Conhecer as fases do ciclo de vida de software

Apresentação

O ciclo de vida de software pode ser conceituado como uma estrutura contendo processos, atividades e tarefas envolvidas na criação, operação ou manutenção de um software. Ele abrange toda a vida de um sistema desde o início, onde as definições de seus requisitos são reconhecidas, até o final, onde ele não é mais utilizado. No começo de um projeto de software, o tipo de ciclo de vida geralmente é a primeira decisão a ser tomada.

Atualmente existe um grande conjunto de ciclos de vida de software, sendo alguns deles: Cascata, Modelo em V, Incremental, Evolutivo, RAD, Prototipagem, Espiral, Modelo de Ciclo de Vida Associado ao RUP. Cada um deles possui características que podem ser uma vantagem ou desvantagem, dependendo do contexto do software que será desenvolvido.

Nesta Unidade de Aprendizagem você irá adquirir conhecimentos fundamentais para avançar no aprendizado sobre o ciclo de vida de software. Você verá como alguns problemas de projeto podem influenciar o ciclo de vida de software, bem como quais são as fases deste ciclo e seu funcionamento.

Bons estudos.

Ao final desta Unidade de Aprendizagem, você deve apresentar os seguintes aprendizados:

- Reconhecer os problemas de um projeto e a influência de ciclo de vida de software.
- Identificar as fases de ciclo de vida de software genérico.
- Caracterizar o funcionamento das fases modelo.

Desafio

A WorldInc. está com dificuldade para organizar as contas. Esta empresa, que trabalha com tradução de sites, tem muitos colaboradores, além de clientes, contas a pagar, fornecedores, impostos, entre outros.

Você está sendo contratado para desenvolver um *software* de gestão financeira. Até o momento, o único contato com o administrador da empresa foi a solicitação do serviço. Logo, você ainda não sabe exatamente o que a WorldInc. precisa.

Para que não haja problemas posteriores, você precisa organizar tudo desde este primeiro contato com o administrador. Portanto, crie um passo a passo que demonstre como você irá gerenciar a criação deste sistema, utilizando uma das técnicas do ciclo de vida de *software*.

Inicialmente deve ser feito o levantamento de requisitos junto aos clientes para verificar todas as necessidades que eles possuem. A segunda etapa seria de análise destes requisitos e também de uma avaliação dos processos, com o objetivo de otimizá-los e facilitar o gerenciamento financeiro do cliente.

Então, inicia-se o processo de planejamento do desenvolvimento, com definições de como o software será implementado, tecnologias, ambientes e hardware. Por fim, uma fase de implementação é realizada para, finalmente, desenvolver o sistema. Além disso, também seria necessário uma fase de testes para certificar o funcionamento do sistema.

Infográfico

O ciclo de vida de um *software* abrange toda a vida de um sistema, desde a definição dos seus requisitos até quando ele não é mais utilizado. Acompanhe, no infográfico a seguir, os principais elementos do ciclo de vida de *software*, além de algumas das suas variações.

O ciclo de vida do software é uma abordagem sistemática e estruturada que descreve as diferentes fases pelas quais passa o projeto de desenvolvimento dele, desde a concepção inicial à entrega do produto final. Engloba todas as etapas envolvidas no processo de criação, evolução e gerenciamento, que começa com a identificação das necessidades dos usuários e vai até a implantação. Neste vídeo, você vai conhecer as fases do ciclo de vida de um software genérico.

Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.

Conteúdo do Livro

A definição do ciclo de vida de um *software* é um item essencial na hora de decidir como um sistema será criado. Essa escolha pode impactar diretamente no sucesso do *software*.

Atualmente, existem diversos ciclos de vida que podem ser utilizados, cada um com diferentes técnicas e metodologias, que em alguns casos podem oferecer vantagem, mas em outros podem levar o projeto ao caos.

