1. Qual a diferença mais importante entre o desenvolvimento de um produto genérico de software e o desenvolvimento de software sob demanda? O que isso pode significar na prática para usuários de produtos de software genérico?

A diferença essencial é que, no desenvolvimento de produtos de software genéricos, especificação é de propriedade do desenvolvedor do produto. Para desenvolvimento de produtos personalizados, a especificação é de propriedade e controlada pelo cliente. As implicações disso são significativas - o desenvolvedor pode decidir rapidamente alterar a especificação resposta a alguma mudança externa (por exemplo, um produto concorrente), mas quando cliente possui a especificação, as mudanças precisam ser negociadas entre o cliente e desenvolvedor e pode ter implicações contratuais. Para usuários de produtos genéricos, isso significa que eles não têm controle sobre especificação de software, não pode controlar a evolução do produto. O desenvolvedor pode decidir incluir / excluir recursos e alterar a interface do usuário. Isso poderia ter implicações para os processos de negócios do usuário e adicionar custos extras de treinamento quando novas versões do sistema são instaladas. Também pode limitar a flexibilidade para mudar seus próprios processos de negócios.

2. Quais são os quatro atributos importantes que todo software profissional deve possuir? Sugira outros quatro atributos que, às vezes, podem ser significantes.

Quatro atributos importantes são capacidade de manutenção, confiabilidade, desempenho e usabilidade. Outros atributos que podem ser significativos: capacidade de reuso (pode ser reutilizado em outras aplicações), capacidade de distribuição (pode ser distribuído através de uma rede de processadores), portabilidade (pode operar em múltiplas plataformas, por exemplo, plataformas móveis) e interoperabilidade (pode funcionar com uma vasta gama de outros sistemas de software).

3. Além dos desafios de heterogeneidade, mudanças sociais e corporativas, confiança e proteção, identifique outros problemas e desafios que a engenharia de software provavelmente enfrentará no século XXI.

   a. Desenvolvimento de sistemas eficientes em termos de energia. Isso os torna mais utilizáveis em dispositivos móveis de baixa potência e ajuda a reduzir a pegada de carbono global de equipamentos de TI.

b. Desenvolvimento de técnicas de validação para sistemas de simulação (que serão essencial para prever a extensão e o planejamento das mudanças climáticas).
c. Desenvolvimento de sistemas para uso multicultural
d. Desenvolvimento de sistemas que podem ser adaptados rapidamente a novas necessidades de negócios
e. Projetar sistemas para desenvolvimento terceirizado
f. Desenvolver sistemas resistentes ao ataque
g. Desenvolvimento de sistemas que podem ser adaptados e configurados pelos usuários finais
h. Encontrar novas maneiras de testar, validar e manter o usuário final envolvido no processo de desenvolvimento de sistemas.

4. Explique como o uso universal da Internet mudou os sistemas de software?

Com o uso universal da Internet, o software passou a ser visto mais como um serviço, apresentando características semelhantes ao funcionamento da Internet. Por exemplo, a possibilidade de armazenamento de dados em nuvem, o aluguel de capacidade de processamento de centros de processamento, ou facilidades de redes. Uma vantagem desta arquitetura e forma de contratação destes serviços é a possibilidade de delegar a operação e atualização do sistema para outras empresas, reduzindo os custos de implantação e operação do software. Uma desvantagem é a dependência direta do funcionamento da Internet.

5. Explique por que o desenvolvimento incremental é o método mais eficaz para o desenvolvimento de software de negócios. Por que esse modelo é menos adequado para a engenharia de sistemas de tempo real?

O desenvolvimento incremental é mais adequado para quando há incertezas e mudanças constantes nas necessidades de negócio. O desenvolvimento incremental é baseado na ideia de desenvolver uma implementação inicial, expô-la a avaliação dos usuários, e planejar os próximos incrementos até que um sistema adequado seja desenvolvido, evoluindo junto com as necessidades e experiência do usuário com o sistema.

Menos adequado para a engenharia de sistemas de tempo real, pois estes precisam de requisitos claramente definidos e estáveis. Isso deve ser planejado com antecedência, e não desenvolvido de forma incremental.

6. Sugira por que é importante, no processo de engenharia de requisitos, fazer uma distinção entre desenvolvimento dos requisitos do usuário e o desenvolvimento de requisitos do sistema.

Os requisitos do usuário destinam-se a descrever as funções do sistema e recursos de uma perspectiva do usuário e é essencial que os usuários entendam esses requisitos. Devem ser expressos em linguagem natural e podem não deve ser expresso em grande detalhe, para permitir alguma flexibilidade de implementação. As pessoas envolvidas no processo devem entender o usuário ambiente e domínio de aplicação.

Os requisitos do sistema são muito mais detalhados do que os requisitos do usuário e destinam-se a ser uma especificação precisa do sistema que pode ser parte de um contrato de sistema. Eles também podem ser usados em situações em que desenvolvimento é terceirizado ea equipe de desenvolvimento precisa de um completo especificação do que deve ser desenvolvido. Os requisitos do sistema são desenvolvidos depois que os requisitos do usuário foram estabelecidos.

7. Explique por que, em sistemas complexos, as mudanças são inevitáveis. Exemplifique as atividades de processo de software que ajudam a prever as mudanças e fazer com que o software seja desenvolvido mais tolerante a mudanças.

