INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE MATO GROSSO DO SUL

KAYKY EDUARDO TONY, YAN VITOR FERREIRA DA SILVA.

MODELOS DE PROCESSO

Três Lagoas, Mato Grosso do Sul
2024

KAYKY EDUARDO TONY, YAN VITOR FERREIRA DA SILVA.

MODELOS DE PROCESSO

Trabalho sobre Modelos de processo e engenharia de software.

Orientador: Elisângela

Três Lagoas, Mato Grosso do Sul

2024

1 **RELATÓRIO SOBRE MODELOS DE PROCESSO.**

Introdução:

Este relatório tem como objetivo apresentar um resumo dos principais conceitos de Engenharia de Software, baseados nos livros "Engenharia de Software" de Roger Pressman e "Engenharia de Software" de Ian Sommerville. Esses autores são referências na área e suas obras são amplamente utilizadas para a formação e atualização de profissionais de software.

Parte 1: Engenharia de Software - Roger Pressman

Definição e Importância

Segundo Pressman, Engenharia de Software é a aplicação de uma abordagem sistemática, disciplinada e quantificável para o desenvolvimento, operação e manutenção de software. A importância de tratar o desenvolvimento de software como uma prática de engenharia é melhorar a qualidade, eficiência e eficácia dos produtos desenvolvidos.

Processos de Desenvolvimento

Pressman descreve vários modelos de processo, incluindo:

Modelo Cascata

O Modelo Cascata, também conhecido como Modelo Sequencial Linear, é uma abordagem tradicional e estruturada para o desenvolvimento de software. Ele segue uma sequência linear de fases, onde cada fase deve ser concluída antes que a próxima comece.

Características Principais:

- Fases Distintas: As fases típicas incluem Requisitos, Design, Implementação, Testes, Implantação e Manutenção.
- Documentação Extensiva: Cada fase gera uma série de documentos que servem como entrada para a próxima fase.
- Rigoroso e Metódico: Não permite voltar a fases anteriores uma vez que estão concluídas, exceto através de um processo formal de revisão.

Vantagens*:*

- Clareza e Estrutura: Facilidade de entendimento e gestão do projeto.
- Controle e Documentação: Bom para projetos com requisitos bem compreendidos e estáveis desde o início.

Desvantagens:

- Rigidez: Dificuldade em acomodar mudanças nos requisitos ao longo do desenvolvimento.
- Feedback Tardio: Problemas e erros muitas vezes só são descobertos nas fases finais de teste.

Modelo Espiral

O Modelo Espiral, desenvolvido por Barry Boehm, é uma abordagem evolutiva que combina elementos do modelo cascata com a prototipagem. Ele enfatiza a avaliação e redução de riscos em cada iteração do ciclo de desenvolvimento.

Características Principais:

- Iterativo e Incremental: O desenvolvimento ocorre em ciclos, cada um passando por várias fases de refinamento.
- Foco em Riscos: Cada ciclo começa com a identificação e análise de riscos, seguida de atividades para reduzir ou eliminar esses riscos.
- Prototipagem: Criação de protótipos para validar requisitos e soluções.

Vantagens:

- Flexibilidade: Permite a revisão e modificação de requisitos durante o desenvolvimento.
- Gerenciamento de Riscos: Identificação e mitigação de riscos antecipadamente.

Desvantagens*:*

- Complexidade: Pode ser mais complexo de gerenciar devido ao foco constante em riscos.
- Custo: Pode ser mais caro devido à necessidade de protótipos e avaliações contínuas.

Modelo Incremental

O Modelo Incremental divide o desenvolvimento do software em incrementos menores e gerenciáveis, onde cada incremento adiciona funcionalidade ao produto final. Cada incremento passa por todas as fases do desenvolvimento (planejamento, análise, design, implementação, teste).

Características Principais:

- Desenvolvimento por Incrementos: Cada incremento fornece uma parte da funcionalidade total.
- Entrega Progressiva: Funcionalidades são entregues em etapas, permitindo o uso parcial do sistema antes que ele esteja completamente desenvolvido.

