Universidade Federal do Ceará
Campus de Crateús
Sistemas de Informação & Ciência da Computação

ENGENHARIA DE REQUISITOS [CRT0393]
TRABALHO EM DUPLA #1

INCLUA ABAIXO DE CADA QUESTÃO A RESPECTIVA RESPOSTA
ENVIAR O DOCUMENTO EM FORMATO PDF

Prof. Allysson Allex Araújo
allysson.araujo@crateus.ufc.br
http://crateus.ufc.br/allysson

Francisco Hartur Lopes de Alcântara

1) Cite exemplos de como a Engenharia de Requisitos está presente em cada uma das cinco etapas de um processo geral de engenharia, como apresentado na Figura abaixo:

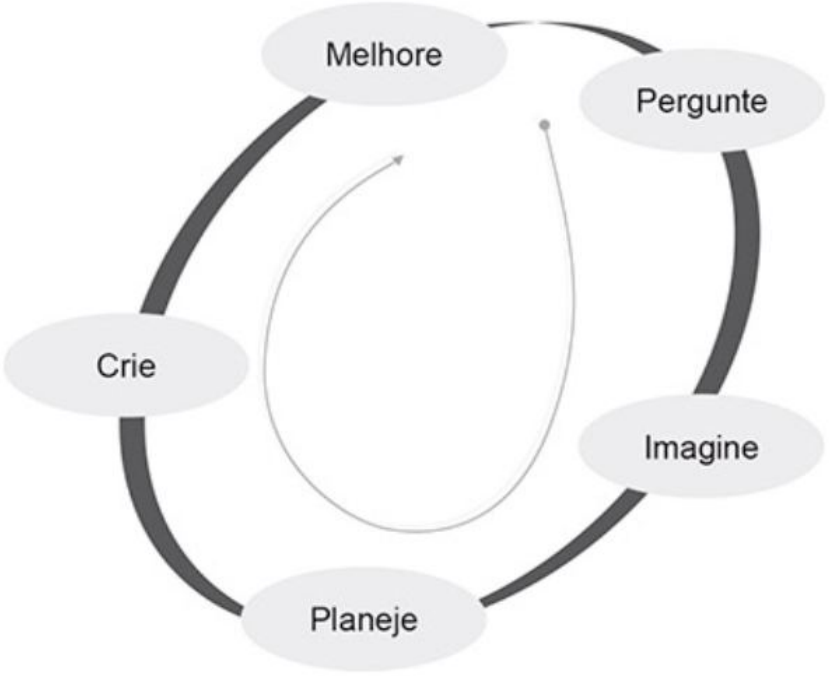

Figura 1.1. Um processo geral de engenharia explicado de forma simples para jovens.

A primeira etapa do processo é “pergunte” e busca identificar qual o problema, o que outros fizeram no sentido de resolvê-lo e quais as restrições que se aplicam. Em seguida, “imagine” quais são algumas soluções, pense em alternativas, escolha a melhor solução. Então, “planeje” desenhando um diagrama e preparando uma lista do que precisa; “crie” seguindo seu plano e teste os resultados. Por fim, “melhore” discutindo o que funciona, o que não funciona e o que poderia ser melhor, modifique o seu projeto para melhorá-lo e testá-lo novamente.

A Engenharia de Requisitos está completamente alinhada a esse processo geral. Em um primeiro momento, pode-se pensar que ela se restringe apenas às primeiras etapas presentes no processo; contudo, isso se revela falso quando se explora melhor um dos principais benefícios da Engenharia de Requisitos: habilitar o entendimento – de forma contínua – das necessidades do cliente para entregar uma solução que atenda aos seus objetivos de negócio, que são dinâmicos e mutáveis.

Exemplos:

Pergunte: é onde se concentram boa parte das atividades de elicitação (ou levantamento) de requisitos. O objetivo é ganhar conhecimento sobre os problemas para poder pensar melhor sobre uma solução.

Imagine: isto pode ser aplicado tanto a algumas atividades de preparação de elicitação (ao imaginar soluções novas, questões surgirão), quanto nas atividades de análise, onde se podem esboçar cenários de soluções para o problema.

