Estratégia
CONCURSOS

Aul

Engenharia de Software p/ TCU (Auditor de TI) Com Videoaulas - 2019.2

Professor: Diego Carvalho, Equipe Informática e TI

SUMÁRIO	PÁGINA

ENGENHARIA DE SOFTWARE

Vamos lá, galera! Apesar de hoje em dia haver milhões de profissionais que mexem com software no mundo inteiro, faz pouco tempo que a Engenharia de Software alcançou o status de profissão reconhecida e de disciplina legítima de engenharia. Pois é, ela ganhou tanta importância que é cobrada até em concursos públicos! *Ok, professor... mas o que é a Engenharia de Software?*

A IEEE define engenharia de software como a aplicação de uma abordagem sistemática, disciplinada e quantificável de desenvolvimento, operação e manutenção de software. Já Friedrich Bauer conceitua como a criação e a utilização de sólidos princípios de engenharia a fim de obter software de maneira econômica, que seja confiável e que trabalhe em máquinas reais.

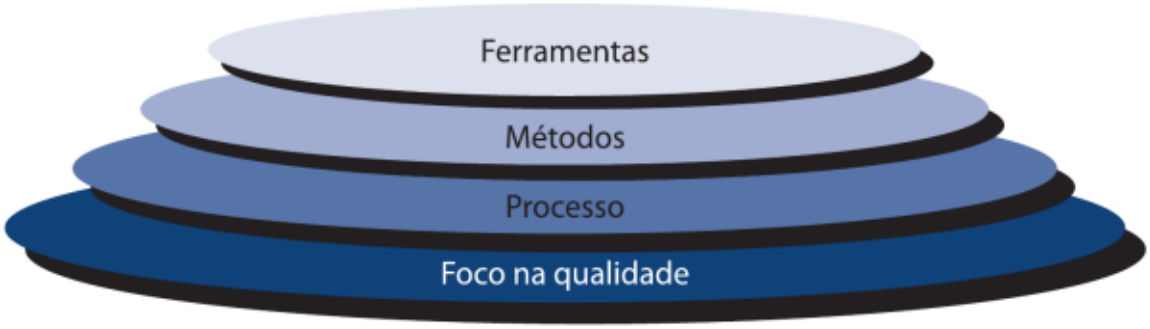
Em suma, é uma disciplina de engenharia que se ocupa de todos os aspectos da produção de software, desde os estágios iniciais de especificação do sistema até a manutenção desse sistema, após sua entrada em produção A meta principal da Engenharia de Software é desenvolver sistemas de software com boa relação custo-benefício. *Bacana?*

Aliás, vamos voltar um pouquinho: *o que seria um software?* Bem, em uma visão restritiva, muitas pessoas costumam associar o termo software aos programas de computador. Software não é apenas o programa, mas também todos os dados de documentação e configuração associados, necessários para que o programa opere corretamente. Vamos prosseguir...

De acordo com Pressman: "*A Engenharia de Software ocorre como consequência de um processo chamado Engenharia de Sistemas. Em vez de se concentrar somente no software, a engenharia de sistemas focaliza diversos elementos, analisando, projetando, e os organizando em um sistema que pode ser um produto, um serviço ou uma tecnologia para transformação da informação ou controle*".

Ademais, nosso renomadíssimo autor afirma que a engenharia de sistemas está preocupada com todos os aspectos do desenvolvimento de sistemas computacionais, incluindo engenharia de hardware, engenharia de software e engenharia de processos. Percebam, então, que a Engenharia de Sistemas está em um contexto com várias outras engenharias. *Entenderam direitinho?*

A Engenharia de Software tem por objetivos a aplicação de teoria, modelos, formalismos, técnicas e ferramentas da ciência da computação e áreas afins para a desenvolvimento sistemático de software. Associado ao desenvolvimento, é preciso também aplicar processos, métodos e ferramentas sendo que a pedra fundamental que sustenta a engenharia de software é o foco na qualidade.



Isto envolve planejamento de custos e prazos, montagem da equipe e garantia de qualidade do produto e do processo. Finalmente, a engenharia de software visa a produção da documentação formal do produto, do processo, dos critérios de qualidade e dos manuais de usuários finais. Todos esses aspectos devem ser levados em consideração.

Aliás, nosso outro renomadíssimo autor (Sommerville) afirma que: *"A engenharia de software não está relacionada apenas com os processos técnicos de desenvolvimento de software, mas também com atividades como o gerenciamento de projeto de software e o desenvolvimento de ferramentas, métodos e teorias que apoiem a produção de software"*.

A Engenharia de Software surgiu em meados da década de sessenta como uma tentativa de contornar a crise do software e dar um tratamento de engenharia ao desenvolvimento de software completo. Naquela época, o processo de desenvolvimento era completamente fora de controle e tinha grandes dificuldades em entregar o que era requisitado pelo cliente.

Já na década de oitenta, surgiu a Análise Estruturada e algumas Ferramentas CASE que permitiam automatizar algumas tarefas. Na década de noventa, surgiu a orientação a objetos, linguagens visuais, processo unificado, entre outros. E na última década, surgiram as metodologias ágeis e diversos paradigmas de desenvolvimento.

A Engenharia de Software possui alguns princípios, tais como: Formalidade, em que o software deve ser desenvolvido de acordo com passos definidos com precisão e seguidos de maneira efetiva; Abstração, preocupa-se com a identificação de um determinado fenômeno da realidade, sem se preocupar com detalhes, considerando apenas os aspectos mais relevantes.

Há a Decomposição, em que se divide o problema em partes, de maneira que cada uma possa ser resolvida de uma forma mais específica; Generalização, maneira usada para resolver um problema, de forma genérica, com o intuito de reaproveitar essa solução em outras situações; Flexibilização é o processo que permite que o software possa ser alterado, sem causar problemas para sua execução.

Certa vez, um aluno me perguntou: *professor, como a formalidade pode reduzir inconsistências?* Cenário 1: Estamos na fase de testes de software. O testador afirma que fez todos os testes, você - líder de projeto - acredita, e passa o software ao cliente. Esse tenta utilizar o software e verifica que ele não está funcionando corretamente. *Bacana?*

Cenário 2: Estamos na fase de testes de software. O testador afirma que fez todos os testes e entrega um documento de testes com tudo que foi verificado formalmente. Você - líder de projeto - lê o documento de testes e verifica que não foram feitos testes de carga e testes de segurança. Retorna para o testador e pede para ele refazer os testes.

Feito isso, ele passa o software ao cliente, que fica feliz e satisfeito com tudo funcionando corretamente. *Vocês percebem que essas formalidades evitam aquele "telefone-sem-fio"?* Quanto mais eu seguir o processo, o passo-a-passo, o que foi definido por várias pessoas a partir de suas experiências com vários projetos, mais eu tenho chance de obter êxito na construção do meu software. *Bacana?*

ESQUEMA

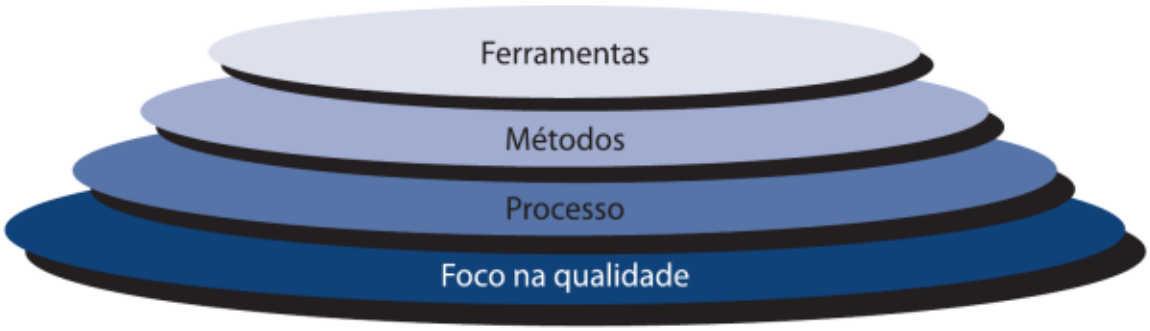

Disciplina que se ocupa de todos os aspectos da produção de software, desde os estágios iniciais de especificação do sistema até a manutenção desse sistema, após sua entrada em produção – passando por aspectos humanos, hardware, etc.
Princípios: Decomposição; Generalização; Flexibilização; Formalidade; Abstração
Eng. de Software
Ferramentas
Métodos
Processo
Foco na qualidade

CAIU na prova!
DIVERSAS BANCAS

1. (CESPE - 2013 - TRT - 10ª REGIÃO (DF e TO) - Analista Judiciário - Tecnologia da Informação) A engenharia de software engloba processos, métodos e ferramentas. Um de seus focos é a produção de software de alta qualidade a custos adequados.

Comentários:

A Engenharia de Software tem por objetivos a aplicação de teoria, modelos, formalismos, técnicas e ferramentas da ciência da computação e áreas afins para a desenvolvimento sistemático de software. Associado ao desenvolvimento, é preciso também aplicar processos, métodos e ferramentas sendo que a pedra fundamental que sustenta a engenharia de software é a qualidade.



Basta visualizar a imagem para responder à questão! Ela não menciona foco na qualidade, mas não restringe também somente a processos, métodos e ferramentas.

Gabarito: C

---

2. (FCC - 2012 - TST - Analista Judiciário - Análise de Sistemas) A Engenharia de Software:

a) é uma área da computação que visa abordar de modo sistemático as questões técnicas e não técnicas no projeto, implantação, operação e manutenção no desenvolvimento de um software.

b) consiste em uma disciplina da computação que aborda assuntos relacionados a técnicas para a otimização de algoritmos e elaboração de ambientes de desenvolvimento.

c) trata-se de um ramo da TI que discute os aspectos técnicos e empíricos nos processos de desenvolvimento de sistemas, tal como a definição de artefatos para a modelagem ágil.

d) envolve um conjunto de itens que abordam os aspectos de análise de mercado, concepção e projeto de software, sendo independente da engenharia de um sistema.

e) agrupa as melhores práticas para o concepção, projeto, operação e manutenção de artefatos que suportam a execução de programas de computador, tais como as técnicas de armazenamento e as estruturas em memória principal.

Comentários:

> *De acordo com Pressman: "A Engenharia de Software ocorre como consequência de um processo chamado Engenharia de Sistemas. Em vez de se concentrar somente no software, a engenharia de sistemas focaliza diversos elementos, analisando, projetando, e os organizando em um sistema que pode ser um produto, um serviço ou uma tecnologia para transformação da informação ou controle".*

(a) Perfeito, observem as palavras-chave: modo sistemático; questões técnicas e não técnicas; projeto, implantação, operação e manutenção de desenvolvimento de software. (b) *Técnicas para otimização de algoritmos e elaboração de ambientes de desenvolvimento?* Não, isso não é Engenharia de Software. (c) Pessoal, discordo do gabarito! Certa vez, um aluno me disse que talvez fosse porque aspectos empíricos são mais voltados para metodologias ágeis. Sim, é verdade! No entanto, a engenharia de software trata também de metodologias ágeis. Se alguém encontrar o erro, avise :-] (d) A Análise de Mercado serve mais como uma técnica para Análise de Interfaces, mas pode ser vista como um dos aspectos que envolvem a Engenharia de Software. Pressman afirma que: *"A Análise de Mercado pode ser inestimável na definição de segmentos de mercado e no entendimento de como cada segmento poderia usar o software de modos sutilmente diferentes"*. De todo modo, a questão está errada porque a Engenharia de Software depende da Engenharia de Sistema

(como é mostrado acima). (e) *Suportam a execução?* Não, suportam o desenvolvimento de programas de computador.

Gabarito: A

---

3. (FCC - 2012 - TRT - 6ª Região (PE) - Técnico Judiciário - Tecnologia da Informação) Considere: *é uma disciplina que se ocupa de todos os aspectos da produção de software, desde os estágios iniciais de especificação do sistema até a manutenção desse sistema, depois que ele entrou em operação. Seu principal objetivo é fornecer uma estrutura metodológica para a construção de software com alta qualidade.* A definição refere-se:

a) ao ciclo de vida do software.
b) à programação orientada a objetos.
c) à análise de sistemas.
d) à engenharia de requisitos.
e) à engenharia de software.

Comentários:

Em suma, é uma disciplina de engenharia que se ocupa de todos os aspectos da produção de software, desde os estágios iniciais de especificação do sistema até a manutenção desse sistema, após sua entrada em produção. A meta principal da Engenharia de Software é desenvolver sistemas de software com boa relação custo-benefício. Bacana?

Conforme vimos em aula, essa é a pura definição de Engenharia de Software!

Gabarito: E

---

4. (CESPE - 2011 - MEC - Gerente de Projetos) A engenharia de software, disciplina relacionada aos aspectos da produção de software, abrange somente os processos técnicos do desenvolvimento de software.

Comentários:

Aliás, nosso outro renomadíssimo autor (Sommerville) afirma que: "A engenharia de software não está relacionada apenas com os processos técnicos de desenvolvimento de software, mas também com atividades como o gerenciamento de projeto de

software e o desenvolvimento de ferramentas, métodos e teorias que apoiem a produção de software".

Conforme vimos em aula, não são apenas processos técnicos!

Gabarito: E

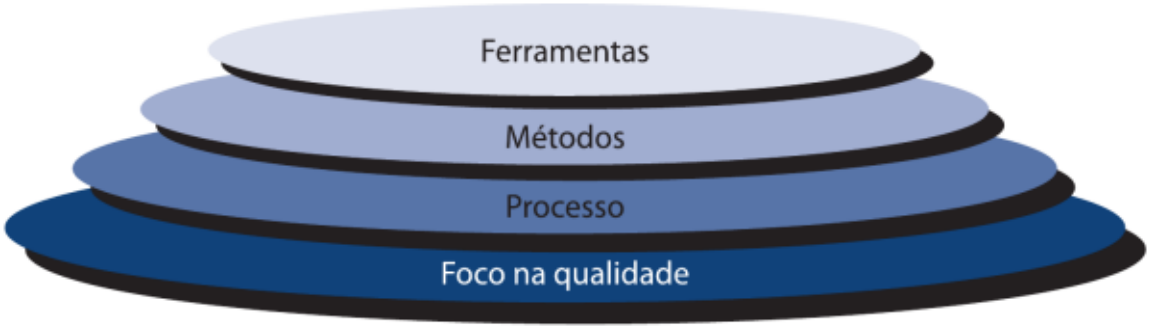
---

5. **(FGV - 2010 - BADESC - Analista de Sistemas - Desenvolvimento de Sistemas)** De acordo com Pressman, a engenharia de software é baseada em camadas, com foco na qualidade.

   Essas camadas são:

   a) métodos, processo e teste.
   b) ferramentas, métodos e processo.
   c) métodos, construção, teste e implantação.
   d) planejamento, modelagem, construção, validação e implantação.
   e) comunicação, planejamento, modelagem, construção e implantação.

Comentários:



Conforme vimos em aula, bastava visualizar a imagem para responder à questão!

Gabarito: B

---

6. **(CESPE - 2010 - Banco da Amazônia - Técnico Científico - Tecnologia da Informação)** Os princípios de engenharia de software definem a necessidade de formalidades para reduzir inconsistências e a decomposição para lidar com a complexidade.

Comentários:

A Engenharia de Software possui alguns princípios, tais como: Formalidade, em que o software deve ser desenvolvido de acordo com passos definidos com precisão e seguidos de maneira efetiva; Há a Decomposição, em que se divide o problema em partes, de maneira que cada uma possa ser resolvida de uma forma mais específica;

Conforme vimos em aula, são princípios: Formalidade e Decomposição.

Gabarito: C

---

7. (FCC - 2010 - TRE-AM - Analista Judiciário - Tecnologia da Informação) A Engenharia de Software:

a) não tem como método a abordagem estruturada para o desenvolvimento de software, pois baseia-se exclusivamente nos modelos de software, notações, regras e técnicas de desenvolvimento.

b) se confunde com a Ciência da Computação quando ambas tratam do desenvolvimento de teorias, fundamentações e práticas de desenvolvimento de software.

c) tendo como foco apenas o tratamento dos aspectos de construção de software, subsidia a Engenharia de Sistemas no tratamento dos sistemas baseados em computadores, incluindo hardware e software.

d) tem como foco principal estabelecer uma abordagem sistemática de desenvolvimento, através de ferramentas e técnicas apropriadas, dependendo do problema a ser abordado, considerando restrições e recursos disponíveis.

e) segue princípios, tais como, o da Abstração, que identifica os aspectos importantes sem ignorar os detalhes e o da Composição, que agrupa as atividades em um único processo para distribuição aos especialistas.

Comentários:

A IEEE define engenharia de software como a aplicação de uma abordagem sistemática, disciplinada e quantificável de desenvolvimento, operação e manutenção de software. Já Friedrich Bauer conceitua como a criação e a utilização de sólidos

princípios de engenharia a fim de obter software de maneira econômica, que seja confiável e que trabalhe em máquinas reais.

(a) Pelo contrário, ela se baseia em uma abordagem estruturada e sistemática!

A Engenharia de Software tem por objetivos a aplicação de teoria, modelos, formalismos, técnicas e ferramentas da ciência da computação e áreas afins para a desenvolvimento sistemático de software. Associado ao desenvolvimento, é preciso também aplicar processos, métodos e ferramentas sendo que a pedra fundamental que sustenta a engenharia de software é a qualidade.

(b) Na verdade, Engenharia de Software é uma disciplina da Ciência da Computação.

(c) *Apenas o tratamento dos aspectos de construção de software? Só construção?* Não!

(d) Perfeito, é exatamente isso!

Há a Decomposição, em que se divide o problema em partes, de maneira que cada uma possa ser resolvida de uma forma mais específica; Generalização, maneira usada para resolver um problema, de forma genérica, com o intuito de reaproveitar essa solução em outras situações; Flexibilização é o processo que permite que o software possa ser alterado, sem causar problemas para sua execução.

A Engenharia de Software possui alguns princípios, tais como: Formalidade, em que o software deve ser desenvolvido de acordo com passos definidos com precisão e seguidos de maneira efetiva; Abstração, preocupa-se com a identificação de um determinado fenômeno da realidade, sem se preocupar com detalhes, considerando apenas os aspectos mais relevantes.

(e) *Composição?* Não, Decomposição! Divide-se o problema em partes para que cada uma possa ser resolvida de uma forma mais específica. Além disso, a abstração ignora detalhes!

Gabarito: D

---

8. **(FCC - 2011 - INFRAERO - Analista de Sistemas - Gestão de TI)** Em relação à Engenharia de Software, é INCORRETO afirmar:

a) O design de software, ao descrever os diversos aspectos que estarão presentes no sistema quando construído, permite que se faça a avaliação prévia para garantir que ele alcance os objetivos propostos pelos interessados.

b) A representação de um design de software mais simples para representar apenas as suas características essenciais busca atender ao princípio da abstração.

c) Iniciar a entrevista para obtenção dos requisitos de software com perguntas mais genéricas e finalizar com perguntas mais específicas sobre o sistema é o que caracteriza a técnica de entrevista estruturada em funil.

d) No contexto de levantamento de requisitos, funcionalidade é um dos aspectos que deve ser levado em conta na abordagem dos requisitos funcionais.

e) A representação é a linguagem do design, cujo único propósito é descrever um sistema de software que seja possível construir.

Comentários:

Galera, descrever um sistema de software que seja possível construir não é o único, mas um dos objetivos da representação. Ela auxilia a comunicação entre as partes interessadas e serve também como documentação.

Gabarito: E

---

9. **(FCC – 2009 – AFR/SP - Analista de Sistemas)** A engenharia de software está inserida no contexto:

a) das engenharias de sistemas, de processo e de produto.
b) da engenharia de sistemas, apenas.
c) das engenharias de processo e de produto, apenas.
d) das engenharias de sistemas e de processo, apenas.
e) das engenharias de sistemas e de produto, apenas.

Comentários:

> *Ademais, nosso renomadíssimo autor afirma que a engenharia de sistemas está preocupada com todos os aspectos do desenvolvimento de sistemas computacionais, incluindo engenharia de hardware, engenharia de software e engenharia de*

processos. Percebam, então, que a Engenharia de Sistemas está em um contexto com várias outras engenharias. Entenderam direitinho?

Observem que a Engenharia de Software está em um contexto em que há também Engenharia de Sistemas, de Processo e de Produto.

Gabarito: A

---

10. (CESPE – 2015 – STJ – Analista de Sistemas) Embora os engenheiros de software geralmente utilizem uma abordagem sistemática, a abordagem criativa e menos formal pode ser eficiente em algumas circunstâncias, como, por exemplo, para o desenvolvimento de sistemas web, que requerem uma mistura de habilidades de software e de projeto.

Comentários:

Galera, basta pensar nas metodologias ágeis! Elas são menos formais e são mais adequadas em determinadas circunstâncias. Notem que isso não vai de encontro ao princípio da formalidade, na medida em que a questão deixa claro que se trata de uma abordagem "*menos formal*" e, não, "*informal*". Além disso, isso consta também do livro do Sommerville:

"No entanto, engenharia tem tudo a ver com selecionar o método mais adequado para um conjunto de circunstâncias, então uma abordagem mais criativa e menos formal pode ser eficiente em algumas circunstâncias. Desenvolvimento menos formal particularmente adequado para o desenvolvimento de sistemas Web, que requerem uma mistura de habilidades de software e de projeto".

Gabarito: C

---

11. (CESPE – 2015 – STJ – Analista de Sistemas) O foco da engenharia de software inclui especificação do sistema, desenvolvimento de hardware, elaboração do projeto de componentes de hardware e software, definição dos processos e implantação do sistema.

Comentários:

Conforme vimos em aula, pode até tratar eventualmente de alguns aspectos de hardware; mas desenvolvimento de hardware, não.

Gabarito: E

12. (FUNIVERSA – 2009 – IPHAN – Analista de Sistemas) A Engenharia de Software resume-se em um conjunto de técnicas utilizadas para o desenvolvimento e manutenção de sistemas computadorizados, visando produzir e manter softwares de forma padronizada e com qualidade. Ela obedece a alguns princípios como (1) Formalidade, (2) Abstração, (3) Decomposição, (4) Generalização e (5) Flexibilização. Assinale a alternativa que apresenta conceito correto sobre os princípios da Engenharia de Software.

a) A flexibilização é o processo que permite que o software possa ser alterado, sem causar problemas para sua execução.

b) A formalidade é a maneira usada para resolver um problema, de forma genérica, com o intuito de poder reaproveitar essa solução em outras situações semelhantes.

c) A generalização preocupa-se com a identificação de um determinado fenômeno da realidade, sem se preocupar com detalhes, considerando apenas os aspectos mais relevantes.

d) Pelo princípio da decomposição, o software deve ser desenvolvido de acordo com passos definidos com precisão e seguidos de maneira efetiva.

e) A abstração é a técnica de se dividir o problema em partes, de maneira que cada uma possa ser resolvida de uma forma mais específica.

Comentários:

A Engenharia de Software possui alguns princípios, tais como: Formalidade, em que o software deve ser desenvolvido de acordo com passos definidos com precisão e seguidos de maneira efetiva; Abstração, preocupa-se com a identificação de um determinado fenômeno da realidade, sem se preocupar com detalhes, considerando apenas os aspectos mais relevantes.

Há a Decomposição, em que se divide o problema em partes, de maneira que cada uma possa ser resolvida de uma forma mais específica; Generalização, maneira usada para resolver um problema, de forma genérica, com o intuito de reaproveitar essa solução em outras situações; Flexibilização é o processo que permite que o software possa ser alterado, sem causar problemas para sua execução.

Conforme vimos em aula, a flexibilização é o processo que permite que o software possa ser alterado, sem causar problemas para sua execução.

Gabarito: A

---

13. (CESPE – 2013 – TCE/RO – Analista de Sistemas) Engenharia de software não está relacionada somente aos processos técnicos de desenvolvimento de softwares, mas também a atividades como gerenciamento de projeto e desenvolvimento de ferramentas, métodos e teorias que apoiem a produção de softwares.

Comentários:

> *Aliás, nosso outro renomadíssimo autor (Sommerville) afirma que: "A engenharia de software não está relacionada apenas com os processos técnicos de desenvolvimento de software, mas também com atividades como o gerenciamento de projeto de software e o desenvolvimento de ferramentas, métodos e teorias que apoiem a produção de software".*

Conforme vimos em aula, a questão está perfeita!

Gabarito: C

---

14. (CESPE – 2010 – DETRAN/ES – Analista de Sistemas) Segundo princípio da engenharia de software, os vários artefatos produzidos ao longo do seu ciclo de vida apresentam, de forma geral, nível de abstração cada vez menor.

Comentários:

Galera, vamos pensar um pouco! A abstração é a subtração de detalhes – focar-se no que é mais relevante e ignorar detalhes menos relevantes. No início do ciclo de vida, os artefatos são bastante abstratos. Temos, por exemplo, uma modelagem do negócio, um documento de requisitos funcionais, a abstração vai diminuindo e temos a modelagem visual de análise, depois a modelagem do projeto, até chegar ao código-fonte e ao executável do software. Nesse ponto, temos o menor nível de abstração (com muitos detalhes!). *Compreenderam?*

Gabarito: C

---

15. (CESPE – 2010 – TRE/BA – Analista de Sistemas) Entre os desafios enfrentados pela engenharia de software estão lidar com sistemas legados, atender à crescente diversidade e atender às exigências quanto a prazos de entrega reduzidos.

Comentários:

Bem, essa questão é bastante intuitiva. De fato, a engenharia de software tem que lidar com sistemas legados, atender à crescente diversidade e atender às exigências quanto aos (curtos) prazos de entrega.

Gabarito: C

---

16. (FCC – 2010 – DPE/SP – Analista de Sistemas) A Engenharia de Software

I. não visa o desenvolvimento de teorias e fundamentações, preocupando-se unicamente com as práticas de desenvolvimento de software.

II. tem como foco o tratamento dos aspectos de desenvolvimento de software, abstraindo-se dos sistemas baseados em computadores, incluindo hardware e software.

III. tem como métodos as abordagens estruturadas para o desenvolvimento de software que incluem os modelos de software, notações, regras e maneiras de desenvolvimento.

IV. segue princípios, tais como, o da Abstração, que identifica os aspectos importantes sem ignorar os detalhes e o da Composição, que agrupa as atividades em um único processo para distribuição aos especialistas.

É correto o que se afirma em:

a) III e IV, apenas.
b) I, II, III e IV.
c) I e II, apenas.
d) I, II e III, apenas.
e) II, III e IV, apenas.

Comentários:

(I) Errado, Sommerville diz: "*Computer science focuses on theory and fundamentals; software engineering is concerned with the practicalities of developing and delivering useful software*". No entanto, ele não diz que a engenharia de software se preocupa unicamente com as práticas de desenvolvimento de software.

(II) Errado, Pressman diz: "*System engineering is concerned with all aspects of computer-based systems development including hardware, software, and process engineering*" – a questão trata da Engenharia de Sistemas.

(III) Correto, de fato ela tem como métodos as abordagens estruturadas para o desenvolvimento de software que incluem os modelos de software, notações, regras e maneiras de desenvolvimento.

(IV) Errado, o princípio da abstração ignora os detalhes; e o princípio da composição não existe – o que existe é o princípio da decomposição. E ele divide o problema em partes menores.

Em suma, nenhuma das opções nos atende! *Vocês sabem qual opção a banca marcou como correta?* A Letra D!!! *E ela voltou atrás com os recursos?* Não!!! Pois é, galera! Acostumem-se com isso :(

Gabarito: D

---

17. (CESPE – 2013 – TCE/RO – Analista de Sistemas) Assim como a Engenharia de Software, existe também na área de informática a chamada Ciência da Computação. Assinale a alternativa que melhor apresenta a diferença entre Engenharia de Software e Ciência da Computação.

a) A Ciência da Computação tem como objetivo o desenvolvimento de teorias e fundamentações. Já a Engenharia de Software se preocupa com as práticas de desenvolvimento de software.

b) A Engenharia de Software trata da criação dos sistemas de computação (softwares) enquanto a Ciência da Computação está ligada ao desenvolvimento e criação de componentes de hardware.

c) A Engenharia de Software trata dos sistemas com base em computadores, que inclui hardware e software, e a Ciência da Computação trata apenas dos aspectos de desenvolvimento de sistemas.

d) A Ciência da Computação trata dos sistemas com base em computadores, que inclui hardware e software, e a Engenharia de Software trata apenas dos aspectos de desenvolvimento de sistemas.

e) A Ciência da Computação destina-se ao estudo e solução para problemas genéricos das áreas de rede e banco de dados e a Engenharia de Software restringe- se ao desenvolvimento de sistemas.

Comentários:

A Engenharia de Software tem por objetivos a aplicação de teoria, modelos, formalismos, técnicas e ferramentas da ciência da computação e áreas afins para a desenvolvimento sistemático de software. Associado ao desenvolvimento, é preciso também aplicar processos, métodos e ferramentas sendo que a pedra fundamental que sustenta a engenharia de software é a qualidade.

Essa questão parece contradizer o que diz Sommerville, mas observem que não! A Engenharia de Software coloca em prática a teoria e fundamentação trazida pela Ciência da Computação. A Engenharia de Software aplica a teoria, mas - grosso modo - ela não tem a finalidade principal de elaborá-la; a finalidade principal é de colocá-la em prática. Já a Ciência da Computação é exatamente o contrário. Enfim, a primeira opção foi retirada literalmente do Sommerville (Pág. 5, 8ª Ed.).

Gabarito: A

---

18. (CESPE – 2009 – ANAC – Analista de Sistemas) O termo engenharia pretende indicar que o desenvolvimento de software submete-se a leis similares às que governam a manufatura de produtos industriais em engenharias tradicionais, pois ambos são metodológicos.

Comentários:

Perfeito! A Engenharia busca os princípios mais consolidados de outras engenharias mais tradicionais – engenharia mecânica, civil, elétrica, etc.

Gabarito: C

---

19. (CESPE - 2016 – TCE/PR – Analista de Sistemas – B) A engenharia de software refere-se ao estudo das teorias e fundamentos da computação, ficando o desenvolvimento de software a cargo da ciência da computação.

Comentários:

A Engenharia de Software tem por objetivos a aplicação de teoria, modelos, formalismos, técnicas e ferramentas da ciência da computação e áreas afins para a desenvolvimento sistemático de software. Associado ao desenvolvimento, é preciso também aplicar processos, métodos e ferramentas sendo que a pedra fundamental que sustenta a engenharia de software é a qualidade.

Conforme vimos em aula, não se trata do estudo – trata-se da aplicação. Além disso, o desenvolvimento também fica a cargo da engenharia de software. Em geral, a ciência da computação trata da teoria e a engenharia de software trata da prática.

Gabarito: E

---

20. (CESPE - 2016 – TCE/PR – Analista de Sistemas – E) O conceito de software se restringe ao desenvolvimento do código em determinada linguagem e seu armazenamento em arquivos.

Comentários:

Aliás, vamos voltar um pouquinho: o que seria um software? Bem, em uma visão restritiva, muitas pessoas costumam associar o termo software aos programas de computador. Software não é apenas o programa, mas também todos os dados de documentação e configuração associados, necessários para que o programa opere corretamente. Vamos prosseguir...

Conforme vimos em aula, o software não é apenas o programa, mas também todos os dados de documentação e configuração associados, necessários para que o programa opere corretamente.

Gabarito: E

ACERTEI	ERREI

PROCESSOS DE DESENVOLVIMENTO DE SOFTWARE

Vamos começar falando sobre ciclo de vida. Esse termo surgiu lá no contexto da biologia. Ele trata do ciclo de vida de um ser vivo, ou seja, trata do conjunto de transformações pelas quais passam os indivíduos de uma espécie durante sua vida. *Vocês se lembram lá no ensino fundamental quando a gente aprende que um ser vivo nasce, cresce, reproduz, envelhece e morre?* Pois é!

Vejam a imagem acima! Nós temos um bebê, que depois se torna uma criança, depois um adolescente, depois um jovem, depois um senhor, depois um velho, depois um ancião e depois morre! Logo, o ciclo de vida de um ser vivo trata das fases pelas quais um ser vivo passa desde seu nascimento até a sua morte. E esse conceito pode ser aplicado a diversos outros contextos.

Exemplos: ciclo de vida de uma organização, ciclo de vida de sistemas, ciclo de vida de produtos, ciclo de vida de projetos, entre vários outros. *E como esse conceito se dá em outros contextos?* Exatamente da mesma forma! Logo, o ciclo de vida de um produto trata da história completa de um produto através de suas fases (Ex: Introdução, Crescimento, Maturidade e Declínio).

Existe também o ciclo de vida de um projeto, que trata do conjunto de fases que compõem um projeto (Ex: Iniciação, Planejamento, Execução, Controle e Encerramento). *O que todos esses ciclos de vida têm em comum?* Eles sempre tratam das fases pelas quais algo passa desde o seu início até o seu fim. Então, é isso que vocês têm que decorar para qualquer contexto de ciclo de vida.

Na Engenharia de Software, esse termo é geralmente aplicado a sistemas de software artificiais com o significado de mudanças que acontecem na vida de um produto de software. O ciclo de vida trata das fases identificadas entre o nascimento e a morte de um software. *Vocês viram?* É exatamente a mesma coisa – são as fases ou atividades pelas quais passa um software durante sua vida.

Uma definição interessante de ciclo de vida de software afirma que se trata das fases de um produto de software que vão desde quando ele é concebido até quando ele não está mais disponível para uso. Já Steve McConnell afirma que um modelo de ciclo de vida de software é uma representação que descreve todas as atividades que tratam da criação de um produto de software.

Portanto, nós podemos concluir que um ciclo de vida de software se refere às fases pelas quais um sistema de software atravessa desde sua concepção até sua retirada de produção. Bem, entender esse conceito não é o problema, esse é um conceito muito simples, convenhamos! O problema é que existem dezenas de fases diferentes para os ciclos de vida de acordo com cada autor.

Como assim, professor? Infelizmente, não há um consenso entre os autores. Cada um que lança um livro decide criar o ciclo de vida com as fases que ele acha mais corretas e isso acaba prejudicando nosso estudo. Na UnB, eu tive um professor de Engenharia de Software chamado Fernando Albuquerque que já tinha escrito livros sobre esse assunto e ele tinha o seu próprio ciclo de vida.

Agora imaginem a quantidade de autores de livros de Engenharia de Software lançados nas últimas três ou quatro décadas. Além disso, acontece de as vezes o próprio autor mudar seu entendimento sobre as fases. Se vocês olharem a quarta edição do livro do Pressman, ele tem um conjunto de fases; se vocês olharem da sexta edição para frente, ele define um novo conjunto de fases. Mudou de ideia!

Então, eu já vi mais vários ciclos de vida diferentes em provas! *Qual a solução para esse problema, professor?* Galera, vocês sempre têm que pensar no custo/benefício. *Vale a pena decorar todos?* Na minha opinião, nenhum pouco! Isso não é algo que cai muito. Guardem o espaço que vocês têm sobrando na cabeça para memorizar coisas que realmente caem.

Portanto, percebam que há autores que descrevem as fases do ciclo de vida de software de maneira mais ampla e abstrata (com poucas e grandes fases) e autores que o fazem de maneira mais diminuta e detalhada (com muitas e pequenas fases).

Vamos começar a ver alguns ciclos de vida que existem por aí para que vocês visualizem isso com maior clareza!

Vamos lá! Do lado esquerdo, temos um ciclo de vida de software só com quatro fases bem genéricas: Definição do Software, Desenvolvimento do Software, Operação do Software e Retirada do Software. Eu só vi cair esse ciclo de vida uma vez, mas ele é útil para que vocês visualizem um ciclo de vida com poucas fases. Observem que ele trata desde o início até a aposentadoria do software.

Do lado direito, eu coloquei um ciclo de vida um pouco mais detalhado. Vejam que ele destrincha mais a fase de desenvolvimento, separando em análise, projeto, implementação, testes, etc. *Por que eu coloquei esse ciclo de vida?* Porque alguns autores afirmam que, em tese, todo ciclo de vida deveria contemplar todas essas atividades ou fases. Vamos ver isso com mais detalhes depois...

Em verde, nós temos um ciclo de vida de software de acordo com nosso querido autor: Ian Sommerville. Ele contém quatro fases: Especificação, Desenvolvimento, Validação e Evolução. Sendo que, atenção aqui!, o desenvolvimento pode ser dividido em Projeto e Implementação. *Ok?* Esse é um ciclo de vida de software que já cai com uma maior frequência.

Por fim, em laranja, nós temos um ciclo de vida de software de acordo com nosso outro querido autor: Roger Pressman. Ele contém cinco fases: comunicação, planejamento, modelagem, construção e implantação. Esse não cai tanto, mas é importante saber também. *Tudo bem até aqui?* Então vejam que não é tão complicado! Esses são os quatro ciclos de vida de software mais comuns em prova...

Recaptulando: primeiro, nós vimos o que é um Ciclo de Vida! Depois nós vimos o que é um Ciclo de Vida de Software. E, agora, nós vamos ver um terceiro conceito, que é o conceito de Modelo de Ciclo de Vida de Software. *Por que, professor?* Porque Ciclo de Vida de Software é diferente de Modelo de Ciclo de Vida de Software. *E o que é um modelo de ciclo de vida de software?*

Bem, é um modelo que apresenta não só as fases do ciclo de vida de software, mas também a forma como essas fases se relacionam. *Vocês entenderam isso bem?* O modelo contém as fases que nós acabamos de ver, mas ele também nos diz como essas fases se relacionam! *Vocês querem um exemplo?* Existe um modelo de ciclo de vida chamado Modelo em Cascata.

Esse modelo foi pioneiro, foi o primeiro a aparecer na Engenharia de Software e nós vamos vê-lo em mais detalhes em outro momento. O importante sobre o Modelo em Cascata é a forma como as fases se relacionam. Nesse modelo, uma fase somente se inicia após o término completo da fase anterior (exceto a primeira, evidentemente). Bem simples, não é?

Logo, cada Modelo de Ciclo de Vida de Software contém suas respectivas fases, mas também contém como essas fases se relacionam. Vejam os conceitos abaixo:

Ciclo de Vida

O Ciclo de Vida trata das fases pelas quais alguma coisa passa desde o seu início até o seu fim.

Ciclo de Vida de Software

O Ciclo de Vida de Software trata das fases pelas quais um software passa desde o seu início até o seu fim.

