AULA 3 – CICLO DE VIDA DO SOFTWARE

OBJETIVO DA AULA

Conhecer as definições do ciclo de vida do desenvolvimento do software, assim como discutir e comparar os benefícios e as desvantagens dos processos incrementais, iterativos, evolutivos e ágeis.

APRESENTAÇÃO

O processo do software, também conhecido como ciclo de vida do software, visa descrever os processos e atividades envolvidas na operação e na evolução do sistema, ao longo de toda a sua existência.

Para facilitar e conduzir esse processo de gestão do software existe modelos que definem como o software será elaborado, guiando o desenvolvedor ao logo de cada fase.

A definição de um modelo vai definir, por exemplo, a ordem de sequenciamento das atividades, dentre muitos quesitos ao longo do desenvolvimento do software.

Esta aula apresentará um pouco sobre esse processo, abordando os principais ciclos de vida existentes e praticados no mercado.

1. CICLO DE VIDA

O ciclo de vida de um software constitui-se na divisão do desenvolvimento em etapas. As etapas por sua são desmembradas em atividades ou disciplinas que possuem importantes passos para a elaboração do software.

Os ciclos de vida são aplicados para conduzir o desenvolvimento e vão produzir artefatos na finalização de cada etapa. Os ciclos de vida são definidos de acordo com vários fatores como familiaridade da organização, experiência em projetos anteriores, filosofia e, principalmente, visto a natureza do projeto, ou seja, sua criticidade de tempo, complexidade e tamanho.

Existem diversos modelos de ciclo de vida utilizados atualmente no mercado, e outros que se tornaram obsoletos ao longo do tempo. Atualmente costumamos dividir os modelos em incrementais e iterativos, mas também consideramos a filosofia de desenvolvimento ágil, que é amplamente adotada nas organizações.

O ciclo de vida de desenvolvimento de um sistema ou software define as etapas que deverão ser seguidas durante a construção do produto, e quais são artefatos que devem ser gerados em cada uma das etapas, desde o conhecimento do problema até a solução instalada num computador.

Consideramos como artefato de software todo produto originário a partir do desenvolvimento de um software. Alguns exemplos são documentações, códigos, planos de testes, diagramas, como os de casos de uso, diagramas de classes, diagramas entidade-relacionamento, e outros modelos da UML, lista de requisitos, protótipos, manuais, documentos de visão, além de uma infinidade de outros objetos oriundos de um esforço pessoal ou coletivo.

Apesar dos modelos de ciclos de vida possuírem a mesma proposta, ou seja, a elaboração de um sistema, o processo de execução difere um do outro. É importante lembrar, no entanto, que a abordagem a ser utilizada é a mesma: conhecer o problema, representar conceitualmente a solução, e desenvolver o produto, procedendo testes e validações para que ele chegue à produção e possa ser utilizado pelos usuários.

2. MODELO EM CASCATA

O modelo em cascata, ou *waterfall*, é um dos modelos mais antigos, ele foi criado em 1966, porém teve a sua formalização apenas em 1970 por Wiston Royce.

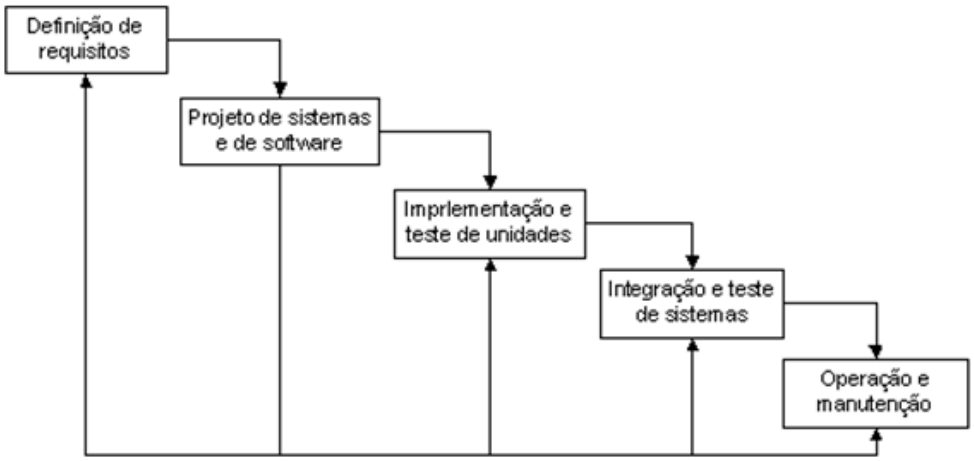
Neste modelo as etapas são executadas de maneira sequencial, na prática, isso quer dizer que uma fase só inicia após o término da fase anterior. Essa característica, no entanto, é uma das grandes desvantagens deste modelo, já que o torna inflexível quanto às mudanças de requisitos, algo comum a todo projeto atualmente se considerarmos a velocidade de mudança nos processos e nas regras de negócios.