Com a leitura do capítulo Conhecer as fases do ciclo de vida de *software*, da obra *Engenharia de Software*, você poderá saber mais sobre cada um dos ciclos de vida de um *software*, suas vantagens e desvantagens.

Boa leitura!

ENGENHARIA DE *SOFTWARE*

Izabelly Soares de Morais

SAGAH
SOLUÇÕES EDUCACIONAIS INTEGRADAS

Conhecer as fases de ciclo de vida do software

Objetivos de aprendizagem

Ao final deste texto, você deve apresentar os seguintes aprendizados:

- Reconhecer os problemas de um projeto e a influência de ciclo de vida de software.
- Identificar as fases de ciclo de vida de software genérico.
- Caracterizar o funcionamento das fases modelo.

Introdução

O ciclo de vida de software pode ser conceituado como uma estrutura contendo processos, atividades e tarefas envolvidas na criação, na operação ou na manutenção de um software. Ele abrange toda a vida de um sistema desde o início, em que a definição de seus requisitos é reconhecida, até o final, quando ele não é mais utilizado. No começo de um projeto de software, o tipo de ciclo de vida geralmente é a primeira decisão a ser tomada.

Neste capítulo, você irá adquirir conhecimentos fundamentais para avançar no aprendizado sobre o ciclo de vida de software. Atualmente, contamos com um grande conjunto de ciclos de vida de software, sendo alguns deles: Cascata, Modelo em V, Incremental, Evolutivo, RAD, Prototipagem, Espiral e Modelo de Ciclo de Vida Associado ao RUP. Cada um deles tem características que podem ser uma vantagem ou uma desvantagem, dependendo do contexto do software que será desenvolvido.

Ciclo de vida de um software

Atualmente, uma enorme indústria de software tornou-se fator dominante nas economias do mundo industrializado. Equipes de especialistas em software, cada qual concentrando-se numa parte da tecnologia necessária para distribuir

uma aplicação complexa, substituíram o programador solitário de antigamente. Esses são conceitos trazidos por Pressman e Maxim (2016, p. 4), que retrará a realidade social e a necessidade de se ter técnicas para produzir um software. Para justificar sua afirmação, o autor selecionou algumas perguntas:

- Por que a conclusão de um software leva tanto tempo?
- Por que os custos de desenvolvimento são tão altos?
- Por que não conseguimos encontrar todos os erros antes de entregarmos o software aos clientes?
- Por que gastamos tanto tempo e esforço realizando a manutenção de programas existentes?
- Por que ainda temos dificuldades de medir o progresso de desenvolvimento e a manutenção de um software?

Um dos objetivos da Engenharia de Software é trazer metodologias para desenvolvimento de software, para isso, os modelos de ciclo de vida de software retratam as técnicas que podem ser utilizadas para desenvolver um sistema na prática. Devido ao avanço dos recursos tecnológicos, podemos notar a evolução pela qual esses modelos passaram, e, com isso, o impacto que esse processo trouxe para o software, até mesmo porque termos como produto final um sistema funcional, o qual deverá atender às necessidades de um cliente.

O cliente é o indivíduo que gostaria que um produto fosse criado (desenvolvido). Os desenvolvedores são os membros de uma equipe responsável pela criação desse produto. Os desenvolvedores podem ser os responsáveis por todos os aspectos do processo, desde o levantamento das necessidades até as etapas subsequentes, ou então ser responsáveis apenas pela implementação de um produto já concebido (SCHACH, 2010, p. 24).

Geralmente um software é concebido para trazer soluções para problemas específicos, dessa forma, são desenvolvidos com base em diversas especificações. Após a definição do objetivo do projeto, ou seja, do software, em que todas as necessidades do cliente são descritas, deve haver uma análise do que é válido ou não para ser implementado, inserido, no sistema. Devemos estar cientes de que nem toda informação passada pelo cliente é válida para solucionar o seu problema, e, dessa forma, para ser inserida no processo de produção.