Modelo de Ciclo de Vida de Software

O Modelo de Ciclo de Vida de Software trata das fases pelas quais um software passa desde o seu início até o seu fim e como essas fases se relacionam (processo).

E o que seria um processo de software? Então, um processo de software pode ser visto como o conjunto de atividades, métodos, práticas e transformações que guiam pessoas na produção de software. Como o Modelo de Ciclo de Vida nos diz como as fases do ciclo de vida se relacionam, nós podemos – em termos de prova – considerar Modelo de Ciclo de Vida como sinônimo de Modelo de Processo!

Processo de Software

Trata-se de um conjunto de atividades, métodos, práticas e transformações que guiam pessoas na produção de software.

Modelo de Processo de Software

Trata-se exatamente do mesmo conceito de Modelo de Ciclo de Vida. Ele é uma representação abstrata de um processo de software.

Um processo de software não pode ser definido de forma universal. Para ser eficaz e conduzir à construção de produtos de boa qualidade, um processo deve ser adequado às especificidades do projeto em questão. Deste modo, processos devem

ser definidos caso a caso, considerando-se diversos aspectos específicos do projeto em questão, tais como:

- Características da aplicação (domínio do problema, tamanho do software, tipo e complexidade, etc);
- Tecnologia a ser adotada na sua construção (paradigma de desenvolvimento, linguagem de programação, mecanismo de persistência, etc);
- Organização onde o produto será desenvolvido e a equipe de desenvolvimento alocada (recursos humanos).

Há vários aspectos a serem considerados na definição de um processo de software. No centro da arquitetura de um processo de desenvolvimento estão as atividades-chave desse processo: análise e especificação de requisitos, projeto, implementação e testes, que são a base sobre a qual o processo de desenvolvimento deve ser construído. *Entendido?*

No entanto, a definição de um processo envolve a escolha de um modelo de ciclo de vida (ou modelo de processo), o detalhamento (decomposição) de suas macro-atividades, a escolha de métodos, técnicas e roteiros (procedimentos) para a sua realização e a definição de recursos e artefatos necessários e produzidos – é bastante coisa!

Podemos dizer que se trata da representação abstrata de um esqueleto de processo, incluindo tipicamente algumas atividades principais, a ordem de precedência entre elas e, opcionalmente, artefatos requeridos e produzidos. Em geral, um modelo de processo descreve uma filosofia de organização de atividades, estruturando-as em fases e definindo como essas fases estão relacionadas.

A escolha de um modelo de ciclo de vida (para concursos, sinônimo de modelo de processo) é o ponto de partida para a definição de um processo de desenvolvimento de software. Um modelo de ciclo de vida, geralmente, organiza as macro-atividades básicas do processo, estabelecendo precedência e dependência entre as mesmas. *Tudo certo?*

Conforme eu disse anteriormente, alguns autores afirmam que os modelos de ciclo de vida básicos, de maneira geral, contemplam pelo menos as fases de: Planejamento; Análise e Especificação de Requisitos; Projeto; Implementação;

Testes; Entrega e Implantação; Operação; e Manutenção. Abaixo eu trago uma descrição genérica sobre cada uma dessas fases.

FASES	DESCRIÇÃO
PLANEJAMENTO	O objetivo do planejamento de projeto é fornecer uma estrutura que possibilite ao gerente fazer estimativas razoáveis de recursos, custos e prazos. Uma vez estabelecido o escopo de software, com os requisitos esboçados, uma proposta de desenvolvimento deve ser elaborada, isto é, um plano de projeto deve ser elaborado configurando o processo a ser utilizado no desenvolvimento de software. À medida que o projeto progride, o planejamento deve ser detalhado e atualizado regularmente. Pelo menos ao final de cada uma das fases do desenvolvimento (análise e especificação de requisitos, projeto, implementação e testes), o planejamento como um todo deve ser revisto e o planejamento da etapa seguinte deve ser detalhado. O planejamento e o acompanhamento do progresso fazem parte do processo de gerência de projeto.
ANÁLISE E ESPECIFICAÇÃO DE REQUISITOS	Nesta fase, o processo de levantamento de requisitos é intensificado. O escopo deve ser refinado e os requisitos mais bem definidos. Para entender a natureza do software a ser construído, o engenheiro de software tem de compreender o domínio do problema, bem como a funcionalidade e o comportamento esperados. Uma vez capturados os requisitos do sistema a ser desenvolvido, estes devem ser modelados, avaliados e documentados. Uma parte vital desta fase é a construção de um modelo descrevendo o que o software tem de fazer (e não como fazê-lo).
PROJETO	Esta fase é responsável por incorporar requisitos tecnológicos aos requisitos essenciais do sistema, modelados na fase anterior e, portanto, requer que a plataforma de implementação seja conhecida. Basicamente, envolve duas grandes etapas: projeto da arquitetura do sistema e projeto detalhado. O objetivo da primeira etapa é definir a arquitetura geral do software, tendo por base o modelo construído na fase de análise de requisitos. Essa arquitetura deve descrever a estrutura de nível mais alto da aplicação e identificar seus principais componentes. O propósito do projeto detalhado é detalhar o projeto do software para cada componente identificado na etapa anterior. Os componentes de software devem ser sucessivamente refinados em níveis maiores de detalhamento, até que possam ser codificados e testados.
IMPLEMENTAÇÃO	O projeto deve ser traduzido para uma forma passível de execução pela máquina. A fase de implementação realiza esta tarefa, isto é, cada unidade de software do projeto detalhado é implementada.

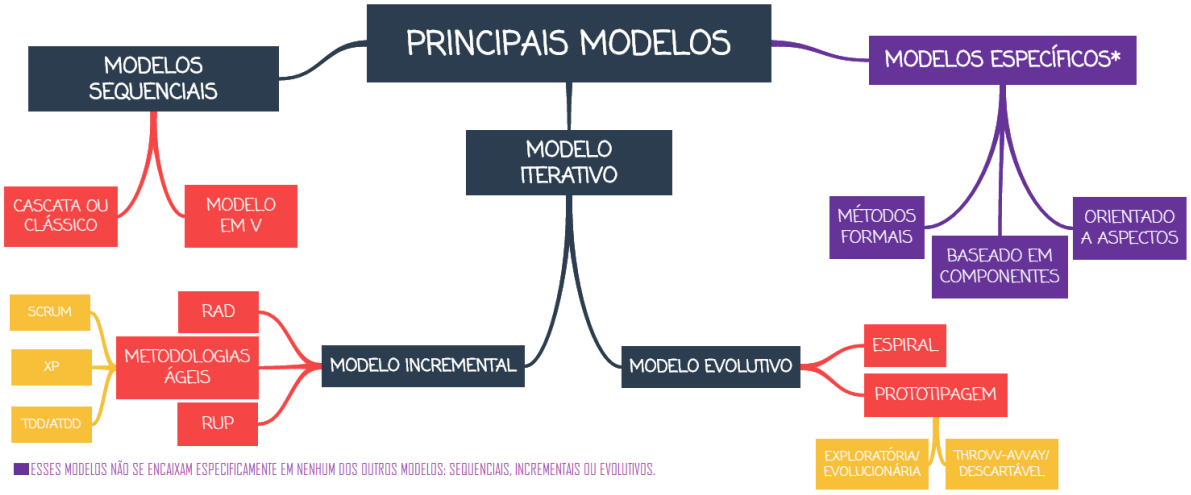
TESTES	Inclui diversos níveis de testes, a saber, teste de unidade, teste de integração e teste de sistema. Inicialmente, cada unidade de software implementada deve ser testada e os resultados documentados. A seguir, os diversos componentes devem ser integrados sucessivamente até se obter o sistema. Finalmente, o sistema como um todo deve ser testado.
ENTREGA E IMPLANTAÇÃO	Uma vez testado, o software deve ser colocado em produção. Para tal, contudo, é necessário treinar os usuários, configurar o ambiente de produção e, muitas vezes, converter bases de dados. O propósito desta fase é estabelecer que o software satisfaz os requisitos dos usuários. Isto é feito instalando o software e conduzindo testes de aceitação. Quando o software tiver demonstrado prover as capacidades requeridas, ele pode ser aceito e a operação iniciada.
OPERAÇÃO	Nesta fase, o software é utilizado pelos usuários no ambiente de produção.
MANUTENÇÃO	Indubitavelmente, o software sofrerá mudanças após ter sido entregue para o usuário. Alterações ocorrerão porque erros foram encontrados, porque o software precisa ser adaptado para acomodar mudanças em seu ambiente externo, ou porque o cliente necessita de funcionalidade adicional ou aumento de desempenho. Muitas vezes, dependendo do tipo e porte da manutenção necessária, essa fase pode requerer a definição de um novo processo, onde cada uma das fases precedentes é reaplicada no contexto de um software existente ao invés de um novo.

Existem outras fases em outros modelos, tais como: análise, responsável por modelar o problema (diferente do projeto, responsável por modelar a solução do problema); homologação, responsável pela aceitação pela parte interessada do produto; gerência de configuração, responsável pela estruturação sistemática dos produtos, artefatos, documentos, modelos, etc.

Sommerville afirma que um processo de software é um conjunto de atividades e resultados associados que produz um produto de software. De acordo com ele, existem quatro atividades fundamentais de processo, que são comuns a todos os processos de software – são elas: Especificação de Software; Desenvolvimento de Software (Projeto e Implementação); Validação de Software; e Evolução de Software.

Também de acordo nosso autor, um modelo de processo de software é uma descrição simplificada desse processo de software que apresenta uma visão dele. Os modelos de processo incluem as atividades, que fazem parte do processo de software, e eventualmente os produtos de software e os papéis das pessoas envolvidas na engenharia de software. E não para por aí...

Ele ainda afirma que a maioria dos processos de software é baseada em três modelos gerais: modelo em cascata; desenvolvimento iterativo e engenharia de software baseada em componentes. Isso entra em contradição com o que dizem outros autores, i.e., os principais modelos podem ser agrupados em três categorias: modelos sequenciais, modelos incrementais e modelos evolutivo.



Existem mais um conceito importante nessa aula! É o conceito de Metodologia de Desenvolvimento de Software (também chamada de Processo de Desenvolvimento de Software). *O que é isso, professor?* É uma caracterização prescritiva ou descritiva de como um produto de software deve ser desenvolvido, i.e., define o quê, como e quando fazer algo – um exemplo clássico é o RUP!

Bem, como o nome indica, os modelos sequenciais organizam o processo em uma sequência linear de fases. O principal modelo desta categoria é o Modelo em Cascata (criado por Winston Royce em 1970), a partir do qual diversos outros modelos foram propostos, inclusive a maioria dos modelos incrementais e evolutivos. Outro representante é o Modelo em V (V-Model).

Há muitas situações em que os requisitos são razoavelmente bem definidos, mas o tamanho do sistema impossibilita a adoção de um modelo sequencial, sobretudo pela necessidade de disponibilizar rapidamente uma versão para o usuário. Nesses

casos, um modelo incremental é indicado. No desenvolvimento incremental, o sistema é dividido em subsistemas ou módulos, tomando por base a funcionalidade.

Os incrementos (ou versões) são definidos, começando com um pequeno subsistema funcional que, a cada ciclo, é acrescido de novas funcionalidades. Além de acrescentar novas funcionalidades, nos novos ciclos, as funcionalidades providas anteriormente podem ser modificadas para melhor satisfazer às necessidades dos clientes/usuários.

Vale destacar que a definição das versões (e a correspondente segmentação e atribuição dos requisitos a essas versões) é realizada antes do desenvolvimento da primeira versão. Os principais representantes são RUP (Rational Unified Process), junto do Modelo RAD (Rapid Application Development) e, inclusive, as famosas Metodologias Ágeis.

Sistemas de software, como quaisquer sistemas complexos, evoluem ao longo do tempo. Seus requisitos, muitas vezes, são difíceis de serem estabelecidos ou mudam com frequência ao longo do desenvolvimento. Assim, é importante ter como opção modelos de ciclo de vida que lidem com incertezas e acomodem melhor as contínuas mudanças.

Alguns modelos incrementais, dado que preconizam um desenvolvimento iterativo, podem ser aplicados a esses casos, mas a grande maioria toma por pressuposto que os requisitos são bem definidos. Modelos evolucionários buscam preencher essa lacuna. Enquanto modelos incrementais têm por base a entrega de versões operacionais desde o primeiro ciclo, modelos evolutivos não têm essa preocupação.

Muito pelo contrário: o que ocorre na maioria das vezes é que os primeiros ciclos produzem protótipos ou até mesmo apenas modelos. À medida que o desenvolvimento avança e os requisitos vão ficando mais claros e estáveis, protótipos vão dando lugar a versões operacionais, até que o sistema completo seja construído. *Bacana?*

Assim, quando o problema não é bem definido e ele não pode ser totalmente especificado no início do desenvolvimento, deve-se optar por um modelo evolutivo. A avaliação ou o uso do protótipo/sistema pode aumentar o conhecimento sobre o produto e melhorar o entendimento que se tem acerca dos requisitos. Entretanto, é necessária uma forte gerência do projeto e de configuração.

1. (CESPE – 2009 – TCE/TO – Analista de Sistemas - A) Quanto maior e mais complexo o projeto de software, mais simples deve ser o modelo de processo a ser adotado.

Comentários:

Galera, não existe essa relação! Em geral, quanto mais complexo o projeto mais complexo o modelo. No entanto, isso também não é uma regra. Hoje em dia, existem projetos megacomplexos feitos utilizando metodologias ágeis, por exemplo.

Gabarito: E

---

2. (CESPE – 2009 – TCE/TO – Analista de Sistemas - B) O modelo de ciclo de vida do software serve para delimitar o alvo do software. Nessa visão, não são consideradas as atividades necessárias e o relacionamento entre elas.

Comentários:

> *A escolha de um modelo de ciclo de vida (para concursos, sinônimo de modelo de processo) é o ponto de partida para a definição de um processo de desenvolvimento de software. Um modelo de ciclo de vida, geralmente, organiza as macro-atividades básicas do processo, estabelecendo precedência e dependência entre as mesmas. Tudo certo?*

Pelo contrário, o alvo do software serve para delimitar o modelo de ciclo de vida a ser escolhido. Primeiro, define-se o alvo do software para, só então, escolher o modelo de ciclo de vida mais adequado. Ademais, são consideradas as atividades necessárias e o relacionamento entre elas.

Gabarito: E

---

3. (CESPE – 2009 – TCE/TO – Analista de Sistemas - C) A escolha do modelo do ciclo de vida não depende de características específicas do projeto, pois o melhor modelo é sempre o mais usado pela equipe do projeto.

Comentários:

Um processo de software não pode ser definido de forma universal. Para ser eficaz e conduzir à construção de produtos de boa qualidade, um processo deve ser adequado às especificidades do projeto em questão. Deste modo, processos devem ser definidos caso a caso, considerando-se diversos aspectos específicos do projeto em questão, tais como:

Conforme vimos em aula, não faz o menor sentido! A escolha depende das características do projeto; além disso, não existe um "*melhor*" modelo.

Gabarito: E

---

4. (ESAF - 2012 – CGU – Analista de Sistemas) A escolha de um modelo é fortemente dependente das características do projeto. Os principais modelos de ciclo de vida podem ser agrupados em três categorias principais:

a) sequenciais, cascata e evolutivos.
b) sequenciais, incrementais e ágeis.
c) sequenciais, incrementais e evolutivos.
d) sequenciais, ágeis e cascata.
e) cascata, ágeis e evolutivos.

Comentários:

Ele ainda afirma que a maioria dos processos de software é baseada em três modelos gerais: modelo em cascata; desenvolvimento iterativo e engenharia de software baseada em componentes. Isso entra em contradição com o que dizem outros autores, i.e., os principais modelos podem ser agrupados em três categorias: modelos sequenciais, modelos incrementais e modelos evolutivo.

Conforme vimos em aula, bastava lembrar da imagem: Sequencial, Incremental e Evolutivo. Galera, ágil é iterativo e incremental e, não, um modelo de ciclo de vida.

Gabarito: C

---

5. (CESPE – 2011 – MEC – Analista de Sistemas) O processo de desenvolvimento de software é uma caracterização descritiva ou prescritiva de como um produto de software deve ser desenvolvido.

Comentários:

Existem mais um conceito importante nessa aula! É o conceito de Metodologia de Desenvolvimento de Software (também chamada de Processo de Desenvolvimento de Software). O que é isso, professor? É uma caracterização prescritiva ou descritiva de como um produto de software deve ser desenvolvido, i.e., define o quê, como e quando fazer algo – um exemplo clássico é o RUP!

Perfeito! Segundo Pressman, um modelo prescritivo consiste em um conjunto específico de atividades de arcabouço, como – por exemplo – a NBR ISSO/IEC 12207 que estabelece uma estrutura comum para os processos de ciclo de vida de software. Já um modelo descritivo apresenta o processo em detalhes, é como se fosse um guia para o desenvolvimento de software. Ambas as formas podem ser utilizadas.

Gabarito: C

---

6. (CESPE – 2013 – TRT/10ª – Analista de Sistemas) As atividades fundamentais relacionadas ao processo de construção de um software incluem a especificação, o desenvolvimento, a validação e a evolução do software.

Comentários:

Sommerville afirma que um processo de software é um conjunto de atividades e resultados associados que produz um produto de software. De acordo com ele, existem quatro atividades fundamentais de processo, que são comuns a todos os processos de software – são elas: Especificação de Software; Desenvolvimento de Software (Projeto e Implementação); Validação de Software; e Evolução de Software.

Conforme vimos em aula, a questão está corretíssima!

Gabarito: C

---

7. (CESPE – 2010 – TRE/BA – Analista de Sistemas) Um modelo de processo de software consiste em uma representação complexa de um processo de software, apresentada a partir de uma perspectiva genérica.

Comentários:

Podemos dizer que se trata da representação abstrata de um esqueleto de processo, incluindo tipicamente algumas atividades principais, a ordem de precedência entre elas e, opcionalmente, artefatos requeridos e produzidos. Em geral, um modelo de processo descreve uma filosofia de organização de atividades, estruturando-as em fases e definindo como essas fases estão relacionadas.

Um modelo de processo de software ou modelo de ciclo de vida trata da representação abstrata/simplificada de um processo de software, apresentada a partir de uma perspectiva genérica.

Gabarito: E

---

8. **(CESPE – 2011 – MEC – Analista de Sistemas)** Atividades comuns a todos os processos de software incluem a especificação, o projeto, a implementação e a validação.

Comentários:

Sommerville afirma que um processo de software é um conjunto de atividades e resultados associados que produz um produto de software. De acordo com ele, existem quatro atividades fundamentais de processo, que são comuns a todos os processos de software – são elas: Especificação de Software; Desenvolvimento de Software (Projeto e Implementação); Validação de Software; e Evolução de Software.

Vejam que ele dividiu a atividade de Desenvolvimento em Projeto e Implementação, mas não invalida a questão. *Professor, ele não falou sobre a evolução!* Sim, mas a questão apenas afirma que essas são atividades comuns e, não, que são as únicas.

Gabarito: C

---

9. **(CESPE – 2015 – STJ – Analista de Sistemas)** As principais atividades de engenharia de software são especificação, desenvolvimento, validação e evolução.

Comentários:

Sommerville afirma que um processo de software é um conjunto de atividades e resultados associados que produz um produto de software. De acordo com ele, existem quatro atividades fundamentais de processo, que são comuns a todos os processos de software – são elas: Especificação de Software; Desenvolvimento de Software (Projeto e Implementação); Validação de Software; e Evolução de Software.

Conforme vimos em aula, a questão peca um pouco ao dizer que são atividades da engenharia de software. O ideal seria dizer que são as atividades principais do processo de software. Não sei se entraram com recurso e a banca recusou e se não entraram com recurso. Percebam também que é a terceira vez esse tipo de questão cai em prova.

Gabarito: C

---

10. (FCC – 2012 – MPE/AP – Analista de Sistemas) Um processo de software é um conjunto de atividades relacionadas que levam à produção de um produto de software. Existem muitos processos de software diferentes, mas todos devem incluir quatro atividades fundamentais: especificação, projeto e implementação, validação e:

a) teste
b) evolução.
c) prototipação.
d) entrega.
e) modelagem.

Comentários:

Sommerville afirma que um processo de software é um conjunto de atividades e resultados associados que produz um produto de software. De acordo com ele, existem quatro atividades fundamentais de processo, que são comuns a todos os processos de software – são elas: Especificação de Software; Desenvolvimento de Software (Projeto e Implementação); Validação de Software; e Evolução de Software.

Conforme vimos em aula, trata-se da evolução!

Gabarito: B

---

11. (CESPE – 2010 – EMBASA – Analista de Sistemas) Ciclo de vida de um software resume-se em eventos utilizados para definir o status de um projeto.

Comentários:

O ciclo de vida de software é um conjunto de estágios (e, não, eventos) utilizados para definir o status de um software (e, não, projeto). Sommerville afirma, por exemplo, que um ciclo de vida é composto por estágios que demonstram atividades fundamentais de desenvolvimento. Questão errada!

Gabarito: E

---

12. (CESPE – 2016 – TCE/PR – Analista de Sistemas) As fases do ciclo de vida de um software são:

a) concepção, desenvolvimento, entrega e encerramento.
b) iniciação, elaboração, construção e manutenção.
c) escopo, estimativas, projeto e processo e gerência de riscos.
d) análise, desenvolvimento, teste, empacotamento e entrega.
e) planejamento, análise e especificação de requisitos, projeto, implementação, testes, entrega e implantação, operação e manutenção.

Comentários:

> *Conforme eu disse anteriormente, alguns autores afirmam que os modelos de ciclo de vida básicos, de maneira geral, contemplam pelo menos as fases de: Planejamento; Análise e Especificação de Requisitos; Projeto; Implementação; Testes; Entrega e Implantação; Operação; e Manutenção. Abaixo eu trago uma descrição genérica sobre cada uma dessas fases.*

Conforme vimos em aula, a questão trata da última opção.

Gabarito: E

---

13. (MPE/RS – 2012 – MPE/RS – Analista de Sistemas) O ciclo de vida básico de um software compreende:

a) a implementação, a implantação e o teste.
b) a análise, a segurança e o controle de usuários.
c) a implementação, a análise e o teste.
d) a implementação, a validação e as vendas.
e) a análise, o projeto, a implementação e o teste.

Comentários:

Conforme eu disse anteriormente, alguns autores afirmam que os modelos de ciclo de vida básicos, de maneira geral, contemplam pelo menos as fases de: Planejamento; Análise e Especificação de Requisitos; Projeto; Implementação; Testes; Entrega e Implantação; Operação; e Manutenção. Abaixo eu trago uma descrição genérica sobre cada uma dessas fases.

Aqui temos que escolher a mais correta, na medida em que a primeira, terceira e última opções estão corretas. Nesse sentido, a última opção é a mais correta!

Gabarito: E

---

14. (CESGRANRIO – 2011 – PETROBRÁS – Analista de Sistemas) A especificação de uma Metodologia de Desenvolvimento de Sistemas tem como pré-requisito indispensável, em relação ao que será adotado no processo de desenvolvimento, a definição do:

a) Engenheiro Responsável pelo Projeto
b) Documento de Controle de Sistemas
c) Software para Desenvolvimento
d) Ciclo de Vida do Software
e) Bloco de Atividades

Comentários:

A escolha de um modelo de ciclo de vida (para concursos, sinônimo de modelo de processo) é o ponto de partida para a definição de um processo de desenvolvimento de software. Um modelo de ciclo de vida, geralmente, organiza as macro-atividades básicas do processo, estabelecendo precedência e dependência entre as mesmas. Tudo certo?

Essa questão foi bem escrita (finalmente)! Percebam que ele diferencia metodologia de desenvolvimento de software do ciclo de vida de software. Por exemplo: primeiro, eu defino que eu vou utilizar um ciclo de vida evolutivo e depois eu decido que eu vou utilizar a metodologia de desenvolvimento em espiral.

Gabarito: D

---

15. (CESPE – 2008 – INPE – Analista de Sistemas) O ciclo de vida do software tem início na fase de projeto.

Comentários:

Conforme eu disse anteriormente, alguns autores afirmam que os modelos de ciclo de vida básicos, de maneira geral, contemplam pelo menos as fases de: Planejamento; Análise e Especificação de Requisitos; Projeto; Implementação; Testes; Entrega e Implantação; Operação; e Manutenção. Abaixo eu trago uma descrição genérica sobre cada uma dessas fases.

Conforme vimos em aula, ele não começa com projeto! *Como você começa com projeto sem sequer levantar os requisitos?* Era possível responder usando só lógica!

Gabarito: E

---

16. (CESPE – 2013 – TRT/10 – Analista de Sistemas) O ciclo de vida de um software, entre outras características, está relacionado aos estágios de concepção, projeto, criação e implementação.

Comentários:

Essa questão é péssima! Ela foi retirada do livro *Sistemas de Informação: Uma Abordagem Gerencial* (Steven R. Gordon, Judith R. Gordon). Nesse livro, os autores de fato tratam o ciclo de vida de software como concepção, projeto, criação e implementação. Essa é uma daquelas quase impossível de acertar! Esse não é um autor consagrado de Engenharia de Software.

Gabarito: C

---

17. (CESPE – 2011 – TJ/ES – Analista de Sistemas) Entre as etapas do ciclo de vida de software, as menos importantes incluem a garantia da qualidade, o projeto e o estudo de viabilidade. As demais atividades do ciclo, como a implementação e os testes, requerem maior dedicação da equipe e são essenciais.

Comentários:

Para a definição completa do processo, cada atividade descrita no modelo de ciclo de vida deve ser decomposta e para suas subatividades, devem ser associados métodos, técnicas, ferramentas e critérios de qualidade, entre outros, formando uma

base sólida para o desenvolvimento. Adicionalmente, outras atividades, tipicamente de cunho gerencial, devem ser definidas, como controle e garantia da qualidade. Não existe uma hierarquia de etapas mais importantes e menos importantes!

Gabarito: E

---

18. (CESPE - 2016 – TCE/PR – Analista de Sistemas – A) A engenharia de software está relacionada aos diversos aspectos de produção de software e inclui as atividades de especificação, desenvolvimento, validação e evolução de software.

Comentários:

Sommerville afirma que um processo de software é um conjunto de atividades e resultados associados que produz um produto de software. De acordo com ele, existem quatro atividades fundamentais de processo, que são comuns a todos os processos de software – são elas: Especificação de Software; Desenvolvimento de Software (Projeto e Implementação); Validação de Software; e Evolução de Software.

Conforme vimos em aula, a questão está correta.

Gabarito: C

---

19. (CESPE - 2016 – TCE/PR – Analista de Sistemas – D) Um processo de software é composto por quatro atividades fundamentais: iniciação, desenvolvimento, entrega e encerramento.

Comentários:

Sommerville afirma que um processo de software é um conjunto de atividades e resultados associados que produz um produto de software. De acordo com ele, existem quatro atividades fundamentais de processo, que são comuns a todos os processos de software – são elas: Especificação de Software; Desenvolvimento de Software (Projeto e Implementação); Validação de Software; e Evolução de Software.

Conforme vimos em aula, as atividades estão incorretas.

Gabarito: E

---

20. (CESPE - 2016 – TCE/PR – Analista de Sistemas – B) A metodologia de processos genérica para a engenharia de software é composta de seis etapas: comunicação, planejamento, modelagem, construção, emprego e aceitação.

Comentários:

Conforme vimos em aula, essas etapas parecem as fases do ciclo de vida de acordo com o Pressman, que são: Comunicação, Planejamento, Modelagem, Construção e Implantação. Percebam que ele troca Implantação por Emprego e Aceitação.

Gabarito: E

---

21. (INSTITUTO CIDADE – 2012 – TCM/GO – Analista de Sistemas) De acordo com a engenharia de software, como todo produto industrial, o software possui um ciclo de vida. Cada fase do ciclo de vida possui divisões e subdivisões. Em qual fase avaliamos a necessidade de evolução dos softwares em funcionamento para novas plataformas operacionais ou para a incorporação de novos requisitos?

a) Fase de operação;
b) Fase de retirada;
c) Fase de definição;
d) Fase de design.
e) Fase de desenvolvimento;

Comentários:

A fase retirada é um grande desafio para os tempos atuais. Diversos softwares que estão em funcionamento em empresas possuem excelente níveis de confiabilidade e de correção. No entanto, eles precisam evoluir para novas plataformas operacionais ou para a incorporação de novos requisitos.

A retirada desses softwares legados em uma empresa é sempre uma decisão difícil: *como abrir mão daquilo que é confiável e ao qual os funcionários estão acostumados, após anos de treinamento e utilização?* Portanto, processos de reengenharia podem ser aplicados para viabilizar a transição ou migração de um software legado para um novo software de forma a proporcionar uma retirada mais suave.

A avaliação da necessidade de evolução do software em funcionamento para novas plataformas operacionais ou para a incorporação de novos requisitos realmente ocorrem na fase de retirada.

Gabarito: B

---

22. (CESPE - 2010 – DETRAN/ES – Analista de Sistemas) Quando um aplicativo de software desenvolvido em uma organização atinge, no fim do seu ciclo de vida, a fase denominada aposentadoria, descontinuação ou fim de vida, todos os dados por ele manipulados podem ser descartados.

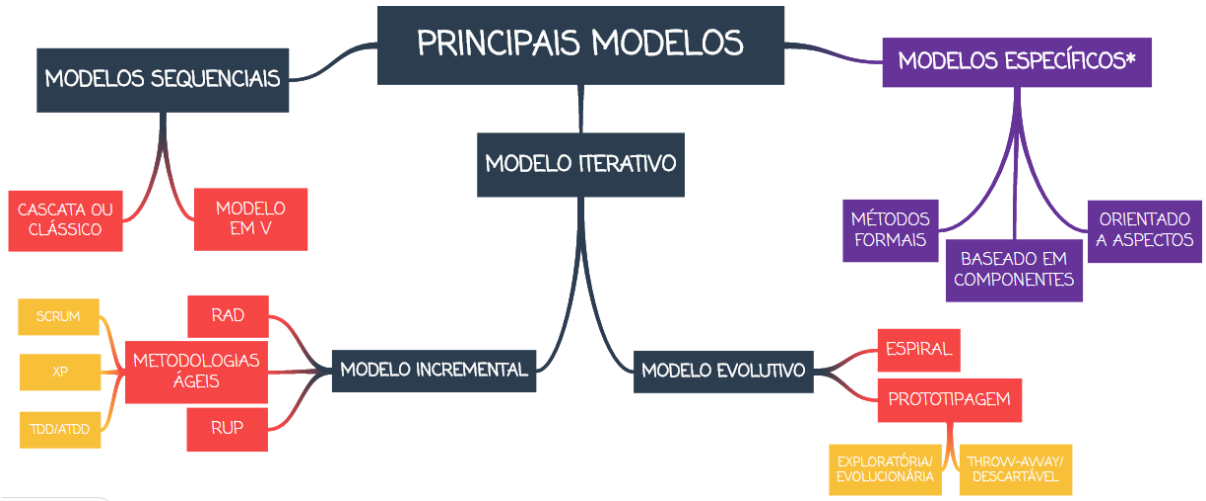
Comentários:

Essa questão é um absurdo! Observem que ela afirma "*podem ser descartados*". Ora, é evidente que podem ser descartados. No entanto, o gabarito oficial é errado!

Gabarito: E

ACERTEI	ERREI

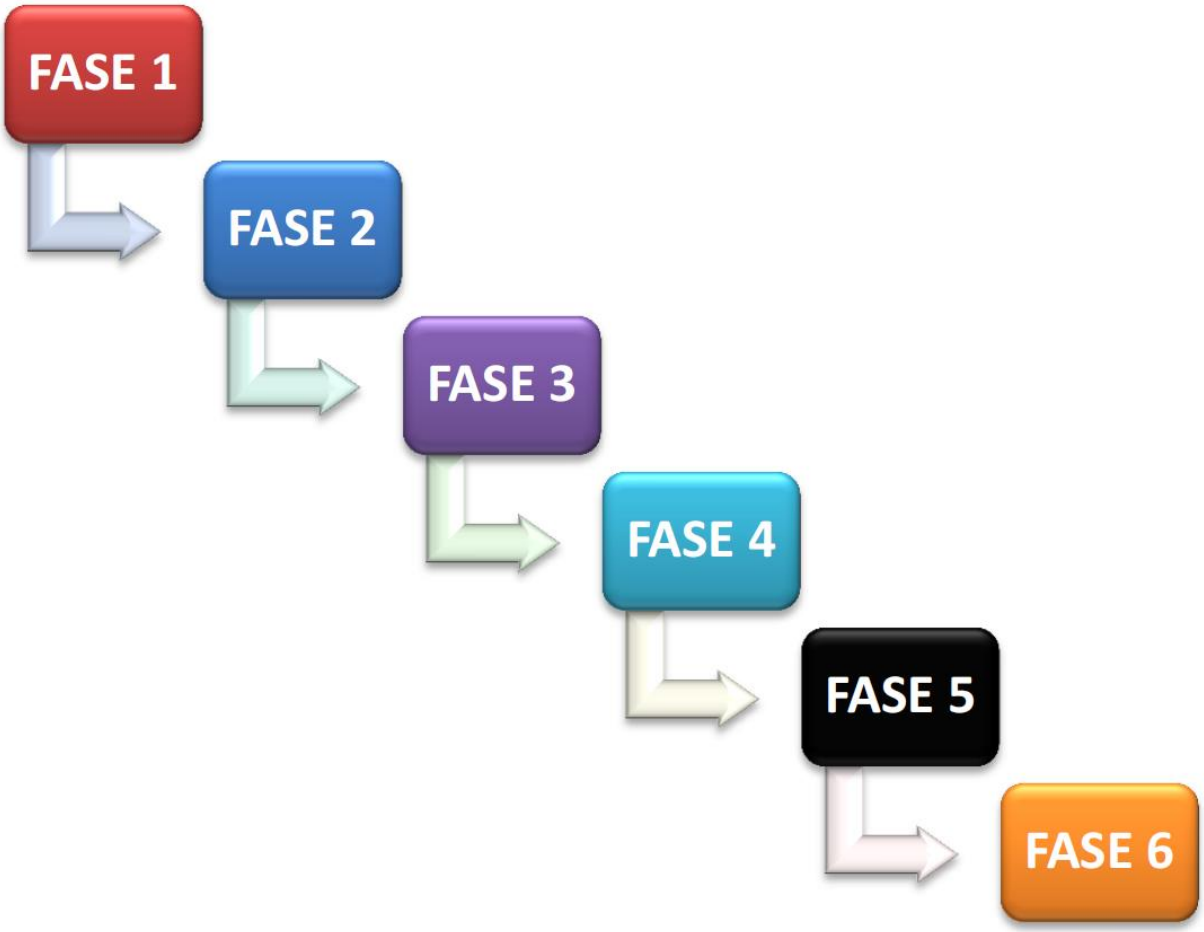
MODELO EM CASCATA



Citado inicialmente em 1970 por W. Royce, também designado Cascata ou Clássico ou Sequencial ou Linear ou Tradicional ou Waterfall ou Rígido ou Monolítico (todos esses nomes já caíram em prova!). Esse nome é devido ao encadeamento simples de uma fase com a outra. Os estágios do modelo demonstram as principais atividades de desenvolvimento. Observem a imagem mais abaixo!

No Modelo em Cascata, uma fase só se inicia após o término e aprovação da fase anterior, isto é, há uma sequência de desenvolvimento do projeto. Por exemplo, a Fase 4 só é iniciada após o término e aprovação da Fase 3. A Fase 5 só é iniciada após o término e aprovação da Fase 4. *Mas que fases são essas?* Bem, agora que complica, porque cada autor resolve criar suas fases!

De acordo com Vasconcelos (2006), no Modelo em Cascata, o projeto segue uma série passos ordenados, ao final de cada fase, a equipe de projeto finaliza uma revisão. Além disso, o desenvolvimento não continua até que o cliente esteja satisfeito com os resultados alcançados. *Vocês conseguem perceber como essas restrições engessam o desenvolvimento?*



Por Sommerville	Por Royce	Por Pressman (4ª Ed.)	Por Pressman (6ª Ed.)
Análise e Definição de Requisitos	Requisitos de Sistema	Modelagem e Engenharia do Sistema/Informação	Comunicação
Projeto de Sistema e Software	Requisitos de Software	Análise de Requisitos de Software	Planejamento
Implementação e Teste de Unidade	Análise	Projeto	Modelagem
Integração e Teste de Sistema	Projeto	Geração de Código	Construção
Operação e Manutenção	Codificação	Teste e Manutenção	Implantação
	Teste		
	Operação		

Percebam que há diferenças gritantes entre os autores! Inclusive, há divergências até entre autor e ele mesmo, dependendo da versão do livro (Exemplo: Pressman mudou as fases na última edição de seu livro). *Professor, você já viu isso cair em prova?* Sim, já vi! *E o que aconteceu?* Bem, polêmica, recursos, etc! Não há o que fazer... minha classificação preferida é a do Royce.

Na prática, esses estágios não são completamente sequenciais, i.e., eles se sobrepõem e trocam informações entre si. Na teoria, são fases sequenciais com o resultado de cada fase consistindo em um ou mais documentos aprovados ou não, dependendo dos problemas. Por exemplo: durante o projeto, são identificados problemas com requisitos.

De modo geral, grande parte dos modelos possuem as seguintes fases:

a) Planejamento: faz-se o esboço do escopo e dos requisitos, além de estimativas razoáveis sobre recursos, custos e prazos.

