Resumo Detalhado dos Tópicos - Engenharia de Software (Prova 1) Este módulo abrange desde conceitos fundamentais de software até práticas modernas de desenvolvimento. Os principais eixos são: 1. Fundamentos: definição de software, engenharia de software, projeto, ciclo de vida, riscos e métricas. 2. Gerenciamento de Projetos (PMBOK): grupos de processos, áreas de conhecimento, artefatos, planos e métricas. 3. Configuração e Versionamento: SCMP, sistemas como Git, boas práticas de versionamento e controle de mudanças. 4. Modelagem OO: classes, objetos, herança, associações, princípios de modelagem, completude, consistência e corretude. 5. Processos de Software: modelos em cascata, incremental, espiral, evolucionário, exploratório, DevOps e Processo Unificado. 6. Software Livre (OSS): organização de comunidades, requisitos de sucesso, comunicação, modelos de gestão. 7. Métodos Ágeis: Scrum, XP (pares, TDD, user stories), Kanban, manifesto ágil e integração de práticas. 8. Prototipação: protótipos exploratórios, throw-away, mockups e prototipagem evolutiva.

1. Defina software, sistema de software e engenharia de software
Software: conjunto de instruções, dados e documentação que permitem a execução de tarefas em sistemas computacionais. Sistema de software: coleção integrada de programas e módulos que trabalham em conjunto para atingir objetivos funcionais e não funcionais. Engenharia de software: disciplina que aplica princípios de engenharia para o desenvolvimento sistemático de software de qualidade, dentro de prazos e orçamentos.

2. Defina projeto no contexto de gerenciamento de projetos
Um projeto é um esforço temporário empreendido para criar um produto, serviço ou resultado único. No gerenciamento de projetos, destaca-se por ter início, fim, escopo e recursos bem definidos.

3. Relacione dificuldades em projeto de software
- Mudanças frequentes de requisitos; - Estimativas imprecisas de prazo e custo; - Complexidade técnica e integração com sistemas legados; - Falhas de comunicação na equipe; - Garantia de qualidade e manutenção pós-entrega.

4. Defina ciclo de vida de projeto e propósitos de suas fases
O ciclo de vida de projeto descreve fases sequenciais: - Iniciação (definir viabilidade e autorização); - Planejamento (detalhar escopo, cronograma, custos); - Execução (implementação das tarefas); - Monitoramento e Controle (acompanhar progresso, corrigir desvios); - Encerramento (formalizar entregas, documentar lições aprendidas).

5. Modelo iterativo e incremental
Modelo Iterativo: permite revisitar requisitos e soluções em ciclos sucessivos para refinar o sistema. Modelo Incremental: entrega o software em partes funcionais (incrementos). - Iteração: um ciclo de desenvolvimento que gera uma versão parcial ou revisada. - Incremento: nova funcionalidade adicionada ao produto.

6. Defina marco em projeto de software
Um marco (milestone) é um ponto de controle ou evento significativo, como a conclusão de uma fase ou entrega de documento importante.

7. Arcabouço de gerenciamento de projetos e elementos integrantes
Arcabouço é um conjunto de metodologias, processos e ferramentas para padronizar a gestão de projetos. Elementos típicos: modelos de ciclo de vida, métricas, templates de documentos, plano de riscos, ferramentas de monitoramento.

8. Grupos de processos do PMBOK
- Iniciação: autorizar o início do projeto; - Planejamento: definir detalhadamente como executar; - Execução: realizar o trabalho planejado; - Monitoramento e Controle: medir desempenho e corrigir desvios; - Encerramento: finalizar o projeto, validar entregas.

9. Áreas de conhecimento do PMBOK
Integração, Escopo, Cronograma, Custos, Qualidade, Recursos, Comunicações, Riscos, Aquisições e Partes Interessadas. Cada área organiza processos e práticas para garantir controle e sucesso do projeto.

10. Exemplo de processo de cada área do PMBOK
Exemplos: - Integração: desenvolver termo de abertura. - Escopo: coletar requisitos. - Cronograma: desenvolver cronograma. - Custos: estimar orçamento. - Qualidade: garantir qualidade. - Recursos: montar equipe. - Comunicação: planejar comunicações. - Riscos: identificar riscos. - Aquisições: conduzir aquisições. - Partes Interessadas: gerenciar engajamento.

11. Cinco artefatos resultantes do PMBOK
- EAP (WBS): decomposição do escopo em entregáveis. - Cronograma: planejamento temporal. - Plano de Riscos: estratégias de mitigação. - Matriz de Comunicação: canais de contato. - Relatório de Desempenho: acompanhamento de progresso.

12. Defina métrica e relacione cinco métricas relevantes
Métrica: medida quantitativa usada para monitorar desempenho do projeto. Exemplos: esforço em horas, custo real x planejado, número de defeitos, produtividade da equipe, valor agregado.

13. Defina risco e exemplifique
Risco é a possibilidade de ocorrência de evento incerto que afeta objetivos. Exemplo: atraso na entrega por indisponibilidade de fornecedor.

14. Categorias de risco em desenvolvimento de software
- Técnicos: falhas em tecnologia nova. - Organizacionais: rotatividade de equipe. - Externos: mudança regulatória. - Gerenciais: estimativas ruins. - De requisitos: mal definidos ou instáveis.

15. Atividades do gerenciamento de riscos
1. Identificação de riscos. 2. Análise qualitativa e quantitativa. 3. Planejamento de respostas. 4. Implementação de estratégias. 5. Monitoramento e revisão contínua.

16. Defina versão
Versão é uma instância específica do software controlada em sistema de versionamento, com identificador único.

