**Engenharia de Requisitos**

Olá pessoal boa ..., hoje vamos dar continuidade à nossa disciplina de Engenharia de Software. Nas aulas passadas vimos os conceitos básicos de engenharia de software, os processos de software com os modelos de software (modelo em cascata, incremental, RUP), e também o desenvolvimento ágil de software. Na nossa aula de hoje vamos aprender sobre Requisitos e o processo de Engenharia de Requisitos, que é uma das áreas de estudo da Engenharia de Software.

Slide de apresentação

O objetivo dessa aula é que ao final possamos saber responder aos seguintes questionamentos:

Questionamentos Iniciais

**- O que são requisitos?** (conceito e características)

**- Como os requisitos são classificados?**

**- O que é Engenharia de Requisitos?**

**- Por que a Engenharia de Requisitos é uma das tarefas mais difíceis no processo de desenvolvimento de um software?**

**- Por que requisitos são importantes?**

Então antes de nos aprofundarmos nos métodos e processos da Engenharia de Requisitos, vamos entender alguns conceitos iniciais.

Conceito de Requisito de Software

“Um requisito é uma característica do sistema ou a descrição de algo que o sistema é capaz de realizar, para atingir os seus objetivos”. Shari Lawrence

“Os requisitos de um sistema são descrições do que o sistema deve fazer, os serviços que oferece e as restrições a seu funcionamento. Esses requisitos refletem as necessidades dos clientes para um sistema que serve a uma finalidade determinada”. Sommervile

“Um requisito de software é uma descrição dos principais recursos de um produto de software, seu fluxo de informações, comportamento e atributos. Em suma, um requisito de software fornece uma estrutura básica para o desenvolvimento de um produto de software” James F. Peters / Witold Pedryez

Em resumo:

- Define o que o sistema deve fazer. (Funcionalidade)

Exemplos:

Software de clínica médica:

O sistema deve permitir o agendamento de consultas dos pacientes de acordo com a disponibilidade da agenda dos médicos.

Software de um Ponto de Venda (PDV):

O sistema deve emitir a NFC-e (nota fiscal do Consumidor eletrônica) de cada venda realizada.

- Define restrições ao funcionamento do sistema:

Exemplo:

O sistema PDV deve funcionar mesmo sem conexão com a internet.

Uma possível solução: o PDV possuir um banco de dados individual, que depois é sincronizam com o banco de dados online na nuvem para centralização todas as vendas.

O sistema deve suportar, pelo menos, 20 transações por segundo.

Infraestrutura

O sistema deve estar disponível, pelo menos, durante 10 horas por dia.

Mecanismos de redundância.

Requisitos Funcionais x Não-Funcionais

“**Requisitos funcionais**: São declarações de serviços que o sistema deve fornecer, de como o sistema deve reagir a entradas especificas e de como o sistema deve se comportar em determinadas situações.” Sommervile

* Dependem do tipo de software
* Dependem dos seus possíveis usuários

Ex.: O sistema deve gerar a cada dia, para cada clínica, a lista dos pacientes para as consultas daquele dia.

“**Requisitos não funcionais:** são requisitos que não estão diretamente relacionados com os serviços específicos oferecidos pelo sistema a seus usuários. ”

“São restrições aos serviços ou funções oferecidas pelo sistema. Incluem restrições de timing, restrições no processo de desenvolvimento e restrições impostas por normas. Muitas vezes, aplicam-se ao sistema como um todo. ” Sommervile

Os requisitos não funcionais surgem por meio das necessidades dos usuários, devido a restrições de orçamento, politicas organizacionais, necessidades de interoperabilidade com outros sistemas de software ou hardware, necessidades de desempenho ou a partir de fatores externos, como regulamentos de segurança ou legislações de privacidade.

**Classificação dos RNF**:

* **Requisitos de produto**: restringem o comportamento do software. Ex.: requisitos de desempenho quanto à rapidez com que o sistema deve executar e quanta memória ele requer, requisitos de confiabilidade que estabelecem a taxa aceitável de falhas, requisitos de usabilidades.
  + Requisitos de Usabilidade
  + Requisito de eficiência
    - Requisitos de desempenho
    - Requisito de espaço
  + Requisitos de confiança
  + Requisitos de proteção
* **Requisitos organizacionais**: são derivados das políticas e procedimentos organizacionais. Exemplos incluem os requisitos do processo operacional, que definem como o sistema será usado, os requisitos do processo de desenvolvimento que especificam a linguagem de programação, o ambiente de desenvolvimento ou normas de processo a serem usadas, bem como os requisitos ambientais que especificam o ambiente operacional do sistema.