Planeje: no planejamento do projeto uma especificação de ao menos parte dos requisitos deve ser concluída para que se possa iniciar a execução. Neste aspecto pode-se pensar também no próprio planejamento das atividades da engenharia de requisitos.

Crie: durante a execução do projeto mudanças podem surgir, e daí exerce-se a gestão de requisitos.

Melhore: ao validar a solução (ou a especificação) com o cliente, será possível perceber diversas oportunidades de melhoria para a solução proposta, Isto pode provocar um novo projeto ou solicitação de mudança sobre o projeto atual.

2) Explique como o exercício da disciplina de Análise e Projeto e da disciplina de Implementação interagem com o exercício da disciplina de Requisitos na iniciação de um projeto de software. Procure fazer isso relacionando decisões

sobre um assunto que influenciam a condução do outro e dando exemplos práticos.

Para o desenvolvimento de um software é necessário seguir algumas fases. Essas fases podem se diferenciar em projetos ou empresas distintas, mas em todos é comum ter as atividades de Levantamento de requisitos, Análise de Requisitos; Projeto e Implementação. Em requisitos, mais necessariamente no levantamento de requisitos é o local onde é compreendido o problema, levando e priorizando as necessidades do cliente. Na Análise de Requisitos (especificação) é quando vai ocorrer a especificação dos requisitos, ou seja, descrevê-los e documentá-los, baseado nas necessidades levantadas do cliente e definir o que o sistema deve fazer, antes de definir como o sistema irá fazer. Todos estes passos estão contidos na Engenharia de Requisitos. Na disciplina de Projeto será definido o detalhamento técnico, como linguagem de programação utilizada, arquitetura do sistema. Nessa fase também é comum utilizar os diagramas de classes, diagramas de sequência e casos de uso. (casos de uso também podem ser usados na fase de especificação) para descrever o comportamento do sistema e como ele irá se comportar. Já na parte de implementação é onde vai acontecer o desenvolvimento do produto, baseado na documentação feita na parte de especificação, utilizando as informações da fase de Projeto. Muitas empresas optam por pular as fases iniciais da engenharia de software, onde acontece o levantamento e especificação de requisitos, o qual é um erro, pois apesar do produto estar bem implementado, pode acontecer de que o que foi desenvolvido não satisfazer as necessidades do cliente, mesmo que esteja bem implementado. Além disso, solicitações de usuários, como a necessidade de controle de acesso, podem já estar atendidas por algum componente da arquitetura corporativa e que devem ser abordadas por tarefas da disciplina de análise e projeto de maneira simultânea com a condução de atividades de requisitos. Desse modo, qualquer fase da construção de um software, tem como base a Engenharia de Requisitos.

3) O que diferencia a estratégia sequencial e iterativa-incremental do ponto de vista da Engenharia de Requisitos?

Na estratégia sequencial se busca esgotar o trabalho de requisitos em uma única fase do projeto. Na interativa-incremental o escopo vai sendo abordado parcialmente ao longo das iterações.

A primeira estratégia tem mais dificuldade em lidar com as mudanças de requisitos. A segunda, quando trabalha com ciclos de entrega curtos, têm mais facilidade.

4) Cite três fatores que podem influenciar o nível de detalhe da especificação de requisitos.

- Desenvolvimento interno ou externo
  Quando o projeto é construído dentro da própria empresa, não é necessário um nível de detalhamento tão alto pois a equipe de desenvolvimento e os clientes são colegas e compartilham de um mesmo interesse, além de estarem no mesmo espaço físico, facilitando o encontro. No desenvolvimento externo a equipe de desenvolvimento e os clientes são de diferentes empresas, com interesses distintos. Além de estarem em diferentes espaços físicos necessitando de mais detalhamento dos requisitos para minimizar problemas de comunicação.

