Revisão de Fachada e Strategy na prática

Se alterarmos a assinatura do método Processar() de uma IStrategy, caso queiramos fazer uma validação de uma lista de objetos. No caso de clientes, teremos que fazer que o validarCliente**s**() chame o validarCliente().

O HashMap é uma classe que implementa a interface IMap, o qual nos permite chamar um valor através de uma chave. No código a seguir, exibe no construtor da classe Fachada o conjunto de validações que sejam validados para cada entidade:

```
public Fachada()
{      // instanciando os mapas de daos e regras de negocio
       daos = new HashMap<String, IDAO>();
       //mapa que possui a lista de STRATEGY
       rns = new HashMap<String, List<IStrategy>>();

       // definindo o dao para o cliente
       daos.put(Cliente.class.getName(), new DAOCliente());

       // denfido o dao para o funcionario
       daos.put(Funcionario.class.getName(), new DAOFuncionario());

       ValidarDadosCliente vCliente = new ValidarDadosCliente();
       ValidarCpf vCpf = new ValidarCpf();
       ValidarCredito vCredito = new ValidarCredito();
       ValidarDependentes vDeps = new ValidarDependentes();
       ValidarExistencia vExistencia = new ValidarExistencia();
       ComplementarDtCadastro cDtCad = new ComplementarDtCadastro();

       List<IStrategy> rnsCliente = new ArrayList<IStrategy>();
       rnsCliente.add(vCliente);
       rnsCliente.add(vCpf);
       rnsCliente.add(vCredito);
       rnsCliente.add(vDeps);
       rnsCliente.add(vExistencia);
       rnsCliente.add(cDtCad);

       List<IStrategy> rnsFuncionario = new ArrayList<IStrategy>();

       rnsFuncionario.add(vCpf);
       rnsFuncionario.add(cDtCad);

       rns.put(Cliente.class.getName(), rnsCliente);
       rns.put(Funcionario.class.getName(), rnsFuncionario);
}
```

A seguir, iremos aplicar uma das regras para o método salvar cliente. Neste caso, iremos ao mapa que foi alimentado e dentro do método está chegando como um parâmetro uma instancia da

Entidade de domínio, através de uma string, jogando-o para o atributo do tipo IStrategy.

```
@Override
public Resultado salvar(EntidadeDominio entidade)
{	resultado = new Resultado();
	sb.setLength(0);
	//pega a entidade mapeada através de uma string
	String nmClasse = entidade.getClass().getName();
	//pega as regras referenciadas a entidade
	List<IStrategy> rnsEntidade = rns.get(nmClasse);
	//executa as regras
	executarRegras(entidade, rnsEntidade);

	if(sb.length()==0)
	{	IDAO dao = daos.get(nmClasse);
		dao.salvar(entidade);
		resultado.addEntidade(entidade);
	}
	else
	{	resultado.addEntidade(entidade);
		resultado.setMsg(sb.toString());
	}
	return resultado;
}

private void executarRegras(EntidadeDominio entidade, List<IStrategy>
                                rnsEntidade)
{	for(IStrategy rn:rnsEntidade)//para cada regra recebida, pois ao
                                    //mesmo tempo pode receber várias
	{	String msg = rn.processar(entidade);
		if(msg!=null) {sb.append(msg);}//retorna mensagem para as
                                               //respectivas aplicações
	}
}
```

Perceba que esta codificação não serve apenas para salvar uma entidade, que no nosso caso o foco é cliente, mas sim é genérico, ou seja, vale para qualquer entidade.