Usar determinada linguagem de programação

Seguir um padrão de codificação

Seguir um processo de desenvolvimento

Seguir uma determinada politica organizacional

* + Requisitos ambientais
  + Requisitos operacionais
  + Requisitos de desenvolvimento
* **Requisitos externos**: abrange todos os requisitos que derivam de fatores externos ao sistema e seu processo de desenvolvimento. Podem incluir requisitos reguladores, que definem o que deve ser feito para que o sistema seja aprovado para uso, por um regulador; Requisitos legais, que devem ser seguidos para garantir que o sistema opere dentro da lei; e requisitos éticos, que asseguram que o sistema será aceitável para seus usuários e o público em geral;
  + Requisitos reguladores
  + Requisitos éticos
  + Requisitos legais
    - Requisitos contábeis
    - Requisitos de segurança/proteção

Exemplo de legais: O sistema deve estar de acordo com as exigências do marco civil da internet.

Requisitos não funcionais são frequentemente mais críticos que requisitos funcionais individuais.

Os usuários do sistema podem, geralmente, encontrar maneiras de contornar uma função do sistema que geralmente não atenda a suas necessidades. No entanto, deixar de atender a um requisito não funcional pode significar a inutilização de todo o sistema.

Ex.:

Se um sistema de aeronaves não comprimir seus requisitos de confiabilidade, não será certificado como um sistema seguro para operar;

Observações sobre RNF:

* Requisitos não funcionais podem afetar a arquitetura geral de um sistema em vez de apenas componentes individuais.

Ex.: para assegurar que sejam cumpridos os requisitos de desempenho, será necessário organizar o sistema para minimizar a comunicação entre os componentes.

* Um único requisito não funcional pode gerar uma série de requisitos funcionais relacionados que definam serviços necessários no sistema.

Sommervile diz que as vezes em um ambiente pratico a distinção entre diferentes tipos de requisitos não é tão clara.

Ex.: requisito não funcional: O sistema deve estar de acordo com o manual de saúde.

Pegar requisito funcional manual de saúde.

* Requisitos Não Funcionais podem ser extremamente difíceis de especificar precisamente.

Ex.: O sistema deve ser de fácil uso pelo pessoal médico e deve ser organizado de tal maneira que os erros dos usuários sejam minimizados.

Essa definição de requisito está muito genérica

Deixa margem para várias interpretações (o que significa “fácil uso”)

Não é “testável”

Uma melhor maneira para definir:

“A equipe médica deve ser capaz de usar todas as funções do sistema após quatro horas de treinamento. Após esse treinamento, o número médio de erros cometidos por usuários experientes não deve exceder dois por hora de uso do sistema. ”

Agora podemos testar.

Temos um requisito escrito quantitativamente que usa uma medida que pode ser testada.

A equipe de desenvolvimento terá que usar técnicas de usabilidade e UX para que o sistema possa ser fácil de ser utilizado e fácil de ser entendido. Deve utilizar padrões de telas, para que ao aprender usar uma tela do sistema, já possa estar de certa forma aprendendo utilizar as outras.

Slide com métricas pra testes.

Na pratica, os clientes geralmente consideram difícil traduzir suas metas em requisitos mensuráveis. Eles não têm experiência para definir os requisitos quantitativamente.

E o custo para testar os requisitos não funcionais pode ser muito elevado.

* Requisitos Não Funcionais frequentemente conflitam entre si!

Atender uma característica não funcional pode significar abrir mão de outra.

Por exemplo:

portabilidade x desempenho

O sistema deve rodar em diversas plataformas pode levar ao uso de tecnologias que possuem perca de desempenho em relação a outras tecnologias que são feitas pra plataformas especificas.

O cliente que quer um aplicativo feito em Ionic pra atender em várias plataformas vai perder em desempenho comparado a soluções especificas para cada plataforma.

usabilidade x manutenibilidade

Uma interface de usuário mais rica (amigável) pode tornar o sistema mais difícil de manter, pois existem mais detalhes para ser implementados.

Os requisitos podem servir a três propósitos:

- Informam aos projetistas que funcionalidades e características o sistema resultante deve ter.

- Permitem aos desenvolvedores explicar seu entendimento de como os clientes querem que o sistema funcione.

- Informam à equipe de testes o que demonstrar para convencer os clientes de que o sistema está sendo entregue de acordo com o que foi solicitado.

Ok. Mas vamos para um pouco pra pensar:

Será se os cliente, usuários, engenheiro de software e desenvolvedores vêm os requisitos da mesma maneira?

