**Engenharia de Requisitos**

O ciclo de vida de um software, entre outras características, está relacionado aos estágios de concepção, projeto, criação e implementação. São quatro atividades fundamentais do processo de software:

* Especificação de Software: são definidas as funcionalidades do software e restrições para sua operação.
* Projeto e Implementação de Software: o software que atenda à especificação deve ser produzido.
* Validação de Software: o software deve ser avaliado para garantir que ele faça o que o cliente deseja.
* Evolução do Software: o software evolui para atender às necessidades de mudança do cliente.

“Processos de engenharia de requisitos podem incluir quatro atividades de alto nível: **Estudo de viabilidade; Elicitação e análise; Especificação e Validação**”. **Após um estudo inicial de viabilidade**, o**próximo estágio** do processo de engenharia de requisitos é a **elicitação** e análise de requisitos. Nessa atividade, os engenheiros de software trabalham com clientes e usuários finais do sistema para obter informações sobre o domínio da aplicação, os serviços que o sistema deve oferecer, o desempenho do sistema, restrições de hardware e assim por diante."

Em uma visão espiral do processo de engenharia de requisitos, a partir do início até sua conclusão, a especificação de requisitos ocorre na seguinte ordem:

* especificação de requisitos de negócio;
* especificação de requisitos usuário;
* especificação de requisitos sistema.

**Requisitos não funcionais:** são restrições aos serviços ou funções oferecidos pelo sistema. Incluem restrições de timing, restrições no processo de desenvolvimento e restrições impostas pelas normas. São requisitos que não estão diretamente relacionados com os serviços oferecidos pelo sistema aos seus usuários. Eles podem estar relacionados às propriedades emergentes do sistema, como confiabilidade, tempo de resposta e ocupação de área.

**Os requisitos de domínio** ***são derivados do domínio da aplicação do sistema que podem ser novos requisitos funcionais em si, podem restringir os requisitos funcionais existentes ou estabelecer como devem ser executados cálculos específicos***. Muitas vezes esses requisitos refletem fundamentos do domínio da aplicação (SOMMERVILLE, 2008). Sem uma compreensão satisfatória desses requisitos pode ser impossível fazer o sistema operar de forma satisfatória (PRESSMAN, 2006). Quando os requisitos derivam do negócio ou da área de aplicação, temos os requisitos de domínio, que podem ser funcionais ou não-funcionais.

Esses requisitos são expressos com o uso de uma linguagem específica do domínio da aplicação e, geralmente, de difícil compreensão para os engenheiros de software. Os especialistas do domínio podem, por julgarem óbvio, deixarem de fornecer informações importantes e como resultado o requisito pode não ser implementado de forma satisfatória (SOMMERVILLE, 2008).

**Métricas para especificar requisitos não funcionais:**

Velocidade

* Transações processadas/segundo;
* Tempo de resposta de usuário/evento;
* Tempo de atualização de tela;

 Tamanho

* Megabytes;
* Número de chips de memória ROM;

Facilidade de uso

* Tempo de treinamento;
* Número de frames de ajuda;

Confiabilidade

* Tempo médio para falha;
* Probabilidade de indisponibilidade;
* Taxa de ocorrência de falhas;
* Disponibilidade;

Robustez

* Tempo de reinicio após falha;
* Percentual de eventos que causam falhas;
* Probabilidade de corrupção de dados em caso de falha;

Portabilidade

* Percentual de declarações dependentes do sistema-alvo;
* Número de sistemas-alvo;

***Sommerville (2008)***classifica os requisitos não funcionais em:

* ***Requisitos de produto*** que especificam o comportamento do produto. Podem restringir, por exemplo, a liberdade dos projetistas a utilizar uma determinada linguagem.
* ***Requisitos organizacionais***que são procedentes de políticas e procedimentos adotados nas organizações do cliente e do desenvolvedor. Especifica que o sistema deve ser de acordo com um processo-padrão da empresa.
* ***Requisitos externos*** que abrange tópicos advindos de fatores externos ao sistema. Dentre eles destacam-se os requisitos éticos e os requisitos legais que devem ser observados a fim de garantir que o sistema opera de acordo com a lei.

Parte superior do formulário

Parte inferior do formulário

Os requisitos **NÃO FUNCIONAIS** podem ser **classificados** em:

**I. Requisitos de Produto** - Características e restrições que são aplicadas diretamente ao produto entregável. Podem ser exemplo: Usabilidade, Requisitos de Eficiência em Desempenho ou Espaço, Confiabilidade e Portabilidade.

* **C**onfiabilidade, **E**ficiência, **D**esempenho, **P**ortabilidade, **E**spaço, **F**acilidade de uso, Requisito de proteção.

**II**. **Requisitos Organizacionais** - Consequência de políticas, processos e padrões organizacionais. Podem ser exemplo: Requisitos de Entrega, Implementação e Padrões.

