

Guilherme da Silva Scher

Liste cinco perguntas que:

- (1) Os projetistas deveriam fazer aos usuários.
- (2) Os usuários deveriam fazer aos projetistas.
- (3) Os usuários deveriam fazer a si mesmos sobre o produto de software a ser desenvolvido.
- (4) Os projetistas deveriam fazer a si mesmos sobre o produto de software a ser construído e sobre o processo que será usado para construí-lo.

Perguntas que os projetistas fariam

O que você espera alcançar com este software?

Que problemas enfrenta atualmente?

Qual sua familiaridade com tecnologia?

Como você imagina a interface ideal?

Com que frequência espera atualizações?

Perguntas que os usuários fariam

Como este software resolverá meus problemas?

Em quanto tempo tu consegue entregar?

Como será o suporte depois que o software for entregue?

Como serão tratadas as mudanças caso eu queira alterar algo no meio do processo de fabricação?

Você vai me ensinar a usar esse software?

Perguntas dos usuários para si mesmos

O que que eu quero resolver?

Eu tô disposto a usar ferramentas que não pretendia na minha cabeça?

Quanto que eu tô disposto a investir?

Com base no que que eu vou saber que tive sucesso?

O que vai acontecer caso eu não consiga realizar direito?

Perguntas dos projetistas para si mesmos

O escopo tá entendível?

Os requisitos fazem sentido?

Que riscos que eu preciso evitar?

O grupo tem as habilidades necessárias?

O processo escolhido se adapta ao projeto do cliente?

2 Discuta as diferenças entre os vários fluxos de processo descritos(linear, iterativo, etc). Identifique os tipos de problemas que poderiam ser aplicáveis a cada um dos fluxos genéricos descritos.

Linear pode ser aplicado a projetos com requisitos certos e definidos.

Iterativo é ideal para projetos que tenham muitas mudanças ou o cliente esteja confuso sobre oq ele quer.

Incremental é útil quando partes do sistema podem ser entregues uma por uma complementando.

Espiral é indicado para projetos complicados e com risco.

Ágil é bom pra ter feedback frequente.

3. Tente desenvolver um conjunto de ações para a atividade de comunicação. Selecione uma ação e defina um conjunto de tarefas para ela.

A comunicação é importante na gestão de um projeto, sempre tendo que ser ressaltada as tarefas de cada participante, definir como será a comunicação durante o processo (se vai ser por meio de logs diários ou algo do tipo) e pegar as expectativas de cada um em relação ao projeto.

4. Durante a comunicação, um problema comum ocorre ao encontrarmos dois envolvidos com ideias conflitantes sobre como o software deve ser. Isto é, há requisitos mutuamente conflitantes.

Eu acho que se deve identificar e analisar os argumentos e ideias dos conflitantes, e depois tentar gerir uma proposta de solução que agrade e tenha a validação de todos, sem deixar ninguém na mão.

5. Dê três exemplos de projetos de software que seriam suscetíveis ao modelo cascata. Seja específico.

Cascata geralmente se usa pra projetos solidos, que não vão ter muitas mudanças e são mais rigorosos, vou dar uns exemplos.

Controle de tráfego aéreo: Sistemas críticos, as mudanças podem ferrar a segurança.

Sistema bancário: Foco em segurança e precisão, sem muitas mudanças.

Software para equipamentos médicos (como o medidor de remédio ou algo do tipo): Porque qualquer dose a mais pode matar um paciente, então precisa ser muito exato.

6. Dê três exemplos de projetos de software que seriam suscetíveis ao modelo de prototipação. Seja específico.

O modelo de prototipação funciona bem quando o que o cliente quer é meio incerto, e ao longo do projeto vão ter várias mudanças.

Aplicativo de rede social que sempre tem que ter feedbacks e sugestões, pois as redes sociais estão sempre adicionando coisas novas como filtros e tudo mais.

Sistema de recomendação, tipo o algoritmo do tiktok, que sempre vai se adaptando com base nas atualizações do aplicativo.

7. À medida que se desloca para fora ao longo do fluxo de processo em espiral, o que pode ser dito em relação ao software que está sendo desenvolvido ou sofrendo manutenção?

A cada ciclo o software fica mais completo, com melhor desempenho e menos risco.

8. É possível combinar modelos de processo? Em caso positivo, dê um exemplo.

Sim, da pra usar prototipação no começo, pra ter algo pronto e que se possa visualizar e cascata no final.

9. É possível combinar modelos de processo? Em caso positivo, dê um exemplo.

É bom pois há a entrega rápida e o custo baixo, mas tem mais bugs e manutenção frequente pra corrigir essa pressa..

10. É possível provar que um componente de software e até mesmo um programa inteiro está correto? Então, por que todo mundo não faz isso?

Completamente eu acho que é muito difícil, pois requiere muito tempo, esforço e dinheiro, então preferem por testes automatizados e correção ao longo do uso do software.

11. Processo Unificado e UML são a mesma coisa? Justifique sua resposta.

Não, o processo unificado é uma abordagem de desenvolvimento, enquanto a UML é uma linguagem de modelagem usada para documentar sistemas.

