Design Patterns

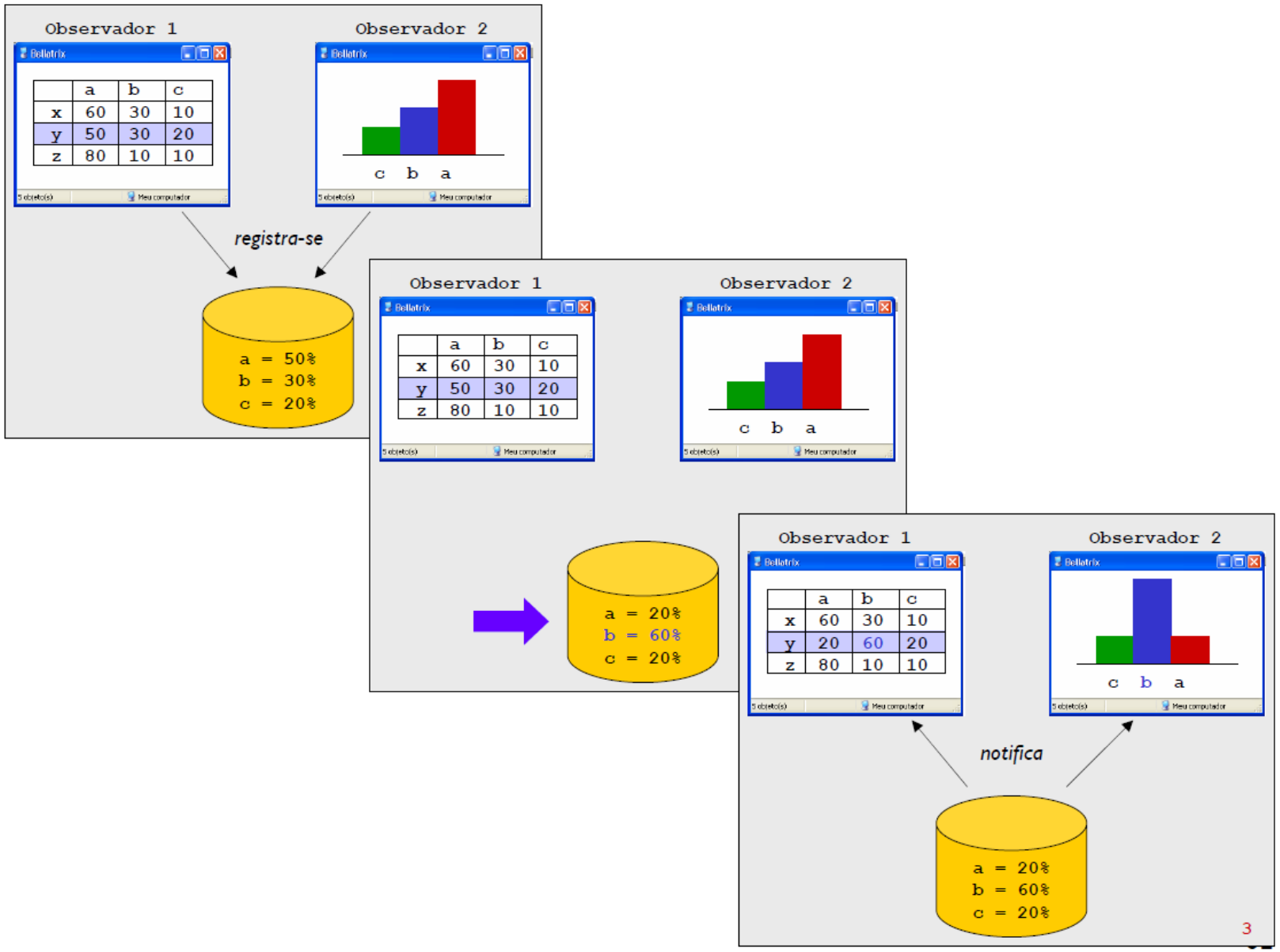
Classificação (GoF)

		Propósito		
		1. Criação	2. Estrutura	3. Comportamento
Escopo	Classe	Factory Method	Class Adapter	Interpreter Template Method
	Objeto	Abstract Factory Builder Prototype Singleton	Object Adapter Bridge Composite Decorator Facade Flyweight Proxy	Chain of Responsibility Command Iterator Mediator Memento Observer State Strategy Visitor