- Conhecimento da equipe no domínio
  Quando a equipe tem experiência em outros projetos da mesma área, dispensa um alto nível de detalhamento, pois como a equipe tem experiência, alguns detalhes são desnecessários especificar. Mas deve haver o cuidado da equipe não pensar que tem toda a solução de que o cliente precisa e não ouvir suas necessidades.
  No desenvolvimento interno, a equipe acumula conhecimento na área, pois desenvolve vários projetos da área.
  No desenvolvimento externo, empresas que atuam exclusivamente com projetos em áreas de negócio específicas, como bancos. Mantendo a equipe com experiência na área.

- Expectativa de rotatividade de mão de obra
  Quanto mais rotatividade ocorrer, mais detalhamento precisará existir, pois quando uma pessoa sai da equipe, existe perda de conhecimento. Então o projeto deve ser bem detalhado para evitar se perca informações com a saída da pessoa. O alto nível de detalhes ajudará a próxima pessoa que realizar o serviço.

5) Ao se trabalhar com metodologias ágeis, a especificação de requisitos torna-se dispensável?

Não. Continua sendo necessário às necessidades dos usuários, documentá-las de alguma forma e manter esta documentação atualizada ao longo do projeto. O que varia são as técnicas usadas e artefatos produzidos nas metodologias ágeis em comparação a metodologias tradicionais. Em geral a documentação de requisitos é menos detalhada, pois se busca no contexto ágil a participação mais intensa do cliente no desenvolvimento do projeto. Como exemplo, no SCRIM, uma das metodologias mais usadas por equipes ágeis, a especificação de requisitos frequentemente ocorre com histórias de usuário, que são especificações mais simples que casos de uso.

6) De acordo com a IEEE 830, os critérios de qualidade mais comuns para uma boa especificação de requisitos são:

- **Correta;**
- **Completa;**
- **Clara;**
- **Consistente;**
- **Modificável;**
- **Priorizada;**
- **Verificável;**
- **Rastreável.**

Dito isso, explique em detalhes cada um dos critérios acima e sugira uma forma de como garantir seu cumprimento.

- Correta - A especificação de requisitos está correta quando ela satisfaz as necessidades do projeto. Um requisito presente na especificação que não tem relação com nenhuma necessidade de negócio é um requisito incorreto. Uma forma para garantir os requisitos corretos é se manter dentro do escopo do projeto.

- Completa - Todos os elementos do domínio do problema devem ser descritos. Não deve existir partes na especificação com termo “a definir”, pois é incompleta, salvo os casos que a especificação ainda se encontra na fase de elaboração. Deve manter a comunicação com o cliente para acertar todos os detalhes e não deixar que os requisitos passem da fase de elaboração com status “a definir”.

- Clara - Uma especificação clara é quando não existe ambiguidade, ou duplo sentido. Possuindo uma única interpretação. Alguns termos podem ser usados para diferentes significados, é importante que exista um glossário definindo os termos e seus significados para evitar o duplo sentido. E ao especificar, pensa como cada envolvido no projeto iria entender a especificação. Outras formas de deixar a linguagem mais clara é o uso de modelos e diagramas para compor sua especificação. Assumir que o leitor NÃO tenha conhecimento sobre o domínio.

- Consistente - Quando não possui contradições entre suas partes. Acontecem principalmente em requisitos. Exemplo: Um requisito tem o objetivo de manter o histórico do cliente ao fazer uma pesquisa, é outro requisito diz para apagar o histórico do cliente. Além de que inconsistências podem ocorrer ao solicitar mudanças na especificação de requisitos. Quando existem mais de uma pessoa especificando os requisitos, aumenta a chance de inconsistência.

- Modificável - Quando as modificações podem ser feitas de maneira fácil, sem comprometer a estrutura da especificação. Minimizando os erros, quando existem mudanças malfeitas. Para facilitar a especificação ser

modificável é importante que não exista redundância, expressar cada requisito de forma individual e que haja a rastreabilidade entre os requisitos.

- Priorizada - cada requisito recebe um valor de importância, com base em alguns critérios, como complexidade, risco e valor para o negócio. Assim a equipe mantém esforços nos requisitos mais críticos do projeto. É importante conversar com a parte interessada para que seja definida a prioridade para cada requisito.

