Resumo do Arquivo: "Aula_4_Texto.pdf"

Este arquivo foca nas etapas de análise e refinamento de requisitos, utilizando técnicas como prototipação e modelagem para garantir que o software a ser desenvolvido atenda às necessidades dos stakeholders.

Tema 1 – Prototipação de Software

A prototipação é uma técnica usada na engenharia de requisitos para **validar e descobrir requisitos** por meio da simulação visual das interfaces e da interação do usuário com o software antes de sua construção. O principal valor da prototipação é o feedback gerado pelo cliente. O ciclo de prototipação envolve obter os requisitos, planejar, construir e validar o protótipo com o usuário, refinando os requisitos em um processo que pode se repetir várias vezes.

Existem diferentes tipos de prototipação:

- **Baixa Fidelidade vs. Alta Fidelidade:** A baixa fidelidade é mais simples e barata, usada no início do projeto para avaliar funcionalidades. A alta fidelidade é mais detalhada, se assemelha ao produto final e é útil para descobrir requisitos de usabilidade, mas pode gerar falsas expectativas no usuário.
- **Horizontal vs. Vertical:** A prototipação horizontal oferece uma visão ampla de várias funcionalidades sem aprofundamento. A vertical detalha funcionalidades específicas, geralmente as mais complexas ou menos compreendidas.
- **Descartável vs. Evolutiva:** A descartável é usada para validar e refinar requisitos e depois é descartada. A evolutiva é refinada ao longo de várias versões até se tornar o sistema final.

É crucial que o analista de requisitos explique ao cliente que o protótipo é uma ferramenta de simulação, e seu desempenho não reflete o do produto final.

Tema 2 – Análise de Requisitos

A análise de requisitos é o processo de **organizar e completar as informações** coletadas na elicitação, com o objetivo de identificar ambiguidades, conflitos, erros e incompletudes. Esta fase é crucial para evitar retrabalho e garantir que a solução

atenda às necessidades primárias do cliente. As tarefas do processo de análise incluem:

- **Checagem da Necessidade:** Garantir que cada requisito contribua para os objetivos do negócio.
- **Checagem de Consistência e Completude:** Verificar se não há contradições (consistência) ou omissões (completude) entre os requisitos.
- **Checagem de Viabilidade:** Assegurar que os requisitos podem ser implementados dentro do orçamento e do tempo disponíveis.

Após a análise, os stakeholders e desenvolvedores negociam as prioridades para criar um plano de projeto realista, resultando em um documento de especificação que representa um acordo contratual.

Tema 3 – Fluxos Operacionais

Este tema aborda como o software em desenvolvimento pode **alterar os fluxos operacionais existentes** em uma organização. A análise de requisitos deve converter requisitos abstratos em tarefas concretas e processáveis pelo software. As tarefas identificadas, que já são de conhecimento dos usuários, devem ser avaliadas para decidir quais serão incorporadas, parcial ou totalmente, pelo novo sistema. O software pode substituir tarefas humanas, trazendo benefícios como agilidade e melhorias, mas é importante notar que nem todas as tarefas podem ser digitalizadas.

Tema 4 – Modelos para Refinamento

A modelagem é o processo de criar esboços de uma solução para **representar o que o cliente solicitou** e fornecer uma base para o projeto do software. O foco da modelagem de requisitos é "o quê" (funções, interações, restrições) e não "como" (implementação). Os princípios de modelagem incluem representar o domínio da informação, definir as funções, descrever o comportamento do software e dividir os modelos em camadas para simplificar a complexidade.

A escolha do modelo depende do público; alguns usuários preferem diagramas, outros protótipos navegáveis ou descrições textuais. Os principais tipos de modelagem são:

- **Baseada em Cenários:** Utiliza a **UML (Linguagem de Modelagem Unificada)** para criar diagramas de casos de uso, de atividades e de sequência, que descrevem a interação do usuário com o sistema.
- **Baseada em Classes:** Identifica classes de análise (objetos ou conjuntos de objetos) a partir de substantivos nos casos de uso, definindo seus atributos e operações.
- **Funcional:** Avalia as funcionalidades do sistema, frequentemente usando diagramas de atividades para funcionalidades complexas.
- **Comportamental:** Mostra como o software responde a eventos externos, utilizando diagramas de sequência e de estados para modelar as transições de estado dos objetos.

Tema 5 – Verificação e Validação de Requisitos

Verificação e validação são etapas para garantir a qualidade do documento de requisitos.

