05

Coesão

Transcrição

Nós escrevemos a funcionalidade de transferir dinheiro de uma conta para outra, fora da classe. Depois, por uma questão de organização, decidimos colocar a refatoração dentro da classe `Conta`, por ser um trecho relacionado a conta. No entanto, existem casos em que percebemos, na elaboração do código, que determinadas partes se encaixam em algumas classes específicas.

Lembrem-se que um código bom costuma ser melhorado ao longo da sua criação e a refatoração faz parte do dia a dia do programador. Assim como existem códigos que primeiro criamos em um lugar e incluímos em outra classe depois, o contrário existe também.

Por exemplo, se trabalhássemos com o método `eh_inadimplente()`, cuja responsabilidade é identificar se alguém é inadimplente, dando como retorno `true` (verdadeiro) e `false` (falso).

Passaremos como parâmetro `cliente`, porque quem é inadiplente é a pessoa e não a conta.

```
def eh_inadimplente(self, cliente):
```

Se neste método, não for utilizado os dados da conta, seria correto extrair a funcionalidade e movê-la para outro lugar, por exemplo, para a classe `cliente` que poderia ser criada. Sempre deveríamos verificar se o método está no local mais apropriado.

Idealmente, uma classe deve ter apenas uma responsabilidade. Se adicionássemos o método `eh_inadimplente`, `Conta` passaria a ter duas funções. E, provavelmente, começaríamos a trabalhar com primeiro e segundo nome do titular, ou talvez, precisaríamos especificar o número da agência. Ou seja, os dados seriam mais detalhados.

Neste caso, se desenvolvêssemos o método referente aos clientes inadimplentes, **faltaria** coesão na nossa classe, por ter mais responsabilidades do que deveria.

Em seguida, removeremos o método `eh_inadimplente()` que foi criada apenas como exemplo.