Observer

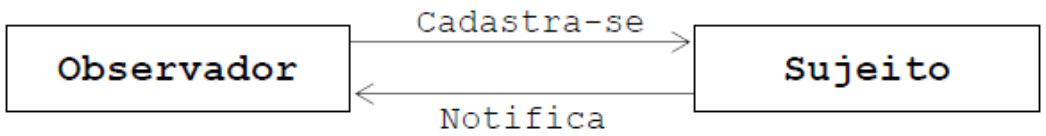
"Definir uma dependência um-para-muitos entre objetos para que quando um objeto mudar de estado, todos os seus dependentes sejam notificados e atualizados automaticamente." [GoF]

Problema

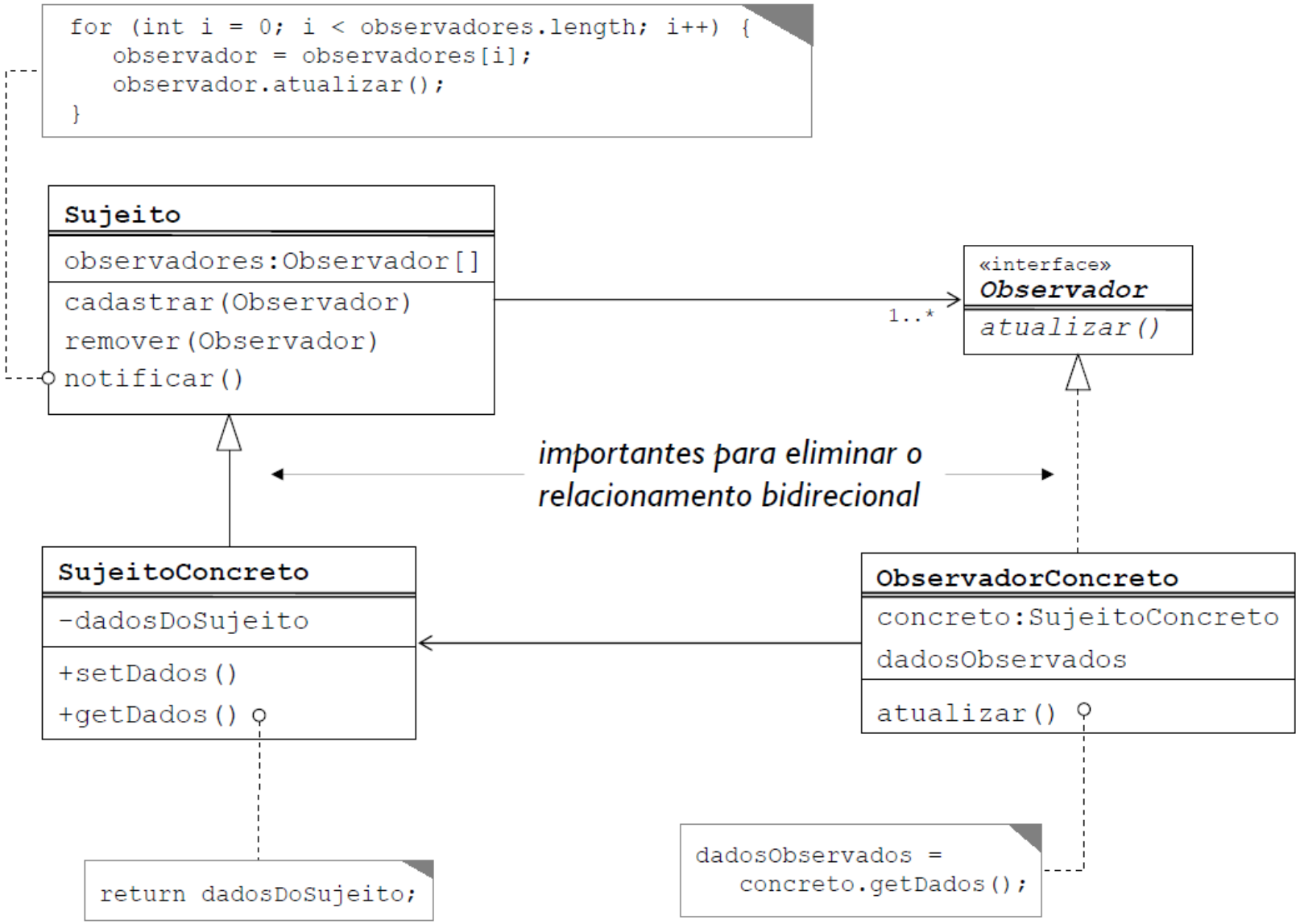

Observador 1
Observador 2
a b c
x 60 30 10
y 50 30 20
z 80 10 10
c b a
registra-se
a = 50%
b = 30%
c = 20%
a = 20%
b = 60%
c = 20%
notifica
3