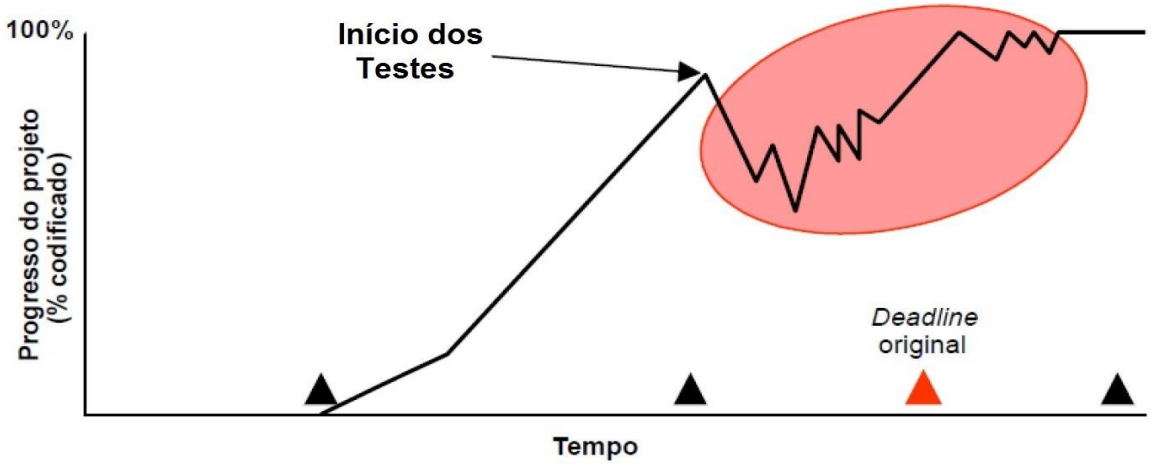
b) Análise e Especificação de Requisitos: durante essa fase, refina-se os requisitos e o escopo e desenha-se o problema em questão.

c) Projeto: durante essa fase, incorpora-se requisitos tecnológicos aos requisitos essenciais do sistema e projeta-se a arquitetura do sistema.

d) Implementação: durante essa fase, codifica-se o software como um conjunto de programas executáveis pela máquina.

e) Teste: o programa é testado como um sistema completo para garantir que os requisitos de software foram atendidos.

f) Implantação, Operação e Manutenção: o sistema de software é liberado para o cliente, treina-se usuários, gerencia serviços e realiza manutenções.



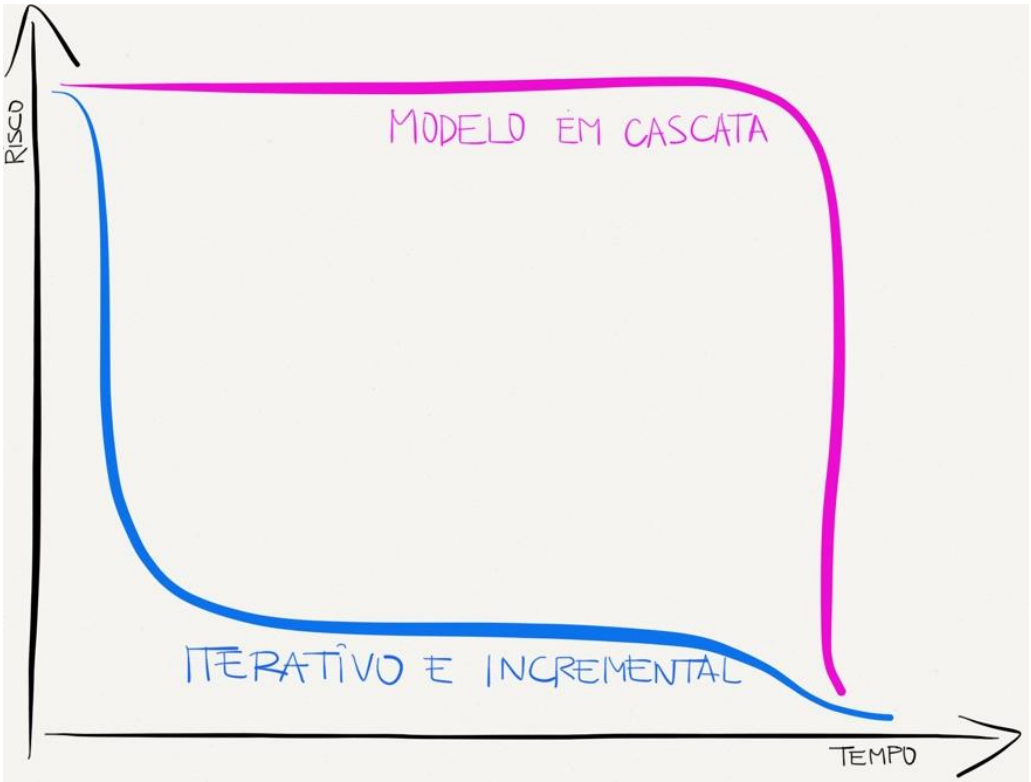
O Modelo em Cascata atrasa a redução de riscos...

Professor, o que você quer dizer com atrasar a redução de riscos? Bem, essa é uma desvantagem recorrente em provas. Como uma fase só se inicia após o término da fase anterior, só é possível em geral verificar se houve erros nas últimas fases – como pode ser visto na imagem abaixo. Em outros modelos, os riscos são reduzidos desde as primeiras fases do processo de desenvolvimento.

Percebam que os riscos deveriam ser descobertos logo no início do processo de desenvolvimento, porém eles são descobertos somente após o início dos testes e integração. Vocês podem notar que, nesse instante (parte vermelha), o progresso do projeto avança e retrai diversas vezes, porque o sistema está sendo corrigido devido a requisitos modificados.

Vejam, também, que o projeto não terminou em seu deadline original. Como a redução dos riscos atrasou, todo andamento do projeto também atrasou. Dessa forma, não se cumpriu nem o prazo do projeto e, provavelmente, nem o orçamento e talvez seu escopo – tendo em vista que, quanto mais ao fim do projeto um erro é identificado, mais caras se tornam as modificações.

Entenderam essa parte direitinho? Um erro na fase de requisitos, por exemplo, que não foi corrigido e foi descoberto no final do processo de desenvolvimento, terá um custo de correção altíssimo, visto que provavelmente terá que se refazer tudo novamente. Ora, se eu peço a construção de um carro e você constrói uma moto, o custo para corrigir esse erro será altíssimo.



Portanto não confundam essas duas coisas: se o erro ocorreu no início e foi identificado no início, terá baixo custo de correção; se o erro ocorreu no início e foi identificado no final, terá alto custo de correção. O custo de correção de um erro está mais focado no momento em que um erro é identificado do que no momento em que ele ocorre. *Bacana?*

Outra maneira de visualizar o atraso é por meio de um gráfico Risco x Tempo, comparando o modelo em cascata com o Modelo Iterativo e Incremental. Observem que o Modelo Iterativo e Incremental já começa a reduzir os riscos desde o início do processo, em contraste com o Modelo em Cascata que acumula os riscos até a fase de testes, integração e implantação do sistema.

Galera, a grande vantagem do Modelo em Cascata é que o desenvolvimento é dividido em fases distintas, padronizadas, etc (antigamente, os softwares eram construídos artesanalmente). É possível colocar pessoas com perfis específicos para cada fase, i.e., quem tem facilidade de se comunicar vai ser analista de requisitos, programadores vão fazer a codificação, etc.

A grande desvantagem é que - em projetos complexos – demora-se muito para chegar até a fase de testes, sendo que o cliente não vê nada rodando até a implantação. Então, pode acontecer de, nas fases finais, os requisitos da organização

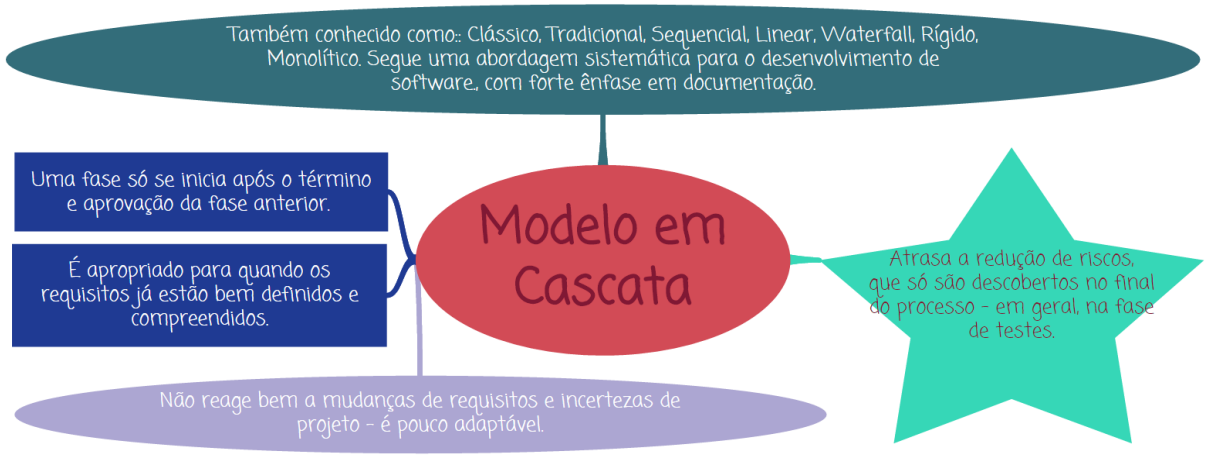
não serem mais os mesmos daqueles do início e o software não ter mais utilidade para organização.

Professor, então o Modelo em Cascata não deve ser usado em nenhuma hipótese? Calma lá, ele pode ser usado! No entanto, sua utilização deve ocorrer preferencialmente quando os requisitos forem bem compreendidos e houver pouca probabilidade de mudanças radicais durante o desenvolvimento do sistema. *Vocês entenderam?*

VANTAGENS	DESVANTAGENS
É simples de entender e fácil de aplicar, facilitando o planejamento.	Divisão inflexível do projeto em estágios distintos.
Fixa pontos específicos para a entrega de artefatos.	Dificuldade em incorporar mudanças de requisitos.
Funciona bem para equipes tecnicamente fracas.	Clientes só visualizam resultados próximos ao final do projeto.
É fácil de gerenciar, devido a sua rigidez.	Atrasa a redução de riscos.
Realiza documentação extensa por cada fase ou estágio.	Apenas a fase final produz um artefato de software entregável.
Possibilita boa aderência a outros modelos de processo.	Cliente deve saber todos os requisitos no início do projeto.
Funciona bem com projetos pequenos e com requisitos bem conhecidos.	Modelo inicial (Royce) não permitia feedback entre as fases do projeto.
	Pressupõe que os requisitos ficarão estáveis ao longo do tempo.
	Não funciona bem com projetos complexos e OO, apesar de compatível.

Podemos afirmar, então, que o Modelo em Cascata é considerado um método tradicional e fortemente prescritivo, i.e., ele busca sempre dizer o que fazer, em geral, por meio de planos definidos no início do projeto. *Que planos, professor?* Escopo, custo, cronograma... tudo isso bem detalhado em uma extensa documentação. *Mudanças são bem-vindas?* Claro que não!

ESQUEMA


Também conhecido como:: Clássico, Tradicional, Sequencial, Linear, Waterfall, Rígido, Monolítico. Segue uma abordagem sistemática para o desenvolvimento de software, com forte ênfase em documentação.
Uma fase só se inicia após o término e aprovação da fase anterior.
É apropriado para quando os requisitos já estão bem definidos e compreendidos.
Modelo em Cascata
Atrasa a redução de riscos, que só são descobertos no final do processo - em geral, na fase de testes.
Não reage bem a mudanças de requisitos e incertezas de projeto - é pouco adaptável.

1. (FCC - 2012 - TJ-RJ - Analista Judiciário - Análise de Sistemas - E) Dos diferentes modelos para o ciclo de vida de desenvolvimento de um software é correto afirmar que o modelo em cascata é o mais recente e complexo.

Comentários:

Citado inicialmente em 1970 por W. Royce, também designado Cascata ou Clássico ou Sequencial ou Linear ou Tradicional ou Waterfall ou Rígido ou Monolítico (todos esses nomes já caíram em prova!). Esse nome é devido ao encadeamento simples de uma fase com a outra. Os estágios do modelo demonstram as principais atividades de desenvolvimento. Observem a imagem mais abaixo!

Mais recente? Não, muito antigo! *Complexo?* Não, possui um encadeamento simples de fases.

Gabarito: E

---

2. (FCC - 2009 - SEFAZ-SP - Agente Fiscal de Rendas - Tecnologia da Informação - Prova 3 - B) O processo de engenharia de software denominado ciclo de vida clássico refere-se ao modelo incremental.

Comentários:

Citado inicialmente em 1970 por W. Royce, também designado Cascata ou Clássico ou Sequencial ou Linear ou Tradicional ou Waterfall ou Rígido ou Monolítico (todos esses nomes já caíram em prova!). Esse nome é devido ao encadeamento simples de uma fase com a outra. Os estágios do modelo demonstram as principais atividades de desenvolvimento. Observem a imagem mais abaixo!

Não, modelo clássico se refere a modelo em cascata, sequencial, linear, tradicional, waterfall, rígido ou monolítico.

Gabarito: E

---

3. (CESPE – 2009 – INMETRO – Analista de Sistemas) Em uma empresa que tenha adotado um processo de desenvolvimento de software em cascata, falhas no levantamento de requisitos têm maior possibilidade de gerar grandes prejuízos do que naquelas que tenham adotado desenvolvimento evolucionário.

Comentários:

VANTAGENS	DESVANTAGENS
É simples de entender e fácil de aplicar, facilitando o planejamento.	Divisão inflexível do projeto em estágios distintos.
Fixa pontos específicos para a entrega de artefatos.	**Dificuldade em incorporar mudanças de requisitos.**
Funciona bem para equipes tecnicamente fracas.	Clientes só visualizam resultados próximos ao final do projeto.
É fácil de gerenciar, devido a sua rigidez.	Atrasa a redução de riscos.
Realiza documentação extensa por cada fase ou estágio.	Apenas a fase final produz um artefato de software entregável.
Possibilita boa aderência a outros modelos de processo.	Cliente deve saber todos os requisitos no início do projeto.
Funciona bem com projetos pequenos e com requisitos bem conhecidos.	Não fornece feedback entre as fases.
	Pressupõe que os requisitos ficarão estáveis ao longo do tempo.
	Não funciona bem com projetos complexos e OO, apesar de compatível.

O Modelo em Cascata, de fato, não reage bem a mudanças. Já Modelo evolucionário é mais adaptável a mudanças por se utilizar de iterações.

Gabarito: C

---

4. (CESPE – 2011 – MEC – Analista de Sistemas) O modelo Waterfall tem a vantagem de facilitar a realização de mudanças sem a necessidade de retrabalho em fases já completadas.

Comentários:

VANTAGENS	DESVANTAGENS
É simples de entender e fácil de aplicar, facilitando o planejamento.	Divisão inflexível do projeto em estágios distintos.
Fixa pontos específicos para a entrega de artefatos.	**Dificuldade em incorporar mudanças de requisitos.**
Funciona bem para equipes tecnicamente fracas.	Clientes só visualizam resultados próximos ao final do projeto.
É fácil de gerenciar, devido a sua rigidez.	Atrasa a redução de riscos.
Realiza documentação extensa por cada fase ou estágio.	Apenas a fase final produz um artefato de software entregável.
Possibilita boa aderência a outros modelos de processo.	Cliente deve saber todos os requisitos no início do projeto.
Funciona bem com projetos pequenos e com requisitos bem conhecidos.	Não fornece feedback entre as fases.
	Pressupõe que os requisitos ficarão estáveis ao longo do tempo.
	Não funciona bem com projetos complexos e OO, apesar de compatível.

Pelo contrário, há dificuldade de lidar com requisitos voláteis, tendo em vista que dependendo do erro, é necessário refazê-lo desde seu início.

Gabarito: E

---

5. (CESPE – 2008 – TST – Analista de Sistemas) No modelo de desenvolvimento sequencial linear, a fase de codificação é a que gera erros de maior custo de correção.

Comentários:

Entenderam essa parte direitinho? Um erro na fase de requisitos, por exemplo, que não foi corrigido e foi descoberto no final do processo de desenvolvimento, terá um custo de correção altíssimo, visto que provavelmente terá que se refazer tudo novamente. Ora, se eu peço a construção de um carro e você constrói uma moto, o custo para corrigir esse erro será altíssimo.

Portanto não confundam essas duas coisas! Percebam o que eu disse: quanto mais tarde se descobre um erro, mais caro se torna sua correção. Dizendo isso de outra forma: erros nas fases iniciais possuem custo de correção altíssimo. Uma coisa é o momento em que o erro ocorre (quanto mais cedo, mais caro); outra coisa é o momento em que um erro é identificado (quanto mais tarde, mais caro). Bacana?

Percebam que erros nas fases iniciais possuem custos de correção mais altos. *Logo, o maior custo está na fase de codificação?* Não, está na fase de requisitos que é a fase inicial!

Gabarito: E

---

6. (CESPE – 2009 – INMETRO – Analista de Sistemas) Em um processo de desenvolvimento em cascata, os testes de software são realizados todos em um mesmo estágio, que acontece após a finalização das fases de implementação.

Comentários:

Por Sommerville	Por Royce	Por Pressman (4ª Ed.)	Por Pressman (6ª Ed.)
Análise e Definição de Requisitos	Requisitos de Sistema	Modelagem e Engenharia do Sistema/Informação	Comunicação
Projeto de Sistema e Software	Requisitos de Software	Análise de Requisitos de Software	Planejamento
Implementação e Teste de Unidade	Análise	Projeto	Modelagem
Integração e Teste de Sistema	Projeto	Geração de Código	Construção
Operação e Manutenção	Codificação	Teste e Manutenção	Implantação
	Teste		
	Operação		

Todos em um mesmo estágio, não. A grande maioria dos testes ocorrem, de fato, após a finalização das fases de implementação. No entanto, podem ocorrer testes unitários durante a própria implementação.

Gabarito: E

---

7. (CESPE – 2008 – SERPRO – Analista de Sistemas) O modelo em cascata consiste de fases e atividades que devem ser realizadas em sequência, de forma que uma atividade é requisito da outra.

Comentários:

> *No Modelo em Cascata, uma fase só se inicia após o término e aprovação da fase anterior, isto é, há uma sequência de desenvolvimento do projeto. Por exemplo, a Fase 4 só é iniciada após o término e aprovação da Fase 3. A Fase 5 só é iniciada após o término e aprovação da Fase 4. Mas que fases são essas? Bem, agora que complica, porque cada autor resolve criar suas fases! Vejamos: (...)*

Vimos isso exaustivamente: no modelo em cascata, uma fase só se inicia após o término e aprovação da fase anterior.

Gabarito: C

---

8. (CESPE – 2005 – AL/ES – Analista de Sistemas - B) O modelo de desenvolvimento em cascata descreve ciclos sequenciais, incrementais e iterativos, possuindo, entre outras, as fases de requisitos e implementação.

Comentários:

Não! Ele não descreve ciclos, muito menos ciclos iterativos. Na verdade, essa é a definição de Modelo Iterativo e Incremental.

Gabarito: E

---

9. (CESPE – 2004 – STJ – Analista de Sistemas) O modelo de desenvolvimento seqüencial linear, também chamado modelo clássico ou modelo em cascata, caracteriza-se por não acomodar adequadamente as incertezas que existem no

início de um projeto de software, em especial as geradas pela dificuldade do cliente de explicitar todos os requerimentos que o programa deve contemplar.

Comentários:

Professor, o que você quer dizer com atrasar a redução de riscos? Bem, essa é uma desvantagem recorrente em provas. Como uma fase só se inicia após o término da fase anterior, só é possível em geral verificar se houve erros nas últimas fases – como pode ser visto na imagem abaixo. Em outros modelos, os riscos são reduzidos desde as primeiras fases do processo de desenvolvimento.

Perfeito, lembrem-se que ele acumula riscos e não lida bem com requisitos voláteis.

Gabarito: C

---

10. (CESPE – 2009 – IPEA – Analista de Sistema) No modelo em cascata de processo de desenvolvimento, os clientes devem definir os requisitos apenas durante a fase de projeto; e os projetistas definem as estratégias de projeto apenas durante a fase de implementação. As fases do ciclo de vida envolvem definição de requisitos, projeto, implementação, teste, integração, operação e manutenção. Em cada fase do ciclo de vida, podem ser produzidos diversos artefatos.

Comentários:

Essa questão não faz sentido! Os clientes definem os requisitos durante a fase de Definição de Requisitos. Já os projetistas definem as estratégias de projeto apenas durante a fase Projeto.

Gabarito: E

---

11. (CESPE – 2008 – TCE/TO – Analista de Sistema – D) No ciclo de vida em cascata, é possível realizar alternadamente e simultaneamente as atividades de desenvolvimento de software.

Comentários:

No Modelo em Cascata, uma fase só se inicia após o término e aprovação da fase anterior, isto é, há uma sequência de desenvolvimento do projeto. Por exemplo, a Fase 4 só é iniciada após o término e aprovação da Fase 3. A Fase 5 só é iniciada

após o término e aprovação da Fase 4. Mas que fases são essas? Bem, agora que complica, porque cada autor resolve criar suas fases! Vejamos: (...)

Não, é sequencial e linear. Não pode ser alternado e simultâneo!

Gabarito: E

---

12. (CESPE – 2004 – TJ/PA – Analista de Sistema – D) A abordagem sistemática estritamente linear para o desenvolvimento de software é denominada modelo em cascata ou modelo sequencial linear.

Comentários:

Citado inicialmente em 1970 por W. Royce, também designado Cascata ou Clássico ou Sequencial ou Linear ou Tradicional ou Waterfall ou Rígido ou Monolítico (todos esses nomes já caíram em prova!). Esse nome é devido ao encadeamento simples de uma fase com a outra. Os estágios do modelo demonstram as principais atividades de desenvolvimento. Observem a imagem mais abaixo!

Perfeito! Modelo em Cascata, Linear, Sequencial, Waterfall, etc.

Gabarito: C

---

13. (CESPE – 2006 – TSE – Analista de Sistema – D) O modelo em cascata organiza o desenvolvimento em fases. Esse modelo encoraja a definição dos requisitos antes do restante do desenvolvimento do sistema. Após a especificação e a análise dos requisitos, têm-se o projeto, a implementação e o teste.

Comentários:

Por Sommerville	Por Royce	Por Pressman (4ª Ed.)	Por Pressman (6ª Ed.)
Análise e Definição de Requisitos	Requisitos de Sistema	Modelagem e Engenharia do Sistema/Informação	Comunicação
Projeto de Sistema e Software	Requisitos de Software	Análise de Requisitos de Software	Planejamento
Implementação e Teste de Unidade	Análise	Projeto	Modelagem

Integração e Teste de Sistema	Projeto	Geração de Código	Construção
Operação e Manutenção	Codificação	Teste e Manutenção	Implantação
	Teste		
	Operação		

Perfeito! De fato, segue essa ordem!

Gabarito: C

---

14. **(CESPE – 2009 – INMTRO – Analista de Sistema)** No desenvolvimento de software, o modelo em cascata é estruturado de tal maneira que as fases que compõem o desenvolvimento são interligadas. Nessa situação, o final de uma fase implica o início de outra.

Comentários:

No Modelo em Cascata, uma fase só se inicia após o término e aprovação da fase anterior, isto é, há uma sequência de desenvolvimento do projeto. Por exemplo, a Fase 4 só é iniciada após o término e aprovação da Fase 3. A Fase 5 só é iniciada após o término e aprovação da Fase 4. Mas que fases são essas? Bem, agora que complica, porque cada autor resolve criar suas fases! Vejamos: (...)

Perfeito, conforme a definição.

Gabarito: C

---

15. **(CESPE – 2010 – BASA – Analista de Sistema)** No modelo em cascata, o projeto segue uma série de passos ordenados. Ao final de cada projeto, a equipe de projeto finaliza uma revisão. O desenvolvimento continua e, ao final, o cliente avalia a solução proposta.

Comentários:

De acordo com Vasconcelos (2006), no Modelo em Cascata, o projeto segue uma série passos ordenados, ao final de cada fase, a equipe de projeto finaliza uma

revisão. Além disso, o desenvolvimento não continua até que o cliente esteja satisfeito com os resultados alcançados. Vocês conseguem perceber como essas restrições engessam o desenvolvimento?

Ao final de cada *projeto*? Não! Ao final de cada fase.

Gabarito: E

---

16. (CESPE – 2009 – TRE/MT – Analista de Sistema - A) O modelo em cascata é apropriado para software em que os requisitos ainda não foram bem compreendidos, pois é focado na criação de incrementos.

Comentários:

Professor, então o Modelo em Cascata não deve ser usado em nenhuma hipótese? Calma lá, ele pode ser usado! No entanto, sua utilização deve ocorrer preferencialmente quando os requisitos forem bem compreendidos e houver pouca probabilidade de mudanças radicais durante o desenvolvimento do sistema. Vocês entenderam?

Pelo contrário, totalmente errado!

Gabarito: E

---

17. (CESPE – 2009 – UNIPAMPA – Analista de Sistema – D) O modelo em cascata sugere uma abordagem sistemática e sequencial para o desenvolvimento de software. Sua natureza linear leva a estados de bloqueio nos quais, para que nova etapa seja iniciada, é necessário que a documentação associada à fase anterior tenha sido aprovada.

Comentários:

No Modelo em Cascata, uma fase só se inicia após o término e aprovação da fase anterior, isto é, há uma sequência de desenvolvimento do projeto. Por exemplo, a Fase 4 só é iniciada após o término e aprovação da Fase 3. A Fase 5 só é iniciada após o término e aprovação da Fase 4. Mas que fases são essas? Bem, agora que complica, porque cada autor resolve criar suas fases! Vejamos: (...)

Perfeito! Não basta terminar uma fase, é necessário que a sua documentação tenha sido aprovada.

Gabarito: C

18. (CESPE – 2004 – ABIN – Analista de Sistema) O modelo de desenvolvimento seqüencial linear, também denominado modelo em cascata, é incompatível com o emprego de técnica de análise orientada a objetos no desenvolvimento de um sistema de informação.

Comentários:

VANTAGENS	DESVANTAGENS
É simples de entender e fácil de aplicar, facilitando o planejamento.	Divisão inflexível do projeto em estágios distintos.
Fixa pontos específicos para a entrega de artefatos.	Dificuldade em incorporar mudanças de requisitos.
Funciona bem para equipes tecnicamente fracas.	Clientes só visualizam resultados próximos ao final do projeto.
É fácil de gerenciar, devido a sua rigidez.	Atrasa a redução de riscos.
Realiza documentação extensa por cada fase ou estágio.	Apenas a fase final produz um artefato de software entregável.
Possibilita boa aderência a outros modelos de processo.	Cliente deve saber todos os requisitos no início do projeto.
Funciona bem com projetos pequenos e com requisitos bem conhecidos.	Não fornece feedback entre as fases.
	Pressupõe que os requisitos ficarão estáveis ao longo do tempo.
	Não funciona bem com projetos complexos e OO, apesar de compatível.

Ele é compatível, mas não é recomendado! *Por que, não?* Imagina um projeto super complexo que utiliza uma análise orientada a objetos (que é um modelo mais sofisticado que a análise estruturada). Lembre-se que, no Modelo em Cascata, você não pode errar, porque se você errar, os riscos de o projeto falhar são enormes! Por essa razão, ele não é recomendável, apesar de compatível!

Gabarito: E

19. (CESPE – 2004 – TRE/AL – Analista de Sistema) O modelo cascata ou ciclo de vida clássico necessita de uma abordagem sistemática, que envolve, em primeiro lugar, o projeto e, em seguida, a análise, a codificação, os testes e a manutenção.

Comentários:

Por Sommerville	Por Royce	Por Pressman (4ª Ed.)	Por Pressman (6ª Ed.)
Análise e Definição de Requisitos	Requisitos de Sistema	Modelagem e Engenharia do Sistema/Informação	Comunicação
Projeto de Sistema e Software	Requisitos de Software	Análise de Requisitos de Software	Planejamento
Implementação e Teste de Unidade	Análise	Projeto	Modelagem
Integração e Teste de Sistema	Projeto	Geração de Código	Construção
Operação e Manutenção	Codificação	Teste e Manutenção	Implantação
	Teste		
	Operação		

Primeiro Projeto e depois Análise? Não, Análise vem antes do Projeto!

Gabarito: E

20. (CESPE – 2008 – MPE/AM – Analista de Sistema) O modelo de desenvolvimento seqüencial linear tem como característica principal a produção de uma versão básica, mas funcional, do software desde as primeiras fases.

Comentários:

VANTAGENS	DESVANTAGENS

É simples de entender e fácil de aplicar, facilitando o planejamento.	Divisão inflexível do projeto em estágios distintos.
Fixa pontos específicos para a entrega de artefatos.	Dificuldade em incorporar mudanças de requisitos.
Funciona bem para equipes tecnicamente fracas.	**Clientes só visualizam resultados próximos ao final do projeto.**
É fácil de gerenciar, devido a sua rigidez.	Atrasa a redução de riscos.
Realiza documentação extensa por cada fase ou estágio.	**Apenas a fase final produz um artefato de software entregável.**
Possibilita boa aderência a outros modelos de processo.	Cliente deve saber todos os requisitos no início do projeto.
Funciona bem com projetos pequenos e com requisitos bem conhecidos.	Não fornece feedback entre as fases.
	Pressupõe que os requisitos ficarão estáveis ao longo do tempo.
	Não funciona bem com projetos complexos e OO, apesar de compatível.

Pelo contrário, somente nas fases finais que se tem uma versão! Essa definição está mais com cara de modelo de desenvolvimento em prototipagem.

Gabarito: E

---

21. (VUNESP - 2012 - SPTrans - Analista de Informática) Uma das abordagens do processo de desenvolvimento da engenharia de software prevê a divisão em etapas, em que o fim de uma é a entrada para a próxima. Esse processo é conhecido como modelo:

a) Transformação.
b) Incremental.
c) Evolutivo.
d) Espiral.
e) Cascata.

Comentários:

No Modelo em Cascata, uma fase só se inicia após o término e aprovação da fase anterior, isto é, há uma sequência de desenvolvimento do projeto. Por exemplo, a Fase 4 só é iniciada após o término e aprovação da Fase 3. A Fase 5 só é iniciada após o término e aprovação da Fase 4. Mas que fases são essas? Bem, agora que complica, porque cada autor resolve criar suas fases!

Conforme vimos em aula, trata-se do Modelo em Cascata.

Gabarito: E

---

22. (CESGRANRIO – 2010 – PETROBRÁS – Analista de Sistemas – Processos de Negócio) No Ciclo de Vida Clássico, também chamado de Modelo Sequencial Linear ou Modelo Cascata, é apresentada uma abordagem sistemática composta pelas seguintes atividades:

a) Análise de Requisitos de Software, Projeto, Geração de Código, Teste e Manutenção.

b) Modelagem e Engenharia do Sistema/Informação, Análise de Requisitos de Software, Projeto, Geração de Código, Teste e Manutenção.

c) Modelagem e Engenharia do Sistema/Informação, Projeto, Geração de Código, Teste e Manutenção.

d) Levantamento de Requisitos de Software, Projeto, Geração de Código e Manutenção e Análise de Requisitos de Software.

e) Levantamento de Requisitos de Software, Projeto, Geração de Código, Teste Progressivo e Manutenção.

Comentários:

Por Sommerville	Por Royce	Por Pressman (4ª Ed.)	Por Pressman (6ª Ed.)
Análise e Definição de Requisitos	Requisitos de Sistema	Modelagem e Engenharia do Sistema/Informação	Comunicação
Projeto de Sistema e Software	Requisitos de Software	Análise de Requisitos de Software	Planejamento

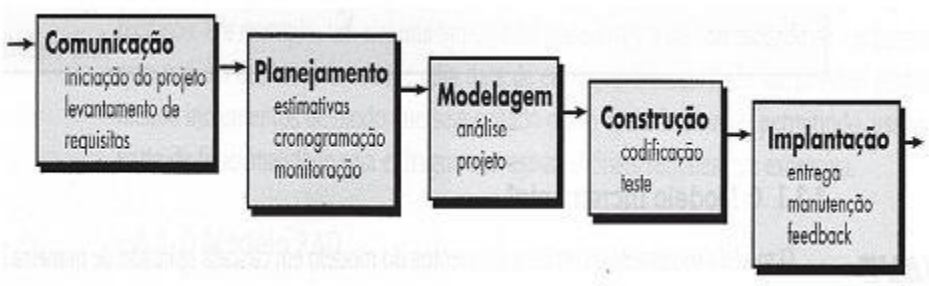
Implementação e Teste de Unidade	Análise	Projeto	Modelagem
Integração e Teste de Sistema	Projeto	Geração de Código	Construção
Operação e Manutenção	Codificação	Teste e Manutenção	Implantação
	Teste		
	Operação		

A Letra B está correta de acordo com o Pressman 4ª Edição, mas está errada de acordo com o Pressman 6ª Edição. Ademais, na questão ele sequer disse que era de acordo com o Pressman. Portanto, percebam que é um assunto polêmico e que as bancas deveriam ignorar, mas eventualmente elas cobram mesmo assim.

Gabarito: B

---

23. (FGV - 2015 – PGE/RO - Análise de Sistemas) A figura abaixo ilustra um modelo de processo, que prescreve um conjunto de elementos de processo como atividades de arcabouço, ações de engenharia de software, tarefas, produtos de trabalho, mecanismos de garantia de qualidade e de controle de modificações para cada projeto.



Esse modelo é conhecido como Modelo:

a) por funções.
b) em cascata.
c) incremental.
d) em pacotes.
e) por módulos.

Comentários:

Por Sommerville	Por Royce	Por Pressman (4ª Ed.)	Por Pressman (6ª Ed.)
Análise e Definição de Requisitos	Requisitos de Sistema	Modelagem e Engenharia do Sistema/Informação	Comunicação
Projeto de Sistema e Software	Requisitos de Software	Análise de Requisitos de Software	Planejamento
Implementação e Teste de Unidade	Análise	Projeto	Modelagem
Integração e Teste de Sistema	Projeto	Geração de Código	Construção
Operação e Manutenção	Codificação	Teste e Manutenção	Implantação
	Teste		
	Operação		

Conforme vimos em aula, trata-se das fases descritas pelo Pressman para o Modelo em Cascata.

Gabarito: B

---

24. (CESPE - 2016 – TCE/PR - Analista de Informática) O modelo de desenvolvimento em cascata é utilizado em caso de divergência nos requisitos de um software, para permitir a evolução gradual do entendimento dos requisitos durante a implementação do software.

Comentários:

Essa questão trata, na verdade, do modelo iterativo e incremental e, não, em cascata.

Gabarito: E

---

25. (CESGRANRIO – 2012 – Transpetro – Analista de Sistemas Júnior – Infra-estrutura) Na engenharia de software, existem diversos modelos de

desenvolvimento de software, e, dentre eles, o modelo em cascata, o qual, no contexto do desenvolvimento de sistemas de software complexos, recomenda:

a) distribuir a elicitação dos requisitos desde o início até o fim do desenvolvimento.

b) dividir o desenvolvimento do produto de software em fases lineares e sequenciais.

c) enfatizar a avaliação e mitigação de riscos durante o desenvolvimento.

d) realizar entregas incrementais do produto de software ao longo do desenvolvimento.

e) usar prototipagem rápida para estimular o envolvimento do usuário no desenvolvimento.

Comentários:

Recomenda dividir o desenvolvimento do produto de software em fases lineares e sequenciais. *Vocês se lembram que o Modelo em Cascata é um modelo sequencial linear?* Pois é!

Gabarito: B

---

26. (CESGRANRIO – 2012 – EPE – Analista de Gestão Corporativa – Tecnologia da Informação) Uma das críticas feitas ao modelo do ciclo de vida do desenvolvimento de software em cascata refere-se a:

a) exigência de conhecimento de avaliação e gerenciamento de risco para evitar grandes surpresas no projeto.

b) comprometimentos na qualidade e nas possibilidades de manutenção a longo prazo, parecendo um protótipo.

c) pouca flexibilidade para mudanças futuras, exigindo compromissos nas fases iniciais do projeto.

d) pouca visibilidade das etapas do processo, tornando cara a documentação de todas as versões dos sistemas.

e) exigências de velocidade as quais levam o engenheiro de software a utilizar linguagens, algoritmos ou ferramentas ineficientes ao longo de todo o projeto.

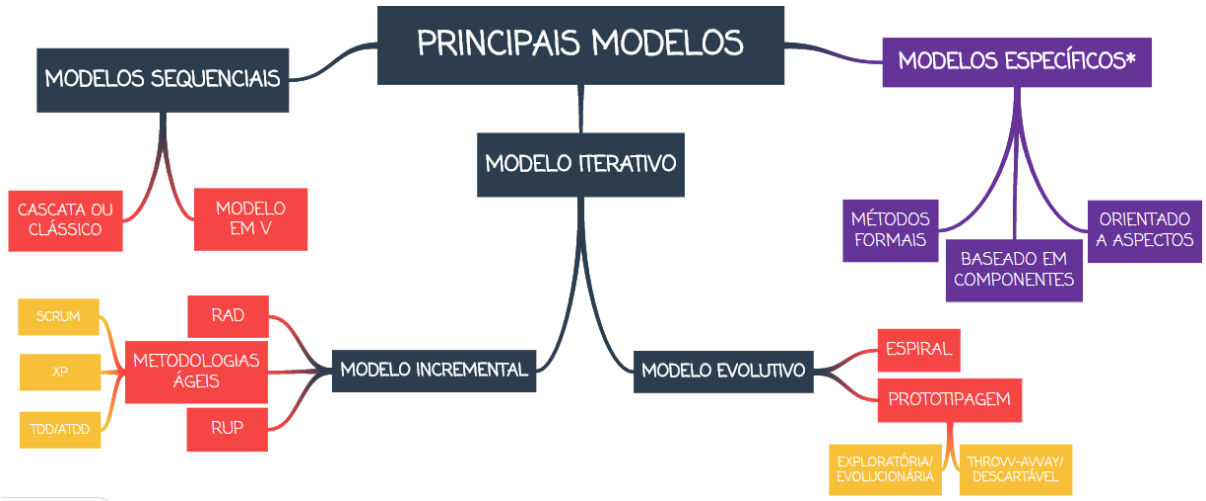
Comentários:

É conhecida a pouca flexibilidade do Modelo em Cascata em se adaptar a mudanças conforme o projeto progride. Mudanças nas fases iniciais do projeto têm menor impacto do que as que são necessárias nas fase finais do projeto. Por isso, o Modelo em Cascata é usado preferencialmente em projetos com requisitos muito bem definidos e com pouca volatilidade.

Gabarito: C

ACERTEI	ERREI

MODELOS ITERATIVOS E INCREMENTAIS



Como foi dito anteriormente, o Modelo em Cascata acumulava riscos e vários projetos começaram a fracassar ao utilizá-lo no mundo inteiro. Então surgiu o Modelo Iterativo como uma tentativa de resolver esse problema de acúmulo de riscos. Vejamos a diferença fundamental entre o Modelo em Cascata e o Modelo Iterativo:

- No Modelo em Cascata, caso haja cem requisitos, analisam-se os cem requisitos, projetam-se os cem requisitos, codificam-se os cem requisitos, testam-se os cem requisitos, e assim, por diante, sequencialmente.