- Verificável - Quando existe um método demonstrar que a solução satisfaz cada requisito especificado. Ou seja, é necessário ter uma métrica para cada requisito. Exemplo: A interface do sistema deve ser amigável. Isso não é rastreável pois não dá para medir, o conceito de amigável é subjetivo para cada pessoa. Para ser verificável precisaria ser “De cada 10 usuários, 9 conseguem utilizar o sistema sem dificuldade.

- Rastreável - Se trata da relação entre requisitos, suas origens e produtos. Consistindo na identificação dos requisitos. Dessa forma a rastreabilidade ajuda a manter a entradas e saídas dos requisitos e que partes do projeto eles irão afetar. Quanto mais requisitos relacionados a outros, mais riscos possuem. É importante que cada requisito receba um código único de identificação, e que as outras partes do projeto, na sua especificação, possam mostrar que o requisito #0000 está associada a ela, mantendo a rastreabilidade.

7) A falha em não detalhar de maneira suficiente as informações na especificação de requisitos pode levar a interpretações equivocadas ou à criação de um número elevado de premissas. Nesse sentido, discorra sobre cada um dos fatores que podem influenciar no nível de detalhe adequado para um projeto de software.

A falha em não detalhar de maneira suficiente as informações na especificação de requisitos pode levar a interpretações equivocadas ou à criação de um número elevado de premissas. Por outro lado, decidir por detalhar demais a especificação de requisitos pode ser um elemento paralisante do projeto (a especificação nunca é finalizada), além de oneroso. O desafio é encontrar o equilíbrio do nível de detalhe adequado da especificação de requisitos para o seu projeto.

A discussão sobre o nível de detalhe na especificação de requisitos deve ser precedida pelo significado que se pretende dar ao termo. Os níveis de detalhe podem ser subdivididos conforme três objetivos diferentes:

- Delimitar o escopo preliminar e definir o escopo final da solução.

- Descrever o funcionamento e as restrições de um item no escopo.

- Mapear os requisitos para design ou implementação.

O nível de detalhe em que o escopo está especificado pode ser descrito em uma escala onde se destacam três pontos, conforme o momento em que o projeto se encontra. No primeiro momento, a especificação de requisitos define o escopo relacionando os processos de negócio impactados pela solução e o seu posicionamento no ambiente, indicando o que será externo a ela. Em geral inclui uma área cinzenta porque várias decisões sobre o escopo ainda devem ser resolvidas.

Em um segundo momento, conforme as decisões sobre o escopo são tomadas, as especificações evoluem, assim relacionando não apenas os processos de mais alto nível, como também tarefas do usuário que devem ser informatizadas.

Por fim, em um terceiro momento, a especificação em sua última versão descreve todo o escopo no nível de detalhe das tarefas do usuário. Todas as decisões sobre o escopo estão tomadas e sabe-se com precisão quais atividades do processo serão incluídas como parte da solução.

8) Leia o artigo "A Utilização de Histórias de Usuários no Levantamento de Requisitos Ágeis" e faça uma análise interpretativa de no mínimo 15 linhas (Arial, 12).

Na década de 90 foi introduzido os casos de uso, onde a ideia era descrever os requisitos funcionais de um sistema de forma simples e principalmente informal. Porém havia certas dificuldades e relutância de alguns no uso dos Casos de Uso, pois havia dúvidas se essa escrita informal estaria sendo executada de maneira correta e se não seria incompleta para descrição dos requisitos. Para tornar os casos de uso mais "rigorosos", foram introduzidas regras de relacionamento, notações e artefatos. Nos anos 2000 as empresas buscavam soluções mais práticas, assim surgiu o manifesto ágil, com métodos informais e pouco documentados, mas com destaque na comunicação verbal e social, também nos anos 2000 surgiu surgiu a ideia de expressar requisitos de software na forma de histórias de usuário (user stories - US), descrições curtas a partir da perspectiva de um usuário. No processo ágil as US são utilizadas para o levantamento e especificação dos requisitos. O objetivo é fornecer à equipe uma capacidade para responder rapidamente o que o usuário quer e precisa. As histórias são escritas em um cartão, que contém 3 perguntas básicas. Quem?, o que? e Por que?. Exemplo. Como aluno, eu quero poder pesquisar as disciplinas ofertadas no semestre no