No modelo cascata são produzidos muitos artefatos antes do processo de desenvolvimento do produto, isso faz com que o cliente e usuários tenham raros contatos com o sistema que está sendo produzido, aumentando as expectativas e incertezas.

Essa característica classifica este modelo como burocrático, pois mesmo que o projeto já tenha iniciado há um bom tempo, muitas vezes o que temos pronto são apenas documentação e diagramas.

Este modelo não oportuniza uma validação mais rápida por parte do usuário, que pode detectar erros de entendimento e de programação antecipadamente, o que ajudaria a reduzir o impacto, caso tais problemas sejam identificados somente no final do projeto.

FIGURA 1 | **O Modelo Cascata**



Fonte: SOMMERVILLE (2019, p.20)

Na Figura 1 observamos o modelo cascata com as suas respectivas atividades, do levantamento de requisitos até o processo de operação e manutenção. É possível observar os

feedbacks ao fim de cada fase, embora na maioria das organizações este modelo é tratado de forma estritamente linear.

Entre outros problemas cabe ressaltar a dificuldade ao necessariamente termos que levantar todos os requisitos e detalhes do projeto já na fase inicial, pois muitas vezes alguns pontos importantes podem ficar de fora desse levantamento, muitas vezes por conta do próprio usuário no momento de levantar as suas próprias necessidades.

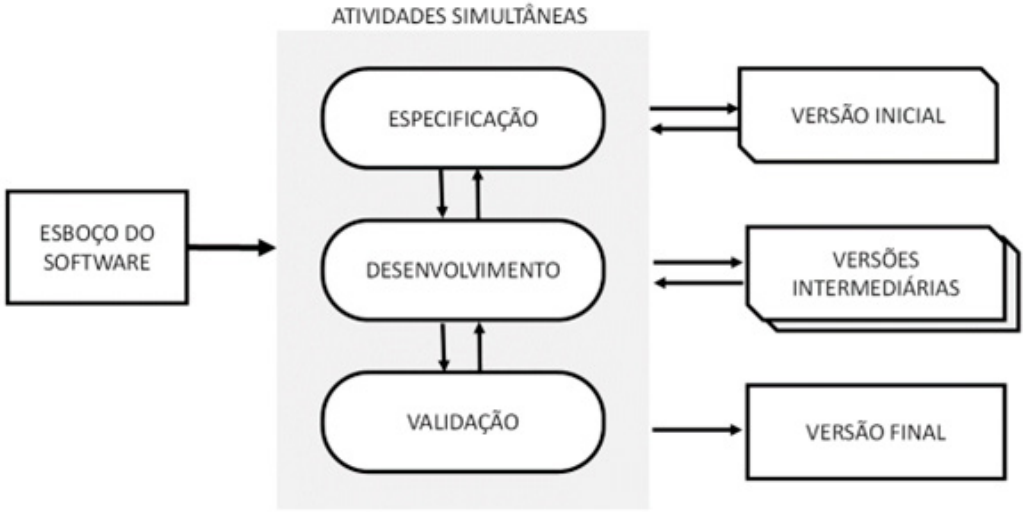
Em um ponto de vista mais otimista, podemos dizer que o cascata tem ao seu favor o fato de ele ser muito simples de ser elaborado e executado, pois a sua estrutura é extremamente simples.

Este modelo deve ser usado apenas em casos em que todos os requisitos são conhecidos e quando houver pouca chance de mudança ao longo do desenvolvimento do produto. Em termos práticos, este modelo favorece pequenos projetos, tornando fácil também as tarefas de testes e validação.

3. ENTREGA INCREMENTAL

Os modelos incrementais são grupos de processos de software que trabalham a partir da geração de incrementos a cada ciclo de desenvolvimento. O trabalho de desenvolvimento prevê a elaboração de uma implementação inicial, que é submetida aos comentários e críticas dos usuários. Após o feedback, são realizadas outras versões, até que uma versão final do sistema seja aprovada.

