# Avaliação

Emily Ramos TBDS14

Metodologias Clássicas

Modelo cascata:

Uma metodologia clássica de gestão de projetos baseada em uma abordagem linear e sequencial, onde cada fase precisa ser concluída antes de avançar para a próxima.

As principais etapas são:

* **Iniciação**: Definição dos requisitos, orçamento e ferramentas necessárias.
* **Planejamento**: Estruturação do projeto, incluindo equipe, cronograma, tarefas e metas.
* **Execução**: Implementação do plano de ação com acompanhamento contínuo.
* **Monitoramento e Controle**: Avaliação de indicadores, prazos e ajustes necessários.
* **Encerramento**: Revisão dos resultados para aprimoramento em futuros projetos.

Ele apresenta vantagens como facilidade de gerenciamento, otimização do tempo e maior disciplina, pois cada fase tem etapas bem definidas.

Para aplicar essa metodologia com eficiência, é essencial definir bem cada etapa, criar um cronograma detalhado e monitorar constantemente o andamento do projeto.

Modelo V-Model:

Uma variação do modelo cascata, utilizada principalmente no desenvolvimento de software. Ele mantém uma abordagem sequencial, mas enfatiza a relação entre cada fase de desenvolvimento e sua respectiva fase de testes. O processo segue um formato de "V", onde a descida representa o desenvolvimento e a subida representa a validação.

As principais etapas são:

* **Definição de Requisitos**: Levantamento das necessidades do sistema.
* **Especificação Funcional**: Detalhamento de como o sistema deve funcionar.
* **Design Arquitetural**: Estrutura do sistema e divisão em módulos.
* **Design de Componentes**: Especificação detalhada de cada módulo.
* **Implementação**: Desenvolvimento do código do sistema.
* **Testes Unitários**: Verificação individual de cada módulo.
* **Testes de Integração**: Validação da interação entre os módulos.
* **Testes de Sistema**: Verificação do funcionamento completo do software.
* **Testes de Aceitação**: Validação final baseada nos requisitos do usuário.

Ele garante maior qualidade ao associar cada fase de desenvolvimento a uma fase de teste correspondente, permitindo a detecção precoce de falhas. Sua principal vantagem é a previsibilidade e a organização, mas exige planejamento rigoroso e pouca flexibilidade para mudanças.

Modelo Espiral:

Modelo Espiral é um modelo de processo de desenvolvimento de software orientado a riscos. É uma combinação de modelo cascata e modelo iterativo. O Spiral Model ajuda a adotar elementos de desenvolvimento de software de vários modelos de processos para o projeto de software com base em padrões de risco exclusivos, garantindo um processo de desenvolvimento eficiente.

## Quando usar o modelo espiral?

* Um modelo espiral em engenharia de software é usado quando o projeto é grande
* Quando é necessário que as liberações sejam frequentes, a metodologia espiral é usada
* Quando a criação de um protótipo é aplicável
* Quando a avaliação de riscos e custos é importante
* A metodologia espiral é útil para projetos de médio a alto risco
* Quando os requisitos não são claros e complexos, o modelo espiral em [SDLC](https://www.guru99.com/pt/software-development-life-cycle-tutorial.html) é útil
* Quando as mudanças podem exigir a qualquer momento
* Quando o compromisso do projecto a longo prazo não é viável devido a mudanças nas prioridades económicas

## Vantagens e desvantagens do modelo espiral

| * Diferenciais |
| --- |
| * Funcionalidades ou alterações adicionais podem ser feitas posteriormente |
| * A estimativa de custos torna-se fácil porque a construção do protótipo é feita em pequenos fragmentos |
| * O desenvolvimento contínuo ou repetido ajuda na gestão de riscos |
| * O desenvolvimento é rápido e os recursos são adicionados de forma sistemática no desenvolvimento em espiral |
|  |
|  |

Desvantagens

* Risco de não cumprir o cronograma ou orçamento
* O desenvolvimento em espiral funciona melhor apenas para grandes projetos e também exige experiência em avaliação de riscos
* Para seu bom funcionamento, o protocolo do modelo espiral precisa ser seguido rigorosamente
* A documentação é mais porque tem fases intermediárias
* O desenvolvimento de software em espiral não é aconselhável para projetos menores, pois pode custar muito caro

Metodologias Ágeis

Scrum

O Scrum não é apenas uma metodologia, mas sim uma filosofia que permeia toda a dinâmica de um projeto. Baseia-se em três pilares fundamentais: transparência, inspeção e adaptação. Esses pilares orientam as equipes na busca pela excelência e eficácia em seus projetos.

