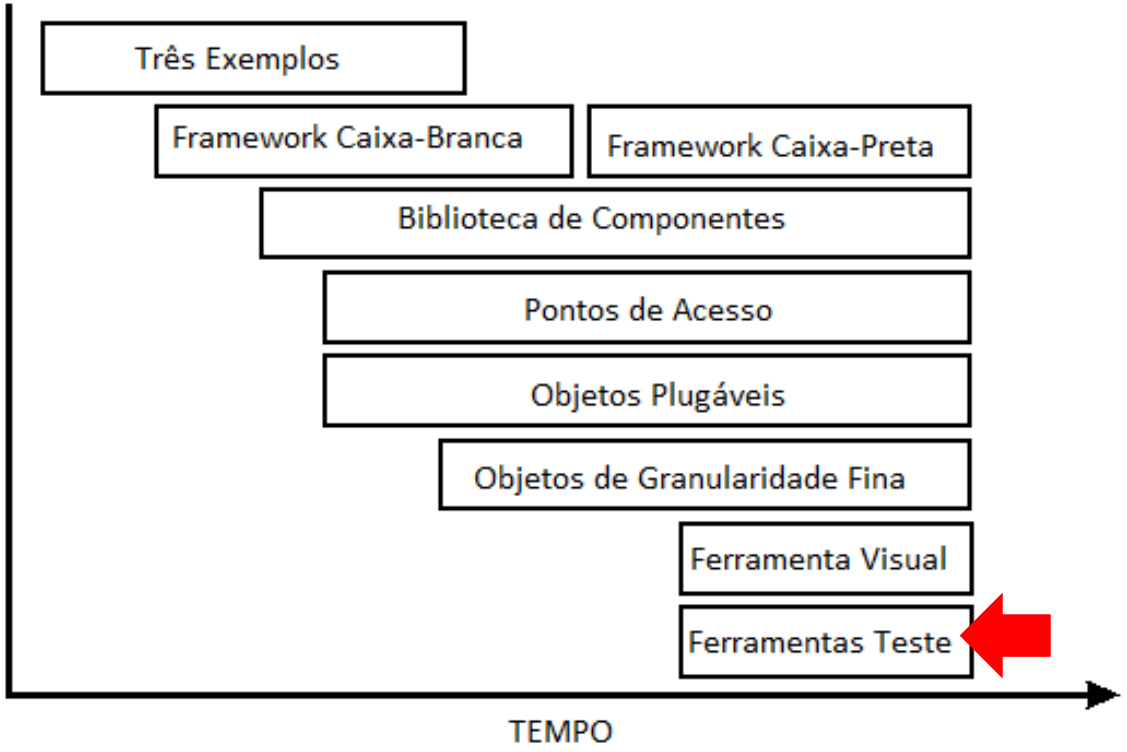
Framework

- **<u>Ferramenta visual</u>: com o framework de caixa-preta desenvolvido queremos prover ferramentas que facilitem a criação de novas aplicações**

- **Problema:** Como simplificar a criação do código para usar o framework ?
- **Solução:**
  - Criar uma ferramenta gráfica (editor) para definir a especificação e gerar código
- **Implementação:**
  - Usar caixas de diálogo e diagramas para definir a relação e o comportamento dos objetos

Framework

- Padrões para Desenvolvimento/Evolução de um Framework



Framework

- **Ferramentas de teste: criamos a ferramenta visual**

- **Problema:** Como facilitar a inspeção e depuração do comportamento dos objetos no nosso framework?
- **Solução:**
  - Criar ferramentas especializadas de teste, simulação, inspeção e depuração
- **Implementação:**
  - Desenvolver ferramentas para acompanhar a evolução da execução em partes complexas do framework