Para Schach (2010, p. 36), "[...] o desenvolvimento de software é consideravelmente diferente na prática por duas razões. Primeiramente, os profissionais de software são seres humanos e, portanto, cometem erros. Em segundo lugar,

as necessidades do cliente podem mudar enquanto o produto de software está sendo desenvolvido". Esse é um dos motivos pelos quais o software passa por diversos processos para ser classificado como pronto. Outro fato importante sobre esse assunto é que um software nunca é finalizado, no sentido de que nunca irá passar por alguma modificação futura. Na verdade, ele está em constante evolução, pois pode muito bem passar por atualizações conforme a necessidade do cliente. Veremos mais adiante, neste capítulo, um ciclo genérico de produção de software.

Fique atento

Há muitos mecanismos de avaliação de processos de software que possibilitam às organizações determinar o nível de "maturidade" de seu processo de software. Entretanto, a qualidade, o cumprimento de prazos e a viabilidade em longo prazo do produto que se desenvolve são os melhores indicadores da eficácia do processo utilizado (PRESSMAN; MAXIM, 2016, p. 41).

Fases do ciclo de vida de um software

O processo pode ser definido como um conjunto de atividades de trabalho, ações e tarefas realizadas quando algum artefato de software deve ser criado. Cada uma dessas atividades, ações e tarefas se alocam dentro de uma metodologia ou um modelo que determina sua relação com o processo e umas com as outras. Cada atividade metodológica é composta por um conjunto de ações de Engenharia de Software. Cada ação é definida por um conjunto de tarefas, o qual identifica as tarefas de trabalho a serem completadas, os artefatos de software que serão produzidos, os fatores de garantia da qualidade que serão exigidos e os marcos utilizados para indicar progresso (PRESSMAN; MAXIM, 2016, p. 35).

Podemos relacionar a Engenharia de Software com um procedimento organizacional, em que o processo é a base para as técnicas trazidas pela Engenharia. Ela é composta por algumas camadas. Dentre elas, podemos destacar: as ferramentas, que fornecem suporte, muitas vezes por meio de software específico automatizando a solução dos processos; os métodos que

trazem as técnicas de desenvolvimento; o processo, o qual já definimos anteriormente; e o foco na qualidade, que demonstrará a eficiência do software diante das necessidades do cliente.

Segundo Sommerville (2011, p. 18), um processo de software é um conjunto de atividades relacionadas que levam à produção de um produto de software. Essas atividades podem envolver o desenvolvimento de software em uma linguagem de programação. No entanto, em razão da exigência do mercado, os sistemas passaram a ser desenvolvidos tendo como base extensões e modificações por meio de integrações entre componentes já existentes ou até mesmo criação de novos. O autor diz que, apesar de existirem diferentes processos de software, todos devem incluir quatro atividades fundamentais:

1. Especificação de software
2. Projeto e implementação de software
3. Validação de software
4. Evolução de software

Essas atividades possuem outras subatividades, tais como validação de requisitos, projeto de arquitetura, testes unitários, dentre outros. As descrições do processo, de acordo com Sommerville (2011, p. 19), podem incluir:

- Produtos, que são os resultados de uma das atividades do processo.
- Papéis, que refletem as responsabilidades das pessoas envolvidas no processo.
- Pré e pós-condições, que são declarações verdadeiras antes e depois de uma atividade do processo ou da produção de um produto.

Assim como o entendimento e a interpretação do modo representativo de um ser humano, que, neste caso, se configura como o papel de nosso cliente, um processo de software é algo bem complexo, tendo em vista que envolve ferramentas práticas para desenvolvimento de software e diversas tomadas de decisões que influenciarão diretamente no resultado final, ou seja, no software.