Requisitos de Usuário x Requisitos de Sistema

“Requisito é uma declaração abstrata de alto nível de um serviço que o sistema deve fornecer ou uma restrição do sistema.” Sommervile

“Requisito é uma definição formal e detalhada de uma função do sistema” Sommervile

“Os requisitos são, primeiro, escritos de tal maneira que nós e nossos clientes possamos chegar a um acordo sobre o que o sistema deverá fazer. Em seguida, os requisitos são reescritos, geralmente em uma representação mais matemática, de tal maneira que os projetistas possam transformar os requisitos em um bom projeto de sistemas”. Shari Lawrence

Para assegurar que a equipe do projeto, clientes e usuários entendemos e utilizamos os requisitos de maneira apropriada, é importante que os requisitos sejam de alta qualidade. Para isso verificamos se eles têm as seguintes características:

Características dos Requisitos

- Os requisitos estão **corretos**? Deve ser feita uma análise para garantir que eles foram definidos sem erros.

- Os requisitos são **consistentes**? Será que não existe requisitos ambíguos ou conflitantes? Dois requisitos são inconsistentes, se for impossível satisfazer aos dois simultaneamente.

Por exemplo, se um requisito diz que apenas 10 usuários podem usar o sistema ao mesmo tempo, e outro diz que, em uma certa ocasião, pode haver 20 usuários simultaneamente, esses requisitos são inconsistentes.

- Os requisitos estão **completos**?

Tudo que é requerido pelo usuário deve ser definido.

Muitas vezes, para sistemas grandes e complexos, é praticamente impossível alcançar completude e consistência dos requisitos. Uma razão pra isso é que ao elaborar especificações para sistemas complexos é fácil cometer erros e omissões.

Outra razão é que em um sistema de grande porte existem muitos stakeholders. Um stakeholder é uma pessoa ou papel que, de alguma maneira, é afetado pelo sistema. Os stkeholders têm necessidades diferentes e, muitas vezes, inconsistentes. Essas inconsistências podem não ser evdentes em um primeiro momento, quando os requisitos são escritos, porém problemas podem surgir após uma análise mais profunda.

“Talvez o maior problema que enfrentamos no desenvolvimento de sistemas de software grandes e complexos seja o da engenharia de requisitos” Sommerville.

“Entender os requisitos de um problema está entre as tarefas mais difíceis enfrentadas por um engenheiro de software” Pressman

A primeira vista, a engenharia de requisitos não parece ser assim tão difícil. Afinal de contas, o cliente não sabe o que é necessário? Os usuários finais não deveriam ter um bom entendimento das características e funções que trarão benefícios? Surpreendentemente, em muitos casos, ou na maioria dos casos, a resposta é NÃO. E mesmo se os clientes e usuários finais fossem explícitos quanto às suas necessidades, elas mudariam ao longo do projeto. Pressman

Papeis

Falar da diferença de Clientes e Usuários finais.

“ Clientes ou usuários finais podem estar situados em uma cidade ou país diferente, podem ter apenas uma vaga ideia daquilo que é necessário, podem ter opiniões conflitantes sobre o sistema a ser construído, podem ter conhecimento técnico limitado ou quem sabe pouco tempo para interagir com o engenheiro que está fazendo o levantamento dos requisitos” Pressman

Nenhuma dessas situações são desejáveis, contudo, todas são relativamente comum e muitas vezes você é forçado a trabalhar de acordo com as limitações impostas pela situação.

Definição de Engenharia de Requisitos

“O processo de descobrir, analisar, documentar e verificar esses serviços e restrições é chamado engenharia de requisitos.” Sommervile

“Engenharia de Requisitos: Está relacionada com a definição do que o sistema deve fazer, suas propriedades emergentes desejáveis e essenciais e as restrições quanto à operação do sistema e quanto aos processos de desenvolvimento de software. ” Sommerville

“O amplo espectro de tarefas e técnicas que levam a um entendimento dos requisitos é denominado engenharia de requisitos” Pressman

“A área de conhecimento de Requisitos de Software está preocupada com a elicitação, análise, especificação, e validação de requisitos de software bem como a gestão dos requisitos durante todo o ciclo de vida do produto de software. ”

SWEBOK - Software Engineering Body of Knowledge

SWEBOK é um guia com melhores práticas de engenharia de software consolidadas, tanto na academia, quanto na indústria.

“A Engenharia de Requisitos fornece o mecanismo apropriado para entender aquilo que o cliente deseja, analisando as necessidades, avaliado a viabilidade, negociando uma solução razoável, especificando a solução sem ambiguidades, validando a especificação e gerenciando as necessidades à medida que são transformadas em um sistema operacional. ” Pressman

“Você pode pensar na engenharia de requisitos como o processo de comunicação entre os clientes e os usuários de software e os desenvolvedores de software” Sommerville

Relação de Engenharia de Software e Engenharia de Requisitos

Mostrar a engenharia de requisitos com uma das áreas do SWEBOK.

Requisito de Software

Plano de Aula:

Colocar objetivos parecidos com os do livro do Sommerville.