- No Modelo Incremental, caso haja cem requisitos, dividem-se os cem requisitos em vinte miniprojetos de cinco requisitos e utiliza-se o modelo em cascata para cada miniprojeto.

- No Modelo Iterativo, caso haja cem requisitos, analisam-se, projetam-se, codificam-se, testam-se os cem requisitos, porém os requisitos são entregues incompletos e eu repito esse ciclo de refinamento até chegar ao produto final.

Galera, pensem só: é possível combinar a abordagem incremental com uma abordagem iterativa para desenvolver os miniprojetos e entregar partes diferentes do projeto. A imagem abaixo apresenta miniprojetos sendo feitos iterativamente em um modelo iterativo e incremental. Observem que cada iteração (passagem pelas fases de desenvolvimento) refinam a funcionalidade.

PRIMEIRA ITERAÇÃO

Assim, os resultados são mais rápidos, há maior interação com o usuário e há um feedback mais intenso – é possível reagir mais facilmente a mudanças. Essa abordagem permite gerenciamento e mitigação de riscos. *Professor, mas eu fiquei com uma dúvida: qual a diferença entre Modelo Iterativo e Modelo Incremental? Ou eles são exatamente a mesma coisa?*

Bem, galera... eu nunca vi nenhuma prova cobrar essa diferença entre modelo iterativo e modelo incremental! Na verdade, quando se fala em modelo iterativo, presume-se que é incremental (ou evolucionário) e quando se fala em modelo incremental, presume-se que é iterativo. Eles frequentemente andam lado a lado, mas há pequenas diferenças.

IMPORTANTE

Galera, eu já vi essas palavras serem trocadas dezenas de vezes (inclusive em edital). Infelizmente, algumas vezes as bancas não acatam os recursos! De todo modo, saibam que:

- **Iterativo:** reiterado ou repetitivo;
- **Interativo:** participação ou ação mútua.

Professor, mas e se cair em prova? Ora, caso caia em prova, a diferença é que, no modelo incremental, há várias equipes desenvolvendo uma parte do software a serem integradas no fim e, no modelo iterativo, lança-se a versão 1.0, adicionam-se funcionalidades, lança uma versão 2.0, adicionam-se mais funcionalidades e assim por diante.

Modelo Incremental: observem que a imagem mostra um artista com uma ideia completa sobre o quadro, mas ele desenvolve cada parte separadamente até integrar as partes em uma imagem completa. É como se fosse um quebra-cabeças em que cada parte é entregue funcionando e depois integrada. Produz builds, i.e., partes do software.

Modelo Iterativo: observem que a imagem mostra um artista com um esboço do quadro, sendo que ele desenvolve várias versões do quadro até chegar ao resultado final. É como se fosse uma visão abstrata da imagem, que em seguida vai sendo melhorada até chegar a uma visão mais concreta. Produz releases, i.e., versões constantemente melhoradas da imagem.

Uma das vantagens do modelo iterativo e incremental é que o cliente pode receber e avaliar as entregas do produto mais cedo, já no início do desenvolvimento do software. Além disso, há maior tolerância a mudanças com consequência direta de redução do risco de falha do projeto, i.e., ele acomoda melhor mudanças. Ele aumenta o reúso e a qualidade.

1. (CESPE - 2011 – TJ/ES - Analista Judiciário - Análise de Sistemas - Específicos) O modelo de processo incremental de desenvolvimento de software é iterativo, assim como o processo de prototipagem. Contudo, no processo incremental, diferentemente do que ocorre no de prototipagem, o objetivo consiste em apresentar um produto operacional a cada incremento.

Comentários:

De fato, no modelo iterativo e incremental, apresenta-se sempre um produto a cada incremento. Já na prototipação, não. Idealmente, ele serve apenas para identificar requisitos.

Gabarito: C

---

2. (CESPE - 2008 – TJ/DF - Analista Judiciário - Análise de Sistemas) No modelo de desenvolvimento incremental, embora haja defasagem entre os períodos de desenvolvimento de cada incremento, os incrementos são desenvolvidos em paralelo.

Comentários:

Questão perfeita. Os incrementos são codificados não exatamente em paralelo – há uma pequena defasagem.

Gabarito: C

---

3. (CESPE - 2009 – UNIPAMPA - Análise de Sistemas) No modelo de desenvolvimento incremental, a cada iteração são realizadas várias tarefas. Na fase de análise, pode ser feito o refinamento de requisitos e o refinamento do modelo conceitual.

Comentários:

Perfeito, é a fase seguinte à fase de requisitos e busca refiná-los.

Gabarito: C

4. (CESPE - 2016 – TCE/PR – Analista de Sistemas – C) No modelo iterativo de desenvolvimento de software, as atividades são dispostas em estágios sequenciais.

Comentários:

A questão trata do modelo em cascata de desenvolvimento de software. No modelo iterativo e incremental, as atividades são dispostas ao longo de diversas iterações

Gabarito: E

ACERTEI	ERREI

NBR ISO/IEC 12207

Pessoal, quando se lê um edital, é preciso analisá-lo friamente para descobrir as suas falhas e encontrar o custo-benefício de cada assunto! Eu sei que vocês têm que decorar dezenas, se não centenas de processos de outras disciplinas e infelizmente essa norma também possui mais uma pancada (no total, são 43!). Eu, particularmente, não considero um bom custo-benefício! Mas vamos lá...

Primeiro de tudo, temos que fazer uma importante observação: a Norma 12207 foi lançada em outubro de 1995 (no Brasil, 1998)! Com nome de ABNT NBR ISO/IEC 12207, ela foi amplamente revisada[1] em março de 2009! Enfim... o curso aqui ministrado será baseado nessa última versão, tendo em vista que a outra está bastante desatualizada e foi cancelada, como afirma a própria norma abaixo:

> *Esta segunda edição* **cancela e substitui** *a edição anterior (ABNT NBR 12207:1998), a qual foi tecnicamente revisada. Ela também incorpora Emendas ISO/IEC 12207:1995/Emenda 1:2002 e ISO/IEC 12207:1995/Emenda 2:2004.*

A ABNT NBR ISO/IEC 12207 estabelece uma estrutura para processos de ciclo de vida de software! *Galera, o que essa norma faz?* Ela estabelece uma estrutura para processos de ciclo de vida de software! *Ela estabelece o que?* Uma estrutura para processos de ciclo de vida de software! *Por que essa insistência?* Porque essa frase minúscula pode responder várias questões de prova, portanto decorem-na!

Como eu já disse, a norma possui diversos processos que são aplicados durante a aquisição e configuração dos serviços do sistema e tem o objetivo de fornecer uma estrutura comum tanto para quem está adquirindo quanto para quem está fornecendo, além dos desenvolvedores, mantenedores, operadores, gerentes, técnicos e outros envolvidos no desenvolvimento.

É uma linguagem comum para todos os interessados estabelecida na forma de processos bem definidos, que cobrem o ciclo de vida desde a concepção de ideias até a descontinuação do software. Ela ajuda organizações a firmarem contratos e executarem projetos de forma mais eficiente, na medida em que ela compreende todos os processos e relacionamentos.

[1] Eventualmente, vocês encontrarão questões sobre a versão anterior, porém como já foi visto ela foi cancelada.

Aplica-se para a aquisição de sistemas e produtos de software e serviços, para a indústria de software. Aplica-se para a aquisição de sistemas e produtos de software e serviços, para o fornecimento, desenvolvimento, operação, manutenção e desativação de produtos de software e parte de software de um sistema, se realizado em uma organização ou fora dela.

Esta norma possui grande consonância com a ISO/IEC 15288. Essa serve como padrão de engenharia de sistemas para cobrir processos e estágios de ciclo de vida. Percebam que a 12207 é para ciclo de vida de software e a 15288 é para ciclo de vida de sistemas. Ela, inclusive, afirma que futuramente pretende-se obter uma visão totalmente harmonizada dos processos do ciclo de vida de sistema e de software.

Esta norma pode ser usada em quatro ocasiões:

Por uma organização – para ajudar a estabelecer um ambiente de processos desejados, podendo ser sustentados por uma infraestrutura de métodos, procedimentos, técnicas, ferramentas e pessoal treinado. A organização pode empregar o ambiente para realizar e gerenciar projetos e sistemas em andamento durante as fases do ciclo de vida, podendo ser usada para avaliar a conformidade de um conjunto estabelecido de processos do ciclo de vida.

Por um projeto – para ajudar a selecionar, estruturar e utilizar os elementos de um conjunto de processos de ciclo de vida estabelecidos que forneçam produtos e serviços. Desse modo, esta norma pode ser usada na avaliação de conformidade do projeto para o ambiente estabelecido e declarado.

Por um adquirente e um fornecedor – para ajudar a estabelecer um acordo em relação aos processos e às atividades. Esse acordo contempla os processos e atividades desta Norma que são selecionados, negociados, acordados e executados. Assim a Norma pode ser usada para orientar a definição do acordo.

Por organizações e avaliadores – para realizar avaliações que possam ser usados para apoiar a melhoria de processos organizacionais.

A norma agrupa as atividades que podem ser executadas durante o ciclo de vida de um sistema de software em sete grupos de processos. Cada um dos processos dentro desses grupos é descrito em termos de seus propósitos e resultados desejados, e lista atividades e tarefas que precisam ser desempenhadas para atingir esses resultados. Os grupos são:

1. Processos Contratuais (2 Processos)

Esses processos definem as atividades necessárias para estabelecer um acordo entre duas organizações. Se o Processo de Aquisição for acionado, fornece meios para conduzir negócios com um fornecedor de produtos, que fornece o sistema operacional a ser usado, serviços de suporte a um sistema operacional, ou de elementos de um sistema desenvolvido pelo projeto.

Se o Processo de Fornecimento for acionado, fornece meios para conduzir um projeto cujo resultado é o produto ou serviço que será entregue ao seu adquirente.

2. Processos Organizacionais Capacitadores de Projeto (5 Processos)

Esses processos gerenciam o potencial da organização em adquirir e fornecer produtos ou serviços através da iniciação, suporte e controle dos projetos. Fornecem recursos e infraestrutura necessários para suporte aos projetos e para garantir que os objetivos organizacionais e acordos estabelecidos sejam atendidos. Não se pretende que sejam conjuntos completos que permitam a gestão dos negócios.

3. Processos de Projeto (7 Processos)

Projetos descrevem processos referentes ao planejamento, avaliação e controle. Os princípios relacionados a esses processos podem ser aplicados a qualquer área de gestão de uma organização. Há duas categorias: Processos de Gestão de Projeto (para planejar, executar, avaliar e controlar o progresso de um projeto) e Processos de Apoio ao Projeto (para suportar os objetivos específicos de gerenciamento).

4. Processos Técnicos (11 Processos)

Esses processos definem as atividades que permitem às funções organizacionais e às de projeto otimizar os benefícios e reduzir os riscos que surgem das decisões e das ações técnicas. Eles definem requisitos de um sistema, para transformar os

requisitos em um produto eficaz, permitir a reprodução consistente de um produto, para utilizar, fornecer, sustentar o fornecimento e descartar o produto ou serviço.

Essas atividades permitem que produtos e serviços possuam atributos de oportunidade/conveniência, disponibilidade, eficiência em custo, funcionalidade, confiabilidade, manutenibilidade, produtividade, usabilidade e outras qualidades exigidas pelas organizações adquirentes e fornecedoras. Permitem que produtos e serviços estejam de acordo com os requisitos legais da sociedade.

5. Processos de Implementação de Software (7 Processos)

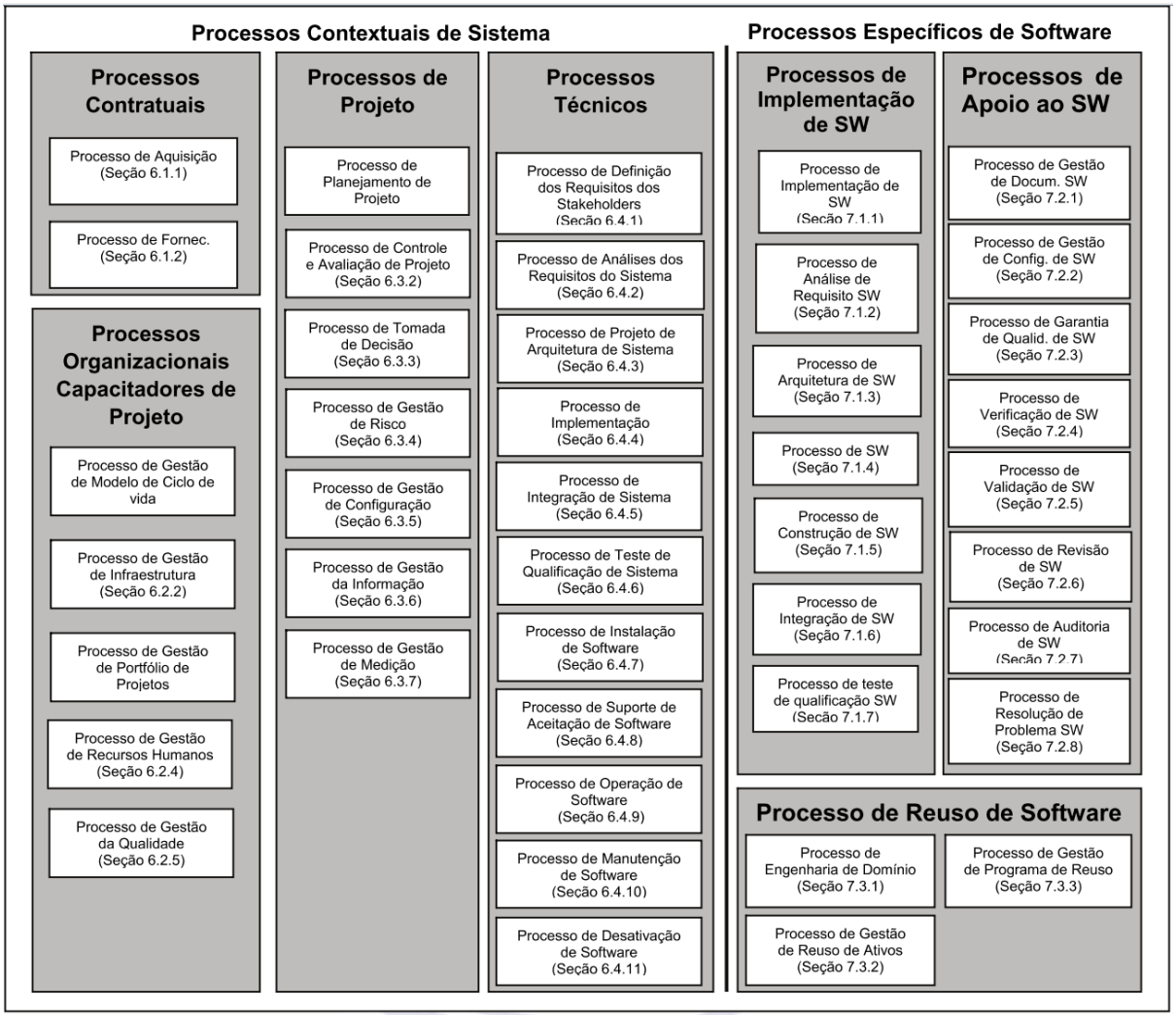
Esses processos são utilizados para produzir um elemento específico do sistema especificado (Item de Software) implementado em software. Estes processos transformam comportamentos, interfaces e restrições de implementação específicas em ações de implementação que resultam em um elemento de sistema que atenda aos requisitos derivados dos requisitos do sistema.

6. Processos de Apoio ao Software (8 Processos)

Esses processos fornecem um conjunto específico e focado de atividades para execução de um processo de software especializado de software. Um processo de apoio dá assistência ao Processo de Implementação de Software como uma parte integral com um propósito distinto, contribuindo para o sucesso e qualidade do projeto de software.

7. Processos de Reúso de Software (3 Processos)

Esse grupo é composto por três processos que apoiam a capacidade da organização em reutilizar os itens de software entre projetos distintos. Esses processos são únicos, porque – pela sua natureza inerente – operam fora dos limites de qualquer projeto particular.



PROCESSOS CONTRATUAIS

- **Processo de Aquisição:** busca obter um produto e/ou serviço que satisfaça a necessidade expressa pelo adquirente. O processo inicia com a identificação de uma necessidade do adquirente e termina com a aceitação do produto e/ou serviço. Possui as seguintes atividades:

1. Preparação para Aquisição
2. Anúncio da Aquisição
3. Seleção do Fornecedor
4. Acordo Contratual
5. Monitoração do Acordo
6. Aceite do Adquirente
7. Fechamento

- Processo de Fornecimento: busca fornecer um produto ou serviço para o adquirente que satisfaça os requisitos combinados. Possui as seguintes atividades:

    1. Identificação de Oportunidade
    2. Proposta de Fornecedores
    3. Acordo Contratual
    4. Execução do Contrato
    5. Entrega e Suporte de Produto/Serviço
    6. Fechamento

PROCESSOS ORGANIZACIONAIS CAPACITADORES DE PROJETO

- Processo de Gestão de Modelo de Ciclo de Vida: busca definir, manter e garantir a disponibilidade de políticas, processos de ciclo de vida, os modelos de ciclo de vida e os procedimentos para utilização pela organização. Possui as seguintes atividades:

    1. Estabelecimento do Processo
    2. Avaliação do Processo
    3. Melhoria do Processo

- Processo de Gestão de Infraestrutura: busca fornecer a infraestrutura capacitadora e serviços a projetos de modo a apoiar os objetivos do projeto e da organização durante o ciclo de vida. Possui as seguintes atividades:

    1. Implementação do Processo
    2. Estabelecimento da Infraestrutura
    3. Manutenção da Infraestrutura

- Processo de Gestão de Portfólio de Projetos: busca iniciar e sustentar projetos necessários, suficientes e adequados, a fim de satisfazer os objetivos estratégicos da organização. Possui as seguintes atividades:

    1. Iniciação de Projeto
    2. Avaliação de Portfólio
    3. Encerramento do Projeto

- Processo de Gestão de Recursos Humanos: busca prover à organização os recursos humanos necessários e manter suas habilidades consistentes com as necessidades do negócio. Possui as seguintes atividades:

    1. Identificação de habilidades
    2. Desenvolvimento de habilidades
    3. Aquisição e fornecimento de habilidades
    4. Gestão de conhecimento

- Processo de Gestão da Qualidade: busca garantir que produtos, serviços e implementações de processos de ciclo de vida alcancem os objetivos de qualidade da organização e a satisfação do cliente. Possui as seguintes atividades:

    1. Gestão da qualidade
    2. Ação corretiva de gestão de qualidade

PROCESSOS DE PROJETO

- Processo de Planejamento de Projeto: busca produzir e comunicar planejamentos de projeto eficazes e viáveis. Este processo determina o objetivo das atividades técnicas e de gestão de projeto, identifica os resultados esperados do processo, tarefas e entregáveis do projeto e estabelece cronogramas para a condução das tarefas do projeto. Possui as seguintes atividades:

    1. Iniciação de Projeto
    2. Planejamento de Projeto
    3. Ativação do Projeto

- Processo de Controle e Avaliação de Projeto: busca determinar o status do projeto e garantir que o projeto seja realizado de acordo com os planos e cronogramas, e dentro dos orçamentos estimados, e que satisfaça os objetivos técnicos. Possui as seguintes atividades:

    1. Monitoração do Projeto
    2. Controle do Projeto
    3. Avaliação do Projeto
    4. Encerramento do Projeto

- Processo de Tomada de Decisão: busca selecionar o curso de ação mais benéfico para o projeto entre as alternativas existentes, respondendo a uma solicitação de decisão encontrada durante o ciclo de vida do sistema, independentemente da natureza ou fonte, a fim de obter os resultados especificados, desejados ou otimizados. Possui as seguintes atividades:

    1. Planejamento da decisão
    2. Análise da decisão
    3. Acompanhamento da decisão

- Processo de Gestão de Risco: busca identificar, analisar, tratar e monitorar os riscos de forma continua, por meio de uma abordagem sistemática durante todo ciclo de vida de um sistema, produto ou serviço de software. Pode ser aplicado para riscos relacionados à aquisição, desenvolvimento, manutenção ou operação de um sistema. Possui as seguintes atividades:

    1. Planejamento de gestão de risco
    2. Gestão de perfil de risco
    3. Análise da decisão
    4. Tratamento do risco
    5. Monitoração de risco
    6. Avaliação do processo de gestão de risco

- Processo de Gestão de Configuração: busca estabelecer e manter a integridade de todos os produtos identificados de um projeto ou processo e torná-los disponíveis às partes interessadas. Possui as seguintes atividades:

    1. Planejamento de gestão de configuração
    2. Execução da gestão de configuração

- Processo de Gestão da Informação: busca fornecer informações relevantes, completas, válidas e, se necessário, confidenciais para as partes designadas e, se pertinente, após o ciclo de vida do sistema. Possui as seguintes atividades:

    1. Planejamento de gestão da informação
    2. Execução de gestão da informação

- Processo de Medição: busca coletar, analisar e reportar dados relativos aos produtos desenvolvidos e processos implementados dentro da unidade

organizacional, para apoiar a gestão eficaz dos processos, e demonstrar de forma objetiva a qualidade dos produtos. Possui as seguintes atividades:

1. Planejamento da medição
2. Desempenho da medição
3. Avaliação da medição

PROCESSOS TÉCNICOS

- Processo de Definição dos Requisitos dos Stakeholders: busca identificar as partes interessadas e suas classes, envolvidas com o sistema durante o ciclo de vida deste, e as necessidades e desejos das partes. Analisa e transforma essas necessidades em um conjunto comum de requisitos que expressam a interação pretendida que o sistema terá com seu ambiente operacional. Possui as seguintes atividades:

  1. Identificação dos stakeholders
  2. Identificação dos requisitos
  3. Avaliação dos requisitos
  4. Acordo dos requisitos
  5. Registro dos requisitos

- Processo de Análise dos Requisitos do Sistema: busca transformar os requisitos dos stakeholders em um conjunto de requisitos técnicos do sistema desejados que proporcionarão orientação ao design do sistema. Possui as seguintes atividades:

  1. Especificação dos requisitos
  2. Avaliação dos requisitos

- Processo de Projeto de Arquitetura de Sistema: busca identificar quais requisitos do sistema são alocados para cada elemento do sistema. Possui as seguintes atividades:

  1. Estabelecimento da arquitetura
  2. Avaliação da arquitetura

- Processo de Implementação: busca executar um elemento específico do sistema, i.e., é a programação ou implementação em si.

- Processo de Integração do Sistema: busca integrar os elementos do sistema (incluindo itens de software, itens de hardware, operações manuais, e outros sistemas, conforme necessário) para produzir um sistema completo que satisfará o design do sistema e as expectativas do cliente expressadas nos requisitos do sistema. Possui as seguintes atividades:

    1. Integração
    2. Prontidão do teste

- Processo de Teste de Qualificação de Sistema: busca assegurar que a implementação de cada requisito do sistema seja testada para a verificação de conformidade e se o sistema está pronto para entrega. Possui as seguintes atividades:

    1. Teste de qualificação

- Processo de Instalação de Software: busca instalar o produto de software que satisfaça os requisitos combinados no ambiente-alvo. Possui as seguintes atividades:

    1. Instalação de software

- Processo de Suporte na Aceitação de Software: busca auxiliar o adquirente a ter confiança de que o produto atenderá aos requisitos. Possui as seguintes atividades:

    1. Processo de aceitação de software

- Processo de Operação de Software: busca operar o produto de software em seu ambiente e fornecer suporte aos clientes desse produto. Possui as seguintes atividades:

    1. Preparação para a operação
    2. Teste operacional e check-out
    3. Uso operacional
    4. Suporte ao cliente
    5. Resolução de problema de operação

- Processo de Manutenção de Software: busca fornecer suporte com boa relação custo-benefício ao produto de software entregue. Possui as seguintes atividades:

1. Implementação do processo
2. Análise de modificação e de problema
3. Implementação da modificação
4. Revisão/Aceitação de manutenção
5. Migração
6. Descontinuação do Software

- Processo de Desativação de Software: busca concluir a existência de uma entidade de software de sistema. Possui as seguintes atividades:

1. Planejamento de desativação de software

PROCESSOS DE IMPLEMENTAÇÃO DE SOFTWARE

- Processo de Implementação de Software: busca produzir um item de sistema especificado implementado como um produto ou serviço de software. Possui as seguintes atividades:

1. Estratégia de implementação de software

- Processo de Análise de Requisitos de Software: busca estabelecer os requisitos dos elementos de software do sistema. Possui as seguintes atividades:

1. Análise de requisitos de software

- Processo de Arquitetura de Software: busca fornecer um projeto para o software que implemente e possa ser verificado com base em seus requisitos. Possui as seguintes atividades:

1. Projeto de arquitetura de software

- Processo de Projeto de Software: busca fornecer um projeto para o software que implemente e possa ser verificado de acordo com seus requisitos e a arquitetura de software e que seja detalhado suficientemente para permitir codificação e testes. Possui as seguintes atividades:

1. Projeto de software

- Processo de Construção de Software: busca produzir unidades de software executáveis que refletem apropriadamente o projeto de software. Possui as seguintes atividades:

    1. Construção de software

- Processo de Integração de Software: busca combinar as unidades de software e componentes de software, produzindo itens de software integrados, consistentes com o projeto de software, que demonstrem que os requisitos funcionais e não-funcionais de software são satisfeitos em uma plataforma operacional completa ou equivalente. Possui as seguintes atividades:

    1. Integração de software

- Processo de Testes de Qualificação de Software: busca confirmar que o produto de software integrado atende a seus requisitos definidos. Possui as seguintes atividades:

    1. Testes de qualificação de software

PROCESSOS DE APOIO AO SOFTWARE

- Processo de Gestão de Documentação de Software: busca desenvolver e manter o registro das informações do software produzidas por um processo. Possui as seguintes atividades:

    1. Projeto e desenvolvimento
    2. Manutenção

- Processo de Gestão de Configuração de Software: busca estabelecer e manter a integridade dos itens de software de um processo ou projeto e disponibilizá-los para as partes envolvidas. Possui as seguintes atividades:

    1. Implementação do processo
    2. Identificação de configuração
    3. Controle de configuração
    4. Relato da situação da configuração
    5. Avaliação de configuração
    6. Gestão de liberação e entrega

- Processo de Garantia da Qualidade de Software: busca fornecer garantia de que os produtos e processos de trabalho estão em conformidade com os planos e condições predefinidos. Possui as seguintes atividades:

    1. Implementação do processo
    2. Garantia de produto
    3. Garantia de processo
    4. Garantia de sistemas de qualidade

- Processo de Verificação de Software: busca confirmar que cada produto de trabalho e/ou serviço de software de um processo ou projeto reflete apropriadamente os requisitos especificados. Possui as seguintes atividades:

    1. Implementação do processo
    2. Verificação

- Processo de Validação de Software: busca confirmar se os requisitos de um uso específico pretendido para o produto de software são atendidos. Possui as seguintes atividades:

    1. Implementação do processo
    2. Validação

- Processo de Revisão de Software: busca manter um entendimento comum com as partes interessadas sobre os progressos obtidos em relação aos objetivos acordados, além do que convém ser feito para garantir que o desenvolvimento do produto satisfaça os seus stakeholders. Possui as seguintes atividades:

    1. Implementação do processo
    2. Revisões de gestão de projeto
    3. Revisões técnicas

- Processo de Auditoria de Software: busca determinar, independentemente, a conformidade dos produtos e processos selecionados com os requisitos, planos e contrato, quando apropriado. Possui as seguintes atividades:

    1. Implementação do processo
    2. Auditoria de software

- Processo de Resolução de Problemas de Software: busca garantir que todos os problemas são identificados, analisados, gerenciados e controlados até sua resolução. Possui as seguintes atividades:

    1. Implementação do processo
    2. Resolução de problemas

PROCESSOS DE REÚSO DE SOFTWARE

- Processo de Engenharia de Domínio: busca desenvolver e manter modelos, arquiteturas e ativos de domínio. Possui as seguintes atividades:

    1. Implementação do processo
    2. Análise de domínio
    3. Design de domínio
    4. Fornecimento de ativos
    5. Manutenção de ativos

- Processo de Gestão de Reúso de Ativos: busca gerenciar a vida de ativos reutilizáveis desde a concepção até sua desativação. Possui as seguintes atividades:

    1. Implementação do processo
    2. Definição do armazenamento e recuperação
    3. Gestão e controle de ativos

- Processo de Gestão do Programa de Reúso: busca planejar, estabelecer, gerenciar, controlar e monitorar o programa de reúso da organização e sistematicamente, explorar oportunidades de reúso. Possui as seguintes atividades:

    1. Iniciação
    2. Identificação de domínio
    3. Avaliação de reúso
    4. Planejamento
    5. Execução e controle
    6. Revisão e avaliação

Pronto, matamos a teoria sobre esse assunto! *Vamos ver um resuminho?* Pois bem, a norma tem o objetivo de estabelecer uma estrutura comum para os processos de

ciclo de vida e de desenvolvimento de software visando ajudar organizações a compreenderem componentes presentes na aquisição e fornecimento de software e, assim, firmar contratos e executarem projetos de forma eficaz.

Ela também estabelece uma arquitetura de alto nível do ciclo de vida de software que é construída a partir de um conjunto de processos e seus inter-relacionamentos. Os processos são descritos tanto em nível de propósito/saídas quanto em termos de atividades. É importante saber também que ela é independente de métodos, ferramentas, treinamentos, métricas ou tecnologias empregadas.

Dessa forma, ela pode ser utilizada mundialmente com qualquer modelo de ciclo de vida, método ou técnica de engenharia de software ou linguagem de programação. É uma norma que diz o que fazer e, não, como fazer! Lembrem-se também que se trata de uma norma cujos processos são modulares, fortemente coesos e fracamente acoplados.

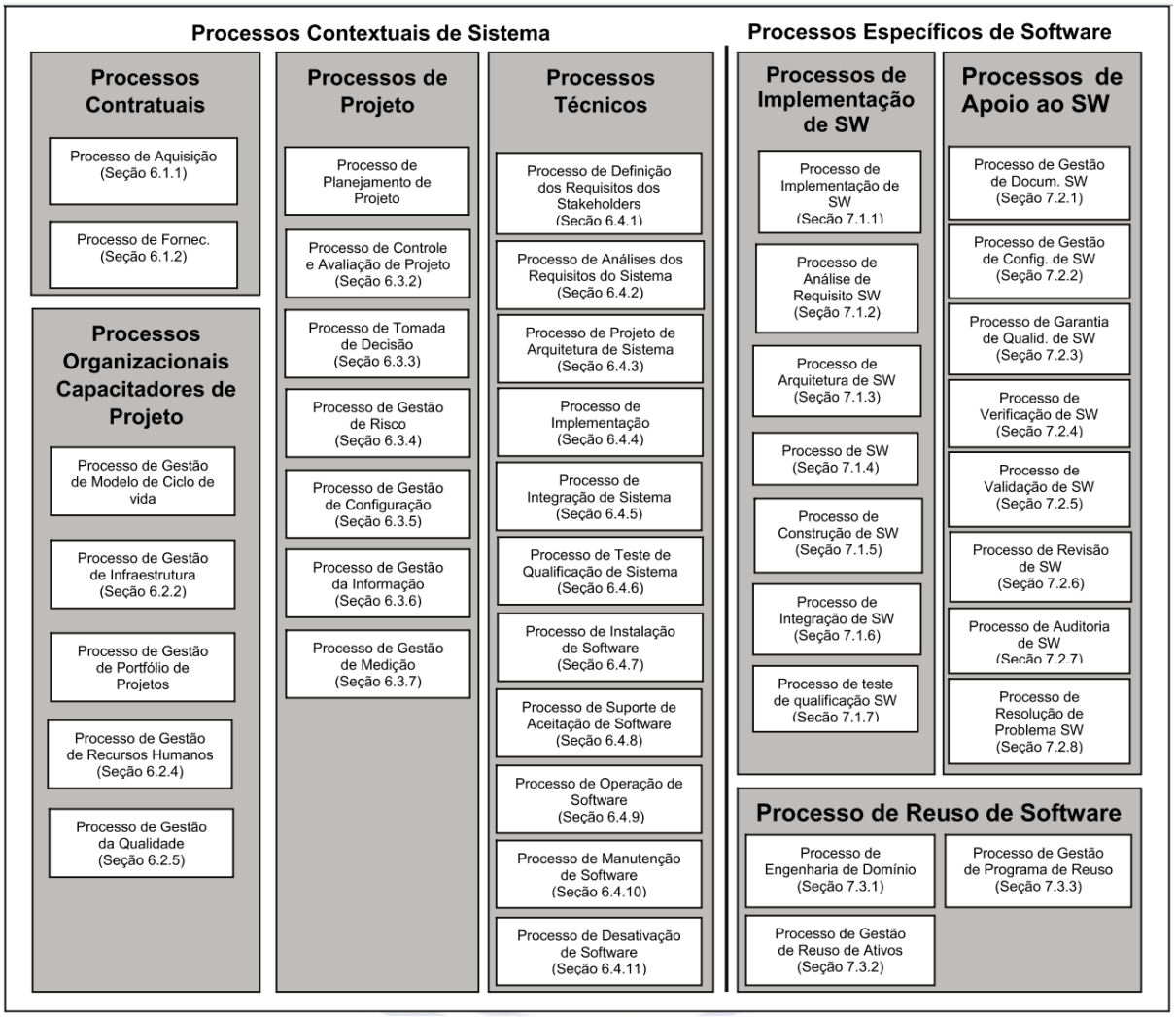
Temos um relacionamento muitos para muitos entre processos e organizações, ou seja, processos podem ser executados por uma ou mais organizações e uma organização pode executar um ou mais processos – não há obrigação de executar todos os processos! A norma apenas fornece uma estrutura para que a organização defina seus processos como achar mais adequado.

Nós temos sete grupos de processos e, a partir daí, é necessário um pouco de decoreba infelizmente. *Bacana, pessoal?*

1. (FCC - 2010 – TRT/PR – Técnico Judiciário – Tecnologia da Informação) Manutenção de software, segundo a norma ISO 12207, trata-se de um processo dentro do grupo de processos:

a) de projeto.
b) de reúso de software.
c) de implementação de software.
d) de suporte de software.
e) técnicos.

Comentários:



São 43 processos divididos em 7 grupos. A questão pede qual grupo se encontra o processo de Manutenção de Software. Ele se encontra no grupo de Processos Técnicos!

Gabarito: E

---

2. **(FCC - 2010 – TRF/04 – Analista Judiciário – Tecnologia da Informação)** Sobre a norma ISO/IEC 12207, considere:

I. Define objetivos, níveis de maturidade organizacional ou de capacidade de processo.

II. Provê uma estrutura para que uma organização defina seus processos.

III. Cobre também a garantia da qualidade, que se estende desde os produtos adquiridos ou fornecidos até a qualidade e melhoria dos processos de implementação.

a) I, apenas.
b) I, II e III.
c) I e II, apenas.
d) II e III, apenas.
e) III, apenas.

Comentários:

Temos um relacionamento muitos para muitos entre processos e organizações, ou seja, processos podem ser executados por uma ou mais organizações e uma organização pode executar um ou mais processos – não há obrigação de executar todos os processos! A norma apenas fornece uma estrutura para que a organização defina seus processos como achar mais adequado.

- *Processo de Garantia da Qualidade de Software: busca fornecer garantia de que os produtos e processos de trabalho estão em conformidade com os planos e condições predefinidos. Possui as seguintes atividades:*

Implementação do processo
Garantia de produto
Garantia de processo
Garantia de sistemas de qualidade

(I) Errado, não define níveis de maturidade nem capacidade; (II) Correto, provê uma estrutura para organização definir seus processos; (III) Correto, cobre a garantia de qualidade, que é um processo do grupo de suporte de software.

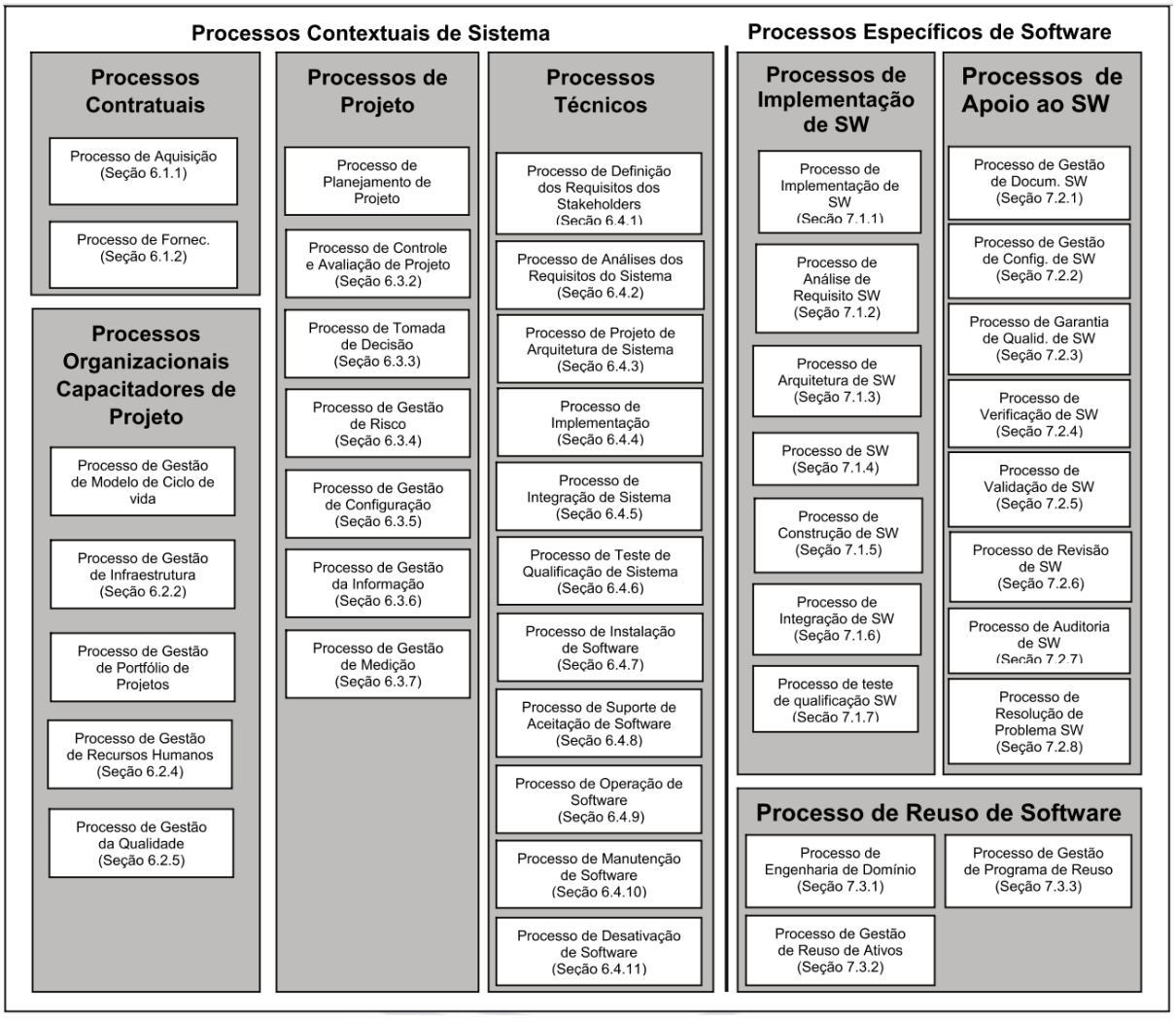
Gabarito: D

---

3. (FCC - 2012 – ARCE – Analista de Regulação – Analista de Sistemas) A Norma ISO/IEC 12207:2008 agrupa as atividades que podem ser realizadas durante o ciclo de vida de um sistema de software em sete grupos de processos. Cada um dos processos do ciclo de vida dentro desses grupos é descrito em termos da sua finalidade e resultados esperados. Dentre estes grupos de processos encontra-se o grupo de Processos de:

a) Manutenção de Segurança.
b) Implementação de Software.
c) Capacitação Profissional.
d) Qualidade e Segurança.
e) Fornecimento e Desenvolvimento.