- **Verificação:** Avalia se há **falhas na especificação**, como inconsistências, ambiguidades e conformidade com os padrões da organização. Geralmente é feita por um membro da equipe que não seja o autor da especificação. A pergunta-chave é: "Temos os requisitos certos?".
- **Validação:** Confirma se a especificação **atende às necessidades do cliente** e exige a participação ativa dele. A pergunta-chave é: "Temos certos os requisitos?".

As técnicas mais comuns para verificação e validação incluem:

- **História do Usuário:** Descrição do que o sistema deve fazer para o usuário, comum em metodologias ágeis.
- **Modelagem de Processos:** Representação dos processos de negócio para facilitar a interpretação dos requisitos.
- **Geração de Casos de Testes:** Desenvolver testes baseados nos requisitos para comprovar sua eficiência e identificar problemas de implementação.
- **Listas de Verificação (Checklist):** Listas de perguntas para avaliar objetivamente se os requisitos estão adequados e completos.

Resumo do Arquivo: "Aula_5_Texto.pdf"

Este arquivo aborda a **gerência de requisitos**, focando em como administrar mudanças, planejar o projeto, garantir a rastreabilidade e priorizar os requisitos para assegurar o sucesso do software.

Tema 1 – Gerência de Requisitos

Gerência de requisitos é o processo de **controlar e documentar as mudanças nos requisitos** de um sistema ao longo do tempo. Mudanças são inevitáveis devido a fatores como a dificuldade dos stakeholders em expressar suas necessidades, mudanças nas prioridades do negócio ou novas regulamentações legais. As tarefas da gerência de requisitos incluem: manter os requisitos em um repositório, priorizá-los, rastreá-los para avaliar o impacto das mudanças e aprovar as alterações junto aos stakeholders.

Tema 2 – Plano de Gerenciamento

O plano de gerenciamento de requisitos detalha como a gerência será executada, geralmente criado pelo gerente de projetos com o apoio do analista de requisitos. Este plano faz parte de um conjunto de planos de projeto que visam garantir a qualidade do produto final. Os principais planos são:

- **Plano de Garantia de Qualidade:** Descreve as verificações e testes para garantir que o software funcione conforme especificado.
- **Plano de Gerência de Riscos:** Identifica potenciais problemas, quantifica suas probabilidades e planeja ações para minimizar seus impactos.
- **Plano de Gerência de Configuração:** Organiza como as alterações em requisitos e outros artefatos serão gerenciadas, controlando versões e o trabalho da equipe.
- **Plano de Mudanças:** Previne o impacto de alterações, por exemplo, através do uso de protótipos.
- **Plano de Testes:** Descreve como os testes serão conduzidos, incluindo estratégias, ferramentas e resultados esperados.

Tema 3 – Gestão de Mudanças de Requisitos

A gestão de mudanças é o processo formal de **avaliar todos os pedidos de alteração**, documentá-los e comunicar as decisões (aceitas ou rejeitadas) à equipe. É crucial que toda solicitação de mudança passe por uma análise de impacto no projeto, considerando como um requisito pode afetar outros, mesmo que não estejam diretamente relacionados. A aprovação das mudanças pelo cliente é fundamental para manter um consenso formal e evitar questionamentos futuros. A técnica de "controle de questões" pode ser usada para garantir que todas as dúvidas e conflitos gerados por uma mudança sejam resolvidos.

Tema 4 – Rastreabilidade de Requisitos

Rastreabilidade é o processo de **identificar e documentar os vínculos entre os requisitos**, sua origem e seus artefatos derivados (como código e testes). Ela é essencial para gerenciar o escopo, avaliar o impacto de mudanças e garantir que o sistema final faça apenas o que foi solicitado. Existem diferentes tipos de rastreabilidade:

- **Horizontal e Vertical:** A rastreabilidade horizontal estabelece dependências entre requisitos no mesmo nível. A vertical conecta requisitos de diferentes fases do ciclo de vida, da origem à implementação.
- **Pré e Pós-rastreabilidade:** A pré-rastreabilidade foca na origem dos requisitos antes de sua inclusão na especificação. A pós-rastreabilidade foca em como eles foram implementados.
- **Matriz de Rastreabilidade:** É uma ferramenta, geralmente uma tabela, que visualiza as dependências entre requisitos e outros artefatos, como casos de uso.

Tema 5 – Priorização

A priorização de requisitos é necessária porque nem todos os requisitos têm a mesma importância, e os recursos do projeto (tempo e custo) são limitados. O objetivo é decidir o que deve ser tratado primeiro, envolvendo negociações com os stakeholders para obter consenso. Algumas técnicas de priorização são:

- **Timeboxing/Budgeting:** Prioriza requisitos que podem ser implementados dentro de um período de tempo (timeboxing) ou orçamento (budgeting) fixos.