Uma metodologia de processo genérica para Engenharia de Software estabelece também cinco atividades metodológicas: comunicação, planejamento, modelagem, construção e entrega. Além disso, um conjunto de atividades de apoio é aplicado ao longo do processo, como o acompanhamento e o controle do projeto, a administração de riscos, a garantia da qualidade, o gerenciamento

das configurações, as revisões técnicas, entre outras. Um fluxo de processo, como apresentado na Figura 1 a seguir, descreve como são organizadas as atividades metodológicas, bem como as ações e as tarefas que ocorrem dentro de cada atividade em relação à sequência e ao tempo (SCHACH, 2010, p. 31).

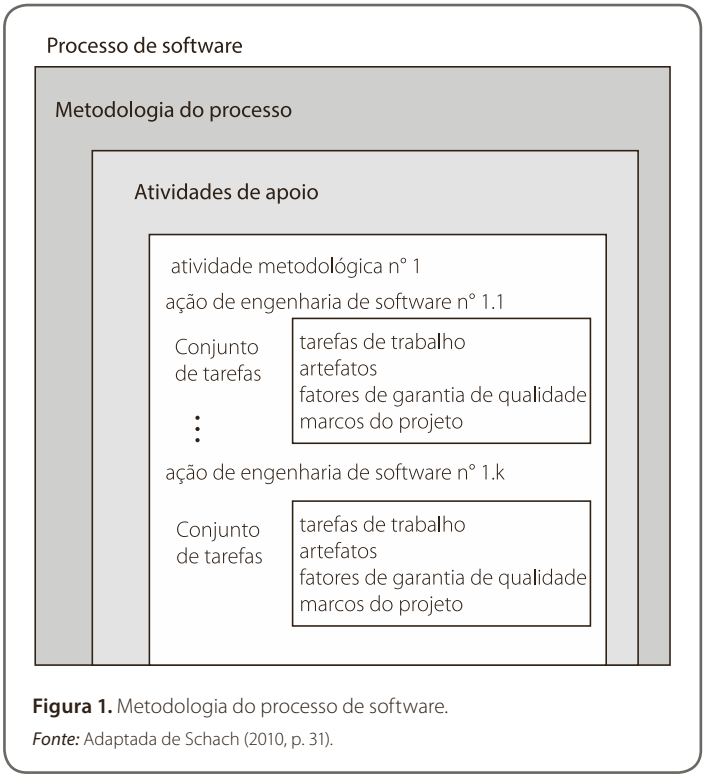


Figura 1. Metodologia do processo de software.
Fonte: Adaptada de Schach (2010, p. 31).

Devemos ter em mente que, apesar de termos todas essas informações, não temos obrigação nenhuma de utilizá-las da maneira que são expostas. Não existe um modelo ou uma técnica ideal, dita como correta, para todas as situações. A escolha dos processos e das métricas deverá ser realizada conforme o contexto do projeto.

Link

Em sua obra, Pressman e Maxim (2016, p. 41) indicam um "jogo de simulação de processos", que inclui os mais importantes modelos de processo prescritivos. Pode ser encontrada em:

https://goo.gl/1pmbqv

Caracterizar o funcionamento das fases modelo

As etapas comuns em todos os modelos de processo de software têm como função nortear as demais fases que vão sendo inseridas conforme os modelos de processo foram sendo criados. Sabemos que a era tecnológica está sempre em movimento, pois cada vez mais recursos estão sendo desenvolvidos. Citamos anteriormente quatro atividades fundamentais da Engenharia de Software diante dos processos de software: especificação, projeto e implementação, validação e evolução de software. Porém, outras atividades vistas como complementares são citadas por Schach (2010, p. 43):

1. Fase de levantamento de necessidades
2. Fase de análise (especificação)
3. Fase de projeto
   - Projeto de arquitetura (extrair os módulos)
   - Projeto detalhado
4. Fase de implementação
   - Codificar os módulos em uma linguagem de programação apropriada
   - Integrar
5. Manutenção pós-entrega
6. Retirada do produto