Comentários:



Fácil, né?! É o Grupo de Processos de Implementação de Software.

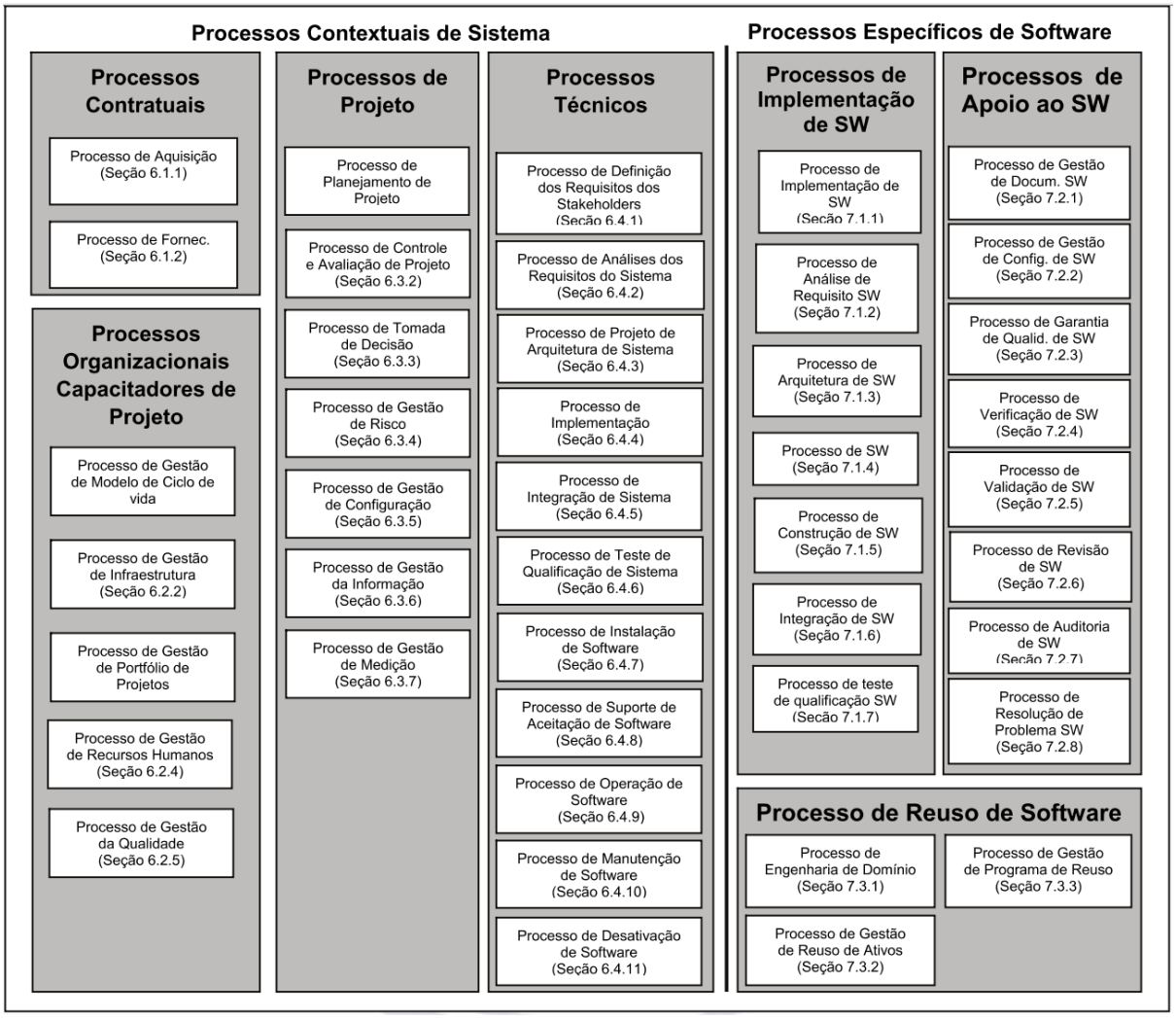
Gabarito: B

---

4. (FCC - 2012 – TRT/11 – Analista Judiciário – Analista de Sistemas) No grupo de processos de projeto, segundo a Norma ISO 12207, dentre outros, encontra-se o processo de gerenciamento:

a) da Infraestrutura.
b) de recursos humanos.
c) do modelo de ciclo de vida.
d) de configuração.
e) de qualidade.

Comentários:



Questão decoreba, infelizmente... enfim, trata-se da Gestão de Configuração!

Gabarito: D

---

5. (FCC - 2013 - TRT - 9ª REGIÃO (PR) - Analista Judiciário - Tecnologia da Informação) A ISO/IEC 12207 objetiva criar um framework que possibilite uma linguagem comum para a criação e o gerenciamento do software. Essa norma:

a) é aplicada para certificação de processos em um esquema formal e é imposta por diversos governos, dentre eles, do Brasil e dos Estados Unidos, como condição para realizar negócios com empresas privadas.

b) descreve os processos para a criação e gerenciamento de software e ainda especifica como implementar e desempenhar as atividades e tarefas incluídas nos processos.

c) define no processo fundamental de fornecimento as atividades do comprador (a organização que adquire o sistema, produto de software ou serviço de software) e compreende as seguintes atividades: iniciação, preparação do request for proposal, preparação de contrato, monitoramento do fornecedor e aceitação do produto ou serviço.

d) cobre o ciclo de vida do software, desde a sua concepção até o seu descarte, os processos para aquisição e suprimento de produtos de software e serviços, assim como os processos para controle e melhoria.

e) define como processos do Grupo de Processos Fundamentais: Documentação, Gerência de Configuração, Garantia da Qualidade, Verificação, Validação, Revisão Conjunta, Treinamento, Auditoria e Resolução de Problemas.

Comentários:

A ABNT NBR ISO/IEC 12207 estabelece uma estrutura para processos de ciclo de vida de software! Galera, o que essa norma faz? Ela estabelece uma estrutura para processos de ciclo de vida de software! Ela estabelece o que? Uma estrutura para processos de ciclo de vida de software! Por que essa insistência? Porque essa frase minúscula pode responder várias questões de prova, portanto decorem-na!

Vocês se lembram que eu falei para decorar que a norma estabelece um processo de ciclo de vida de software? Foi por conta de questões como essa! A banca tenta confundir o candidato, então é preciso saber algumas palavras-chaves! Ficou fácil, né?! Cobre o ciclo de vida do software, desde a sua concepção até o seu descarte, os processos para aquisição e suprimento de produtos de software e serviços, assim como os processos para controle e melhoria.

Gabarito: D

6. (FCC - 2011 – TRT/1 - Analista Judiciário - Tecnologia da Informação) Uma das atividades recomendadas no processo de manutenção da NBR ISO/IEC 12207 é a:

a) implementação do processo.
b) gerência de liberação e distribuição.
c) descontinuação do software.
d) avaliação da configuração.
e) identificação da configuração.

Comentários:

- *Processo de Manutenção de Software: busca fornecer suporte com boa relação custo-benefício ao produto de software entregue. Possui as seguintes atividades:*

Implementação do processo
Análise de modificação e de problema
Implementação da modificação
Revisão/Aceitação de manutenção
Migração
Descontinuação do Software

Decoreba demaaaaaais, galera! Pede a atividade do processo de manutenção e ainda por cima foi anulada porque contém duas respostas: implementação do processo e descontinuação do software.

Gabarito: X

---

7. (FCC - 2011 – TRT/4 - Analista Judiciário - Tecnologia da Informação) A arquitetura de alto nível do ciclo de vida de software estabelecida pela Norma ISO/IEC 12207 está embasada em:

a) métodos, treinamentos e tecnologias empregadas.
b) métricas, ferramentas e tecnologias empregadas.
c) processo e seus inter-relacionamentos.
d) processos, ferramentas e métricas.
e) métodos, processos, ferramentas e treinamentos.

Comentários:

Ela também estabelece uma arquitetura de alto nível do ciclo de vida de software que é construída a partir de um conjunto de processos e seus inter-relacionamentos. Os processos são descritos tanto em nível de propósito/saídas quanto em termos de atividades. É importante saber também que ela é independente de métodos, ferramentas, treinamentos, métricas ou tecnologias empregadas.

A Norma diz *o que* fazer e, não, *como fazer*. Portanto, não está embasada em tecnologias empregadas ou métodos ou ferramentas ou treinamento, apesar de que ela com certeza pode ser sustentada por eles. Ela trata apenas de processos e seus inter-relacionamentos.

Gabarito: C

---

8. (FCC - 2013 – TRT/15 - Analista Judiciário - Tecnologia da Informação) André trabalha no desenvolvimento de um software para o Tribunal Regional do Trabalho da 15a Região. Recentemente seu chefe cogitou adotar uma Norma que se aplica ao desenvolvimento de produtos de software. André foi o encarregado de escolher a Norma adequada. Pesquisou então a norma ABNT NBR ISO/IEC 12207:2009, que se aplica à aquisição de sistemas e produtos de software e serviços para o fornecimento, desenvolvimento, operação, manutenção e descontinuidade de produtos de software. André descobriu que esta Norma pode ser usada:

(I) Em um projeto, para ajudar a selecionar, estruturar e utilizar os elementos de um conjunto de processos de ciclo de vida estabelecidos que forneçam produtos e serviços. Desse modo, esta Norma pode ser usada na avaliação de conformidade do projeto para o ambiente estabelecido e declarado.

(II) Por uma organização, para ajudar a estabelecer um ambiente de processos desejados. Esses processos podem ser sustentados por uma infraestrutura de métodos, procedimentos, técnicas, ferramentas e pessoal treinado. A organização pode empregar esse ambiente para realizar e gerenciar seus projetos e seus sistemas em andamento durante as fases do ciclo de vida. Desse modo, essa Norma pode ser usada para avaliar a conformidade de um conjunto declarado e estabelecido de processos do ciclo de vida de acordo com as necessidades.

(III) Por um adquirente e um fornecedor, para ajudar a estabelecer um acordo em relação aos processos e às atividades. Esse acordo contempla os processos e atividades desta Norma que são selecionados, negociados, acordados e

executados. Desse modo, esta Norma pode ser usada para orientar a definição do acordo.

(IV) Por organizações avaliadoras e avaliadores credenciados, para realizar avaliações que possam ser usadas para obtenção de certificação oficial. Esta Norma fornece um conjunto definido de processos para que a organização obtenha certificação ISO/IEC no prazo máximo de 1 ano.

Está correto o que se afirma APENAS em:

a) I, II e III.
b) I e II.
c) III e IV.
d) II, III e IV.
e) IV.

Comentários:

As três primeiras foram tiradas integralmente da 12207. A última diz que a norma pode ser usada por organizações avaliadoras e avaliadores credenciados, para realizar avaliações que possam ser usadas para obtenção de certificação oficial. Esta Norma fornece um conjunto definido de processos para que a organização obtenha certificação ISO/IEC no prazo máximo de 1 ano. A norma diz que ela pode ser usada por organizações e avaliadores – para realizar avaliações que possam ser usados para apoiar a melhoria de processos organizacionais.

Gabarito: A

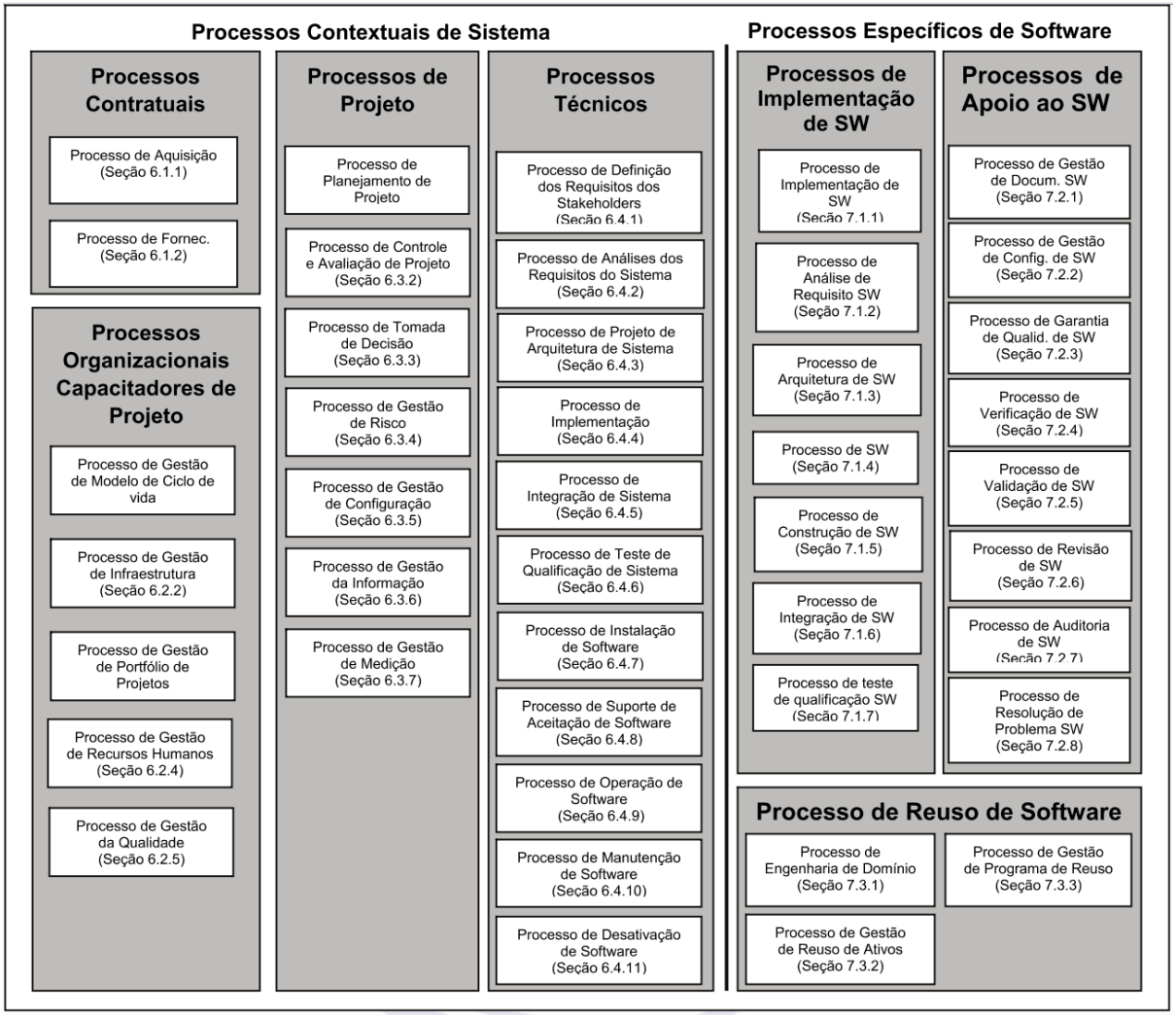
---

9. (FCC - 2013 – TRT/15 - Analista Judiciário - Tecnologia da Informação) Após escolher a norma ABNT NBR ISO/IEC 12207:2009 para ser adotada na organização onde trabalha, André verificou que a Norma é dividida em sete grupos de processos. Como sua especialidade é em análise de requisitos, verificou que o Processo de Análise de Requisitos do Sistema e o Processo de Análise de Requisitos de Software estavam, respectivamente, nos grupos de Processos:

a) Organizacionais Capacitadores de Projeto e Técnicos.
b) de Implementação de Software e de Apoio ao Software.
c) Técnicos e de Implementação de Software.
d) de Projeto e Técnicos.

e) de Projeto e de Implementação de Software.

Comentários:



Basta consultar a tabela: Processos Técnicos e Processos de Implementação de Software.

Gabarito: C

---

10. **(FCC - 2014 – TRF/3 - Analista Judiciário - Tecnologia da Informação)** Na Norma ABNT ISO/IEC 12207:2009, um dos Processos Técnicos é o Processo de Definição dos Requisitos dos Stakeholders. Com relação a este Processo, analise as tarefas e atividades a seguir:

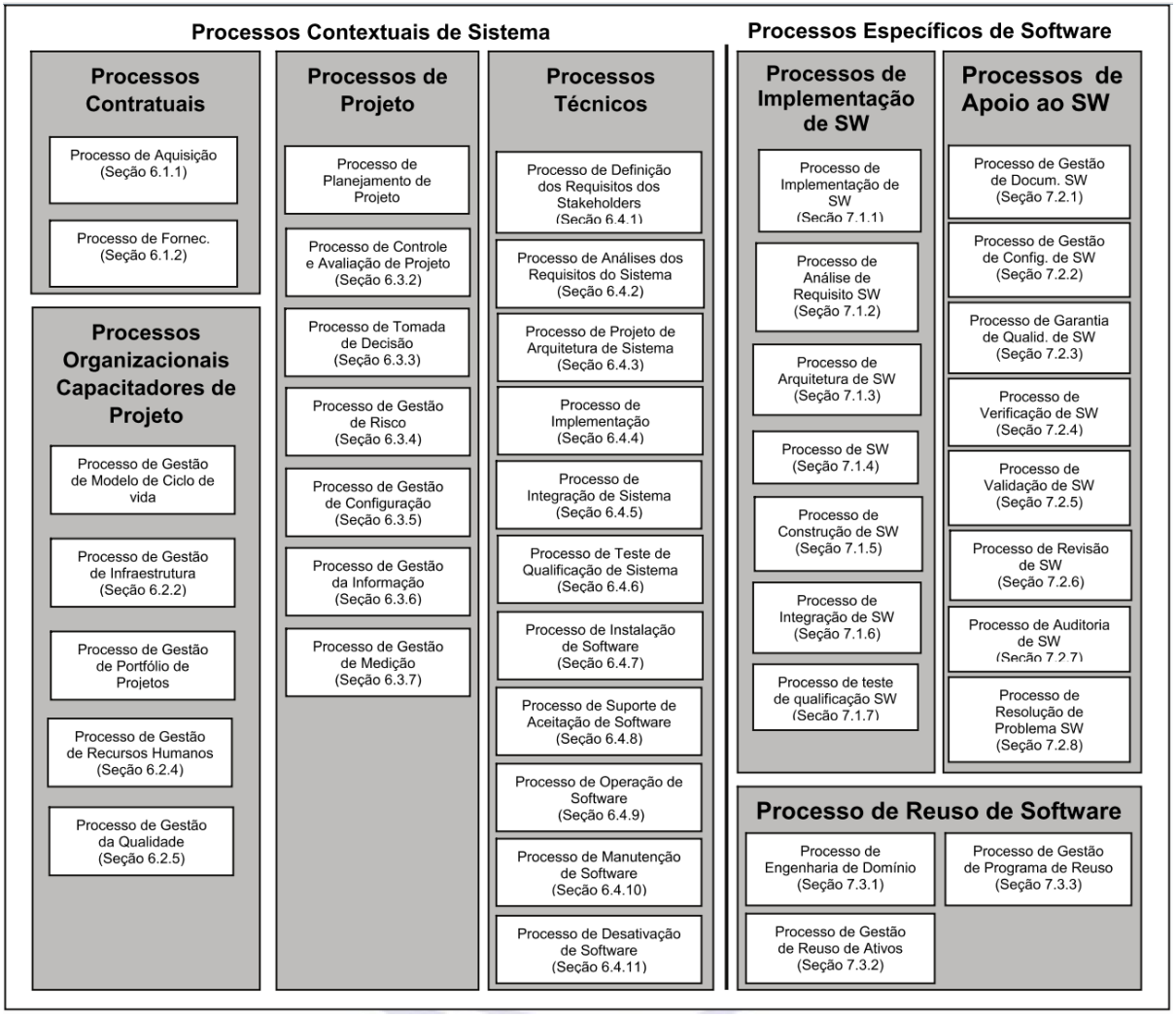
1. Identificação dos Stakeholders.
2. Classificação dos Stakeholders.

3. Identificação dos Requisitos.
4. Classificação dos Requisitos.
5. Avaliação dos Requisitos.
6. Acordo dos Requisitos.
7. Mediação de Conflitos.
8. Registro dos Requisitos.
9. Produção do Documento de Requisitos.

De acordo com a Norma supracitada, as atividades e tarefas que devem ser implementadas em consonância com as políticas e procedimentos organizacionais aplicáveis, com relação ao Processo de Definição dos Requisitos dos Stakeholders, são as que constam APENAS em 1,

a) 2, 3, 4, 5, 6, 8 e 9.
b) 3, 5, 8 e 9.
c) 3, 5, 6 e 8.
d) 2, 3, 4, 6, 7 e 8.
e) 4, 6, 8 e 9.

Comentários:



Mais decoreba que isso, impossível! Trata-se do 1, 3, 5, 6 e 8.

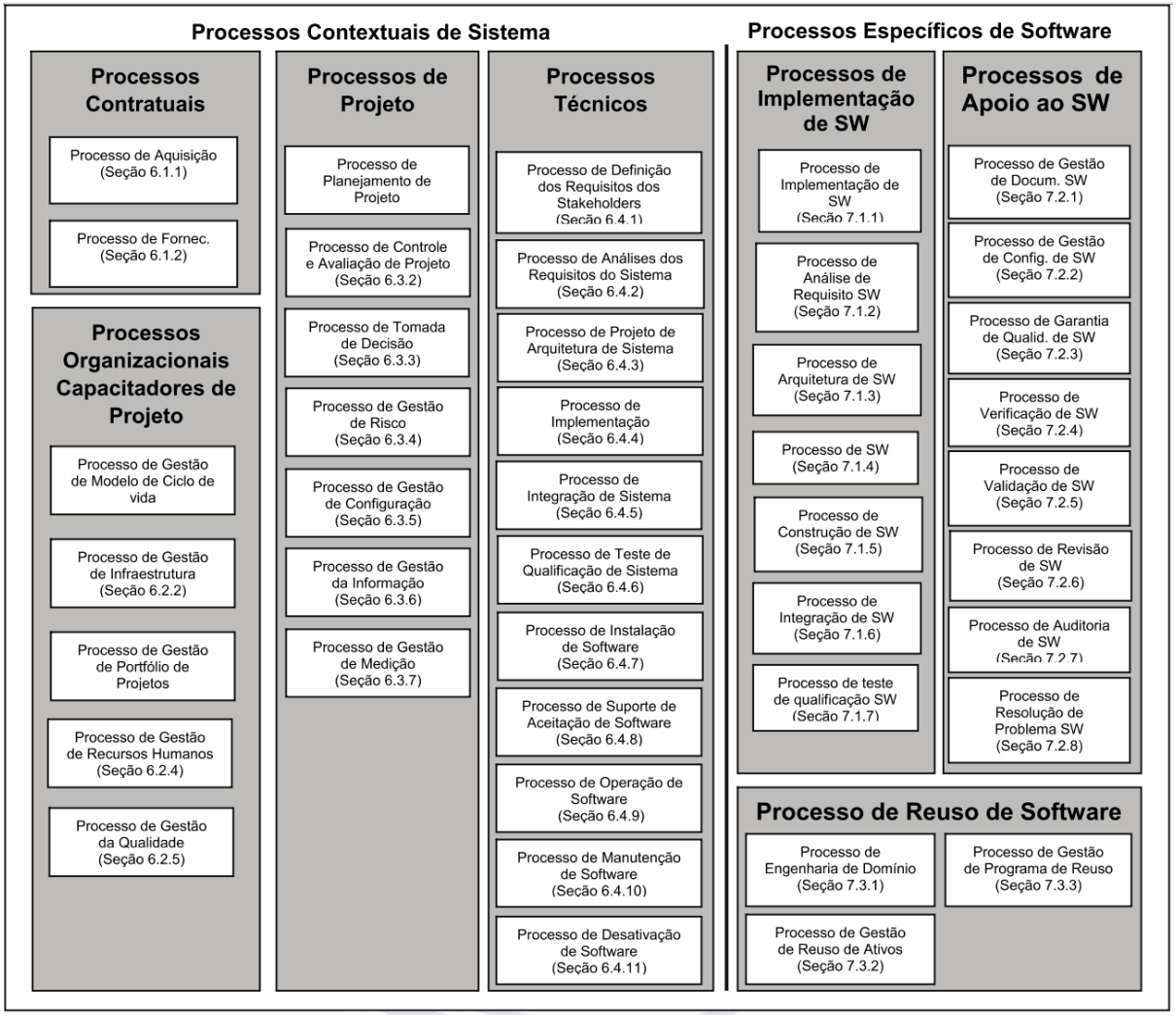
Gabarito: C

---

11. **(FCC - 2014 – TRF/3 - Analista Judiciário - Tecnologia da Informação)** Na ABNT ISO/IEC 12207:2009 os processos estão agrupados. Dentre os processos contextuais do sistema constam os Processos Contratuais, os Processos Organizacionais Capacitadores de Projeto, os Processos de Projeto e os Processos:

a) de Apoio ao Software.
b) de Auditoria de Software.
c) de Implementação de Software.
d) de Reuso de Software.
e) Técnicos.

Comentários:



Observem que as categorias de processos estão divididas em: Processos Contextuais de Sistema e Processos Específicos de Software.

- Processos Contextuais de Sistema: Processos Contratuais; Processos Organizacionais Capacitadores de Projeto; Processos de Projeto; *e Processos Técnicos*.
- Processos Específicos de Software: Processos de Implementação de Software; Processos de Apoio ao Software; e Processos de Reuso de Software.

Gabarito: E

---

12. **(CESPE – 2016 – TCE/PR – Analista de Sistema – A)** De acordo com a norma NBR ISO/IEC 12207, o fornecedor de um sistema ou software deve ser uma entidade externa à organização.

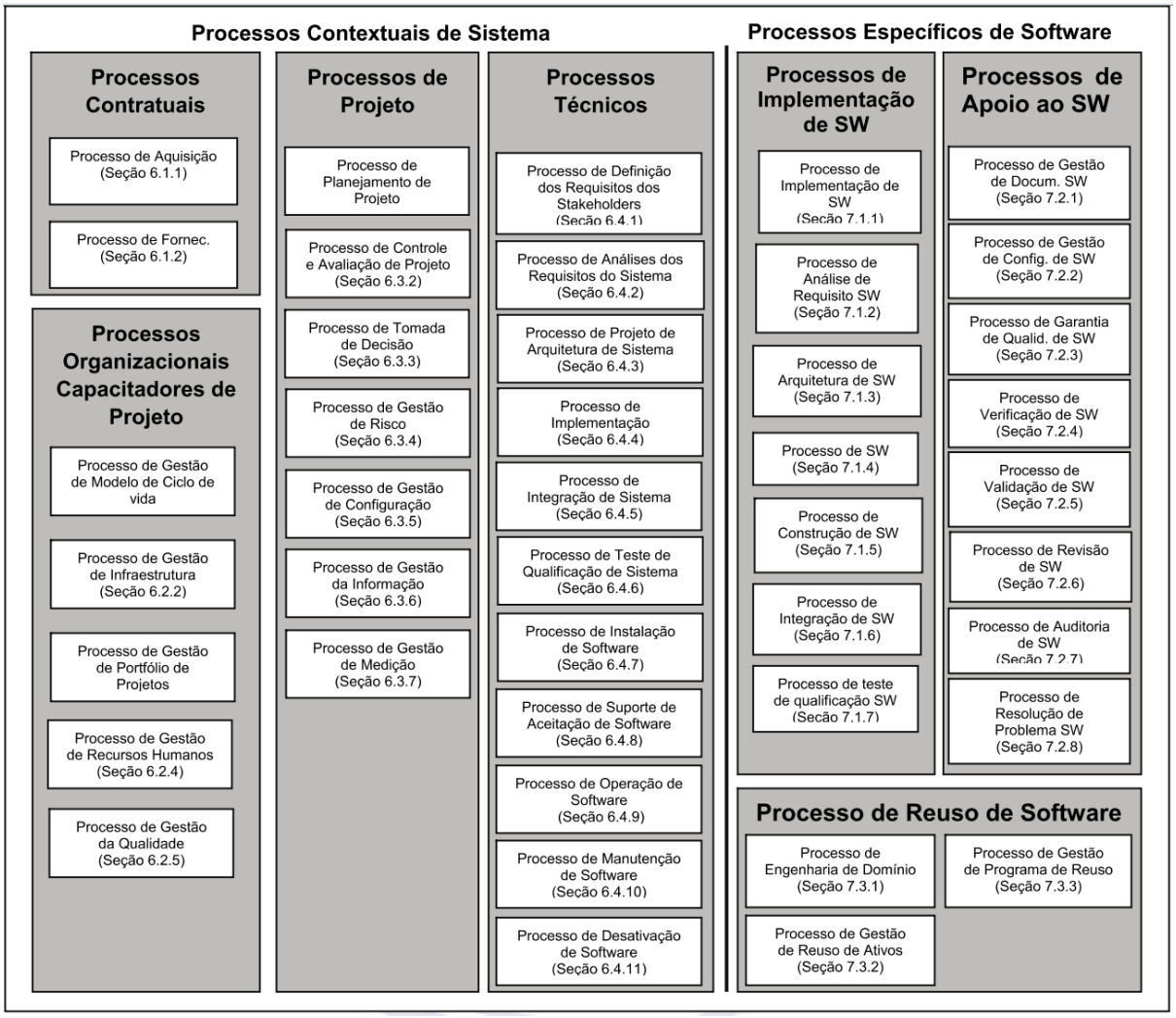
Comentários:

Na verdade, não é necessariamente uma entidade externa.

Gabarito: E

13. (CESPE – 2016 – TCE/PR – Analista de Sistema – D) A norma NBR ISO/IEC 12207 dispõe que a gerência de melhoria e o treinamento compõem o conjunto de processos de apoio do ciclo de vida de um software.

Comentários:



Conforme vimos em aula, eles não compõem os processos de apoio.

Gabarito: E

14. **(CESPE – 2016 – TCE/PR – Analista de Sistema – A)** Conforme a norma NBR ISO/IEC 12207, as tarefas de descontinuação, migração ou desativação de um software incluem-se no processo de operação do ciclo de vida do software.

Comentários:

- ***Processo de Manutenção de Software:*** *busca fornecer suporte com boa relação custo-benefício ao produto de software entregue. Possui as seguintes atividades:*

7. *Implementação do processo*
8. *Análise de modificação e de problema*
9. *Implementação da modificação*
10. *Revisão/Aceitação de manutenção*
11. *Migração*
12. *Descontinuação do Software*

Conforme vimos em aula, descontinuação e migração fazem parte do processo de manutenção (a desativação faz parte do processo de desativação do software).

Gabarito: E

---

15. **(CESPE – 2016 – TCE/PR – Analista de Sistema – E)** De acordo com a norma NBR ISO/IEC 12207, o ciclo de vida de um software é constituído de cinco processos fundamentais: aquisição, fornecimento, desenvolvimento, operação e manutenção.

Comentários:

Não existe mais o termo "*processos fundamentais*" na ISO/IEC 12207:2008, logo eu acho que essa questão deveria ser considerada errada.

Gabarito: C

---

16. **(FCC – 2016 – TRT/SE – Analista de Sistema)** A norma NBR ISO/IEC 12207:2009 estabelece uma estrutura comum para os processos de ciclo de vida de software, que pode ser referenciada pela indústria de software. A norma agrupa as atividades que podem ser executadas durante o ciclo de vida de um sistema que contém software em sete grupos de processo. No grupo de Processos de Projeto encontra-se o processo de:

a) Gestão de Modelo de Ciclo de Vida, que tem como propósito definir, manter e garantir disponibilidade das políticas e modelos de ciclo de vida de software.

b) Gestão de Risco, cujo propósito é identificar, analisar, tratar e monitorar riscos de forma contínua.

c) Gestão de Configuração e Ativos de Serviço, cujo propósito é garantir a instalação e configuração dos softwares e controlar os ativos de serviço.

d) Definição dos Requisitos dos Stakeholders, cujo propósito é definir os requisitos funcionais e não funcionais de um sistema.

e) Aquisição, que tem como propósito obter um produto que satisfaça a necessidade expressa pelo adquirente.

Comentários:

- *Processo de Gestão de Risco: busca identificar, analisar, tratar e monitorar os riscos de forma continua, por meio de uma abordagem sistemática durante todo ciclo de vida de um sistema, produto ou serviço de software. Pode ser aplicado para riscos relacionados à aquisição, desenvolvimento, manutenção ou operação de um sistema. Possui as seguintes atividades:*

Conforme vimos em aula, trata-se do processo de gestão de riscos.

Gabarito: B

ACERTEI	ERREI

NBR ISO/IEC 9126

Esta Norma versa sobre as características que definem um produto de software de qualidade. Após diversas revisões, Norma foi dividida em quatro partes:

1. ISO/IEC 9126-1: Modelo de Qualidade;
2. ISO/IEC 9126-2: Métricas Externas;
3. ISO/IEC 9126-3: Métricas Internas;
4. ISO/IEC 9126-4: Métricas de Qualidade em Uso.

Ela permite que a qualidade do produto de software seja especificada e avaliada em diferentes perspectivas pelos envolvidos com aquisição, requisitos, desenvolvimento, uso, avaliação, apoio, manutenção, garantia de qualidade e auditoria de software. Ademais, descreve um modelo de qualidade do produto de software, composto de duas partes: qualidade interna e externa e qualidade em uso.

E qual a diferença, professor? A Qualidade Interna é a totalidade das características do produto de software do ponto de vista interno. A qualidade interna é medida e avaliada com relação aos requisitos de qualidade interna. Detalhes da qualidade do produto de software podem ser melhorados durante a implementação do código, revisão e teste.

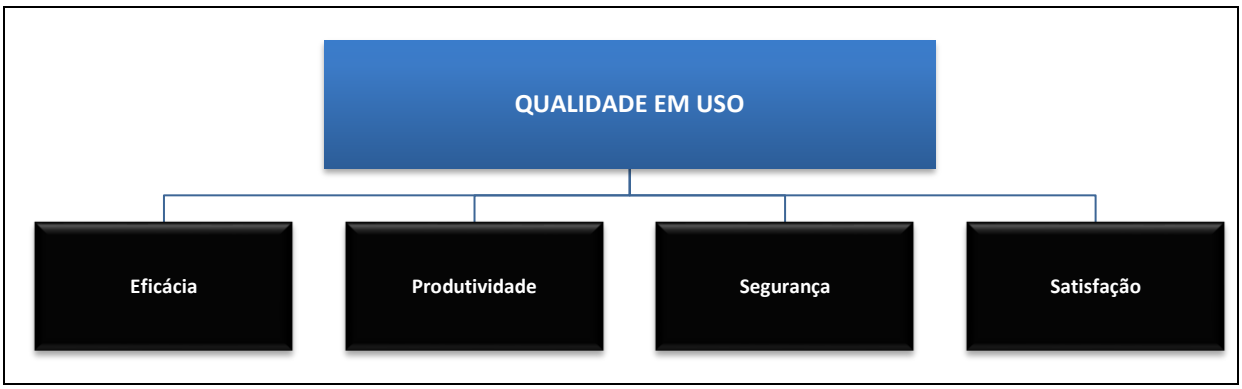
No entanto, a natureza fundamental da qualidade do produto de software representada pela qualidade interna mantém-se inalterada, a menos que seja reprojetada. *E a Qualidade Externa, professor?* Bem, a Qualidade Externa é a totalidade das características do produto de software do ponto de vista externo. *Como assim? Não entendi!*

É a qualidade quando o software é executado, o qual é tipicamente medido e avaliado enquanto está sendo testado num ambiente simulado, com dados simulados e usando métricas externas. Durante os testes, convém que a maioria dos defeitos seja descoberta e eliminada. Entretanto, alguns defeitos podem permanecer após o teste.

Como é difícil corrigir a arquitetura do software ou outro aspecto básico do projeto do software, a base do projeto usualmente permanece inalterada ao longo do teste. Por fim, a Qualidade em Uso é a visão da qualidade do produto de software do

ponto de vista do usuário, quando este produto é usado em um ambiente e um contexto de uso especificados.

Ela mede o quanto usuários podem atingir seus objetivos num determinado ambiente e não as propriedades do software em si. *Entendido?* Então, nós vimos a qualidade interna, a qualidade externa e a qualidade em uso. Existe um modelo para a qualidade interna e externa e outro para a qualidade em uso, que é apresentado na imagem abaixo composta de quatro características.



