--> Modelos de desenvolvimento:

-Tradicional:



-Ágil: Produto subdivido em entregas menores, que passam sempre pelo processo acima. Um release grande no ágil, portanto, é rápido já que há vários pedaços do programa sendo criados juntos.



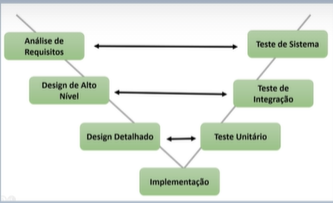
-Entregas menores possibilitam lidar com a rápida concorrência do mercado, pois as mudanças entram já no próximo lançamento.

* O método ágil tem muito menos documentação, muito mais centrada e enxuta. O ágil foca em entregar o produto, com foco no cliente.

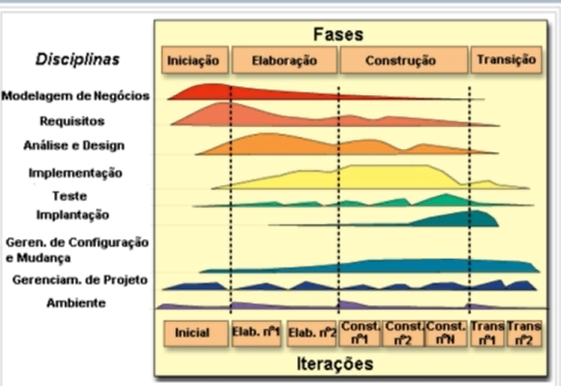


--> Tipos de teste:

* Sequencial em V/cascata:



* Incremental Processo Unificado Racional: Cada interação do ágil passa por todas as fases, com TESTES EM TODO O PROCESSO.



-Métodos ágeis têm testes durante toda sprint (entrega de valor)

-O Kanban coloca uma coluna só para os testes, facilitando o trabalho de todos.

=> Importante: Os modelos de teste devem ser adaptados ao ambiente de trabalho.

-> Os níveis de teste:

-Componente: Porção definida do sistema

-Integração entre os componentes testados

-Teste do sistema integrado

-Aceitação final do cliente

* Cada um desses níveis tem objetivos específicos, bases de teste próprias, objetos de teste específicos, defeitos e falhas típicos e abordagens próprias.

1) O teste de componente:

-Realizados pelo desenvolvedor;

-Busca reduzir os riscos, verificando aspectos funcionais e não funcionais; Garantir a qualidade geral do componente, encontrando defeitos; Evitar que esses defeitos se espalhem para níveis mais altos do teste (Identificar e resolver problemas tão antes quanto possível)



2) Teste de integração:

-Feitos pelo desenvolvedor e pelo testador, garante que todos os componentes se conversem da forma esperada



-Pode haver também integração entre sistemas diversos

-> Integração big bang: Todos os módulos são integrados individualmente, após serem construídos e testador

-> Integração top-down: Testa as unidades mais altas, usando stubs (mocks) para os componentes que ainda não foram criados.

-> Integração bottom-up: Componentes mais baixos são construídos primeiro. O chamado driver é um componente mais alto, mockado, que testa a integração entre eles, realizando chamadas de interfaces.

3) Teste de sistema:

-Garantir que o sistema todo está de acordo com os requisitos e funciona como deveria, evitando que defeitos migrem para produção.



4) Teste de aceite:

-Verificar se o sistema atende os requisitos, normas legais,etc e funciona como deveria.



* Nesse nível, deve-se cuidar muito de performance.

-> Teste de caixa preta: Preocupação com as funcionalidades, sem olhar o código; O software atende as necessidades do usuário?

* Existe também um teste de caixa preta não funcional, que verifica usabilidade e segurança.

-> Teste de caixa branca: Foca na cobertura do código em si, verificando se ele todo é executado, na arquitetura do sistema/modelagem de dados.

-> Teste de regressão e confirmação: Verifica se as mudanças feitas não afetam outros componentes; Verificar se a mudança foi implementada corretamente.

-> Teste de manutenção: Feita em produção, para testar atualizações e migrações para um novo sistema.; Testar o impacto de mudanças. Ênfase nos testes de integração e segurança.