Vantagens:

- Flexibilidade: Fácil de adicionar novas funcionalidades de forma incremental.
- Risco Reduzido: Menor risco de falha total, já que partes funcionais do sistema são entregues e testadas ao longo do tempo.

Desvantagens:

- Planejamento Inicial: Requer um planejamento inicial detalhado para identificar os incrementos.
- Integração Contínua: Necessidade de integração contínua dos incrementos pode ser desafiadora.

Engenharia de Requisitos

A engenharia de requisitos é um componente crucial que envolve a elicitação, análise, especificação e validação dos requisitos do sistema. Pressman enfatiza que requisitos bem definidos são fundamentais para o sucesso de qualquer projeto. A falha em capturar requisitos corretamente pode levar a retrabalho significativo e a problemas durante o desenvolvimento.

Design e Arquitetura

Pressman detalha técnicas de design modular e padrões de projeto, destacando que um bom design facilita a manutenção e a evolução do software. Ele também

discute a importância da arquitetura de software, que serve como uma estrutura para o desenvolvimento e manutenção do sistema.

Testes de Software

Pressman discute métodos de teste como caixa-branca e caixa-preta. Testes caixa-branca envolvem a verificação da lógica interna do código, enquanto testes caixa-preta se concentram na funcionalidade externa do software. A importância de testes exaustivos é ressaltada para garantir a qualidade e a confiabilidade do software.

Manutenção

Ele classifica a manutenção em quatro tipos:

Corretiva: Corrige defeitos encontrados após a entrega.

Adaptativa: Adapta o software para um novo ambiente ou novas condições.

Perfectiva: Melhora o desempenho ou adiciona novas funcionalidades.

Preventiva: Realiza melhorias para evitar problemas futuros.

2 **- RELATÓRIO SOBRE MODELOS DE PROCESSO.**

Parte 2: Engenharia de Software - Ian Sommerville

Definição e Importância

Sommerville também define a Engenharia de Software como uma abordagem sistemática para o desenvolvimento de software. Ele destaca a engenharia de software como essencial para melhorar a qualidade, o custo e a eficiência dos processos de desenvolvimento.

Processos de Desenvolvimento

Sommerville discute tanto métodos tradicionais quanto metodologias ágeis:

Scrum: Enfatiza iterações curtas, chamadas sprints, e reuniões diárias para monitorar o progresso.

Extreme Programming (XP): Foca em práticas de desenvolvimento como programação em par e entrega contínua de pequenas releases.

Engenharia de Requisitos

Sommerville amplia a abordagem de Pressman, focando na comunicação constante com os stakeholders e na evolução contínua dos requisitos ao longo do ciclo de vida do software. Ele sugere técnicas como entrevistas, workshops e análise de documentos para a elicitação de requisitos.

Design e Arquitetura

Ele destaca a importância de arquiteturas de software robustas e escaláveis, que suportem a evolução e manutenção a longo prazo. Sommerville também discute

a importância do design orientado a objetos e do uso de padrões de design para criar sistemas flexíveis e reutilizáveis.

Testes de Software

Sommerville enfatiza os testes automatizados e contínuos como parte integral das metodologias ágeis, garantindo que o software atenda aos requisitos funcionais e não funcionais. Ele aborda diversas técnicas de teste, incluindo testes unitários, de integração e de sistema.

Manutenção e Gerência de Configuração

Sommerville aborda a gerência de configuração como uma prática essencial para controlar mudanças e assegurar a rastreabilidade de versões de software. Ele descreve processos para gerenciar mudanças de requisitos e controlar versões, garantindo que todas as alterações sejam documentadas e rastreáveis.

Conclusão

A Engenharia de Software, conforme discutida por Pressman e Sommerville, é uma disciplina multifacetada que requer uma abordagem estruturada para garantir o desenvolvimento de sistemas de alta qualidade. A compreensão dos princípios e práticas discutidos por esses autores é fundamental para o sucesso na criação de software eficiente, eficaz e adaptável.

3 **REFERÊNCIAS:**

- PRESSMAN, Roger S. Engenharia de Software. 6. ed. São Paulo: McGraw Hill, 2005.

- SOMMERVILLE, Ian. Engenharia de Software. 9. ed. São Paulo: Pearson, 2011.