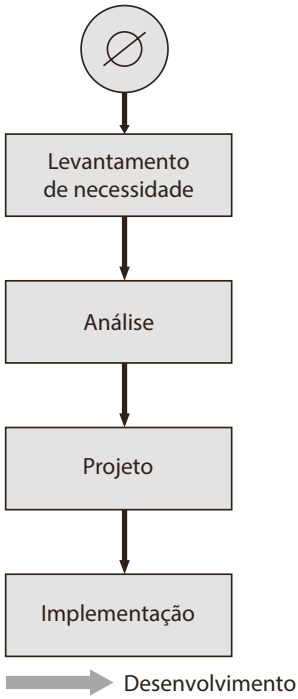


Figura 2. Desenvolvimento ideal de software.

Fonte: Pressman e Maxim (2016, p. 36).

Apesar de obviamente ter algumas etapas em comum, Schach (2010) traz algumas atividades complementares a cada uma das fases, como podemos visualizar na Figura 2. Porém, vamos debater um pouco sobre as principais, que são comuns a todos os modelos, seguindo as definições de Sommerville (2011):

1. Especificação de software: "[...] é o processo de compreensão e definição dos serviços requisitados do sistema e identificação de restrições relativas à operação e ao desenvolvimento do sistema." (SOMMERVILLE, 2011, p. 24). Os requisitos são bem importantes nesta fase, pois, por estar presente na fase inicial do desenvolvimento, todas as demais etapas irão depender dela. Algumas informações complementares podem ser visualizadas na Figura 3.

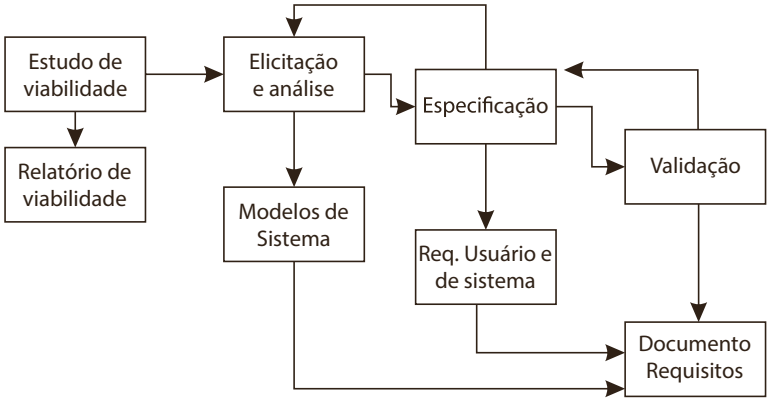


Figura 3. Requisitos da Engenharia de Processos.
Fonte: Sommerville (2011, p. 25).

2. Projeto e implementação de software: "[...] é o processo de conversão de uma especificação do sistema em um sistema executável. Sempre envolve processos de projeto e programação de software, mas, se for usada uma abordagem incremental para o desenvolvimento, também pode envolver o refinamento da especificação do software." (SOMMERVILLE, 2011, p. 25). Após essa definição dada por Sommerville (2011), já começa a ficar mais claro que uma etapa depende, de alguma maneira, da sua etapa anterior. Como já dito, pode haver outras etapas adicionais em cada uma delas, o que chamamos de subatividades, porém, as principais sempre serão utilizadas. Podemos visualizá-las na Figura 4.