Vamos falar sobre a Qualidade Interna e Externa – esse é de longe o modelo que mais cai em prova! *Professor, precisa decorar?* Difícil dizer isso, se você tiver tempo sobrando, eu recomendo. Bem, ela categoriza os atributos de qualidade de software em seis características que são, por sua vez, subdivididas em subcaracterísticas (que podem ser medidas por meio de métricas externas e internas)[2].

[2] MNEMÔNICO SUGERIDO: **EFI**GÊNIO **FUMA POUCO**! EM ORDEM: **EFICIÊNCIA**, **FUNCIONALIDADE**, **MANUTENIBILIDADE**, **PORTABILIDADE**, **USABILIDADE E CONFIABILIDADE**.

Define-se, a seguir, cada característica e subcaracterística do software que influencia a característica de qualidade. A capacidade do software é determinada por um conjunto de atributos internos que podem ser medidos para cada característica e subcaracterística. As características e subcaracterísticas podem ser medidas externamente pelo grau da capacidade do sistema contendo o software.

FUNCIONALIDADE	Capacidade do produto de software de prover funções que atendam às necessidades explícitas e implícitas, quando o software estiver sendo utilizado sob condições especificadas.

SUBCARACTERÍSTICA	DESCRIÇÃO
ADEQUAÇÃO	Capacidade do produto de software de prover um conjunto apropriado de funções para tarefas e objetivos do usuário especificados.
ACURÁCIA	Capacidade do produto de software de prover, com o grau de precisão necessário, resultados ou efeitos corretos ou conforme acordados.
INTEROPERABILIDADE	Capacidade do produto de software de interagir com um ou mais sistemas especificados.
SEGURANÇA DE ACESSO	Capacidade do produto de software de proteger informações e dados, tal que pessoas ou sistemas não autorizados não possam lê-los nem os modificar e que não seja negado o acesso às pessoas ou sistemas autorizados.
CONFORMIDADE RELACIONADA À FUNCIONALIDADE	Capacidade do produto de software de estar de acordo com normas, convenções ou regulamentações previstas em leis e prescrições similares relacionadas à funcionalidade.

CONFIABILIDADE	Capacidade do produto de software de manter um nível de desempenho especificado, quando usado em condições especificadas.

SUBCARACTERÍSTICA	DESCRIÇÃO
MATURIDADE	Capacidade do produto de software de evitar falhas decorrentes de defeitos no software.
TOLERÂNCIA A FALHAS	Capacidade do produto de software de manter um nível de desempenho especificado em casos de defeitos no software ou de violação de sua interface especificada.
RECUPERABILIDADE	Capacidade do produto de software de restabelecer seu nível de desempenho especificado e recuperar os dados diretamente afetados no caso de uma falha.
CONFORMIDADE RELACIONADA À CONFIABILIDADE	Capacidade do produto de software de estar de acordo com normas, convenções ou regulamentações relacionadas à confiabilidade.

USABILIDADE	Capacidade do produto de software de ser compreendido, aprendido, operado e atraente ao usuário, quando usado sob condições especificadas.

SUBCARACTERÍSTICA	DESCRIÇÃO
INTELIGIBILIDADE	Capacidade do produto de software de possibilitar ao usuário compreender se o software é apropriado e como ele pode ser usado para tarefas e condições de uso específicas.
APREENSIBILIDADE	Capacidade do produto de software de possibilitar ao usuário aprender sua aplicação.
OPERACIONALIDADE	Capacidade do produto de software de possibilitar ao usuário operá-lo e controlá-lo.

ATRATIVIDADE	Capacidade do produto de software de ser atraente ao usuário.
CONFORMIDADE RELACIONADA À USABILIDADE	Capacidade do produto de software de estar de acordo com normas, convenções, guias de estilo ou regulamentações relacionadas à usabilidade.

EFICIÊNCIA	Capacidade do produto de software de apresentar desempenho apropriado, relativo à quantidade de recursos usados, sob condições especificadas.

SUBCARACTERÍSTICA	DESCRIÇÃO
COMPORTAMENTO EM RELAÇÃO AO TEMPO	Capacidade do produto de software de fornecer tempos de resposta e de processamento, além de taxas de transferência, apropriados, quando o software executa suas funções, sob condições estabelecidas.
UTILIZAÇÃO DE RECURSOS	Capacidade do produto de software de usar tipos e quantidades apropriados de recursos, quando o software executa suas funções sob condições estabelecidas.
CONFORMIDADE RELACIONADA À EFICIÊNCIA	Capacidade do produto de software de estar de acordo com normas e convenções relacionadas à eficiência.

MANUTENIBILIDADE	Capacidade do produto de software de ser modificado. As modificações podem incluir correções, melhorias ou adaptações devido a mudanças no ambiente e nos seus requisitos ou especificações funcionais.

SUBCARACTERÍSTICA	DESCRIÇÃO

ANALISABILIDADE	Capacidade do produto de software de permitir o diagnóstico de deficiências ou causas de falhas no software, ou a identificação de partes a serem modificadas.
MODIFICABILIDADE	Capacidade do produto de software de permitir que uma modificação especificada seja implementada.
ESTABILIDADE	Capacidade do produto de software de evitar efeitos inesperados decorrentes de modificações no software.
TESTABILIDADE	Capacidade do produto de software de permitir que o software, quando modificado, seja validado.
CONFORMIDADE RELACIONADA À MANUTENIBILIDADE	Capacidade do produto de software de estar de acordo com normas ou convenções relacionadas à manutenibilidade.

PORTABILIDADE	Capacidade do produto de software de ser transferido de um ambiente para outro.

SUBCARACTERÍSTICA	DESCRIÇÃO
ADAPTABILIDADE	Capacidade do produto de software de ser adaptado para diferentes ambientes especificados, sem necessidade de aplicação de outras ações ou meios além daqueles fornecidos para essa finalidade pelo software.
CAPACIDADE PARA SER INSTALADO	Capacidade do produto de software para ser instalado em um ambiente especificado.

COEXISTÊNCIA	Capacidade do produto de software de coexistir com outros produtos de software independentes, em um ambiente comum, compartilhando recursos comuns.
CAPACIDADE PARA SUBSTITUIR	Capacidade do produto de software de ser usado em substituição a outro produto de software especificado, com o mesmo propósito e no mesmo ambiente.
CONFORMIDADE RELACIONADA À PORTABILIDADE	Capacidade do produto de software de estar de acordo com normas ou convenções relacionadas à portabilidade.

Vamos detalhar agora a Qualidade em Uso e suas características:

CARACTERÍSTICA	DESCRIÇÃO
EFICÁCIA	Capacidade do produto de software de permitir que usuários atinjam metas especificadas com acurácia e completude, em um contexto de uso especificado.
PRODUTIVIDADE	Capacidade do produto de software de permitir que seus usuários empreguem quantidade apropriada de recursos em relação à eficácia obtida, em um contexto de uso especificado.
SEGURANÇA	Capacidade do produto de software de apresentar níveis aceitáveis de riscos de danos a pessoas, negócios, software, propriedades ou ao ambiente, em um contexto de uso especificado.
SATISFAÇÃO	Capacidade do produto de software de satisfazer usuários, em um contexto de uso especificado.

Pessoal, nós agora vamos falar brevemente sobre as outras partes da norma. Primeiro, a NBR ISO/IEC 9126-2: Métricas Externas. Ela se apoia na definição de atributos externos de qualidade correlacionados com uma determinada

característica e define indicadores e métricas externas para avaliar um produto de software. *Bacana?*

As Métricas Externas referem-se a medições indiretas de um produto de software a partir do comportamento do Sistema Computacional ou do seu efeito no ambiente, quando da execução de seus programas. Elas devem ser usadas para avaliar o comportamento do software quando usado em situações específicas; para predizer a qualidade real durante o uso.

Também será usada para avaliar e indicar se o produto satisfaz as verdadeiras necessidades durante a operação real pelo usuário. *Vamos dar um exemplo?* Se escolhermos uma característica (Funcionalidade) e uma subcaracterística (Adequação). Uma Métrica Externa poderia ser a quantidade de funções atendidas, que poderão ser subdivididas em desejáveis e obrigatórias.

Já a NBR ISO/IEC 9126-3 (Parte 3): Métricas Internas define indicadores e métricas internas para avaliar um produto de software. Elas referem-se a medições de um produto de software a partir de suas características internas, sem a necessidade de execução nos programas, por exemplo: linhas de código, número de erros encontrados, etc.

O documento nos informa que as Métricas Internas fornecem aos usuários a possibilidade de medir a qualidade dos artefatos intermediários e de prever a qualidade do produto final. Isto permite que o usuário identifique problemas de qualidade e inicie a ação corretiva assim que possível no ciclo de vida do desenvolvimento.

Por fim, a NBR ISO/IEC 9126-4 (Parte 4) valida a qualidade do produto em cenários e tarefas comuns ao usuário. Os atributos da qualidade em uso são categorizados pelas características: eficácia, produtividade, segurança e satisfação. Usuários também podem desenvolver e aplicar métricas para seus domínios particulares de aplicação. *Bacana?*

Pessoal, nós devemos ter em mente sempre que a Qualidade Internas e a Qualidade Externa são aplicáveis a um produto de software. Já a Qualidade em Uso é aplicável ao efeito do produto de software em um cenário específico – é importante notar essa diferença importante. Ademais, as Métricas Internas podem ser aplicadas a um produto de software não executável.

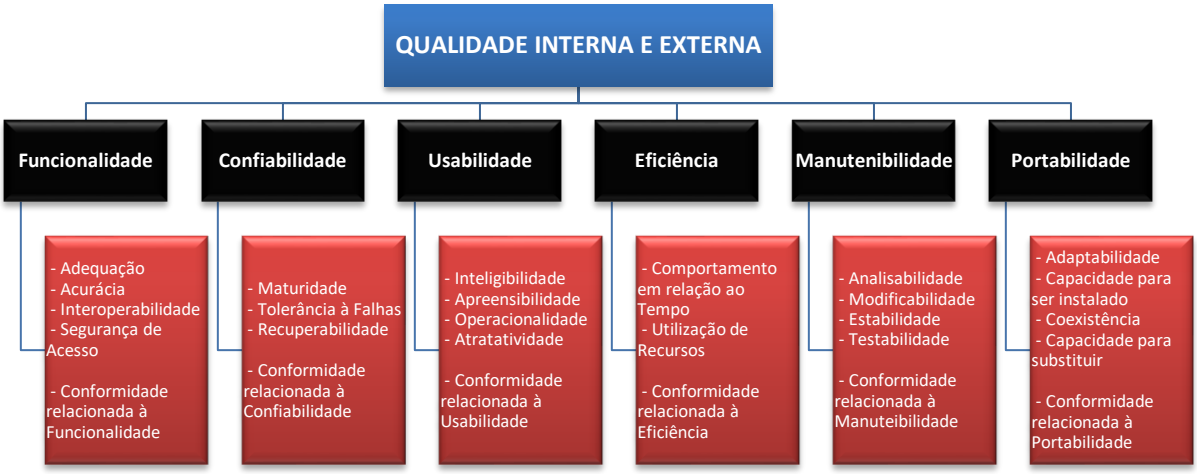
As Métricas Externas podem ser usadas para medir a qualidade do produto de software através da medição de seu comportamento em um sistema do qual ele faça parte. E as Métricas de Qualidade em Uso medem o quanto o produto agrega às necessidades de usuários específicos. Temos, então, três categorias de qualidade e cada uma tem sua especificidade.

Findando essa aula, nós podemos ratificar o que diz a norma, i.e., ela trata de um conjunto de atributos que têm impacto na capacidade do software de manter o seu nível de desempenho dentro de condições estabelecidas por um dado período de tempo – ela é responsável pela confiabilidade do software! *Bacana?* Chegamos ao fim da nossa aula sobre essa norma.

1. (CONSULPLAN - 2010 – Prefeitura de Santa Maria Madalena – Analista de Sistemas) São categorias de características principais de qualidade de software, segundo a Norma (ISO/IEC 9126: NBR 13596), EXCETO:

a) Eficiência.
b) Segurança.
c) Confiabilidade.
d) Usabilidade.
e) Funcionalidade.

Comentários:



Conforme vimos em aula, Segurança não faz parte das seis características principais de qualidade de software: Funcionalidade, Confiabilidade, Usabilidade, Eficiência, Manutenibilidade e Portabilidade.

Gabarito: B

---

2. (COVEST - 2013 – UFPE – Analista de Sistemas) A norma ISO/IEC 9126 é um padrão internacional para a qualidade de software, a qual é composta de uma série de características. Sobre essa norma, é correto afirmar que:

a) descreve seis características relacionadas à qualidade de software, e como elas devem ser medidas.

b) compatibilidade é uma das características, que é composta por duas subcaracterísticas: coexistência e interoperabilidade.

c) funcionalidade é a característica que visa verificar se, durante um período de tempo, o sistema funciona de acordo com as condições preestabelecidas.

d) foi atualizada pela norma ISO/IEC 25010, a qual só adicionou a característica de segurança.

e) a característica que determina métricas para avaliar o comportamento do sistema em relação a tempo e a recursos utilizados é a eficiência.

Comentários:

(a) Não, não descreve como as características devem ser medidas. Na verdade, ela descreve como as métricas devem ser medidas, inclusive apresenta diversas fórmulas. No entanto, percebam que isso é bem sutil; (b) Não, a compatibilidade não é uma das características; (c) Não, descreve a capacidade do produto de software de prover funções que atendam às necessidades explícitas e implícitas, quando o software estiver sendo utilizado sob condições especificadas; (d) Não, o item adicionou Segurança, Compatibilidade, Operabilidade e retirou Usabilidade; (e) Perfeito, essa é a opção verdadeira!

Gabarito: E

---

3. (FGV - 2009 – MEC – Analista de Sistemas) Analise a citação a seguir.

"Um conjunto de atributos que têm impacto na capacidade do software de manter o seu nível de desempenho dentro de condições estabelecidas por um dado período de tempo."

A Norma que integra os conceitos de ambiente, estratégias e planejamento de testes, é conhecida por:

a) ISO 12207
b) ISO 15504
c) ISO 9126
d) IEEE 829
e) MPS.BR.

Comentários:

Findando essa aula, nós podemos ratificar o que diz a norma, i.e., ela trata de um conjunto de atributos que têm impacto na capacidade do software de manter o seu nível de desempenho dentro de condições estabelecidas por um dado período de tempo – ela é responsável pela confiabilidade do software! Bacana? Chegamos ao fim da nossa aula sobre essa norma.

A questão fala de atributos que têm impacto na capacidade do software de manter seu nível de desempenho. *O que isso quer dizer, galera?* Qualidade de Software! Um software que mantém seu nível de desempenho é um software de qualidade!

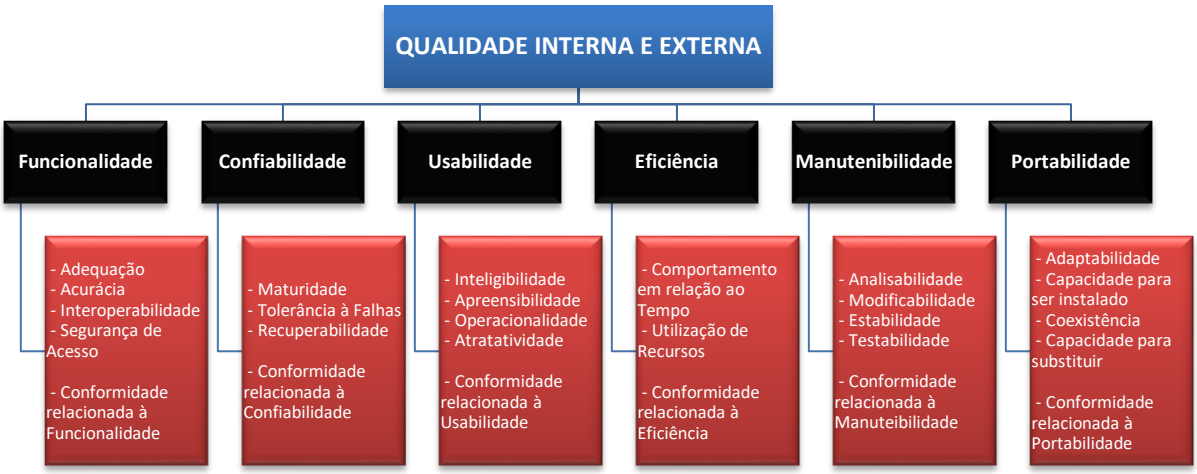
Gabarito: X

---

4. **(FGV - 2009 – MEC – Analista de Sistemas)** Entre os critérios de qualidade da Norma ISO 9126 não se inclui:

a) a manutenibilidade.
b) a funcionalidade.
c) a confiabilidade.
d) a utilizabilidade.
e) a eficicácia.

Comentários:



Conforme vimos em aula, não existe utilizabilidade nem eficicácia e, por isso, a questão foi anulada.

Gabarito: X

---

5. **(IADES - 2013 - EBSERH - Analista de Tecnologia da Informação - Teste e Qualidade)** De acordo com o padrão de qualidade ISO 9126, são identificados seis atributos fundamentais da qualidade. Sobre o tema, assinale a alternativa correta.

a) A usabilidade diz respeito à quantidade de tempo, que o software fica disponível para uso.

b) A eficiência é o grau com que o software satisfaz às necessidades declaradas.

c) A disponibilidade é o grau de tempo em que o software permanece no ar para utilização

d) A portabilidade é a facilidade com a qual um software pode ser transportado de um ambiente para outro.

e) A confidencialidade é a capacidade de manter partes do software, em sigilo, só sendo permitido o conhecimento, por parte de pessoas autorizadas.

Comentários:

(a) Não, isso é confiabilidade; (b) Não, isso é funcionalidade; (c) Não, disponibilidade não é um dos atributos de qualidade da ISO/IEC 9126; (d) Perfeito, essa é a resposta; (e) Não, confidencialidade também não é um dos atributos da ISO/IEC 9126.

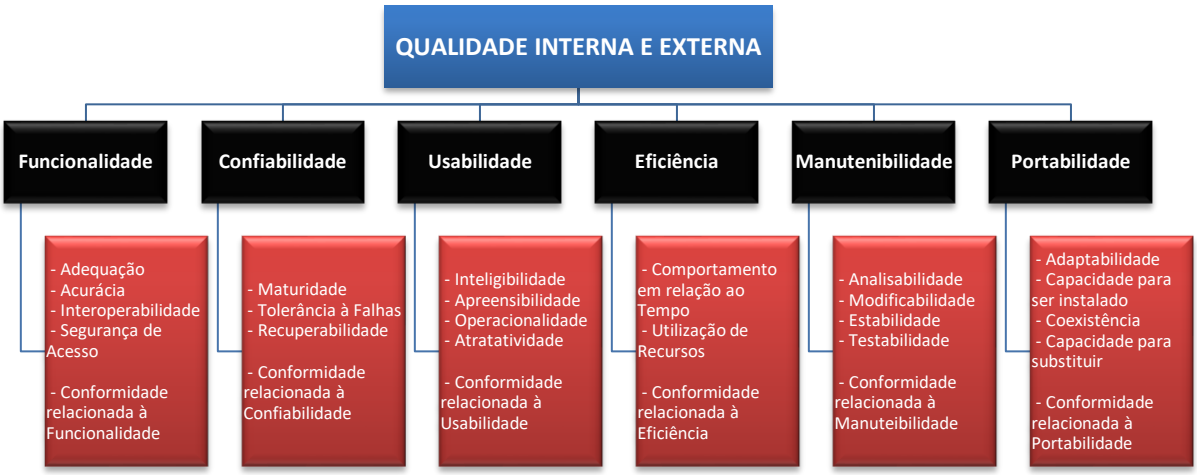
Gabarito: D

---

6. **(FGV - 2010 - DETRAN-RN – Programador)** Assinale a alternativa que NÃO contém somente atributos para características externas e internas do modelo de qualidade de software, definido na ISO/IEC 9126-1:

a) Funcionalidade, confiabilidade, usabilidade.
b) Funcionalidade, confiabilidade, eficiência.
c) Funcionalidade, confiabilidade, alta gerência.
d) Funcionalidade, usabilidade, portabilidade.
e) Eficiência, manutenibilidade, portabilidade.

Comentários:



Conforme vimos em aula, não existe Alta Gerência.

Gabarito: C

---

7. **(FCC - 2007 - TRE-SE - Analista Judiciário - Tecnologia da Informação)** Considere as questões chave apresentadas na seguinte tabela, com o enfoque da ISO 9126 (NBR 13596) ? Qualidade de Software

Questão chave	
I.	Propõe-se a fazer o que é apropriado?
II.	Faz o que foi proposto de forma correta?
III.	Com que freqüência apresenta falhas?
IV.	Há grande risco quando se faz alterações?

As seguintes sub características aplicáveis à avaliação da qualidade do software: Maturidade, Estabilidade, Acurácia e Adequação são aplicáveis, respectivamente, às questões chave:

a) I, II, IV e III.
b) II, I, III e IV.
c) III, IV, II e I.
d) IV, III, II e I.
e) IV, III, I e II.

Comentários:

Maturidade é a capacidade do produto de software de evitar falhas decorrentes de defeitos no software; Estabilidade é a capacidade do produto de software de evitar efeitos inesperados decorrentes de modificações no software; Acurácia é a capacidade do produto de software de prover, com o grau de precisão necessário, resultados ou efeitos corretos ou conforme acordados; Adequação é a capacidade do produto de software de prover um conjunto apropriado de funções para tarefas e objetivos do usuário especificados.

Gabarito: C

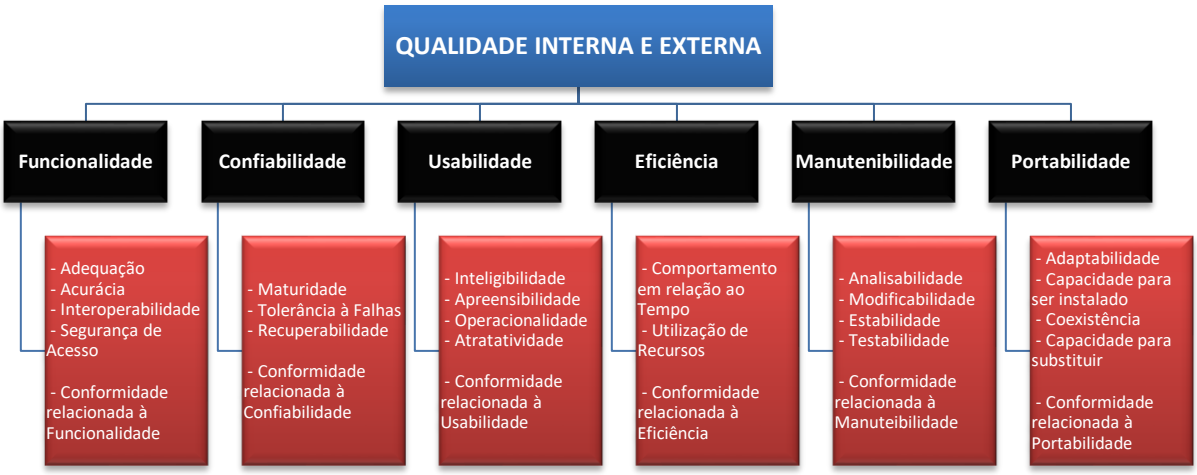
---

8. (UFF – 2008 – DATAPREV – Analista de Sistemas) A norma ISO 9.126 foi desenvolvida para identificar atributos de qualidade para software de computador. O período de tempo em que o software está disponível para uso, indicado pelos sub-atributos maturidade, tolerância à falha e recuperabilidade, é caracterizado pelo atributo-chave:

a) portabilidade;
b) confiabilidade;

c) manutenibilidade;
d) eficiência;
e) funcionalidade.

Comentários:



Conforme vimos em aula, a questão fala em período de tempo disponível para uso, maturidade, recuperabilidade e tolerância a falhas, logo trata-se da Confiabilidade!

Gabarito: B

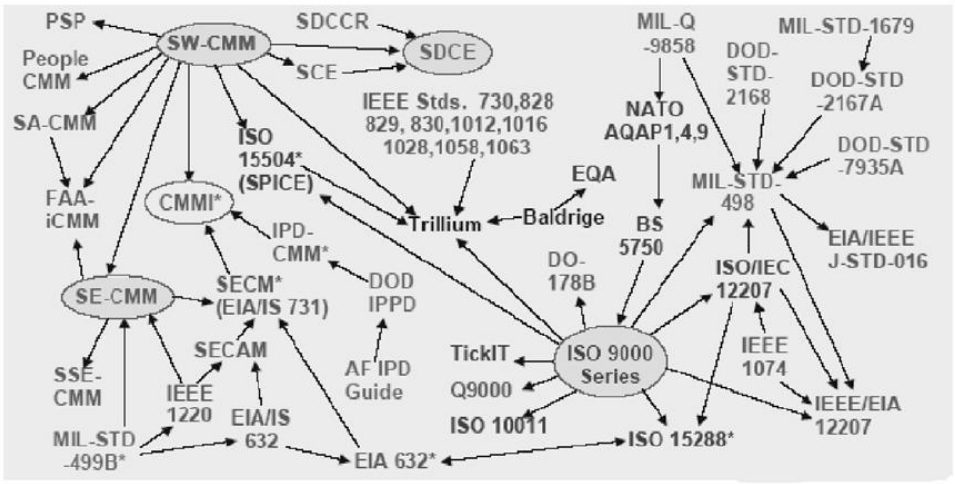
---

9. **(CESPE - 2009 – INMETRO – Analista de Sistemas)** Um modelo para a avaliação contínua de capacidade de processos é descrito na norma NBR ISO/IEC 9126, que deriva do ciclo da melhoria contínua presente na norma ISO 9001.

Comentários:

Não! A Norma NBR ISO/IEC 15504 que é responsável pela avaliação contínua de capacidade de processos.

Gabarito: E



A figura acima, elaborada no ano de 1998, apresenta uma proposta de relacionamento de precedência de criação e de incorporação de conceitos entre diversos modelos de qualidade de processo de engenharia de software e de sistemas. Não estão representados no modelo as normas e os modelos NBR ISO/IEC 27001, MPS.BR e NBR ISO/IEC 9126.

10. (CESPE - 2009 – INMETRO – Analista de Sistemas) A norma NBR ISO/IEC 9126-1, que define atributos de qualidade externa e interna, como funcionalidade, confiabilidade, usabilidade, eficiência, manutenibilidade e portabilidade, não figura no mapa porque, principalmente, não estava disponível à época.

Comentários:

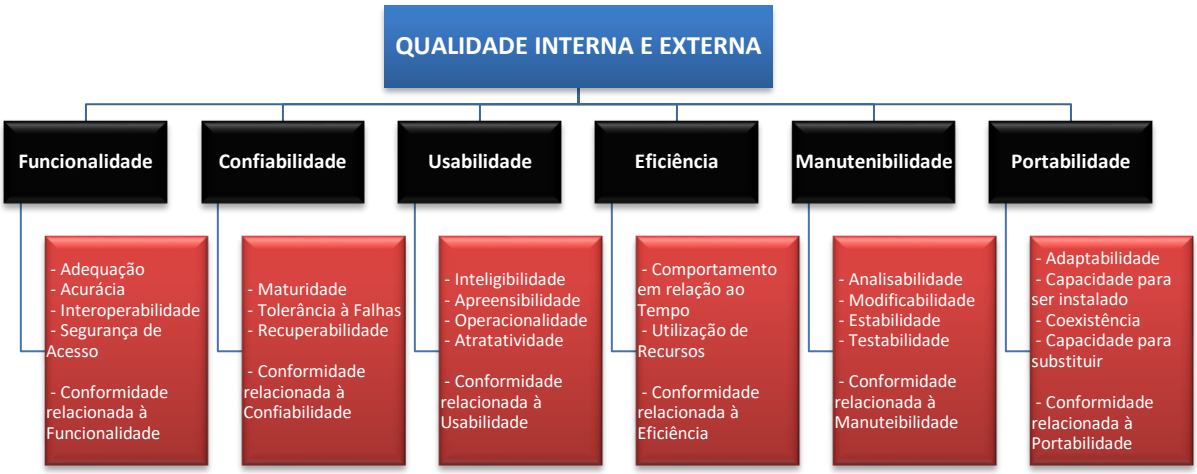
Na verdade, estava disponível - ela é de 1991! É ridículo cobrar datas em prova :(

Gabarito: E

---

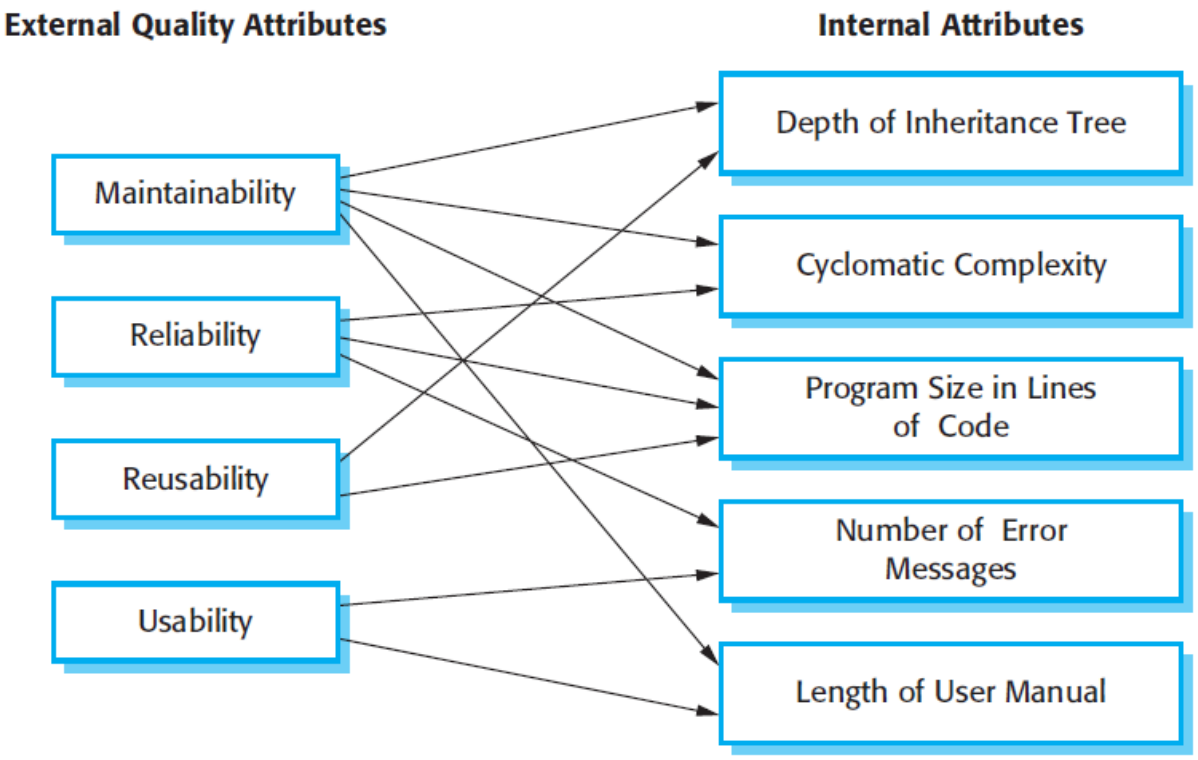
11. (CESPE - 2015 – STJ – Analista de Sistemas) A manutenibilidade é atributo de qualidade externa que pode ser medida por atributos internos, como a profundidade da árvore de herança e a complexidade ciclomática.

Comentários:



Conforme vimos em aula, a manutenibilidade é um atributo de qualidade externa (e interna) e pode ser medida por atributos internos (ou subcaracterísticas). No entanto, não encontrei nada na ISO 9126 que fale sobre profundidade da árvore de herança e complexidade ciclomática. Inclusive, essas duas são métricas de complexidade e, não, de manutenibilidade.

Qual é o lance dessa questão? Ela foi retirada do Sommerville, que afirma que o Atributo de Qualidade Externa chamado Manutenibilidade pode ser medido por meio de atributos internos, tais como: Complexidade Ciclomática, Profundidade da Árvore de Herança; Quantidade de Linhas de Código; e Tamanho do Manuel de Usuário. *Entendido?*



No entanto, o enunciado dessa questão fala efetivamente sobre ISO 9126! Logo, a questão deveria ter sido anulada! Vejam a imagem sobre esses atributos acima.

Gabarito: C

---

12. (CESPE - 2015 – STJ – Analista de Sistemas) A apreensibilidade cuida da capacidade de o usuário compreender se o software é apropriado e como este pode ser usado para a tarefa e as condições específicas.

Comentários:

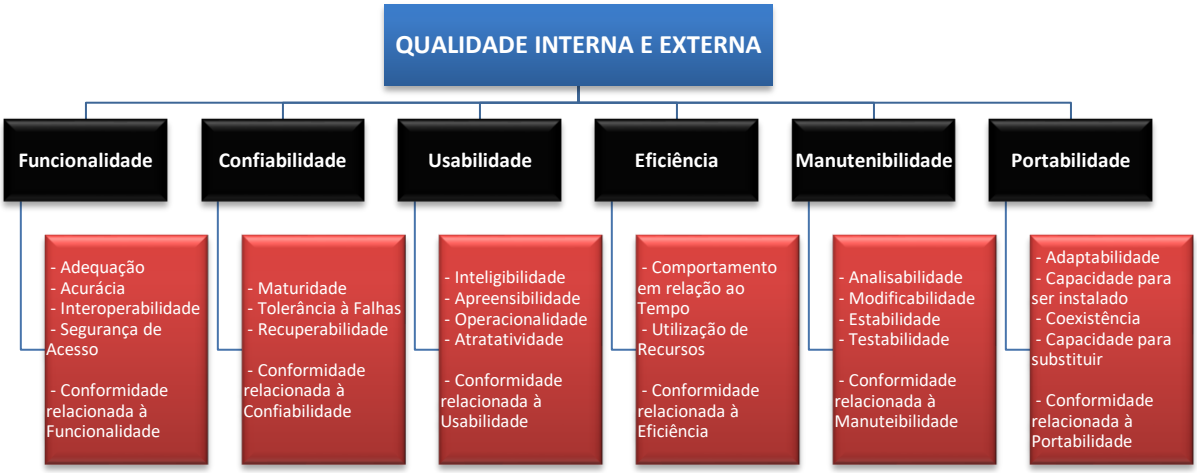
Conforme vimos em aula, é uma subcaracterística da Usabilidade, mas a questão descreveu a Inteligibilidade, que é outra subcaracterística da Usabilidade. A Apreensibilidade é a capacidade do produto de software de possibilitar ao usuário aprender sua aplicação.

Gabarito: E

---

13. **(CESPE - 2015 – STJ – Analista de Sistemas)** A funcionalidade e a usabilidade, características dos atributos de qualidade de software, possuem como subcaracterísticas, respectivamente, a operacionalidade e a interoperabilidade.

Comentários:



Conforme vimos em aula, a questão inverteu as subcaracterísticas.

Gabarito: E

---

14. **(QUADRIX - 2015 – COBRA – Analista de Sistemas)** De acordo com a norma ISO/IEC 9126, a qualidade do produto software está relacionada às seguintes características: Funcionalidade, Confiabilidade, Usabilidade, Eficiência, Manutenibilidade e Portabilidade. Sobre o tema, assinale a afirmação correta.

a) A Manutenibilidade diz que o produto de software deve ser capaz de manter seu nível de desempenho, ao longo do tempo, nas condições estabelecidas.

b) A Confiabilidade está relacionada ao esforço necessário para a utilização do sistema, baseado em um conjunto de implicações e de condições do usuário.

c) A Usabilidade refere-se à compatibilidade dos recursos e os tempos envolvidos compatíveis com o nível de desempenho requerido pelo software.

d) A Funcionalidade refere-se à existência de funções e propriedades específicas do produto, que satisfazem as necessidades do usuário.

e) A Eficiência diz respeito à facilidade de o software poder ser transferido de um ambiente para outro.

Comentários:

FUNCIONALIDADE	Capacidade do produto de software de prover funções que atendam às necessidades explícitas e implícitas, quando o software estiver sendo utilizado sob condições especificadas.

Conforme vimos em aula, a funcionalidade trata da capacidade do produto de software de prover funções que atendam às necessidades explícitas e implícitas, quando o software estiver sendo utilizado sob condições especificadas.

Gabarito: D

---

15. (IDECAN - 2014 – AGU – Analista de Sistemas) "Detalhes da qualidade do produto de software podem ser melhorados durante a implementação do código, revisão e teste, mas a natureza fundamental da qualidade do produto de software representada pela qualidade_____ mantém-se inalterada, a menos que seja reprojetada." Assinale a alternativa que completa corretamente a afirmativa anterior.

a) interna
b) em uso
c) externa
d) em uso estimada (ou prevista)
e) externa estimada (ou prevista)

Comentários:

E qual a diferença, professor? A Qualidade Interna é a totalidade das características do produto de software do ponto de vista interno. A qualidade interna é medida e avaliada com relação aos requisitos de qualidade interna. Detalhes da qualidade do

produto de software podem ser melhorados durante a implementação do código, revisão e teste.

No entanto, a natureza fundamental da qualidade do produto de software representada pela qualidade interna mantém-se inalterada, a menos que seja reprojetada. E a Qualidade Externa, professor? Bem, a Qualidade Externa é a totalidade das características do produto de software do ponto de vista externo. Como assim? Não entendi!

Conforme vimos em aula, a questão trata da qualidade interna.

Gabarito: A

---

16. (VUNESP - 2014 – DESENVOLVESP – Analista de Sistemas) A norma ISO 9126 (Engenharia de Software – Qualidade do Produto) estabelece um modelo de qualidade com 6 atributos. Dentre eles, está o atributo eficiência, que visa medir:

a) a facilidade de se fazer manutenções corretiva e adaptativa no software.
b) a facilidade de transportar o software de um computador para outro.
c) o número de erros detectados por dia de operação.
d) o nível no qual o software utiliza, de forma otimizada, os recursos do sistema computacional.
e) o tempo máximo decorrido entre duas paradas simultâneas do software.

Comentários:

EFICIÊNCIA	Capacidade do produto de software de apresentar desempenho apropriado, relativo à quantidade de recursos usados, sob condições especificadas.

Conforme vimos em aula, a eficiência trata do nível no qual o software utiliza, de forma otimizada, os recursos do sistema computacional.