Os sistemas devem mudar porque, como estão instalados em um ambiente, ambiente se adapta a eles e esta adaptação naturalmente gera novos / diferentes requisitos de sistema. Além disso, o ambiente do sistema é dinâmico e gera constantemente novos requisitos como consequência de mudanças nos negócios, objetivos de negócios e políticas de negócios. A menos que o sistema esteja adaptado para refletir esses requisitos, sua implantação ficará fora de sintonia com os requisitos para apoiar o negócio e, portanto, se tornará menos útil.

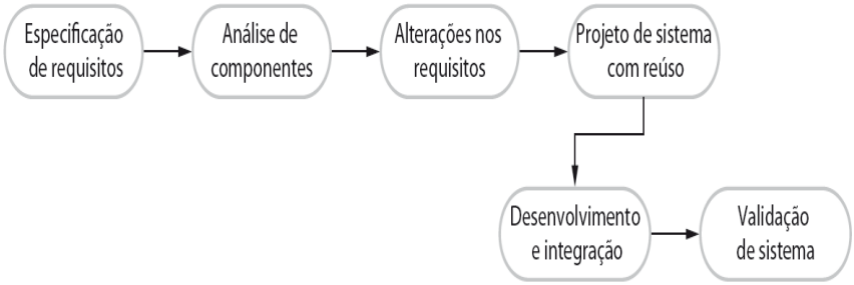
Exemplos de práticas de processo que ajudam a prever as mudanças:

a. Justificar cada requisito. Isso ajuda com a mudança futura.
b. Manter a rastreabilidade de requisitos, realçando a dependências entre requisitos e entre os requisitos e o projeto/código do sistema.
c. Documentar a estrutura e projeto do software.
d. Refatorar o código buscando a melhoria na qualidade do código e, portanto, torna-o mais fácil mudar.

8. Explique por que os sistemas desenvolvidos como protótipos normalmente não devem ser usados como sistemas de produção.

    a. Pode ser impossível ajustar o protótipo para atender aos requisitos não funcionais, como requisitos de desempenho, proteção, robustez e confiabilidade, que foram ignorados durante o desenvolvimento do protótipo.
    b. Mudanças rápidas durante o desenvolvimento inevitavelmente significam que o protótipo não está documentado. A única especificação de projeto é o código do protótipo. Para a manutenção a longo prazo, isso não é bom o suficiente.
    c. As mudanças durante o desenvolvimento do protótipo provavelmente terão degradado a estrutura do sistema. O sistema será difícil e custoso de ser mantido.
    d. Padrões de qualidade organizacional geralmente são relaxados para o desenvolvimento do protótipo.

9. Considere o modelo da Engenharia de software orientada a reuso (Figura abaixo). Explique por que, nesse processo, é essencial ter duas atividades distintas de engenharia de requisitos?



Em um processo baseado em reutilização, você precisa de duas atividades de engenharia de requisitos, porque é essencial adaptar os requisitos do sistema de acordo com as capacidades do sistema/componentes a serem reutilizados.

10. Explique por que o modelo em espiral de Boehm é um modelo adaptável, que apoia tanto as atividades de prevenção de mudanças quanto as de tolerância a mudanças. Na prática, esse modelo não tem sido amplamente usado. Sugira as possíveis razões para isso.

O modelo de Boehm foi um dos primeiros modelos de desenvolvimento de software a reconhecer o risco no processo de desenvolvimento de software. Contudo, a avaliação do

risco requer um grande esforço e experiência do time de desenvolvimento, o que nem sempre é uma atividade trivial, principalmente quando se trata de um produto (software) inovador, sem um histórico consistente de ações pregressas que possa ser utilizado como parâmetro da avaliação do risco. O modelo de Boehm pode funcionar melhor em grandes projetos.

11. Quais são as vantagens de proporcionais visões estáticas e dinâmicas do processo de software, assim como no Rational Unified Process?

Os processos podem ser descritos como uma sequencia de atividades estáticas, tais como, levantamento de requisitos, projeto, implementação, etc. A perspectiva dinâmica incluída no RUP descreve como cada uma destas atividades estáticas poderiam ser usadas em cada fase do processo. A visão dinâmica do RUP pode variar de uma organização para outra, sem a imposição de um processo particular ao modelo.

12. Historicamente a introdução de tecnologia provocou mudanças profundas no mercado de trabalho e, pelo menos temporariamente, deixou muitas pessoas desempregadas. Discuta se a introdução da automação extensiva em processos pode vir a ter as mesmas consequências para os engenheiros de software. Se sua resposta for não, justifique. Se você acha que sim, que vai reduzir as oportunidades de emprego, é ética a resistência passiva ou ativa, pelos engenheiros afetados, à introdução dessa tecnologia?

De uma certa forma, as novas tecnologias trazem mudanças de forma ampla em todas as atividades e profissões, incluindo a Engenharia de Software. Em relação a automação, hoje temos ferramentas que automatizam diversas etapas da engenharia de software. É o caso das ferramentas de gerenciamento dos itens de configuração de um projeto de software. Há também uma maior oferta de componentes de reuso nos processos de engenharia de software, incluindo a tendência de determinadas funcionalidades do software serem tratadas como serviço. É o caso do gerenciamento do banco de dados e segurança da informação. Neste cenário, eu imagino a necessidade de atualização constante do engenheiro de software quanto as novas tecnologias e tendências de mercado, representando um desafio e uma oportunidade de elevar a produtividade a um novo padrão de escala.