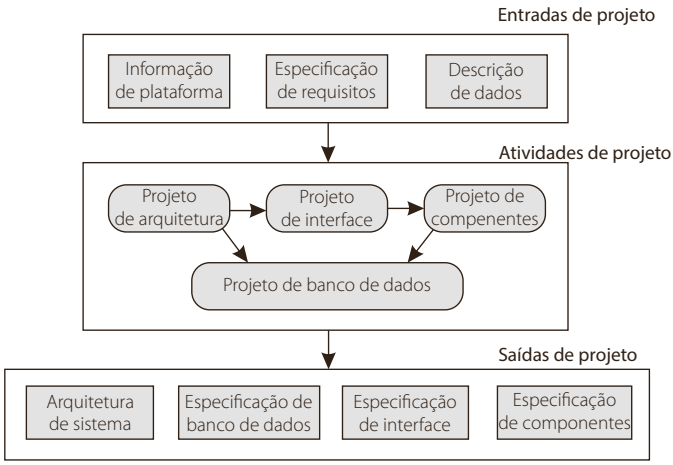


Figura 4. Um modelo geral do processo de projeto.
Fonte: Sommerville (2011, p. 26).

3. Validação de software: É uma fase conhecida também como de verificação e validação(V&V), "[...] tem a intenção de mostrar que um software se adequa a suas especificações ao mesmo tempo em que satisfaz as especificações do cliente do sistema.[...] A validação também pode envolver processos de verificação, como inspeções e revisões, em cada estágio do processo de software, desde a definição dos requisitos de usuários até o desenvolvimento do programa." (SOMMERVILLE, 2011, p. 27). A Figura 5 traz os estágios do processo de teste.

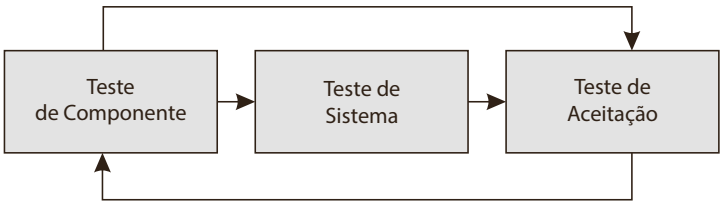


Figura 5. Estágio de processo de teste.
Fonte: Adaptada de Sommerville (2011, p. 27).

4. Evolução do software: "Historicamente, sempre houve uma separação entre o processo de desenvolvimento e o de evolução do software, ou seja, a manutenção de software. [...] Poucos sistemas de software são completamente novos, e faz muito mais sentido ver o desenvolvimento e a manutenção como processos contínuos." (SOMMERVILLE, 2011, p. 29).

As mudanças são inevitáveis. O que podemos aprender diante dos conceitos vistos é que temos essas ferramentas para auxiliar a equipe desenvolvedora a produzir o software com qualidade no processo e, por fim, no sistema final.

Referências

PRESSMAN, R. S.; MAXIM, B. R. *Engenharia de software*: uma abordagem profissional. 8. ed. Porto Alegre: AMGH, 2016.

SCHACH, S. R. *Engenharia de software*: os paradigmas clássicos e orientado a objetos. 7. ed. Porto Alegre: AMGH, 2010.

SOMMERVILLE, I. *Engenharia de software*. 9. ed. São Paulo: Pearson, 2011.

Leituras recomendadas

OLIVEIRA. R. C. *Engenharia de software*. 2011. Disponível em: <http://www.facom.ufu.br/~ronaldooliveira/ESOF-2011-2/Aula8-ESOF-AnaliseEstruturada.pdf>. Acesso em: 13 ago. 2017.

SLACK, N. et al. *Gerenciamento de operações e de processos*: princípios e práticas de impacto estratégico. 2. ed. Porto Alegre: Bookman, 2013.

YOURDON, E.; CONSTANTINE, L. L. *Structured design: fundamentals of a discipline of computer program and systems design*. Englewood Cliffs: Prentice-Hall, 1979.

Encerra aqui o trecho do livro disponibilizado para esta Unidade de Aprendizagem. Na Biblioteca Virtual da Instituição, você encontra a obra na íntegra.

Conteúdo:

SAGAH
SOLUÇÕES EDUCACIONAIS INTEGRADAS

Dica do Professor

Nesta dica do professor iremos introduzir o ciclo de vida de *software* e apresentar dois modelos clássicos.

