**Construção de sistemas**

No desenvolvimento de sistemas, primeiramente, se deve ter a noção de separar o seu uso, de sua construção, pois embora ao utilizar, se tenha a impressão de que uma ação tenha um caminho direto a um objetivo por baixo dos panos, na maioria das vezes, em projetos de alto nível, há uma grande separação de tarefas para que tal ação tenha um retorno correto, por mais simples que seja.

**Separação de tarefas do time de desenvolvimento**

Quando desenvolvemos um software, um conceito muito importante e básico, mas que muitas vezes é ignorado, é a divisão de tarefas no time de desenvolvimento.

A gerência de projetos tem um papel grande de separar uniformemente cada etapa de desenvolvimento, acoplando tarefas que combinem com as habilidades de cada desenvolvedor, para que nenhum conhecimento seja desperdiçado, mas sim aproveitado para um software coerente e bem produzido.

**Separação de tarefas no código**

Porém o conceito de separação de tarefas não se aplica somente a equipe de desenvolvimento, mas também na própria codificação. Seguindo o conceito de programação orientada à objetos, dividimos cada etapa e tarefa em determinadas classes e objetos, seguindo um padrão que compõe um software bem dividido com cada papel bem definido.

Dentro da POB temos conceitos que nos auxiliam a obter um código mais limpo e que obedeça a algumas regras e padrões seguidos, como por exemplo o princípio da responsabilidade única, que nos diz que um método não poderá ter mais de uma responsabilidade.

**Desenvolvimento gradual dirigido a pequenas implementações a cada atualização, e testes constantes**

No início de planejamento, mais especificamente, é normal que se tenha várias ideias de implementação num sistema. Porém invés de se implementar várias funcionalidades ao mesmo tempo, é preferível que se comece implementando apenas algumas funções básicas do software, a fim de resolver todos os problemas e validar todos os testes, para que depois o foco da equipe de desenvolvimento mude para implementar funcionalidades novas.

**Refatoração para um código limpo**

Um conceito muito utilizado no clean code é a refatoração de código, que tem como premissa primeiramente o desenvolvimento de forma rápida e sem muitas preocupações com clean code, a fim de validar um funcionamento, para depois “refatorar” todo o código de forma legível e de fácil entendimento, minimizando blocos de código e comentando a funcionalidade de cada trecho.

**Uso de frameworks de POA para utilizar proxies e controlar acessos**

A fim de auxiliar na divisão de tarefas, foram criados frameworks de POA para a utilização de proxies em java por exemplo quando se tem objetos grandes que só serão acessados raramente.

Estas estruturas criam um objeto falso que representa o objeto grande utilizado, e são a principal comunicação do usuário com aquilo, se passando pelo objeto real, com o propósito de deixar todo o trabalho de lidar com um objeto grande para o proxy.