sistema, para que eu possa planejar meu horário. Além de colocar uma identificação, título, estimativa e a importância da US. A principal diferença entre os casos de uso e as US, é os casos de Uso tem o objetivo de descrever o conjunto de interações e eventos entre os usuários ou sistemas externos, eles contém informações específicas sobre os passos que o usuário deve seguir para usar com sucesso a nova funcionalidade do sistema. Já as US têm o objetivo de capturar a perspectiva que o usuário tem sobre o sistema, descrevendo suas necessidades. Cada um contém suas particularidades, cada método pode atender a diferentes necessidades de levantamento e especificação de requisitos de uma empresa. Dito isto, não há um método certo ou mais útil que o outro. Inclusive, os Casos de Uso e as Histórias de Usuário se complementam, de forma que as US são eficazes para a captura e interação, e os casos de uso são necessários para a implementação quando a documentação formal é necessária.

9) Leia os artigos "Uma Análise Crítica dos Desafios para Engenharia de Requisitos em Manutenção de Software" e "Desafios da Engenharia de Requisitos Ágeis Centrada no Usuário" e, dessa forma, faça uma análise interpretativa relacionando-os de no mínimo 15 linhas (Arial, 12).

Ao ler os dois artigos vemos que o primeiro artigo ("Desafios da Engenharia de Requisitos Ágeis Centrada no Usuário") possui um objetivo claro que é identificar os desafios mais importantes na indústria e na academia referentes a Engenharia de Requisitos ágil integrada à UCD. Enquanto o segundo artigo ("Uma Análise Crítica dos Desafios para Engenharia de Requisitos em Manutenção de Software") tem como objetivo apresentar uma análise crítica da influência das principais dificuldades encontradas na manutenção de sistemas legados sobre os processos da Engenharia de Requisitos. No primeiro artigo percebemos uma dificuldade inerente às organizações referente ao incentivo à colaboração entre as equipes de UCD e Ágeis. Por mais que o desenvolvimento híbrido torne o desenvolvimento mais centrado no usuário, ainda temos problemas com o envolvimento do usuário e do cliente, principalmente com a inclusão do usuário final no processo, e ainda existe a dificuldade em como utilizar o conhecimento adquirido através das técnicas na elicitação dos requisitos. Desse modo, por mais que os métodos ágeis possam ser combinados com UCD, e essa integração seja benéfica e facilite o envolvimento do usuário no processo, ainda não está claro como incorporar de forma eficiente o usuário e as práticas de UCD no ciclo de vida de desenvolvimento de software ágil, principalmente na Engenharia de Requisitos ágil. No segundo artigo temos que a análise realizada por eles demonstrou a criticidade de algumas das atividades da Engenharia de Requisitos, quando realizadas em um contexto tão delicado e importante como a manutenção de software. Identificou-se também uma certa dificuldade no uso dos modelos e processos de Engenharia de Requisitos quando analisados sob a ótica do mantenedor de software e de suas necessidades, principalmente no que se refere aos aspectos de rastreamento de requisitos. Portanto, ao tentarmos estabelecer uma relação desses dois artigos podemos ver

que a compreensão das necessidades dos stakeholders é fundamental para qualquer tipo de projeto de software, isso fica evidente nos dois artigos. No primeiro, por mais que os métodos ágeis possam ser combinados com UCD, e essa integração seja benéfica e facilite o envolvimento do usuário no processo, ainda não está claro como incorporar de forma eficiente o usuário e as práticas de UCD no ciclo de vida de desenvolvimento de software ágil. No segundo, a manutenção de software é considerada uma fase de grande importância em engenharia de software. Esta fase ocupa a maior parte do ciclo de vida do software, sendo fundamental para garantir a qualidade geral do produto à medida que as necessidades de seus usuários e do ambiente de negócios evoluem.