Gabarito: D

---

17. (CESGRANRIO - 2014 – EPE – Analista de Sistemas) Com a evolução das pesquisas na área de qualidade, ficou cada vez mais claro para os pesquisadores que este é um conceito complexo e multifacetado. Muitos autores desenvolveram modelos de qualidade baseados na ideia de descrever qualidade como um conjunto de características ou atributos, organizadas de forma

hierárquica. Esse movimento também aconteceu na área de qualidade de software, resultando em múltiplos modelos.

Um marco importante nessa discussão foi o estabelecimento de um modelo padrão de qualidade de software, representado na norma ISO/IEC 9126, que identificou seis características da qualidade de software, cada uma delas com um conjunto de subcaracterísticas.

Com relação a esse padrão, a acurácia, ou seja, a capacidade de o produto de software prover com o grau de precisão necessário resultados ou efeitos corretos ou conforme acordados é uma subcaracterística de:

a) Portabilidade
b) Usabilidade
c) Confiabilidade
d) Eficiência
e) Funcionalidade

Comentários:

Conforme vimos em aula, trata-se de uma subcaracterística de funcionalidade.

Gabarito: E

---

18. (CESPE - 2013 – STF – Analista de Sistemas) A qualidade de software abrange apenas os aspectos internos e externos decorrentes do uso e, portanto, pode ser medida durante a utilização do software por parte do usuário.

Comentários:

Ela mede o quanto usuários podem atingir seus objetivos num determinado ambiente e não as propriedades do software em si. Entendido? Então, nós vimos a qualidade interna, a qualidade externa e a qualidade em uso. Existe um modelo para a qualidade interna e externa, e outro para a qualidade em uso, que é apresentado na imagem abaixo composta de quatro características.

Calma, são coisas diferentes! A qualidade de software abrange a qualidade interna, a qualidade externa e a qualidade em uso. Seria correto dizer que abrange os aspectos internos e externos e os decorrentes de uso. *Ela pode ser medida durante a utilização do software por parte do usuário?* Sim, essa é a qualidade em uso!

Gabarito: E

---

19. **(CESPE - 2013 – STF – Analista de Sistemas)** Do ponto de vista histórico, o termo usabilidade evoluiu a partir do termo qualidade em uso, que, por sua vez, substituiu o termo interface amigável, principalmente devido à pouca abrangência e subjetividade que estes últimos sugeriam.

Comentários:

Ela mede o quanto usuários podem atingir seus objetivos num determinado ambiente e não as propriedades do software em si. Entendido? Então, nós vimos a qualidade interna, a qualidade externa e a qualidade em uso. Existe um modelo para a qualidade interna e externa, e outro para a qualidade em uso, que é apresentado na imagem abaixo composta de quatro características.

Primeira, o termo qualidade em uso que evoluiu a partir do termo usabilidade. Segundo, esse termo no substituiu esse outro termo. Terceiro, é ridículo cobrar em prova a origem de termos.

Gabarito: E

---

20. **(IADES - 2013 – EBSERH – Analista de Sistemas)** De acordo com o padrão de qualidade ISO 9126, são identificados seis atributos fundamentais da qualidade. Sobre o tema, assinale a alternativa correta.

a) A usabilidade diz respeito à quantidade de tempo, que o software fica disponível para uso.

b) A eficiência é o grau com que o software satisfaz às necessidades declaradas.

c) A disponibilidade é o grau de tempo em que o software permanece no ar para utilização.

d) A portabilidade é a facilidade com a qual um software pode ser transportado de um ambiente para outro.

e) A confidencialidade é a capacidade de manter partes do software, em sigilo, só sendo permitido o conhecimento, por parte de pessoas autorizadas.

Comentários:

PORTABILIDADE	Capacidade do produto de software de ser transferido de um ambiente para outro.

Conforme vimos em aula, trata-se da penúltima opção.

Gabarito: D

---

21. (FCC - 2012 – TJ/PE – Analista de Sistemas) No contexto dos atributos de qualidade de software, considere:

I. A resiliência é a capacidade de o sistema voltar ao nível de desempenho anterior a falhas ou comportamento imprevisto de usuários, software ou hardware e recuperar os dados afetados, caso existam.

II. O desempenho e uso de recursos referem-se à capacidade do sistema de alcançar tempos de resposta, latência, tempo de processamento, vazão, etc dentro do período de tempo especificado e ao fato do software exigir mais ou menos recursos de acordo com suas condições de uso.

III. A analisabilidade é o grau de facilidade, com qual seja possível procurar por deficiências no software ou por partes que devem ser modificadas para algum fim.

As subcaracterísticas contidas nos itens I, II e III referem-se, respectivamente, aos atributos de qualidade:

a) funcionabilidade, confiabilidade e usabilidade.
b) eficiência, manutenibilidade e portabilidade.
c) funcionabilidade, usabilidade e manutenibilidade.
d) confiabilidade, eficiência e manutenibilidade
e) confiabilidade, eficiência e portabilidade.

Comentários:

RECUPERABILIDADE	**Capacidade do produto de software de restabelecer seu nível de desempenho especificado e recuperar os dados diretamente afetados no caso de uma falha.**
COMPORTAMENTO EM RELAÇÃO AO TEMPO	Capacidade do produto de software de fornecer tempos de resposta e de processamento, além de taxas de transferência, apropriados, quando o software executa suas funções, sob condições estabelecidas.
UTILIZAÇÃO DE RECURSOS	Capacidade do produto de software de usar tipos e quantidades apropriados de recursos, quando o software executa suas funções sob condições estabelecidas.
ANALISABILIDADE	Capacidade do produto de software de permitir o diagnóstico de deficiências ou causas de falhas no software, ou a identificação de partes a serem modificadas.

Conforme vimos em aula, trata-se da confiabilidade, eficiência e manutenibilidade. Porém, observem que a questão dá alguns nomes diferentes da norma: resiliência

e recuperabilidade; e comportamento em relação ao tempo e desempenho. Logo, eu acredito que caberia recurso nessa questão.

Gabarito: D

---

22. (CESGRANRIO - 2012 – CHESF – Analista de Sistemas) Dentre os atributos de um software de qualidade, incluem-se:

a) controlabilidade, dependabilidade e eficiência
b) controlabilidade, eficiência e manutenibilidade
c) eficiência, imutabilidade e manutenibilidade
d) eficiência, manutenibilidade e usabilidade
e) imutabilidade, manutenibilidade e usabilidade

Comentários:

Conforme vimos em aula, trata-se da eficiência, manutenibilidade e usabilidade.

Gabarito: D

---

23. (CESPE - 2016 – TCE/PR – Analista de Sistemas) De acordo com a norma ISO/IEC 9126, os atributos de qualidade de software referentes às características de usabilidade são:

a) inteligibilidade, analisabilidade, conformidade e adaptabilidade.

b) estabilidade, testabilidade, utilização de recursos e acessibilidade.

c) inteligibilidade, comportamento com relação ao tempo, atratividade e operacionalidade.

d) acessibilidade, estética, atratividade, inteligibilidade e apreensibilidade.

e) segurança de acesso, maturidade, atratividade e adaptabilidade.

Comentários:

(a) Errado, visto que Analisabilidade e Adaptabilidade não são atributos de usabilidade; (b) Errado, visto que Estabilidade, Testabilidade, Utilização de Recursos e Acessibilidade não são atributo de usabilidade; (c) Errado, visto que Comportamento com relação ao Tempo não é um atributo de usabilidade; (d) Errado, visto que Acessibilidade e Estética não são atributos de usabilidade; (e) Errado, visto que Segurança de Acesso, Maturidade e Adaptabilidade não são atributos de usabilidade. Para mim, essa questão deveria ser anulada, na medida em que não apresenta nenhuma resposta correta.

Gabarito: D

---

24. (CESPE – 2016 – TCE/PR – Analista de Sistema – C) A norma NBR ISO/IEC 9126 define acurácia como a capacidade de um software fornecer resultados com o grau necessário de precisão, sendo, por isso, considerada parte integrante da funcionalidade de um software.

Comentários:

ACURÁCIA	Capacidade do produto de software de prover, com o grau de precisão necessário, resultados ou efeitos corretos ou conforme acordados.

Conforme vimos em aula, acurácia trata da capacidade do produto de software de prover, com o grau de precisão necessário, resultados ou efeitos corretos ou conforme acordados. No entanto, não é parte integrante da funcionalidade de um software - é uma subcaracterística da categoria de funcionalidade da qualidade. Logo, são coisas diferentes.

Gabarito: E

---

25. (CESPE – 2016 – TCE/PR – Analista de Sistema – E) Em vez de ser estimada com base na qualidade interna, a qualidade externa do software deve ser avaliada a partir das características do produto pelo ponto de vista externo, segundo a norma NBR ISO/IEC 9126.

Comentários:

No entanto, a natureza fundamental da qualidade do produto de software representada pela qualidade interna mantém-se inalterada, a menos que seja reprojetada. E a Qualidade Externa, professor? Bem, a Qualidade Externa é a totalidade das características do produto de software do ponto de vista externo. Como assim? Não entendi!

Conforme vimos em aula, a questão está correta.

Gabarito: C

---

26. (FGV – 2017 – ALERJ – Analista de Sistema) Um sistema está sendo desenvolvido por uma empresa terceirizada para apoiar as vendas de um mercado varejista da Grande São Paulo denominado "Mendes Sá Colão". Após o desenvolvimento do sistema, a empresa terceirada deverá passar o código fonte para a área de TI da "Mendes Sá Colão", que passará a ser a necessidade de que o sistema, caso ocorra uma falha, se recupere de forma automática e rapidamente. Nesse caso, os atributos de qualidade do sistema com maior peso são:

a) Portabilidade e confiabilidade;
b) Manutenibilidade e confiabilidade;
c) Portabilidade e Eficiência;
d) Confiabilidade e Usabilidade;
e) Manutenibilidade e eficiência.

Comentários:

MANUTENIBILIDADE	Capacidade do produto de software de ser modificado. As modificações podem incluir correções, melhorias ou adaptações devido a mudanças no ambiente e nos seus requisitos ou especificações funcionais.
CONFIABILIDADE	Capacidade do produto de software de manter um nível de desempenho especificado, quando usado em condições especificadas.

Galera, sempre que uma equipe desenvolve um código-fonte e o repassa para que outra equipe passe a mantê-lo, há um risco manutenibilidade. Por que? Porque o código pode conter "gambiarras", pode estar pouco comentado, pode ter baixa qualidade, entre outros. Além disso, a questão destaca que, caso ocorra uma falha, o software deve ser recuperar de forma automática e rápida. Logo, há uma necessidade inerente de que o código seja confiável.

Gabarito: B

---

27. (CESPE – 2017 – TRE/PE – Analista de Sistema) A ISO barra IEC 9126 descreve uma das características do modelo de qualidade de software como capacidade do produto de software de apresentar desempenho apropriado, relativo à quantidade de recursos usados, sob condições especificadas. Essa característica corresponde à:

a) confiabilidade.
b) eficiência.
c) manutenibilidade.
d) funcionalidade.
e) usabilidade.

Comentários:

EFICIÊNCIA	Capacidade do produto de software de apresentar desempenho apropriado, relativo à quantidade de recursos usados, sob condições especificadas.

Conforme vimos em aula, a questão está perfeita!

Gabarito: B

28. (CESPE – 2017 – TRE/BA - Analista de Sistemas) As normas da série ISO/IEC 9126 estabelecem como medidas da qualidade de software características como: funcionalidade, confiabilidade, usabilidade, eficiência, manutenibilidade e portabilidade. Já a série ISO/IEC 14598 estabelece métricas para mensurar o grau de qualidade, bem como requisitos e orientações para a avaliação do produto de software. Com relação às orientações dessas séries, assinale a opção correta.

a) A série 9126 considera as métricas de qualidade internas e externas, mas a série 14598 não considera as métricas de qualidade em uso.

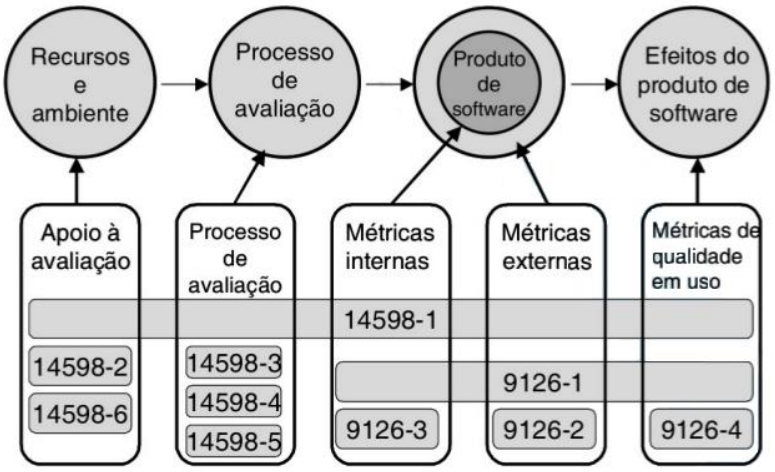
b) A série 14598 considera as métricas de qualidade internas e externas, mas a série 9126 não considera as métricas de qualidade em uso.

c) A série 9126 considera as métricas de qualidade internas, mas a série 14598 não considera as métricas externas nem as de qualidade em uso.

d) A série 14598 considera as métricas de qualidade internas, mas a série 9126 não considera as métricas externas nem as de qualidade em uso.

e) As séries 9126 e 14598 consideram tanto as métricas de qualidade internas e externas quanto as métricas de qualidade em uso.

Comentários:



Relação entre ISO 9126 e ISO 14.598

Conforme vimos em aula, as séries NBR ISO/ IEC 9126 e NBR ISO/ IEC 14598 consideram tanto as métricas de qualidade internas e externas quanto as métricas de qualidade em uso – a imagem explicita isso!

Gabarito: E

---

29. (CESPE – 2017 – TRE/BA - Analista de Sistemas) Um gestor de desenvolvimento de software ficou responsável por avaliar a qualidade de determinado software. Nessa avaliação, ele utilizou atributos categorizados em características, como, por exemplo, a funcionalidade. Para essa característica — funcionalidade —, o usuário do software pode utilizar como métricas as subcaracterísticas:

a) eficiência e interoperabilidade.
b) manutenibilidade e portabilidade.
c) adequação e acurácia.
d) maturidade e confiabilidade.
e) inteligibilidade e usabilidade.

Comentários:

Questão decoreba! Tinha que lembrar da nossa tabelinha da Norma ISO/IEC 9126:

A Funcionalidade é a capacidade do produto de software de prover funções que atendam às necessidades explícitas e implícitas, quando o software estiver sendo utilizado sob condições especificadas.

A Adequação é a capacidade do produto de software de prover um conjunto apropriado de funções para tarefas e objetivos do usuário especificados; Acurácia é

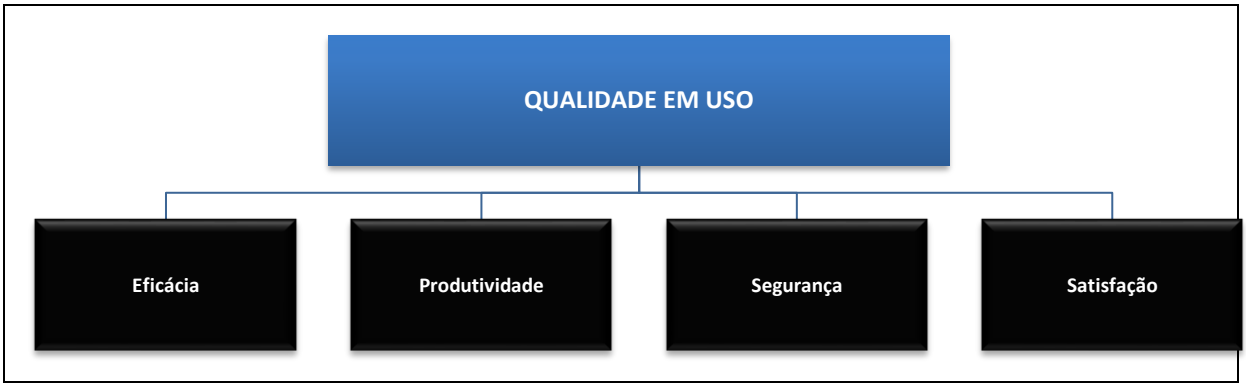
capacidade do produto de software de prover, com o grau de precisão necessário, resultados ou efeitos corretos ou conforme acordados.

Gabarito: C

30. (CESPE – 2017 – TRE/BA - Analista de Sistemas) Um usuário avaliou um software sob o ponto de vista da qualidade em uso, em complemento à medição de qualidade interna e externa do referido software. O produto, sob a perspectiva do usuário, falhou em lhe permitir o atingimento das metas especificadas com acurácia e completude em um contexto de uso especificado. Nessa situação, o software avaliado falhou no atributo:

a) segurança.
b) eficácia.
c) satisfação.
d) analisabilidade.
e) produtividade.

Comentários:



CARACTERÍSTICA	DESCRIÇÃO
EFICÁCIA	Capacidade do produto de software de permitir que usuários atinjam metas especificadas com acurácia e completude, em um contexto de uso especificado.
PRODUTIVIDADE	Capacidade do produto de software de permitir que seus usuários empreguem quantidade apropriada de recursos em relação à eficácia obtida, em um contexto de uso especificado.

SEGURANÇA	Capacidade do produto de software de apresentar níveis aceitáveis de riscos de danos a pessoas, negócios, software, propriedades ou ao ambiente, em um contexto de uso especificado.
SATISFAÇÃO	Capacidade do produto de software de satisfazer usuários, em um contexto de uso especificado.

Conforme vimos em aula, a Eficácia é a capacidade do produto de software de permitir que usuários atinjam metas especificadas com acurácia e completude, em um contexto de uso especificado.

Gabarito: B

ACERTEI	ERREI

LISTA DE EXERCÍCIOS COMENTADOS (DIVERSAS BANCAS)
CONCEITOS BÁSICOS DE ENGENHARIA DE SOFTWARE

1. (CESPE - 2013 - TRT - 10ª REGIÃO (DF e TO) - Analista Judiciário - Tecnologia da Informação) A engenharia de software engloba processos, métodos e ferramentas. Um de seus focos é a produção de software de alta qualidade a custos adequados.

2. (FCC - 2012 - TST - Analista Judiciário - Análise de Sistemas) A Engenharia de Software:

   a) é uma área da computação que visa abordar de modo sistemático as questões técnicas e não técnicas no projeto, implantação, operação e manutenção no desenvolvimento de um software.

   b) consiste em uma disciplina da computação que aborda assuntos relacionados a técnicas para a otimização de algoritmos e elaboração de ambientes de desenvolvimento.

   c) trata-se de um ramo da TI que discute os aspectos técnicos e empíricos nos processos de desenvolvimento de sistemas, tal como a definição de artefatos para a modelagem ágil.

   d) envolve um conjunto de itens que abordam os aspectos de análise de mercado, concepção e projeto de software, sendo independente da engenharia de um sistema.

   e) agrupa as melhores práticas para o concepção, projeto, operação e manutenção de artefatos que suportam a execução de programas de computador, tais como as técnicas de armazenamento e as estruturas em memória principal.

3. (FCC - 2012 - TRT - 6ª Região (PE) - Técnico Judiciário - Tecnologia da Informação) Considere: *é uma disciplina que se ocupa de todos os aspectos da produção de software, desde os estágios iniciais de especificação do sistema até a manutenção desse sistema, depois que ele entrou em operação. Seu principal objetivo é fornecer uma estrutura metodológica para a construção de software com alta qualidade.* A definição refere-se:

a) ao ciclo de vida do software.
b) à programação orientada a objetos.
c) à análise de sistemas.
d) à engenharia de requisitos.
e) à engenharia de software.

4. (CESPE - 2011 - MEC - Gerente de Projetos) A engenharia de software, disciplina relacionada aos aspectos da produção de software, abrange somente os processos técnicos do desenvolvimento de software.

5. (FGV - 2010 - BADESC - Analista de Sistemas - Desenvolvimento de Sistemas) De acordo com Pressman, a engenharia de software é baseada em camadas, com foco na qualidade.

   Essas camadas são:

   a) métodos, processo e teste.
   b) ferramentas, métodos e processo.
   c) métodos, construção, teste e implantação.
   d) planejamento, modelagem, construção, validação e implantação.
   e) comunicação, planejamento, modelagem, construção e implantação.

6. (CESPE - 2010 - Banco da Amazônia - Técnico Científico - Tecnologia da Informação) Os princípios de engenharia de software definem a necessidade de formalidades para reduzir inconsistências e a decomposição para lidar com a complexidade.

7. (FCC - 2010 - TRE-AM - Analista Judiciário - Tecnologia da Informação - A) A Engenharia de Software:

   a) não tem como método a abordagem estruturada para o desenvolvimento de software, pois baseia-se exclusivamente nos modelos de software, notações, regras e técnicas de desenvolvimento.

   b) se confunde com a Ciência da Computação quando ambas tratam do desenvolvimento de teorias, fundamentações e práticas de desenvolvimento de software.

c) tendo como foco apenas o tratamento dos aspectos de construção de software, subsidia a Engenharia de Sistemas no tratamento dos sistemas baseados em computadores, incluindo hardware e software.

d) tem como foco principal estabelecer uma abordagem sistemática de desenvolvimento, através de ferramentas e técnicas apropriadas, dependendo do problema a ser abordado, considerando restrições e recursos disponíveis.

e) segue princípios, tais como, o da Abstração, que identifica os aspectos importantes sem ignorar os detalhes e o da Composição, que agrupa as atividades em um único processo para distribuição aos especialistas.

8. **(FCC - 2011 - INFRAERO - Analista de Sistemas - Gestão de TI)** Em relação à Engenharia de Software, é INCORRETO afirmar:

a) O design de software, ao descrever os diversos aspectos que estarão presentes no sistema quando construído, permite que se faça a avaliação prévia para garantir que ele alcance os objetivos propostos pelos interessados.

b) A representação de um design de software mais simples para representar apenas as suas características essenciais busca atender ao princípio da abstração.

c) Iniciar a entrevista para obtenção dos requisitos de software com perguntas mais genéricas e finalizar com perguntas mais específicas sobre o sistema é o que caracteriza a técnica de entrevista estruturada em funil.

d) No contexto de levantamento de requisitos, funcionalidade é um dos aspectos que deve ser levado em conta na abordagem dos requisitos funcionais.

e) A representação é a linguagem do design, cujo único propósito é descrever um sistema de software que seja possível construir.

9. **(FCC – 2009 – AFR/SP - Analista de Sistemas)** A engenharia de software está inserida no contexto:

a) das engenharias de sistemas, de processo e de produto.
b) da engenharia de sistemas, apenas.
c) das engenharias de processo e de produto, apenas.
d) das engenharias de sistemas e de processo, apenas.
e) das engenharias de sistemas e de produto, apenas.

10. (CESPE – 2015 – STJ – Analista de Sistemas) Embora os engenheiros de software geralmente utilizem uma abordagem sistemática, a abordagem criativa e menos formal pode ser eficiente em algumas circunstâncias, como, por exemplo, para o desenvolvimento de sistemas web, que requerem uma mistura de habilidades de software e de projeto.

11. (CESPE – 2015 – STJ – Analista de Sistemas) O foco da engenharia de software inclui especificação do sistema, desenvolvimento de hardware, elaboração do projeto de componentes de hardware e software, definição dos processos e implantação do sistema.

12. (FUNIVERSA – 2009 – IPHAN – Analista de Sistemas) A Engenharia de Software resume-se em um conjunto de técnicas utilizadas para o desenvolvimento e manutenção de sistemas computadorizados, visando produzir e manter softwares de forma padronizada e com qualidade. Ela obedece a alguns princípios como (1) Formalidade, (2) Abstração, (3) Decomposição, (4) Generalização e (5) Flexibilização. Assinale a alternativa que apresenta conceito correto sobre os princípios da Engenharia de Software.

    a) A flexibilização é o processo que permite que o software possa ser alterado, sem causar problemas para sua execução.

    b) A formalidade é a maneira usada para resolver um problema, de forma genérica, com o intuito de poder reaproveitar essa solução em outras situações semelhantes.

    c) A generalização preocupa-se com a identificação de um determinado fenômeno da realidade, sem se preocupar com detalhes, considerando apenas os aspectos mais relevantes.

    d) Pelo princípio da decomposição, o software deve ser desenvolvido de acordo com passos definidos com precisão e seguidos de maneira efetiva.

    e) A abstração é a técnica de se dividir o problema em partes, de maneira que cada uma possa ser resolvida de uma forma mais específica.

13. (CESPE – 2013 – TCE/RO – Analista de Sistemas) Engenharia de software não está relacionada somente aos processos técnicos de desenvolvimento de softwares,

mas também a atividades como gerenciamento de projeto e desenvolvimento de ferramentas, métodos e teorias que apoiem a produção de softwares.

14. **(CESPE – 2010 – DETRAN/ES – Analista de Sistemas)** Segundo princípio da engenharia de software, os vários artefatos produzidos ao longo do seu ciclo de vida apresentam, de forma geral, nível de abstração cada vez menor.

15. **(CESPE – 2010 – TRE/BA – Analista de Sistemas)** Entre os desafios enfrentados pela engenharia de software estão lidar com sistemas legados, atender à crescente diversidade e atender às exigências quanto a prazos de entrega reduzidos.

16. **(FCC – 2010 – DPE/SP – Analista de Sistemas)** A Engenharia de Software

I. não visa o desenvolvimento de teorias e fundamentações, preocupando-se unicamente com as práticas de desenvolvimento de software.

II. tem como foco o tratamento dos aspectos de desenvolvimento de software, abstraindo-se dos sistemas baseados em computadores, incluindo hardware e software.

III. tem como métodos as abordagens estruturadas para o desenvolvimento de software que incluem os modelos de software, notações, regras e maneiras de desenvolvimento.

IV. segue princípios, tais como, o da Abstração, que identifica os aspectos importantes sem ignorar os detalhes e o da Composição, que agrupa as atividades em um único processo para distribuição aos especialistas.

É correto o que se afirma em:

a) III e IV, apenas.
b) I, II, III e IV.
c) I e II, apenas.
d) I, II e III, apenas.
e) II, III e IV, apenas.

17. **(CESPE – 2013 – TCE/RO – Analista de Sistemas)** Assim como a Engenharia de Software, existe também na área de informática a chamada Ciência da

Computação. Assinale a alternativa que melhor apresenta a diferença entre Engenharia de Software e Ciência da Computação.

a) A Ciência da Computação tem como objetivo o desenvolvimento de teorias e fundamentações. Já a Engenharia de Software se preocupa com as práticas de desenvolvimento de software.

b) A Engenharia de Software trata da criação dos sistemas de computação (softwares) enquanto a Ciência da Computação está ligada ao desenvolvimento e criação de componentes de hardware.

c) A Engenharia de Software trata dos sistemas com base em computadores, que inclui hardware e software, e a Ciência da Computação trata apenas dos aspectos de desenvolvimento de sistemas.

d) A Ciência da Computação trata dos sistemas com base em computadores, que inclui hardware e software, e a Engenharia de Software trata apenas dos aspectos de desenvolvimento de sistemas.

e) A Ciência da Computação destina-se ao estudo e solução para problemas genéricos das áreas de rede e banco de dados e a Engenharia de Software restringe- se ao desenvolvimento de sistemas.

18. (CESPE – 2009 – ANAC – Analista de Sistemas) O termo engenharia pretende indicar que o desenvolvimento de software submete-se a leis similares às que governam a manufatura de produtos industriais em engenharias tradicionais, pois ambos são metodológicos.

19. (CESPE - 2016 – TCE/PR – Analista de Sistemas – B) A engenharia de software refere-se ao estudo das teorias e fundamentos da computação, ficando o desenvolvimento de software a cargo da ciência da computação.

20. (CESPE - 2016 – TCE/PR – Analista de Sistemas – E) O conceito de software se restringe ao desenvolvimento do código em determinada linguagem e seu armazenamento em arquivos.

LISTA DE EXERCÍCIOS COMENTADOS (DIVERSAS BANCAS)

CICLO DE VIDA E PROCESSOS DE DESENVOLVIMENTO DE SOFTWARE

1. (CESPE – 2009 – TCE/TO – Analista de Sistemas - A) Quanto maior e mais complexo o projeto de software, mais simples deve ser o modelo de processo a ser adotado.

2. (CESPE – 2009 – TCE/TO – Analista de Sistemas - B) O modelo de ciclo de vida do software serve para delimitar o alvo do software. Nessa visão, não são consideradas as atividades necessárias e o relacionamento entre elas.

3. (CESPE – 2009 – TCE/TO – Analista de Sistemas - C) A escolha do modelo do ciclo de vida não depende de características específicas do projeto, pois o melhor modelo é sempre o mais usado pela equipe do projeto.

4. (ESAF - 2012 – CGU – Analista de Sistemas) A escolha de um modelo é fortemente dependente das características do projeto. Os principais modelos de ciclo de vida podem ser agrupados em três categorias principais:

   a) sequenciais, cascata e evolutivos.
   b) sequenciais, incrementais e ágeis.
   c) sequenciais, incrementais e evolutivos.
   d) sequenciais, ágeis e cascata.
   e) cascata, ágeis e evolutivos.

5. (CESPE – 2011 – MEC – Analista de Sistemas) O processo de desenvolvimento de software é uma caracterização descritiva ou prescritiva de como um produto de software deve ser desenvolvido.

6. (CESPE – 2013 – TRT/10ª – Analista de Sistemas) As atividades fundamentais relacionadas ao processo de construção de um software incluem a especificação, o desenvolvimento, a validação e a evolução do software.

7. (CESPE – 2010 – TRE/BA – Analista de Sistemas) Um modelo de processo de software consiste em uma representação complexa de um processo de software, apresentada a partir de uma perspectiva genérica.

8. (CESPE – 2011 – MEC – Analista de Sistemas) Atividades comuns a todos os processos de software incluem a especificação, o projeto, a implementação e a validação.

9. (CESPE – 2015 – STJ – Analista de Sistemas) As principais atividades de engenharia de software são especificação, desenvolvimento, validação e evolução.

10. (FCC – 2012 – MPE/AP – Analista de Sistemas) Um processo de software é um conjunto de atividades relacionadas que levam à produção de um produto de software. Existem muitos processos de software diferentes, mas todos devem incluir quatro atividades fundamentais: especificação, projeto e implementação, validação e:

    a) teste
    b) evolução.
    c) prototipação.
    d) entrega.
    e) modelagem.

11. (CESPE – 2010 – EMBASA – Analista de Sistemas) Ciclo de vida de um software resume-se em eventos utilizados para definir o status de um projeto.

12. (CESPE – 2016 – TCE/PR – Analista de Sistemas) As fases do ciclo de vida de um software são:

    a) concepção, desenvolvimento, entrega e encerramento.
    b) iniciação, elaboração, construção e manutenção.
    c) escopo, estimativas, projeto e processo e gerência de riscos.
    d) análise, desenvolvimento, teste, empacotamento e entrega.
    e) planejamento, análise e especificação de requisitos, projeto, implementação, testes, entrega e implantação, operação e manutenção.

13. (MPE/RS – 2012 – MPE/RS – Analista de Sistemas) O ciclo de vida básico de um software compreende:

    a) a implementação, a implantação e o teste.
    b) a análise, a segurança e o controle de usuários.
    c) a implementação, a análise e o teste.
    d) a implementação, a validação e as vendas.

e) a análise, o projeto, a implementação e o teste.

14. (CESGRANRIO – 2011 – PETROBRÁS – Analista de Sistemas) A especificação de uma Metodologia de Desenvolvimento de Sistemas tem como pré-requisito indispensável, em relação ao que será adotado no processo de desenvolvimento, a definição do:

    a) Engenheiro Responsável pelo Projeto
    b) Documento de Controle de Sistemas
    c) Software para Desenvolvimento
    d) Ciclo de Vida do Software
    e) Bloco de Atividades

15. (CESPE – 2008 – INPE – Analista de Sistemas) O ciclo de vida do software tem início na fase de projeto.

16. (CESPE – 2013 – TRT/10 – Analista de Sistemas) O ciclo de vida de um software, entre outras características, está relacionado aos estágios de concepção, projeto, criação e implementação.

17. (CESPE – 2011 – TJ/ES – Analista de Sistemas) Entre as etapas do ciclo de vida de software, as menos importantes incluem a garantia da qualidade, o projeto e o estudo de viabilidade. As demais atividades do ciclo, como a implementação e os testes, requerem maior dedicação da equipe e são essenciais.

18. (CESPE - 2016 – TCE/PR – Analista de Sistemas – A) A engenharia de software está relacionada aos diversos aspectos de produção de software e inclui as atividades de especificação, desenvolvimento, validação e evolução de software.

19. (CESPE - 2016 – TCE/PR – Analista de Sistemas – D) Um processo de software é composto por quatro atividades fundamentais: iniciação, desenvolvimento, entrega e encerramento.

20. (CESPE - 2016 – TCE/PR – Analista de Sistemas – B) A metodologia de processos genérica para a engenharia de software é composta de seis etapas: comunicação, planejamento, modelagem, construção, emprego e aceitação.

21. (INSTITUTO CIDADE – 2012 – TCM/GO – Analista de Sistemas) De acordo com a engenharia de software, como todo produto industrial, o software possui um ciclo de vida. Cada fase do ciclo de vida possui divisões e subdivisões. Em qual

fase avaliamos a necessidade de evolução dos softwares em funcionamento para novas plataformas operacionais ou para a incorporação de novos requisitos?

a) Fase de operação;
b) Fase de retirada;
c) Fase de definição;
d) Fase de design.
e) Fase de desenvolvimento;

22. (CESPE - 2010 – DETRAN/ES – Analista de Sistemas) Quando um aplicativo de software desenvolvido em uma organização atinge, no fim do seu ciclo de vida, a fase denominada aposentadoria, descontinuação ou fim de vida, todos os dados por ele manipulados podem ser descartados.

LISTA DE EXERCÍCIOS COMENTADOS (DIVERSAS BANCAS)

MODELO EM CASCATA

1. (FCC - 2012 - TJ-RJ - Analista Judiciário - Análise de Sistemas - E) Dos diferentes modelos para o ciclo de vida de desenvolvimento de um software é correto afirmar que o modelo em cascata é o mais recente e complexo.

2. (FCC - 2009 - SEFAZ-SP - Agente Fiscal de Rendas - Tecnologia da Informação - Prova 3 - B) O processo de engenharia de software denominado ciclo de vida clássico refere-se ao modelo incremental.

3. (CESPE – 2009 – INMETRO – Analista de Sistemas) Em uma empresa que tenha adotado um processo de desenvolvimento de software em cascata, falhas no levantamento de requisitos têm maior possibilidade de gerar grandes prejuízos do que naquelas que tenham adotado desenvolvimento evolucionário.

4. (CESPE – 2011 – MEC – Analista de Sistemas) O modelo Waterfall tem a vantagem de facilitar a realização de mudanças sem a necessidade de retrabalho em fases já completadas.

5. (CESPE – 2008 – TST – Analista de Sistemas) No modelo de desenvolvimento sequencial linear, a fase de codificação é a que gera erros de maior custo de correção.

6. (CESPE – 2009 – INMETRO – Analista de Sistemas) Em um processo de desenvolvimento em cascata, os testes de software são realizados todos em um mesmo estágio, que acontece após a finalização das fases de implementação.

7. (CESPE – 2008 – SERPRO – Analista de Sistemas) O modelo em cascata consiste de fases e atividades que devem ser realizadas em sequência, de forma que uma atividade é requisito da outra.

8. (CESPE – 2005 – AL/ES – Analista de Sistemas - B) O modelo de desenvolvimento em cascata descreve ciclos sequenciais, incrementais e iterativos, possuindo, entre outras, as fases de requisitos e implementação.

9. (CESPE – 2004 – STJ – Analista de Sistemas) O modelo de desenvolvimento seqüencial linear, também chamado modelo clássico ou modelo em cascata, caracteriza-se por não acomodar adequadamente as incertezas que existem no início de um projeto de software, em especial as geradas pela dificuldade do cliente de explicitar todos os requerimentos que o programa deve contemplar.

10. (CESPE – 2009 – IPEA – Analista de Sistema) No modelo em cascata de processo de desenvolvimento, os clientes devem definir os requisitos apenas durante a fase de projeto; e os projetistas definem as estratégias de projeto apenas durante a fase de implementação. As fases do ciclo de vida envolvem definição de requisitos, projeto, implementação, teste, integração, operação e manutenção. Em cada fase do ciclo de vida, podem ser produzidos diversos artefatos.

11. (CESPE – 2008 – TCE/TO – Analista de Sistema – D) No ciclo de vida em cascata, é possível realizar alternadamente e simultaneamente as atividades de desenvolvimento de software.

12. (CESPE – 2004 – TJ/PA – Analista de Sistema – D) A abordagem sistemática estritamente linear para o desenvolvimento de software é denominada modelo em cascata ou modelo sequencial linear.

13. (CESPE – 2006 – TSE – Analista de Sistema – D) O modelo em cascata organiza o desenvolvimento em fases. Esse modelo encoraja a definição dos requisitos antes do restante do desenvolvimento do sistema. Após a especificação e a análise dos requisitos, têm-se o projeto, a implementação e o teste.

14. (CESPE – 2009 – INMTRO – Analista de Sistema) No desenvolvimento de software, o modelo em cascata é estruturado de tal maneira que as fases que

compõem o desenvolvimento são interligadas. Nessa situação, o final de uma fase implica o início de outra.

15. (CESPE – 2010 – BASA – Analista de Sistema) No modelo em cascata, o projeto segue uma série de passos ordenados. Ao final de cada projeto, a equipe de projeto finaliza uma revisão. O desenvolvimento continua e, ao final, o cliente avalia a solução proposta.

16. (CESPE – 2009 – TRE/MT – Analista de Sistema - A) O modelo em cascata é apropriado para software em que os requisitos ainda não foram bem compreendidos, pois é focado na criação de incrementos.

17. (CESPE – 2009 – UNIPAMPA – Analista de Sistema – D) O modelo em cascata sugere uma abordagem sistemática e sequencial para o desenvolvimento de software. Sua natureza linear leva a estados de bloqueio nos quais, para que nova etapa seja iniciada, é necessário que a documentação associada à fase anterior tenha sido aprovada.

18. (CESPE – 2004 – ABIN – Analista de Sistema) O modelo de desenvolvimento seqüencial linear, também denominado modelo em cascata, é incompatível com o emprego de técnica de análise orientada a objetos no desenvolvimento de um sistema de informação.