Problema

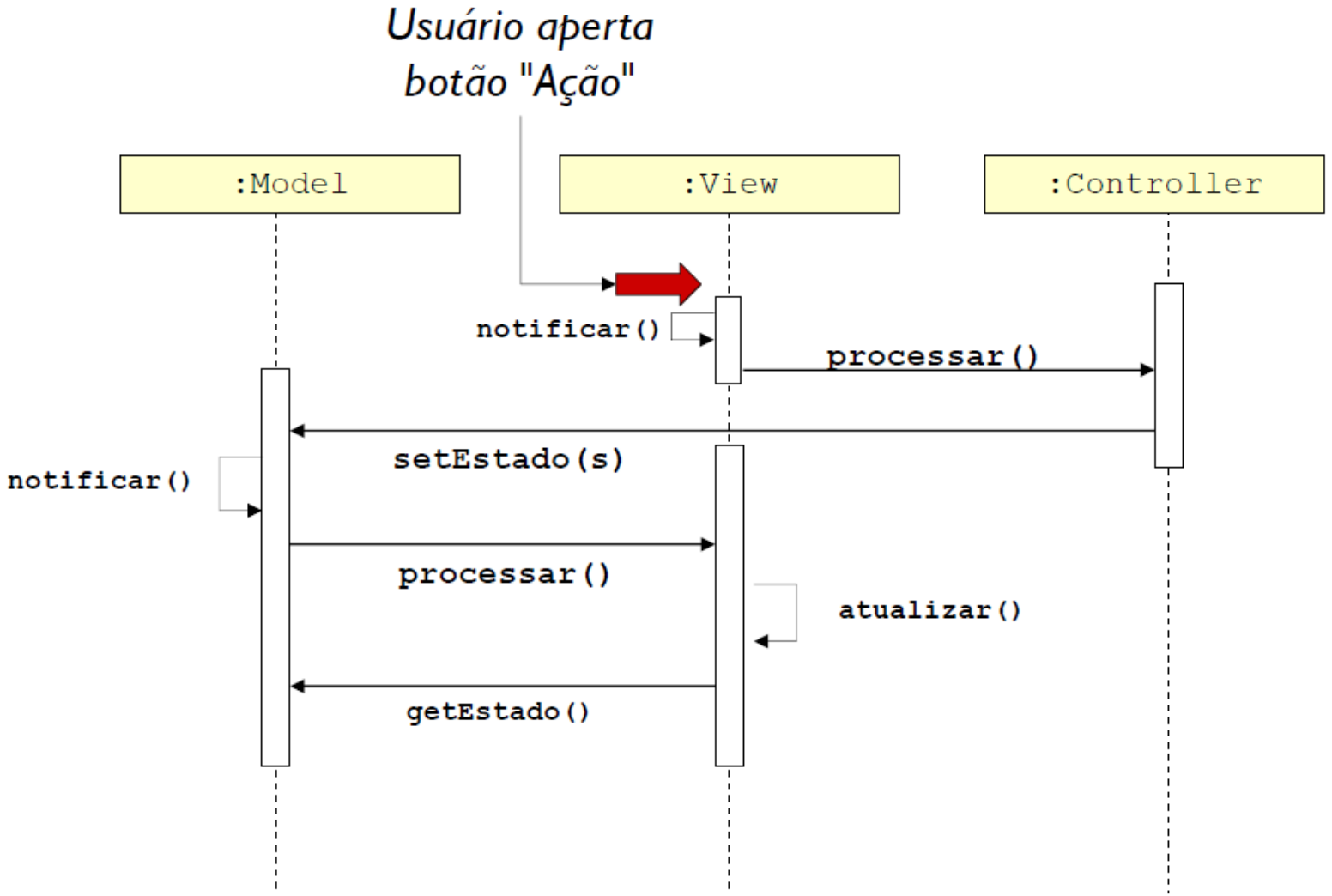
- *Como garantir que objetos que dependem de outro objeto fiquem em dia com mudanças naquele objeto?*
  - *Como fazer com que os observadores tomem conhecimento do objeto de interesse?*
  - *Como fazer com que o objeto de interesse atualize os observadores quando seu estado mudar?*
- *Possíveis riscos*
  - *Relacionamento (bidirecional) implica alto acoplamento. Como podemos eliminar o relacionamento bidirecional?*



