

## Apresentações: Avaliação Geral

### **RAD (Rapid Application Development):**

**Nota: 7.0**

- Criado em 1991.
- Ciclo rápido com cerca de 60 a 90 dias.
- Baixo custo.
- Usuário com forte envolvimento.
- Reutilização de componentes.
- Tempo de teste reduzido.
- Frisa o a reutilização de código.

Caracterizado como um modelo sequencial, sem marcos clássicos, que enfatiza o reuso do código para atingir sua máxima: um ciclo de desenvolvimento do projeto extremamente curto.

---

### **SOA (Service-oriented architecture):**

**Nota: 6.0**

- Serve de “amálgama” entre os processos do negócio de forma proativa.
- Focada para o meio corporativo e suas atividades internas.
- Baixo acoplamento.
- Desempenho irá depender de hardware.
- Reutilizável.
- Cerca de 43% dos executivos desconfiam da segurança desta arquitetura.

Tem como objetivo integrar as aplicações, disponibilizar maior flexibilidade para mudanças, suportar serviços independentes de plataforma e protocolos. Por isso é

reutilizável e conecta diferentes departamentos dentro de um mesmo meio corporativo afim de oferecer uma forma ágil e coesa de gerenciamento da mesma.

---

### TDD (Test-driven development):

#### Nota 7.0

- Mesmo criador do XP (Kent Beck).
- Derivada do XP.
- Refatoração do código.
- Grande camada de testes.
- Realização de testes automatizados

Baseado em ciclos curtos de repetições, na qual o desenvolvedor primeiramente escreve o caso de testes para depois começar a desenvolver. Após isto é feita a refatoração, melhorias no código, realizada nova bateria de testes.

---

### Crystal (Crystal Family of Methodologies):

#### Nota 7.0

- Ciclos de 4 meses no máximo de forma incremental.
- Organizado por cores para definir “intensidade do método”.
- Inclui um grande número de métodos diferentes.
- Metodologia focada em pessoas.

Os membros da “família Crystal” são identificados por cores que indicam a intensidade do método. Neste caso, quanto mais escura a cor, maior é a complexidade do projeto, enquanto as cores mais claras indentificam projetos de menor complexidade. Esta intensidade linear também irá incidir no contingente e custo estimado do projeto.

---

**FDD (Feature Driven Development):****Nota 9.0**

- Criado entre 1997 e 1999 para um grande projeto em Java.
- Existia antes do manifesto de metodologias ágeis.
- Busca o desenvolvimento por funcionalidade.
- Duas fases definidas: Concepção e Planejamento.
- Faz teste de software.
- Fornece forma de saber se o plano e a estimativa são sólidos.
- Resultados em até duas semanas.

Focada em funcionalidades do sistema, com resultados iterativos a cada duas semanas, atua no processo de desenvolvimento baseado em orientação a objetos. Pode atuar em parceria com o Scrum, sendo este, direcionado ao gerenciamento.

---

**BDD (Behavior Driven Development):****Nota 9.0**

- Utiliza User Stories.
- Criada em 2003, sendo mais “recente”.
- Desenvolvimento guiado a comportamento.
- I.N.V.E.S.T (Independent, Negotiable, Valuable, Estimable, Small, Testable).

Prioriza a utilização de User Stories para cada funcionalidade, focando na linguagem e as interações usadas no processo de desenvolvimento de software.

---

**Kanban (sinalização visual):****Nota 8.0**

- Criada em 1940 no Japão.
- Surgiu para controlar a produção na empresa Toyota

- Utilização massiva de post-its.
- Melhorias constantes.
- Auxilia na gestão visual do trabalho.
- Idealizada para gestão de projetos em nível industrial.

Fundamentada para transparecer o progresso de cada setor envolvido na produção, podendo incluir tempo, alterações, pendentes. Desta forma fica mais evidente onde o problema deve ser resolvido, gerando uma constante melhoria e qualidade nos processos envolvidos. Por depender de post-its, estes manipulados por mão de obra humana, pode resultar em erros crassos extremamente prejudiciais ao projeto.

---

### ASD (Adaptive Software Development):

#### Nota 7.0

- Prazos pré-fixados
- Iterativo e incremental
- Orientado a missões
- Orientado a riscos
- Tolerante a mudanças

Com ciclo de vida definido, focado no cliente, utilizada em especial para desenvolvimentos complexos, depende da colaboração humana para atingir este objetivo em comum, de forma auto organizada.