19. (CESPE – 2004 – TRE/AL – Analista de Sistema) O modelo cascata ou ciclo de vida clássico necessita de uma abordagem sistemática, que envolve, em primeiro lugar, o projeto e, em seguida, a análise, a codificação, os testes e a manutenção.

20. (CESPE – 2008 – MPE/AM – Analista de Sistema) O modelo de desenvolvimento seqüencial linear tem como característica principal a produção de uma versão básica, mas funcional, do software desde as primeiras fases.

21. (VUNESP - 2012 - SPTrans - Analista de Informática) Uma das abordagens do processo de desenvolvimento da engenharia de software prevê a divisão em etapas, em que o fim de uma é a entrada para a próxima. Esse processo é conhecido como modelo:

a) Transformação.
b) Incremental.
c) Evolutivo.
d) Espiral.

e) Cascata.

22. (CESGRANRIO – 2010 – PETROBRÁS – Analista de Sistemas – Processos de Negócio) No Ciclo de Vida Clássico, também chamado de Modelo Sequencial Linear ou Modelo Cascata, é apresentada uma abordagem sistemática composta pelas seguintes atividades:

a) Análise de Requisitos de Software, Projeto, Geração de Código, Teste e Manutenção.

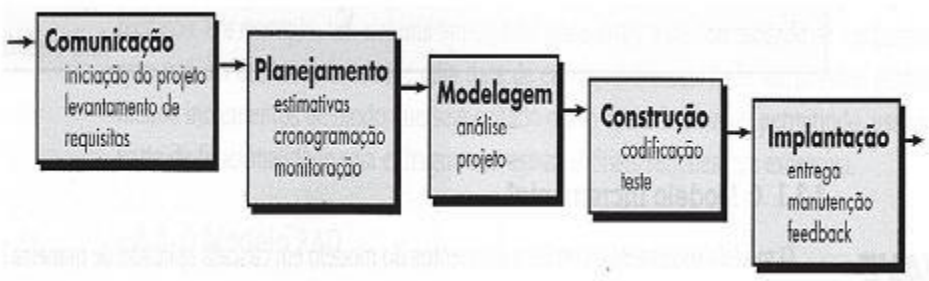
b) Modelagem e Engenharia do Sistema/Informação, Análise de Requisitos de Software, Projeto, Geração de Código, Teste e Manutenção.

c) Modelagem e Engenharia do Sistema/Informação, Projeto, Geração de Código, Teste e Manutenção.

d) Levantamento de Requisitos de Software, Projeto, Geração de Código e Manutenção e Análise de Requisitos de Software.

e) Levantamento de Requisitos de Software, Projeto, Geração de Código, Teste Progressivo e Manutenção.

23. (FGV - 2015 – PGE/RO - Análise de Sistemas) A figura abaixo ilustra um modelo de processo, que prescreve um conjunto de elementos de processo como atividades de arcabouço, ações de engenharia de software, tarefas, produtos de trabalho, mecanismos de garantia de qualidade e de controle de modificações para cada projeto.



Esse modelo é conhecido como Modelo:

a) por funções.
b) em cascata.

c) incremental.
d) em pacotes.
e) por módulos.

24. (CESPE - 2016 – TCE/PR - Analista de Informática) O modelo de desenvolvimento em cascata é utilizado em caso de divergência nos requisitos de um software, para permitir a evolução gradual do entendimento dos requisitos durante a implementação do software.

25. (CESGRANRIO – 2012 – Transpetro – Analista de Sistemas Júnior – Infra-estrutura) Na engenharia de software, existem diversos modelos de desenvolvimento de software, e, dentre eles, o modelo em cascata, o qual, no contexto do desenvolvimento de sistemas de software complexos, recomenda:

a) distribuir a elicitação dos requisitos desde o início até o fim do desenvolvimento.

b) dividir o desenvolvimento do produto de software em fases lineares e sequenciais.

c) enfatizar a avaliação e mitigação de riscos durante o desenvolvimento.

d) realizar entregas incrementais do produto de software ao longo do desenvolvimento.

e) usar prototipagem rápida para estimular o envolvimento do usuário no desenvolvimento.

26. (CESGRANRIO – 2012 – EPE – Analista de Gestão Corporativa – Tecnologia da Informação) Uma das críticas feitas ao modelo do ciclo de vida do desenvolvimento de software em cascata refere-se a:

a) exigência de conhecimento de avaliação e gerenciamento de risco para evitar grandes surpresas no projeto.

b) comprometimentos na qualidade e nas possibilidades de manutenção a longo prazo, parecendo um protótipo.

c) pouca flexibilidade para mudanças futuras, exigindo compromissos nas fases iniciais do projeto.

d) pouca visibilidade das etapas do processo, tornando cara a documentação de todas as versões dos sistemas.

e) exigências de velocidade as quais levam o engenheiro de software a utilizar linguagens, algoritmos ou ferramentas ineficientes ao longo de todo o projeto.

LISTA DE EXERCÍCIOS COMENTADOS (DIVERSAS BANCAS)

MODELOS ITERATIVOS E INCREMENTAIS

1. (CESPE - 2011 – TJ/ES - Analista Judiciário - Análise de Sistemas - Específicos) O modelo de processo incremental de desenvolvimento de software é iterativo, assim como o processo de prototipagem. Contudo, no processo incremental, diferentemente do que ocorre no de prototipagem, o objetivo consiste em apresentar um produto operacional a cada incremento.

2. (CESPE - 2008 – TJ/DF - Analista Judiciário - Análise de Sistemas) No modelo de desenvolvimento incremental, embora haja defasagem entre os períodos de desenvolvimento de cada incremento, os incrementos são desenvolvidos em paralelo.

3. (CESPE - 2009 – UNIPAMPA - Análise de Sistemas) No modelo de desenvolvimento incremental, a cada iteração são realizadas várias tarefas. Na fase de análise, pode ser feito o refinamento de requisitos e o refinamento do modelo conceitual.

4. (CESPE - 2016 – TCE/PR – Analista de Sistemas – C) No modelo iterativo de desenvolvimento de software, as atividades são dispostas em estágios sequenciais.

LISTA DE EXERCÍCIOS COMENTADOS (DIVERSAS BANCAS)

NBR ISSO.IEC 12207

1. (FCC - 2010 – TRT/PR – Técnico Judiciário – Tecnologia da Informação) Manutenção de software, segundo a norma ISO 12207, trata-se de um processo dentro do grupo de processos:

   a) de projeto.
   b) de reúso de software.
   c) de implementação de software.
   d) de suporte de software.
   e) técnicos.

2. (FCC - 2010 – TRF/04 – Analista Judiciário – Tecnologia da Informação) Sobre a norma ISO/IEC 12207, considere:

   I. Define objetivos, níveis de maturidade organizacional ou de capacidade de processo.

   II. Provê uma estrutura para que uma organização defina seus processos.

   III. Cobre também a garantia da qualidade, que se estende desde os produtos adquiridos ou fornecidos até a qualidade e melhoria dos processos de implementação.

   a) I, apenas.
   b) I, II e III.
   c) I e II, apenas.
   d) II e III, apenas.
   e) III, apenas.

3. (FCC - 2012 – ARCE – Analista de Regulação – Analista de Sistemas) A Norma ISO/IEC 12207:2008 agrupa as atividades que podem ser realizadas durante o ciclo de vida de um sistema de software em sete grupos de processos. Cada um dos processos do ciclo de vida dentro desses grupos é descrito em termos da sua finalidade e resultados esperados. Dentre estes grupos de processos encontra-se o grupo de Processos de:

a) Manutenção de Segurança.
b) Implementação de Software.
c) Capacitação Profissional.
d) Qualidade e Segurança.
e) Fornecimento e Desenvolvimento.

4. (FCC - 2012 – TRT/11 – Analista Judiciário – Analista de Sistemas) No grupo de processos de projeto, segundo a Norma ISO 12207, dentre outros, encontra-se o processo de gerenciamento:

a) da Infraestrutura.
b) de recursos humanos.
c) do modelo de ciclo de vida.
d) de configuração.
e) de qualidade.

5. (FCC - 2013 - TRT - 9ª REGIÃO (PR) - Analista Judiciário - Tecnologia da Informação) A ISO/IEC 12207 objetiva criar um framework que possibilite uma linguagem comum para a criação e o gerenciamento do software. Essa norma:

a) é aplicada para certificação de processos em um esquema formal e é imposta por diversos governos, dentre eles, do Brasil e dos Estados Unidos, como condição para realizar negócios com empresas privadas.

b) descreve os processos para a criação e gerenciamento de software e ainda especifica como implementar e desempenhar as atividades e tarefas incluídas nos processos.

c) define no processo fundamental de fornecimento as atividades do comprador (a organização que adquire o sistema, produto de software ou serviço de software) e compreende as seguintes atividades: iniciação, preparação do request for proposal, preparação de contrato, monitoramento do fornecedor e aceitação do produto ou serviço.

d) cobre o ciclo de vida do software, desde a sua concepção até o seu descarte, os processos para aquisição e suprimento de produtos de software e serviços, assim como os processos para controle e melhoria.

e) define como processos do Grupo de Processos Fundamentais: Documentação, Gerência de Configuração, Garantia da Qualidade, Verificação, Validação, Revisão Conjunta, Treinamento, Auditoria e Resolução de Problemas.

6. **(FCC - 2011 – TRT/1 - Analista Judiciário - Tecnologia da Informação)** Uma das atividades recomendadas no processo de manutenção da NBR ISO/IEC 12207 é a:

a) implementação do processo.
b) gerência de liberação e distribuição.
c) descontinuação do software.
d) avaliação da configuração.
e) identificação da configuração.

7. **(FCC - 2011 – TRT/4 - Analista Judiciário - Tecnologia da Informação)** A arquitetura de alto nível do ciclo de vida de software estabelecida pela Norma ISO/IEC 12207 está embasada em:

a) métodos, treinamentos e tecnologias empregadas.
b) métricas, ferramentas e tecnologias empregadas.
c) processo e seus inter-relacionamentos.
d) processos, ferramentas e métricas.
e) métodos, processos, ferramentas e treinamentos.

8. **(FCC - 2013 – TRT/15 - Analista Judiciário - Tecnologia da Informação)** André trabalha no desenvolvimento de um software para o Tribunal Regional do Trabalho da 15a Região. Recentemente seu chefe cogitou adotar uma Norma que se aplica ao desenvolvimento de produtos de software. André foi o encarregado de escolher a Norma adequada. Pesquisou então a norma ABNT NBR ISO/IEC 12207:2009, que se aplica à aquisição de sistemas e produtos de software e serviços para o fornecimento, desenvolvimento, operação, manutenção e descontinuidade de produtos de software. André descobriu que esta Norma pode ser usada:

(I) Em um projeto, para ajudar a selecionar, estruturar e utilizar os elementos de um conjunto de processos de ciclo de vida estabelecidos que forneçam produtos e serviços. Desse modo, esta Norma pode ser usada na avaliação de conformidade do projeto para o ambiente estabelecido e declarado.

(II) Por uma organização, para ajudar a estabelecer um ambiente de processos desejados. Esses processos podem ser sustentados por uma infraestrutura de métodos, procedimentos, técnicas, ferramentas e pessoal treinado. A organização pode empregar esse ambiente para realizar e gerenciar seus projetos e seus sistemas em andamento durante as fases do ciclo de vida. Desse modo, essa Norma pode ser usada para avaliar a conformidade de um conjunto declarado e estabelecido de processos do ciclo de vida de acordo com as necessidades.

(III) Por um adquirente e um fornecedor, para ajudar a estabelecer um acordo em relação aos processos e às atividades. Esse acordo contempla os processos e atividades desta Norma que são selecionados, negociados, acordados e executados. Desse modo, esta Norma pode ser usada para orientar a definição do acordo.

(IV) Por organizações avaliadoras e avaliadores credenciados, para realizar avaliações que possam ser usadas para obtenção de certificação oficial. Esta Norma fornece um conjunto definido de processos para que a organização obtenha certificação ISO/IEC no prazo máximo de 1 ano.

Está correto o que se afirma APENAS em:

a) I, II e III.
b) I e II.
c) III e IV.
d) II, III e IV.
e) IV.

9. (FCC - 2013 – TRT/15 - Analista Judiciário - Tecnologia da Informação) Após escolher a norma ABNT NBR ISO/IEC 12207:2009 para ser adotada na organização onde trabalha, André verificou que a Norma é dividida em sete grupos de processos. Como sua especialidade é em análise de requisitos, verificou que o Processo de Análise de Requisitos do Sistema e o Processo de Análise de Requisitos de Software estavam, respectivamente, nos grupos de Processos:

a) Organizacionais Capacitadores de Projeto e Técnicos.
b) de Implementação de Software e de Apoio ao Software.
c) Técnicos e de Implementação de Software.
d) de Projeto e Técnicos.

e) de Projeto e de Implementação de Software.

10. **(FCC - 2014 – TRF/3 - Analista Judiciário - Tecnologia da Informação)** Na Norma ABNT ISO/IEC 12207:2009, um dos Processos Técnicos é o Processo de Definição dos Requisitos dos Stakeholders. Com relação a este Processo, analise as tarefas e atividades a seguir:

1. Identificação dos Stakeholders.
2. Classificação dos Stakeholders.
3. Identificação dos Requisitos.
4. Classificação dos Requisitos.
5. Avaliação dos Requisitos.
6. Acordo dos Requisitos.
7. Mediação de Conflitos.
8. Registro dos Requisitos.
9. Produção do Documento de Requisitos.

De acordo com a Norma supracitada, as atividades e tarefas que devem ser implementadas em consonância com as políticas e procedimentos organizacionais aplicáveis, com relação ao Processo de Definição dos Requisitos dos Stakeholders, são as que constam APENAS em 1,

a) 2, 3, 4, 5, 6, 8 e 9.
b) 3, 5, 8 e 9.
c) 3, 5, 6 e 8.
d) 2, 3, 4, 6, 7 e 8.
e) 4, 6, 8 e 9.

11. **(FCC - 2014 – TRF/3 - Analista Judiciário - Tecnologia da Informação)** Na ABNT ISO/IEC 12207:2009 os processos estão agrupados. Dentre os processos contextuais do sistema constam os Processos Contratuais, os Processos Organizacionais Capacitadores de Projeto, os Processos de Projeto e os Processos:

a) de Apoio ao Software.
b) de Auditoria de Software.
c) de Implementação de Software.
d) de Reuso de Software.
e) Técnicos.

12. (CESPE – 2016 – TCE/PR – Analista de Sistema – A) De acordo com a norma NBR ISO/IEC 12207, o fornecedor de um sistema ou software deve ser uma entidade externa à organização.

13. (CESPE – 2016 – TCE/PR – Analista de Sistema – D) A norma NBR ISO/IEC 12207 dispõe que a gerência de melhoria e o treinamento compõem o conjunto de processos de apoio do ciclo de vida de um software.

14. (CESPE – 2016 – TCE/PR – Analista de Sistema – A) Conforme a norma NBR ISO/IEC 12207, as tarefas de descontinuação, migração ou desativação de um software incluem-se no processo de operação do ciclo de vida do software.

15. (CESPE – 2016 – TCE/PR – Analista de Sistema – E) De acordo com a norma NBR ISO/IEC 12207, o ciclo de vida de um software é constituído de cinco processos fundamentais: aquisição, fornecimento, desenvolvimento, operação e manutenção.

16. (FCC – 2016 – TRT/SE – Analista de Sistema) A norma NBR ISO/IEC 12207:2009 estabelece uma estrutura comum para os processos de ciclo de vida de software, que pode ser referenciada pela indústria de software. A norma agrupa as atividades que podem ser executadas durante o ciclo de vida de um sistema que contém software em sete grupos de processo. No grupo de Processos de Projeto encontra-se o processo de:

    a) Gestão de Modelo de Ciclo de Vida, que tem como propósito definir, manter e garantir disponibilidade das políticas e modelos de ciclo de vida de software.

    b) Gestão de Risco, cujo propósito é identificar, analisar, tratar e monitorar riscos de forma contínua.

    c) Gestão de Configuração e Ativos de Serviço, cujo propósito é garantir a instalação e configuração dos softwares e controlar os ativos de serviço.

    d) Definição dos Requisitos dos Stakeholders, cujo propósito é definir os requisitos funcionais e não funcionais de um sistema.

    e) Aquisição, que tem como propósito obter um produto que satisfaça a necessidade expressa pelo adquirente.

LISTA DE EXERCÍCIOS COMENTADOS (DIVERSAS BANCAS)
NBR ISO/IEC 9126

1. (CONSULPLAN - 2010 – Prefeitura de Santa Maria Madalena – Analista de Sistemas) São categorias de características principais de qualidade de software, segundo a Norma (ISO/IEC 9126: NBR 13596), EXCETO:

   a) Eficiência.
   b) Segurança.
   c) Confiabilidade.
   d) Usabilidade.
   e) Funcionalidade.

2. (COVEST - 2013 – UFPE – Analista de Sistemas) A norma ISO/IEC 9126 é um padrão internacional para a qualidade de software, a qual é composta de uma série de características. Sobre essa norma, é correto afirmar que:

   a) descreve seis características relacionadas à qualidade de software, e como elas devem ser medidas.

   b) compatibilidade é uma das características, que é composta por duas subcaracterísticas: coexistência e interoperabilidade.

   c) funcionalidade é a característica que visa verificar se, durante um período de tempo, o sistema funciona de acordo com as condições preestabelecidas.

   d) foi atualizada pela norma ISO/IEC 25010, a qual só adicionou a característica de segurança.

   e) a característica que determina métricas para avaliar o comportamento do sistema em relação a tempo e a recursos utilizados é a eficiência.

3. (FGV - 2009 – MEC – Analista de Sistemas) Analise a citação a seguir.

   "Um conjunto de atributos que têm impacto na capacidade do software de manter o seu nível de desempenho dentro de condições estabelecidas por um dado período de tempo."

A Norma que integra os conceitos de ambiente, estratégias e planejamento de testes, é conhecida por:

a) ISO 12207
b) ISO 15504
c) ISO 9126
d) IEEE 829
e) MPS.BR.

4. (FGV - 2009 – MEC – Analista de Sistemas) Entre os critérios de qualidade da Norma ISO 9126 não se inclui:

a) a manutenibilidade.
b) a funcionalidade.
c) a confiabilidade.
d) a utilizabilidade.
e) a eficicácia.

5. (IADES - 2013 - EBSERH - Analista de Tecnologia da Informação - Teste e Qualidade) De acordo com o padrão de qualidade ISO 9126, são identificados seis atributos fundamentais da qualidade. Sobre o tema, assinale a alternativa correta.

a) A usabilidade diz respeito à quantidade de tempo, que o software fica disponível para uso.

b) A eficiência é o grau com que o software satisfaz às necessidades declaradas.

c) A disponibilidade é o grau de tempo em que o software permanece no ar para utilização

d) A portabilidade é a facilidade com a qual um software pode ser transportado de um ambiente para outro.

e) A confidencialidade é a capacidade de manter partes do software, em sigilo, só sendo permitido o conhecimento, por parte de pessoas autorizadas.

6. (FGV - 2010 - DETRAN-RN – Programador) Assinale a alternativa que NÃO contém somente atributos para características externas e internas do modelo de qualidade de software, definido na ISO/IEC 9126-1:

a) Funcionalidade, confiabilidade, usabilidade.
b) Funcionalidade, confiabilidade, eficiência.
c) Funcionalidade, confiabilidade, alta gerência.
d) Funcionalidade, usabilidade, portabilidade.
e) Eficiência, manutenibilidade, portabilidade.

7. (FCC - 2007 - TRE-SE - Analista Judiciário - Tecnologia da Informação) Considere as questões chave apresentadas na seguinte tabela, com o enfoque da ISO 9126 (NBR 13596) ? Qualidade de Software

Questão chave	
I.	Propõe-se a fazer o que é apropriado?
II.	Faz o que foi proposto de forma correta?
III.	Com que freqüência apresenta falhas?
IV.	Há grande risco quando se faz alterações?

As seguintes sub características aplicáveis à avaliação da qualidade do software: Maturidade, Estabilidade, Acurácia e Adequação são aplicáveis, respectivamente, às questões chave:

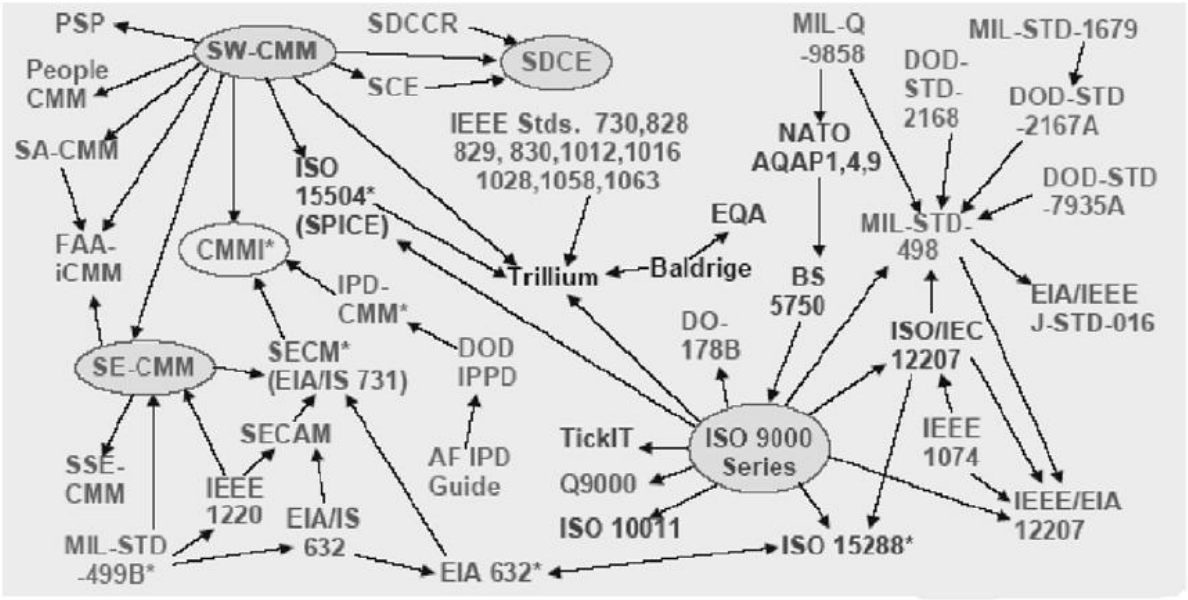
a) I, II, IV e III.
b) II, I, III e IV.
c) III, IV, II e I.
d) IV, III, II e I.
e) IV, III, I e II.

8. (UFF – 2008 – DATAPREV – Analista de Sistemas) A norma ISO 9.126 foi desenvolvida para identificar atributos de qualidade para software de computador. O período de tempo em que o software está disponível para uso, indicado pelos sub-atributos maturidade, tolerância à falha e recuperabilidade, é caracterizado pelo atributochave:

a) portabilidade;
b) confiabilidade;
c) manutenibilidade;

d) eficiência;
e) funcionalidade.

9. (CESPE - 2009 – INMETRO – Analista de Sistemas) Um modelo para a avaliação contínua de capacidade de processos é descrito na norma NBR ISO/IEC 9126, que deriva do ciclo da melhoria contínua presente na norma ISO 9001.



Internet: <www.software.org> (com adaptações).

A figura acima, elaborada no ano de 1998, apresenta uma proposta de relacionamento de precedência de criação e de incorporação de conceitos entre diversos modelos de qualidade de processo de engenharia de software e de sistemas. Não estão representados no modelo as normas e os modelos NBR ISO/IEC 27001, MPS.BR e NBR ISO/IEC 9126.

10. (CESPE - 2009 – INMETRO – Analista de Sistemas) A norma NBR ISO/IEC 9126-1, que define atributos de qualidade externa e interna, como funcionalidade, confiabilidade, usabilidade, eficiência, manutenibilidade e portabilidade, não figura no mapa porque, principalmente, não estava disponível à época.

11. (CESPE - 2015 – STJ – Analista de Sistemas) A manutenibilidade é atributo de qualidade externa que pode ser medida por atributos internos, como a profundidade da árvore de herança e a complexidade ciclomática.

12. (CESPE - 2015 – STJ – Analista de Sistemas) A apreensibilidade cuida da capacidade de o usuário compreender se o software é apropriado e como este pode ser usado para a tarefa e as condições específicas.

13. (CESPE - 2015 – STJ – Analista de Sistemas) A funcionalidade e a usabilidade, características dos atributos de qualidade de software, possuem como subcaracterísticas, respectivamente, a operacionalidade e a interoperabilidade.

14. (QUADRIX - 2015 – COBRA – Analista de Sistemas) De acordo com a norma ISO/IEC 9126, a qualidade do produto software está relacionada às seguintes características: Funcionalidade, Confiabilidade, Usabilidade, Eficiência, Manutenibilidade e Portabilidade. Sobre o tema, assinale a afirmação correta.

a) A Manutenibilidade diz que o produto de software deve ser capaz de manter seu nível de desempenho, ao longo do tempo, nas condições estabelecidas.

b) A Confiabilidade está relacionada ao esforço necessário para a utilização do sistema, baseado em um conjunto de implicações e de condições do usuário.

c) A Usabilidade refere-se à compatibilidade dos recursos e os tempos envolvidos compatíveis com o nível de desempenho requerido pelo software.

d) A Funcionalidade refere-se à existência de funções e propriedades específicas do produto, que satisfazem as necessidades do usuário.

e) A Eficiência diz respeito à facilidade de o software poder ser transferido de um ambiente para outro.

15. (IDECAN - 2014 – AGU – Analista de Sistemas) "Detalhes da qualidade do produto de software podem ser melhorados durante a implementação do código, revisão e teste, mas a natureza fundamental da qualidade do produto de software representada pela qualidade_____ mantém-se inalterada, a menos que seja reprojetada." Assinale a alternativa que completa corretamente a afirmativa anterior.

a) interna
b) em uso
c) externa
d) em uso estimada (ou prevista)
e) externa estimada (ou prevista)

16. (VUNESP - 2014 – DESENVOLVESP – Analista de Sistemas) A norma ISO 9126 (Engenharia de Software – Qualidade do Produto) estabelece um modelo de qualidade com 6 atributos. Dentre eles, está o atributo eficiência, que visa medir:

a) a facilidade de se fazer manutenções corretiva e adaptativa no software.
b) a facilidade de transportar o software de um computador para outro.
c) o número de erros detectados por dia de operação.
d) o nível no qual o software utiliza, de forma otimizada, os recursos do sistema computacional.
e) o tempo máximo decorrido entre duas paradas simultâneas do software.

17. (CESGRANRIO - 2014 – EPE – Analista de Sistemas) Com a evolução das pesquisas na área de qualidade, ficou cada vez mais claro para os pesquisadores que este é um conceito complexo e multifacetado. Muitos autores desenvolveram modelos de qualidade baseados na ideia de descrever qualidade como um conjunto de características ou atributos, organizadas de forma hierárquica. Esse movimento também aconteceu na área de qualidade de software, resultando em múltiplos modelos.

Um marco importante nessa discussão foi o estabelecimento de um modelo padrão de qualidade de software, representado na norma ISO/IEC 9126, que identificou seis características da qualidade de software, cada uma delas com um conjunto de subcaracterísticas.

Com relação a esse padrão, a acurácia, ou seja, a capacidade de o produto de software prover com o grau de precisão necessário resultados ou efeitos corretos ou conforme acordados é uma subcaracterística de:

a) Portabilidade
b) Usabilidade
c) Confiabilidade
d) Eficiência
e) Funcionalidade

18. (CESPE - 2013 – STF – Analista de Sistemas) A qualidade de software abrange apenas os aspectos internos e externos decorrentes do uso e, portanto, pode ser medida durante a utilização do software por parte do usuário.

19. (CESPE - 2013 – STF – Analista de Sistemas) Do ponto de vista histórico, o termo usabilidade evoluiu a partir do termo qualidade em uso, que, por sua vez,

substituiu o termo interface amigável, principalmente devido à pouca abrangência e subjetividade que estes últimos sugeriam.

20. (IADES - 2013 – EBSERH – Analista de Sistemas) De acordo com o padrão de qualidade ISO 9126, são identificados seis atributos fundamentais da qualidade. Sobre o tema, assinale a alternativa correta.

a) A usabilidade diz respeito à quantidade de tempo, que o software fica disponível para uso.

b) A eficiência é o grau com que o software satisfaz às necessidades declaradas.

c) A disponibilidade é o grau de tempo em que o software permanece no ar para utilização.

d) A portabilidade é a facilidade com a qual um software pode ser transportado de um ambiente para outro.

e) A confidencialidade é a capacidade de manter partes do software, em sigilo, só sendo permitido o conhecimento, por parte de pessoas autorizadas.

21. (FCC - 2012 – TJ/PE – Analista de Sistemas) No contexto dos atributos de qualidade de software, considere:

I. A resiliência é a capacidade de o sistema voltar ao nível de desempenho anterior a falhas ou comportamento imprevisto de usuários, software ou hardware e recuperar os dados afetados, caso existam.

II. O desempenho e uso de recursos referem-se à capacidade do sistema de alcançar tempos de resposta, latência, tempo de processamento, vazão, etc dentro do período de tempo especificado e ao fato do software exigir mais ou menos recursos de acordo com suas condições de uso.

III. A analisabilidade é o grau de facilidade, com qual seja possível procurar por deficiências no software ou por partes que devem ser modificadas para algum fim.

As subcaracterísticas contidas nos itens I, II e III referem-se, respectivamente, aos atributos de qualidade:

a) funcionabilidade, confiabilidade e usabilidade.
b) eficiência, manutenibilidade e portabilidade.
c) funcionabilidade, usabilidade e manutenibilidade.
d) confiabilidade, eficiência e manutenibilidade
e) confiabilidade, eficiência e portabilidade.

22. (CESGRANRIO - 2012 – CHESF – Analista de Sistemas) Dentre os atributos de um software de qualidade, incluem-se:

a) controlabilidade, dependabilidade e eficiência
b) controlabilidade, eficiência e manutenibilidade
c) eficiência, imutabilidade e manutenibilidade
d) eficiência, manutenibilidade e usabilidade
e) imutabilidade, manutenibilidade e usabilidade

23. (CESPE - 2016 – TCE/PR – Analista de Sistemas) De acordo com a norma ISO/IEC 9126, os atributos de qualidade de software referentes às características de usabilidade são:

a) inteligibilidade, analisabilidade, conformidade e adaptabilidade.

b) estabilidade, testabilidade, utilização de recursos e acessibilidade.

c) inteligibilidade, comportamento com relação ao tempo, atratividade e operacionalidade.

d) acessibilidade, estética, atratividade, inteligibilidade e apreensibilidade.

e) segurança de acesso, maturidade, atratividade e adaptabilidade.

24. (CESPE – 2016 – TCE/PR – Analista de Sistema – C) A norma NBR ISO/IEC 9126 define acurácia como a capacidade de um software fornecer resultados com o grau necessário de precisão, sendo, por isso, considerada parte integrante da funcionalidade de um software.

25. (CESPE – 2016 – TCE/PR – Analista de Sistema – E) Em vez de ser estimada com base na qualidade interna, a qualidade externa do software deve ser avaliada a partir das características do produto pelo ponto de vista externo, segundo a norma NBR ISO/IEC 9126.

26. (FGV – 2017 – ALERJ – Analista de Sistema) Um sistema está sendo desenvolvido por uma empresa terceirizada para apoiar as vendas de um mercado varejista da Grande São Paulo denominado "Mendes Sá Colão". Após o desenvolvimento do sistema, a empresa terceirada deverá passar o código fonte para a área de TI da "Mendes Sá Colão", que passará a ser a necessidade de que o sistema, caso ocorra uma falha, se recupere de forma automática e rapidamente. Nesse caso, os atributos de qualidade do sistema com maior peso são:

a) Portabilidade e confiabilidade;
b) Manutenibilidade e confiabilidade;
c) Portabilidade e Eficiência;
d) Confiabilidade e Usabilidade;
e) Manutenibilidade e eficiência.

27. (CESPE – 2017 – TRE/PE – Analista de Sistema) A ISO barra IEC 9126 descreve uma das características do modelo de qualidade de software como capacidade do produto de software de apresentar desempenho apropriado, relativo à quantidade de recursos usados, sob condições especificadas. Essa característica corresponde à:

a) confiabilidade.
b) eficiência.
c) manutenibilidade.
d) funcionalidade.
e) usabilidade.

28. (CESPE – 2017 – TRE/BA - Analista de Sistemas) As normas da série ISO/IEC 9126 estabelecem como medidas da qualidade de software características como: funcionalidade, confiabilidade, usabilidade, eficiência, manutenibilidade e portabilidade. Já a série ISO/IEC 14598 estabelece métricas para mensurar o grau de qualidade, bem como requisitos e orientações para a avaliação do produto de software. Com relação às orientações dessas séries, assinale a opção correta.

a) A série 9126 considera as métricas de qualidade internas e externas, mas a série 14598 não considera as métricas de qualidade em uso.

b) A série 14598 considera as métricas de qualidade internas e externas, mas a série 9126 não considera as métricas de qualidade em uso.

c) A série 9126 considera as métricas de qualidade internas, mas a série 14598 não considera as métricas externas nem as de qualidade em uso.

d) A série 14598 considera as métricas de qualidade internas, mas a série 9126 não considera as métricas externas nem as de qualidade em uso.

e) As séries 9126 e 14598 consideram tanto as métricas de qualidade internas e externas quanto as métricas de qualidade em uso.

29. (CESPE – 2017 – TRE/BA - Analista de Sistemas) Um gestor de desenvolvimento de software ficou responsável por avaliar a qualidade de determinado software. Nessa avaliação, ele utilizou atributos categorizados em características, como, por exemplo, a funcionalidade. Para essa característica — funcionalidade —, o usuário do software pode utilizar como métricas as subcaracterísticas:

a) eficiência e interoperabilidade.
b) manutenibilidade e portabilidade.
c) adequação e acurácia.
d) maturidade e confiabilidade.
e) inteligibilidade e usabilidade.

30. (CESPE – 2017 – TRE/BA - Analista de Sistemas) Um usuário avaliou um software sob o ponto de vista da qualidade em uso, em complemento à medição de qualidade interna e externa do referido software. O produto, sob a perspectiva do usuário, falhou em lhe permitir o atingimento das metas especificadas com acurácia e completude em um contexto de uso especificado. Nessa situação, o software avaliado falhou no atributo:

a) segurança.
b) eficácia.
c) satisfação.
d) analisabilidade.
e) produtividade.

GABARITO DOS EXERCÍCIOS COMENTADOS (DIVERSAS BANCAS)
CONCEITOS BÁSICOS DE ENGENHARIA DE SOFTWARE

1	2	3	4	5	6	7	8	9	10
C	A	E	E	B	C	D	E	A	C
11	12	13	14	15	16	17	18	19	20
E	A	C	C	C	D	A	C	E	E

GABARITO DOS EXERCÍCIOS COMENTADOS (DIVERSAS BANCAS)
CICLO DE VIDA E PROCESSOS DE DESENVOLVIMENTO DE SOFTWARE

1	2	3	4	5	6	7	8	9	10
E	E	E	C	C	C	E	C	C	B
11	12	13	14	15	16	17	18	19	20
E	E	E	D	E	C	E	C	E	E
21	22	23	24	25	26	27	28	29	30
B	E								

GABARITO DOS EXERCÍCIOS COMENTADOS (DIVERSAS BANCAS)
MODELO EM CASCATA

1	2	3	4	5	6	7	8	9	10
E	E	C	E	E	E	C	E	C	E
11	12	13	14	15	16	17	18	19	20
E	C	C	C	E	E	C	E	E	E
21	22	23	24	25	26	27	28	29	30
E	B	B	E	B	C				

GABARITO DOS EXERCÍCIOS COMENTADOS (DIVERSAS BANCAS)
MODELOS ITERATIVOS E INCREMENTAIS

1	2	3	4	5	6	7	8	9	10
C	C	C	E						

GABARITO DOS EXERCÍCIOS COMENTADOS (DIVERSAS BANCAS)
NBR ISSO.IEC 12207

1	2	3	4	5	6	7	8	9	10
E	C	B	D	D	X	C	A	C	C
11	12	13	14	15	16	17	18	19	20
E	E	E	E	C	B				

GABARITO DOS EXERCÍCIOS COMENTADOS (DIVERSAS BANCAS)
NBR ISO/IEC 9126

1	2	3	4	5	6	7	8	9	10
B	E	X	X	D	C	C	B	E	E
11	12	13	14	15	16	17	18	19	20
C	E	E	D	A	D	E	E	E	D
21	22	23	24	25	26	27	28	29	30
D	D	D	E	C	B	B	E	C	B

ESSA LEI TODO MUNDO CONHECE:
PIRATARIA É CRIME.

Mas é sempre bom revisar o porquê e como você pode ser prejudicado com essa prática.

1. 

Professor investe seu tempo para elaborar os cursos e o site os coloca à venda.

2. 

Pirata divulga ilicitamente (grupos de rateio), utilizando-se do anonimato, nomes falsos ou laranjas (geralmente o pirata se anuncia como formador de "grupos solidários" de rateio que não visam lucro).

3. 

Pirata cria alunos fake praticando falsidade ideológica, comprando cursos do site em nome de pessoas aleatórias (usando nome, CPF, endereço e telefone de terceiros sem autorização).

4. 

Pirata compra, muitas vezes, clonando cartões de crédito (por vezes o sistema anti-fraude não consegue identificar o golpe a tempo).

5. 

Pirata fere os Termos de Uso, adultera as aulas e retira a identificação dos arquivos PDF (justamente porque a atividade é ilegal e ele não quer que seus fakes sejam identificados).

6. 

Pirata revende as aulas protegidas por direitos autorais, praticando concorrência desleal e em flagrante desrespeito à Lei de Direitos Autorais (Lei 9.610/98).

7. 

Concurseiro(a) desinformado participa de rateio, achando que nada disso está acontecendo e esperando se tornar servidor público para exigir o cumprimento das leis.

8. 

O professor que elaborou o curso não ganha nada, o site não recebe nada, e a pessoa que praticou todos os ilícitos anteriores (pirata) fica com o lucro.

Deixando de lado esse mar de sujeira, aproveitamos para agradecer a todos que adquirem os cursos honestamente e permitem que o site continue existindo.