Acompanhe!

Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.

Exercícios

1) **O que é um ciclo de vida de *software*?**

A) Ciclo de vida de *software* refere-se aos estágios de concepção, projeto, criação e implementação de um *software*.

B) Ciclo de vida de *software* refere-se aos estágios de levantamento de requisitos.

C) Ciclo de vida de *software* refere-se ao tempo de implementação estimado pelo analista.

D) Ciclo de vida de *software* refere-se aos estágios de análise do *software*.

E) Ciclo de vida de *software* ocorreu antes da Crise do *Software*, em 1970.

2) **Em qual fase do ciclo de vida de *software* são definidas as questões técnicas, como banco de dados, localização, *hardware* e linguagens de programação?**

A) Na fase de projeto.

B) Na fase de levantamento de requisitos.

C) Na fase de implementação.

D) Na fase de testes e manutenção.

E) Em nenhuma fase, estas questões são decididas pelo programador.

3) **No ciclo de vida de *software*, o que é realizado na etapa de "levantamento das necessidades"?**

A) É realizada a implementação do sistema.

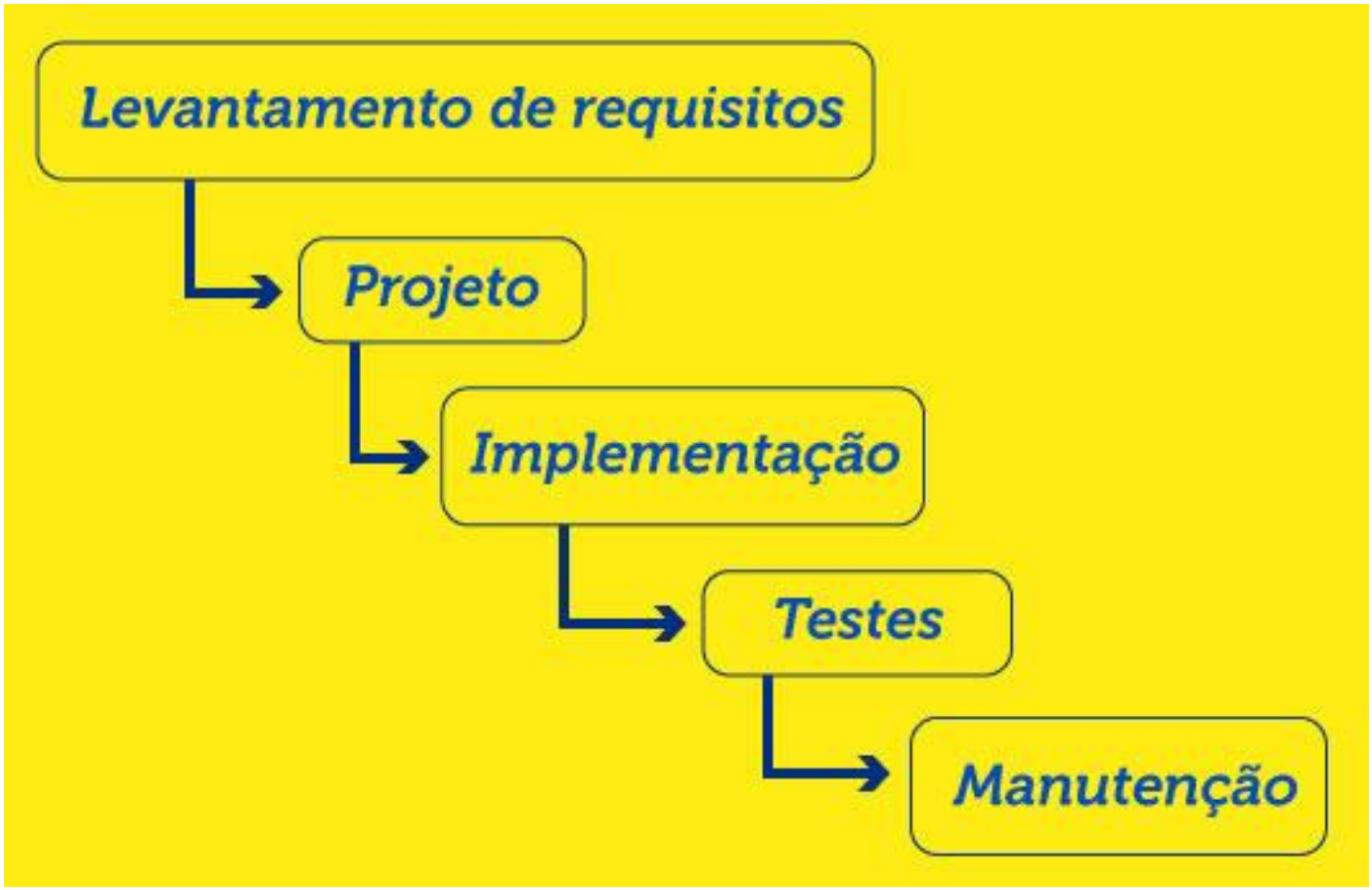
B) É realizada uma verificação de todas as necessidades do cliente.