Estrutura


for (int i = 0; i < observadores.length; i++) {
observador = observadores[i];
observador.atualizar();
}
Sujeito
observadores:Observador[]
cadastrar(Observador)
remover(Observador)
notificar()
1..*
«interface»
Observador
atualizar()
importantes para eliminar o
relacionamento bidirecional
SujeitoConcreto
-dadosDoSujeito
+setDados()
+getDados()
ObservadorConcreto
concreto:SujeitoConcreto
dadosObservados
atualizar()
return dadosDoSujeito;
dadosObservados =
concreto.getDados();

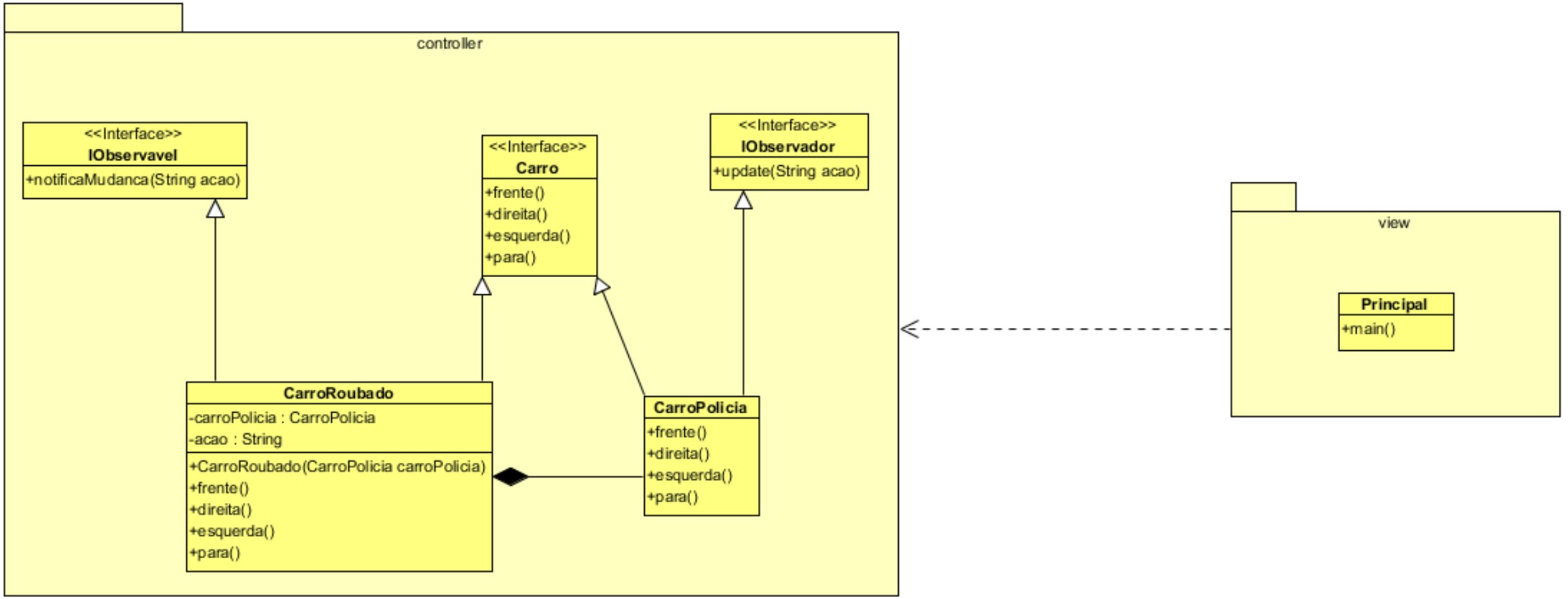
Sequência de Operação



Prática e aplicação em Java

- Considere um jogo que, quando um carro é roubado pelo jogador principal e entra num raio de visão de um policial, ele será perseguido. Estando nesse raio de visão, sempre que ele seguir em frente, a viatura seguirá em frente. Quando ele virar para a direita ou a esquerda, a viatura acompanhará o movimento. Caso ele pare, será preso pela viatura. Tratar em Java essa situação de perseguição.

Prática e aplicação em Java



Exercício

- Em uma rede de publicação de conteúdo, toda vez que você se inscreve (“Ativa o sininho”), você recebe uma mensagem a cada vez que um novo conteúdo é postado. Considerando a estrutura Observer, onde o inscrito é o observador e o criador de conteúdo é o observado, construir uma solução em Java que simule uma nova postagem de um criador de conteúdo e 3 inscritos, em um List de Inscritos.