Princípio de Responsabilidade Única – SRP

Uma classe deveria ter apenas uma única razão para mudar, ou seja, a classe deve ter apenas um trabalho.

Qual é a necessidade da existência desse princípio?

Quando se cria uma classe com diversas funcionalidades, podemos acabar criando uma grande quantidade de dependências dentro da própria classe. Com isso caso seja necessário alterar alguma dessas funcionalidades futuramente, é possível que essa mudança tenha impacto em outras funcionalidades, o que faz ser necessário testar todas as outras responsabilidades da classe após a mudança.

Digamos que nós temos uma classe que é utilizada para que um usuário possa fazer alterações nas suas configurações, mas antes disso, ele deve se autenticar no sistema. Uma das formas de implementar essa classe, seria essa: (slide)

Porém, dessa forma, caso precisemos utilizar o checarAcesso() em algum outro lugar, caso precisemos fazer mudanças na forma que a checagem esteja sendo feita ou caso queiramos alterar a forma na qual os e-mails estão sendo aprovados, nos últimos 2 casos teríamos que alterar a mesma classe, e no primeiro caso, teríamos que puxar a classe ServicoConfigUsuario para autenticar, o que é desnecessário.

Uma forma simples de corrigir isso, é decompondo a classe ServicoConfigUsuario em duas outras, como mostrado no exemplo: (slides).