FIGURA 2 | **Desenvolvimento Incremental**

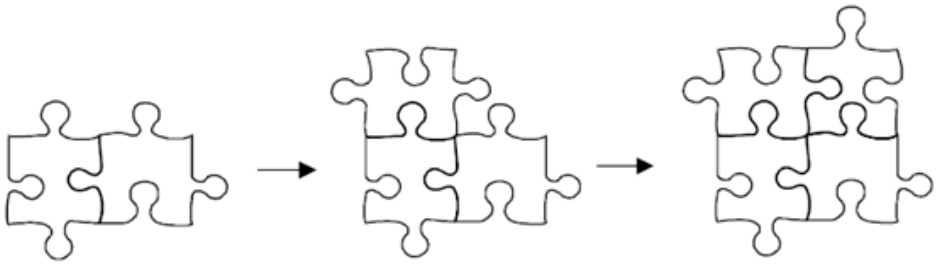


Fonte: SOMMERVILLE (2011, p. 22)

Na Figura 2 observamos o modus operandi do desenvolvimento incremental, em que há um esboço inicial do software e vemos as fases de especificação, desenvolvimento e validação intercaladas, e não separadas, como no cascata. Por fim, observamos as várias versões do software sendo construídas até chegarmos à versão final.

Esse tipo de desenvolvimento ganhou muita popularidade com a adoção dos métodos ágeis, já que ele é a base desse tipo de abordagem. Ele reflete a forma como normalmente resolvemos problemas me nosso dia a dia. No geral, este tipo de desenvolvimento é mais barato e facilita eventuais mudanças de requisitos ao longo do desenvolvimento do projeto.

FIGURA 3 | **Entrega Incremental**



Fonte: IFSC. Disponível em: wiki.sj.ifsc.edu.br/wiki/index.php/Ciclo_de_Vida_Iterativo_e_Incremental. Acesso em: 02 nov. 2022.

A Figura 3 mostra o produto sendo construído de maneira incremental, até chegar ao produto completo.

A ideia é que seja incrementadas novas funcionalidades a cada nova versão elaborada pela equipe de desenvolvimento, isso permite ao cliente priorizar aquelas atividades mais críticas, por exemplo, além de dar ao cliente uma visão exata de como está o processo de desenvolvimento, pelo fato de que ele precisa estar acompanhando de perto a equipe de desenvolvimento.

Apesar de apresentar visíveis melhorias se comparado ao modelo cascata, o modelo de entrega incremental apresenta alguns pontos de atenção.

Há alguma dificuldade no gerenciamento do projeto por parte do gerente, já que a documentação não acompanha de forma viável a elaboração das versões.

Em outro prisma, conforme os incrementos vão sendo gerados ao longo da vida do software, a sua estrutura tende a se degradar, ou seja, sistemas de vida muito longa não são favorecidos com este modelo de desenvolvimento.

4. MODELOS ITERATIVOS

Definimos iteração como o processo em que um conjunto de instruções ou estruturas são repetidas em uma sequência por um número específico de vezes ou até que uma condição predefinida seja alcançada.

Assim, os modelos ditos como iterativos possuem uma repetição de atividades de desenvolvimento até que o sistema esteja pronto. Um exemplo de desenvolvimento iterativo é a Prototipação, modelo no qual é gerado um novo protótipo a cada iteração realizada.

FIGURA 4 | **Ciclo de Vida Iterativo**

Fonte: IFSC. Disponível em: wiki.sj.ifsc.edu.br/wiki/index.php/Ciclo_de_Vida_Iterativo_e_Incremental. Acesso em: 02 nov. 2022.

A Figura 4 mostra um produto sendo construído iterativamente, tendo uma nova versão a cada ciclo.

A ideia central do modelo iterativo é a de refinar o sistema a cada iteração, ou seja, melhorá-lo pouco a pouco. Em cada iteração os desenvolvedores identificam junto ao cliente os requisitos relevantes, criando um projeto que utilize uma arquitetura que será escolhida como guia, o projeto é implementado então a partir destes componentes.

Assim que uma iteração atinge os seus objetivos, o desenvolvimento passa para a próxima iteração, caso contrário, a equipe volta ao ciclo para correções, realizando os devidos ajustes e submetendo o produto a uma nova avaliação. Dessa forma, podemos dizer que não é o sistema como um todo que é alterado, mas, sim, a sua composição, o seu detalhe, em cada iteração.

