**Modelos de Processo**

Construídos de acordo com modelos ou estilo, mais conhecidos quando aplicados como ciclo de vida.

Existindo duas famílias: prescritivos e os ágeis.

Não existe um modelo melhor que os outros.

O engenheiro de software deve saber escolher aquele mais adequado com a equipe que vai desenvolver o projeto.

Escolhendo bem, será mais fácil ter controle sobre os riscos, erros e o processo serão mais eficientes.

A má escolha poderá gerar trabalhos repetitivos e frustrantes.

De acordo com Ellis (2010) o engenheiro de software deve responder as seguintes perguntas para escolher um ciclo de vida:

Qual bem os analistas e o cliente podem conhecer os requisitos do sistema?

Qual bem e compreendida a arquitetura do sistema?

E provável que sejam feitas grandes mudanças de rumo ao longo do projeto?

Qual o grau de confiabilidade em relação ao cronograma? Por exemplo, se programar para o tempo decorrente necessário para finalizar o projeto e seus atrasos previstos.

O grau de risco que o projeto apresenta?

Qual o grau de adaptação necessária para a equipe poder utilizar o ciclo de vida mais adequado? Etc...

**Codificar e Consertar (CODE AND FIX)**

Passos: Construa um entendimento junto ao cliente sobre o sistema a ser desenvolvido.

Programar uma primeira versão do sistema.

Interagir com o cliente para que se corrija a versão preliminar em favor da satisfação do mesmo.

Faca testes e corrija os inevitáveis erros.

Entregue o produto.

Sem previsibilidade.

Uma forma bastante ingênua de modelo de processo.

Pode-se dizer que nem sequer parece ser um processo, pois não há previsibilidade e seus resultados obtidos.

Modelo simples, por isso e utilizado e não porque funciona bem.

Utilizados em empresas que não usam modelo de processo algum, utilizando provavelmente Codificar e Consertar por default.

Utilizado muito provavelmente em empresas de pequeno porte, entendem=do mais como uma maneira de pressionar programadores do que um processo organizado (McConnell), 1996).

Quando o sistema não satisfaz o cliente e cobrado do programador uma versão funcional adequada no menos tempo possível.

**Vantagens**

Não se perde tempo com documentação, planejamento ou projeto.

Facilmente visível à medida que o programa vai ficando pronto.

Não existe necessidade de treinamento ou conhecimentos especiais. Qualquer pessoa que programe pode desenvolver software com esse modelo de ciclo.

**Desvantagens**

A técnica apresenta problemas, pois o software sofreu varias correção assim fica cada vez mais difícil de modificar.

O cliente certamente precisa mais de certas necessidades atendidas do que códigos ruins produzidos rapidamente.

Muitas vezes os requisitos podem ser levantados às pressas assim muitos necessidades não são percebidas, deixando de atingir os objetivos.

Existe certa dificuldade de avaliar o progresso ate que o programa esteja funcionando, sendo assim difícil também de avaliar os riscos e a qualidade do projeto.

Durante o projeto for identificado que algumas decisões tomadas da arquitetura estavam erradas, não existe solução a não ser começar tudo de novo.

Assim sem testes executados sistematicamente podem ocorrer à produção de sistemas com grandes probabilidades de erros e instáveis.

**CASCATA ENTRELACADO (SASHIMI)**

Modelo de 1990 e uma tentativa de atenuar a característica BDUF do modelo cascata.

Ao invés de se produzir documentação completa fase por fase, ele propõe que se iniciem questões da fase seguinte e continue resolvendo as da fase anterior.

SCRUM

E uma das evoluções mais importantes do modelo Sashimi, a ideia e levar as fases entrelaçadas ao extremo na qual em apenas uma fase são realizadas todas as fases do modelo Cascata paralelamente.

**Vantagens**

Esse modelo indica, por exemplo, que uma fase só estará completa depois que o projeto estiver terminado.

Reduz a quantidade de documentação, pois as equipes independentes das fases irão trabalhar juntas em grande parte do tempo.

**Desvantagens**

Fica mais difícil definir marcos (millestones) devido não ficar claramente visível quando exatamente termina uma fase.

Atividade paralela realizada ao mesmo tempo pode gerar falhas de comunicação, causando assim erros com hipóteses erradas tornando o trabalho ineficiente.

**CASCATA COM SUBPROJETOS (WATERFALL WITH SUBPROJECTS)**

Da pra se executar algumas fases do modelo cascata em paralelo.

Após o projeto de arquitetura, e possível subdividir em vários subsistemas para serem desenvolvidos paralelamente por diferentes equipes.

Vantagens

Modelo razoável para se utilizar do que o modelo cascata puro e permite que subprojeto mais rápido e fácil para se gerenciar sejam realizados.

Com esta técnica o progresso e mais visível, e possível se produzir varias entregas de partes funcionais do sistema na medida em que vão ficando prontas.

Desvantagens

A dificuldade maior existe na possibilidade de surgir imprevistos entre subsistemas.

O projeto arquitetônico deve ser bem feito para minimizar os problemas.

E um modelo que exige muito mais gerencia para impedir inconsistências entre subsistemas.