- **Votação:** Os stakeholders votam nos requisitos propostos, e os que recebem mais votos são priorizados.
- **Análise de MoSCoW:** Classifica os requisitos em quatro categorias: **M**ust have (obrigatório), **S**hould have (deveria ter), **C**ould have (poderia ter) e **W**on't have (não terá agora).

Resumo do Arquivo: "Aula_6_Texto.pdf"

Este arquivo introduz o desenvolvimento de software com **metodologias ágeis**, contrastando-o com abordagens tradicionais e detalhando como a elicitação e o gerenciamento de requisitos são adaptados a esse contexto dinâmico.

Tema 1 – Requisitos Ágeis

Os processos tradicionais de desenvolvimento (em cascata) são lentos para lidar com requisitos que mudam constantemente. As **metodologias ágeis** surgiram para produzir software de forma rápida e incremental, intercalando especificação, projeto e implementação em ciclos curtos. A filosofia ágil, expressa no **Manifesto Ágil**, valoriza "indivíduos e interações mais que processos e ferramentas" e "responder a mudanças mais que seguir um plano". O **Scrum** é um dos frameworks ágeis mais populares, conhecido por suas práticas como o *backlog do produto* (lista de tarefas), *reuniões diárias* e *Sprints* (ciclos de desenvolvimento de até quatro semanas).

Tema 2 – Características e Refinamento de Requisitos

Na abordagem ágil, a coleta de requisitos é colaborativa e utiliza ferramentas como:

- **Wireframes:** Esboços de interfaces para alinhar o entendimento.
- **Personas:** Personagens fictícios que representam os usuários para identificar suas necessidades.
- **Jogos Ágeis:** Atividades lúdicas para facilitar a colaboração e o planejamento.
- **Histórias de Usuário:** Descrições curtas e informais do ponto de vista do usuário. Histórias grandes ou relacionadas podem ser agrupadas em **Épicos**. Um conjunto de histórias que entrega uma funcionalidade maior é chamado de **Feature**. O backlog do produto deve ser constantemente refinado, questionando se os requisitos ainda são relevantes, prioritários e se agregam valor ao negócio.

Tema 3 – Canvas e Storyboard

O **Project Model Canvas (PMC)** é uma ferramenta de planejamento ágil que representa visualmente os componentes de um projeto em uma única "tela" (Canvas). Seus objetivos incluem engajar stakeholders, facilitar a colaboração e criar um plano de projeto realista. O **Storyboard** é outra ferramenta visual, semelhante a uma história em quadrinhos, que ajuda a planejar a narrativa, o design e as cenas de interação do usuário com o software.

Tema 4 – Backlog e Priorização

O **backlog do produto** é a lista de todas as tarefas pendentes do projeto, que muda constantemente com novos refinamentos e requisitos. Para gerenciar o fluxo de trabalho, é comum o uso do **Kanban**, um quadro visual com colunas que representam as etapas do processo (ex: "A Fazer", "Em Andamento", "Concluído"). Os cartões no quadro representam as tarefas.

Existem várias técnicas para priorizar as tarefas do backlog:

- **Valor de Negócio vs. Risco:** Itens de alto valor e alto risco são feitos primeiro; itens de baixo valor e alto risco são evitados.
- **Testes de Suposição:** Prioriza com base na validação de hipóteses e sua relevância para o usuário.
- **Business User Cost (BUC):** Analisa os benefícios para o negócio e o usuário em relação aos custos.
- **Scorecard:** Utiliza uma pontuação baseada em critérios ponderados para classificar as funcionalidades.
- **MoSCoW:** Classifica as tarefas como *Must-have, Should-have, Could-have* ou *Won't have*.

Tema 5 – Histórias do Usuário

As **histórias de usuário** são a principal forma de documentar requisitos em abordagens ágeis. Elas são descrições curtas e informais de uma funcionalidade sob a perspectiva do usuário, seguindo o formato: "COMO UM `<usuário>`, DESEJO `<necessidade>` PARA QUE `<objetivo>`". Uma boa história de usuário segue o conceito **INVEST** (Independente, Negociável, Valorosa, Estimável, Pequena e Testável). Embora simples, as histórias precisam ser complementadas com critérios de aceitação para detalhar o comportamento esperado do software. A equipe de

desenvolvimento as utiliza para estimar o esforço e planejar as tarefas a serem executadas.