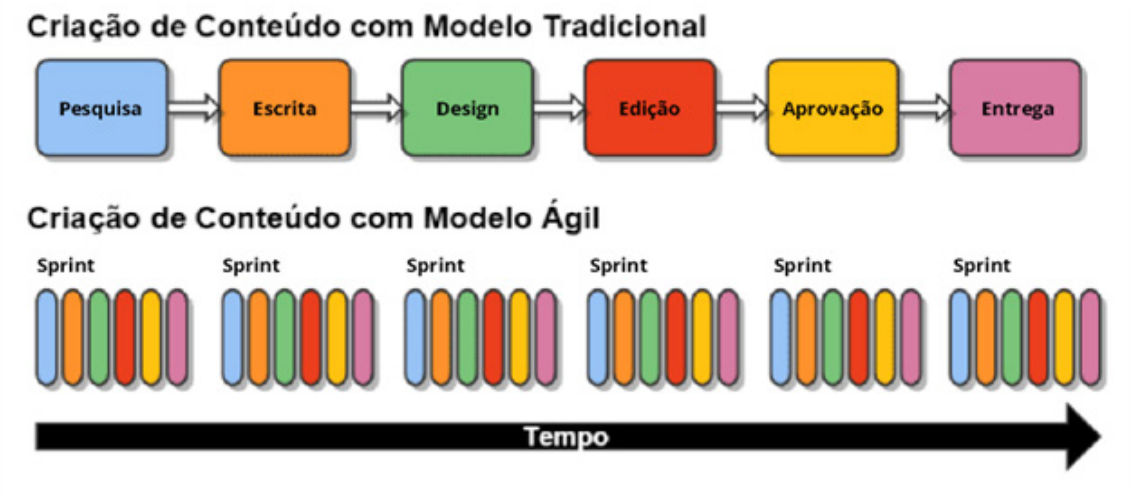
O modelo iterativo também é conhecido como evolucionário, e tem como exemplos os modelos Prototipação e Espiral. Ele é atualmente é empregado juntamente como complemento de outro modelo de ciclo de vida como o incremental.

5. DESENVOLVIMENTO ÁGIL

O desenvolvimento ágil é uma filosofia de desenvolvimento de software que utiliza normalmente os modelos iterativos e incrementais, mas especificado a partir de uma abordagem que visa uma postura de maior colaboração com o cliente e maior integração entre a equipe desenvolvedora.

Esta abordagem oferece algumas vantagens importantes ao processo de resolução de problemas de software. Uma delas é realizar o desenvolvimento de uma maneira que permita que os clientes possam ver os resultados mais rapidamente, e outra importante vantagem é a entrega do produto funcional de maneira mais rápida, tendo incrementos agregados, de forma flexível, a cada novo ciclo.

FIGURA 5 | **Comparação do Modelo Tradicional X Modelo Ágil**



Fonte: Blog Cdlfor. Disponível em: https://blog.cdlfor.com.br/dicas/metodologia-agil/. Acesso em: 02 nov. 2022.

Na Figura 5 distinguimos a diferença entre as abordagens, onde no modelo tradicional a construção se dá etapa por etapa, enquanto no modelo ágil cada nova iteração (Sprint) produz uma série de novos incrementos ao sistema.

Os métodos ágeis são focados em entregas incrementais contínuas, com redução considerável na documentação gerada ao longo do desenvolvimento do software. Os exemplos de modelos mais comuns nessa abordagem são o XP (*Extreme Programming*) e o Scrum.

CONSIDERAÇÕES FINAIS

O ciclo de vida estabelece um guia para a elaboração de software seguindo as atividades propostas por cada modelo. É importante para o analista conhecer o problema que deseja resolver para escolher a abordagem de construção ideal.

O modela mais antigo é o chamado cascata e, embora a sua utilização e implantação seja muito simples, ele possui diversos problemas que o tornam ultrapassado para a maioria dos casos.

Os modelos de entrega incremental e iterativos hoje são utilizados como base para outras abordagens de desenvolvimento como os métodos ágeis de desenvolvimento, trazendo alguns benefícios interessantes no ponto de vista de agilidade e otimização em relação aos artefatos produzidos ao longo do projeto.

MATERIAIS COMPLEMENTARES

Vídeo: *Ciclo de Vida do Sofware*. Disponível em: https://www.youtube.com/watch?v=6_fV-cpC0cxE. Acesso em: 02 nov. 2022.

REFERÊNCIAS

PRESSMAN, R. G. *Engenharia de Software*. 9ª ed. Rio de Janeiro: McGraw-Hill, 2021.

SOMMERVILLE, I. *Engenharia de Software*. 10ª ed. São Paulo: Pearson Addison. Wesley, 2019.