17. Recursos típicos de sistema de versionamento
Controle de versões, criação de branches, merges, histórico completo de mudanças, rastreabilidade de autor e data.

18. Elementos de plano de gestão de configuração (SCMP)
Inclui: procedimentos de identificação de itens, políticas de controle de mudanças, auditorias de configuração, relatórios de status.

19. Completude, consistência e corretude em modelos
- Completude: cobre todos requisitos necessários. - Consistência: não apresenta contradições. - Corretude: representa fielmente os requisitos.

20. Princípios de modelagem
Simplicidade, clareza, abstração, verificabilidade, reutilização de modelos.

21. Defina classe em OO e exemplifique
Classe: define atributos e comportamentos. Ex.: Classe 'Carro' com atributos cor e modelo, métodos acelerar() e frear().

22. Defina objeto em OO e exemplifique
Objeto: instância de uma classe. Ex.: Carro meuCarro = new Carro("Vermelho").

23. Associação, agregação, composição e herança
Associação: relação genérica (Aluno-Curso). Agregação: 'tem-um' fraca (Time-Jogador). Composição: 'parte-de' forte (Casa-Cômodos). Herança: especialização (Pessoa-Aluno).

24. Ciclo de vida de desenvolvimento de software
Etapas: levantamento de requisitos, análise, design, implementação, testes, manutenção. Exemplo: modelo cascata tradicional.

25. Produto e produto de software
Produto: resultado final tangível ou intangível. Produto de software: solução digital construída para atender requisitos.

26. Ciclo de vida de produto de software
Introdução (lançamento), Crescimento (adoção), Maturidade (estabilidade), Declínio (substituição). Exemplo: evolução de aplicativos móveis.

27. Classes de processos de software
- Primários: diretamente ligados ao desenvolvimento. - Organizacionais: gestão e suporte. - De apoio: qualidade, configuração e documentação.

28. Modelo cascata
Sequencial: requisitos → análise → design → implementação → testes → manutenção. Vantagem: simplicidade e clareza. Desvantagem: pouca flexibilidade a mudanças.

29. Modelo evolucionário
Sistema evolui em versões sucessivas. Vantagem: adaptação a mudanças. Desvantagem: complexidade e controle difícil.

30. Modelo incremental
Entrega de funcionalidades em incrementos. Vantagem: valor rápido ao cliente. Desvantagem: exige planejamento robusto de integração.

31. Modelo espiral
Iterativo com foco em análise de riscos. Vantagem: adequado para projetos críticos. Desvantagem: alto custo e complexidade.

32. Modelo exploratório
Pouco planejamento inicial, ajustes constantes. Vantagem: flexibilidade. Desvantagem: risco de desorganização.

33. Modelo DevOps e práticas
Integra desenvolvimento e operações. Práticas: integração contínua, entrega contínua, automação de testes, infraestrutura como código, monitoramento contínuo.

34. Responsabilidades no Processo Unificado
- Concepção: viabilidade e visão. - Elaboração: arquitetura base. - Construção: implementação. - Transição: entrega e treinamento.

35. Workflow no UP
Fluxo de atividades relacionadas. Ex.: workflow de requisitos (captura, análise, validação).

36. Software de código aberto
Software distribuído com código-fonte disponível para uso, modificação e redistribuição.

37. Requisitos para sucesso de OSS
Comunidade ativa, governança clara, documentação, licença aberta, ferramentas colaborativas.

38. Modelo bug driven development
Foco na correção rápida de defeitos detectados, orientando o desenvolvimento.

39. Canais de comunicação OSS
Listas de e-mail, fóruns, IRC, Slack, GitHub/GitLab Issues, wikis.

40. Gestão centralizada OSS
Um líder ou grupo controla direção do projeto.

41. Gestão OSS baseada em mérito
Contribuintes ganham influência conforme a qualidade de suas contribuições.

42. Cinco princípios do manifesto ágil
Indivíduos e interações, software funcionando, colaboração com cliente, resposta a mudanças, entregas frequentes.

43. Programação em pares no XP

Dois programadores trabalham juntos no mesmo código. Vantagens: qualidade, revisão imediata, compartilhamento de conhecimento. Desvantagens: uso intensivo de recursos, possível conflito.

44. Test Driven Design (TDD) no XP
Escrever testes antes do código. Força design modular, garante código testável e reduz defeitos.

45. História de usuário e exemplo
Descrição simples do ponto de vista do usuário. Exemplo: 'Como cliente, quero acompanhar meu pedido para saber o status da entrega.'

46. Uso de histórias de usuário no XP
Guiam backlog, priorizam funcionalidades e organizam o desenvolvimento iterativo.

47. Defina eventos do Scrum
- Sprint: iteração fixa (2-4 semanas). - Sprint Planning: definir metas. - Daily Scrum: reunião curta diária. - Sprint Review: revisar entregas. - Retrospective: melhorar processo.

48. Método Kanban
Método visual com colunas (A fazer, Fazendo, Feito), limites de WIP e fluxo contínuo.

49. Integração Kanban + Scrum
Scrum organiza ciclos fixos, Kanban melhora fluxo e transparência.

50. Processo Pessoal (PSP/PXP)
Envolve planejamento individual, coleta de dados de tempo, análise e melhoria contínua.

51. Protótipo e exemplo
Versão inicial ou modelo de sistema. Exemplo: mockup de interface com botões e fluxos básicos.

52. Protótipo throw-away
Criado apenas para validar requisitos ou ideias e depois descartado. Exemplo: rascunho de interface em papel para validar design.