Vantagens:

* Flexibilidade para adaptações rápidas.
* Transparência e colaboração entre os membros da equipe.
* Entregas incrementais aumentam a previsibilidade do progresso.

Desvantagens:

* Requer alto comprometimento da equipe.
* Pode ser difícil de implementar corretamente sem experiência.
* Nem sempre adequado para projetos com requisitos muito estáveis.

XP (Extreme Programming)

Uma das suas principais características é a programação em pares, onde dois desenvolvedores trabalham juntos em um único computador, um escrevendo o código e o outro revisando simultaneamente. Isso aumenta a qualidade do código e reduz o número de bugs.

Outros aspectos notáveis incluem integração contínua, refatoração regular do código para melhorar sua estrutura, e a criação de testes automatizados que são executados constantemente para garantir que as mudanças não quebrem funcionalidades existentes.

O XP também valoriza o feedback do cliente, incorporando-o através de releases curtas que permitem que o software seja testado e melhorado em ciclos rápidos.

Vantagens:

* Alta qualidade do código devido à revisão constante.
* Rapidez na entrega e resposta às mudanças.
* Maior envolvimento do cliente.

Desvantagens:

* Exige alto nível de disciplina e colaboração.
* Pode ser difícil de aplicar em equipes grandes.
* Necessita de testes automatizados eficazes.

**L**ean

A metodologia lean é uma filosofia que visa a zerar desperdícios, diminuir custos e aumentar a produtividade dentro de uma empresa. Aliado ao uso de tecnologias, esse método fornece os meios para uma gestão eficiente com foco em garantir uma entrega de máxima qualidade para o cliente final.

Vantagens:

* Redução de desperdícios e custos.
* Processo ágil e eficiente.
* Foco na melhoria contínua.

Desvantagens:

* Difícil de implementar em empresas com cultura tradicional.
* Exige envolvimento ativo da equipe e do cliente.
* Pouca estrutura formal, o que pode levar a dificuldades na gestão.

Kanban

O Kanban é um sistema de controle e fluxo, desenvolvido pela Toyota na década de 40 para controlar o estoque de peças. A montadora japonesa o classificou como uma parte do Sistema Toyota de Produção. Naquela época, o objetivo do método era promover o controle do estoque de materiais para que os produtos não excedessem e nem faltassem, trazendo equilíbrio para o seu estoque e linha de produção. David Anderson adaptou o método utilizado pela Toyota para desenvolver um controle detalhado do desenvolvimento de software com informações sobre quando, quanto e o que desenvolver. A base do Kanban é utilizar a gestão visual das atividades para melhorar o fluxo dos processos.

Vantagens:

* Visualização clara do progresso das tarefas.
* Fácil de implementar em qualquer equipe.
* Permite mudanças contínuas sem impacto significativo.

Desvantagens:

* Pode gerar gargalos se mal gerenciado.
* Não define papéis e responsabilidades de forma estruturada.
* Menos adequado para projetos muito complexos.

Crystal

Método ágil de desenvolvimento de software e gerenciamento de projetos que foi criado para atender as necessidades especificas de cada projeto, com foco nas características da equipe e no contexto do trabalho

Vantagens:

* Flexível e adaptável a diferentes contextos.
* Enfatiza comunicação e colaboração.
* Valoriza pessoas e interação sobre processos rigidamente definidos.

Desvantagens:

* Falta de estrutura pode dificultar a aplicação em equipes grandes.
* Requer alto nível de maturidade da equipe.
* Pode ser menos eficaz em projetos com muitas regras e requisitos formais.

Metodologias Ágeis

Go Horse

Go Horse é uma abordagem não estruturada e improvisada ao desenvolvimento de software que, muitas vezes, resulta em soluções temporárias e de baixa qualidade, também conhecidas como “gambiarras”. Esta abordagem tende a priorizar a entrega rápida de soluções, mesmo que isso signifique comprometer a qualidade do design, a arquitetura do software e a manutenibilidade do código. Vamos detalhar como o Go Horse afeta o design de sistemas, a arquitetura de software e a qualidade em geral.