C) É realizada a análise de requisitos.

D) São realizados testes no sistema para verificar quais as necessidades de implementação.

E) É a etapa onde o sistema é entregue para o usuário/cliente.

4) A figura ilustra um modelo de desenvolvimento de *software* no qual o fluxo é visto como um fluir constante através das fases. Esse modelo utiliza como entrada as informações obtidas nas fases anteriores e cada fase só inicia após o término da que antecede (não existindo fases em paralelo). Com base nessas informações, qual é o modelo apresentado na figura?



A) Modelo V.

B) Cascata.

C) Espiral.

D) Prototipagem.

E) Incremental.

5) Qual é o maior problema encontrado no modelo cascata?

A) Nenhum. O sistema cascata foi utilizado durante anos e até o momento não precisou de correções.

B) É um modelo bastante simples.

C) O sistema prevê a revisão das fases e é totalmente iterativo.

D) Apresenta o problema de reatividade a mudanças.

E) O modelo cascata é top-down e isso faz com que o *software* seja construído de maneira incorreta.

Na prática

Os modelos de ciclo de vida são muito importantes para a criação de sistemas de alta qualidade e que resolvam os problemas do cliente/usuário. Definir adequadamente o ciclo de vida no início do projeto pode ser o que levará o projeto ao sucesso pleno ou ao total fracasso.

A empresa XXYZ irá desenvolver um app e um site para uma ONG focada em atender pessoas com deficiência visual e auditiva. Até o momento, você conheceu o modelo Cascata e o modelo V. Porém, para os materiais direcionados ao público desta ONG, qual seria o modelo ideal?

Conteúdo interativo disponível na plataforma de ensino!

Devido as inúmeras interações com os usuários no início do projeto, o modelo evita que seja entregue um *software* ao final que não atenda as expectativas. A escolha da prototipagem é muito interessante em situações como a da empresa XXYZ, em que o design, arquitetura e a maioria dos componentes do sistema dependem bastante da usabilidade e interface. Caso a empresa tivesse utilizado outros modelos, após a criação de todo o *software*, os usuários poderiam ter dificuldades no uso, precisando reimplementar boa parte do *software*.

Saiba mais

Para ampliar o seu conhecimento a respeito desse assunto, veja abaixo as sugestões do professor:

O modelo Cascata, que foi uma das primeiras iniciativas para solucionar a crise de Software, é ilustrado na animação seguir. Acompanhe!

Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.

Amplie seus conhecimentos sobre os fundamentos teóricos da engenharia de software, bem como sobre o ciclo de vida de um software, a partir da próxima dica de leitura.

Conteúdo interativo disponível na plataforma de ensino!