* **P**adrão, **E**ntrega, **I**mplementação.

**III**. **Requisitos Externos** - Decorrente de fatores que são externos ao sistema, como legislação. Podem ser exemplo: Requisitos Legais de Privacidade e Segurança, Éticos e de Interoperabilidade.

* **L**egais, **É**ticos, **P**rivacidade.

**Classificação dos requisitos quanto à Evolução e Manutenção**

* **Requisitos Permanentes**: também chamados de Requisitos **Estáveis**, estão diretamente ligados à atividade principal da organização. São concebidos com a essência de um sistema e seu domínio da aplicação, e mudam mais lentamente que requisitos voláteis. Em geral, eles são derivados do Modelo de Domínio.
* **Requisitos Voláteis**: também chamados de Requisitos **Instáveis**, são específicos para a instanciação de um sistema em um ambiente ou um cliente particular e são mais propensos à mudança. Modificam-se quando o sistema está em desenvolvimento ou em uso. Podem ser:
* **Mutáveis**: são os requisitos que se modificam em função de mudanças no ambiente no qual o sistema opera. Por exemplo, os requisitos para um sistema que calcula taxas de dedução que evolui conforme as leis fiscais são atualizadas (muito comum no Brasil).
* **Emergentes**: são os requisitos que não podem ser completamente definidos quando o sistema é especificado e ***emergem (olha a dica!) à medida que a compreensão do cliente sobre o sistema se desenvolve***. Em geral, eles só aparecerão durante o desenvolvimento.
* **Consequentes**: são os requisitos baseados em suposições de como o sistema será utilizado, isto é, são resultados da introdução do sistema no ambiente do usuário. O usuário percebe as necessidades enquanto utiliza o sistema e esses requisitos são uma consequência (olha a dica!) do uso.
* **De Compatibilidade**: são os requisitos que dependem de outro equipamento, processo, componente ou elemento. Conforme outros elementos mudam, esses requisitos também mudam. Esses são requisitos menos comuns, mas que também ocorrem.

**Requisitos de usuários**. São declarações, em uma linguagem natural com diagramas, de quais serviços o sistema deverá fornecer a seus usuários e as restrições com as quais este deve operar.

**Requisitos de sistema**. São descrições mais detalhadas das funções, serviços e restrições operacionais do sistema de software. O documento de requisitos do sistema (às vezes, chamado especificação funcional) deve definir exatamente o que deve ser implementado. Pode ser parte do contrato entre o comprador do sistema e os desenvolvedores de software.

As principais atividades do processo de elicitação e análise de requisitos são:

* **Obtenção de Requisitos:**processo de interação com os stakeholders para coletar requisitos. Os requisitos de domínio também são descobertos durante essa atividade.
* **Classificação e organização de requisitos:** esta atividade envolve a coleção de requisitos não estruturados, agrupa os requisitos relacionados e os organiza em conjuntos coerentes.
* **Priorização e negociação de requisitos:** inevitavelmente, os requisitos serão conflitantes. Assim, busca-se priorizar os requisitos e resolver conflitos por meio da negociação.
* **Documentação de requisitos:** os requisitos são documentados e colocados na próxima volta da espiral. Podem ser produzidos documentos de requisitos formais ou informais.

***As atividades do processo de elicitação de requisitos são:***

**1.  Descoberta de requisitos.** Essa é a atividade de interação com os stakeholders do sistema para descobrir seus requisitos. Os requisitos de domínio dos stakeholders e da documentação também são descobertos durante essa atividade. Existem várias técnicas complementares que podem ser usadas para descoberta de requisitos, que discuto mais adiante.

**2.  Classificação e organização de requisitos**. Essa atividade toma a coleção de requisitos não estruturados, agrupa requisitos relacionados e os organiza em grupos coerentes. A forma mais comum de agrupar os requisitos é o uso de um modelo de arquitetura do sistema para identificar subsistemas e associar requisitos a cada subsistema. Na prática, a engenharia de requisitos e projeto da arquitetura não podem ser atividades completamente separadas.

**3.  Priorização e negociação de requisitos**. Inevitavelmente, quando os vários stakeholders estão envolvidos, os requisitos entram em conflito. Essa atividade está relacionada com a priorização de requisitos e em encontrar e resolver os conflitos por meio da negociação de requisitos. Normalmente, os stakeholders precisam se encontrar para resolver as diferenças e chegar a um acordo sobre os requisitos.

**4.  Especificação de requisitos**. Os requisitos são documentados e inseridos no próximo ciclo da espiral. Documentos formais ou informais de requisitos podem ser produzidos. *Se visualizado na forma de uma espiral, o processo de engenharia de requisitos, especialmente na fase de especificação de requisitos, inicia-se com a especificação de requisitos de negócio, depois de usuário e, por último, especificação e modelagem de sistema.*

No que se refere à atividade de **GERENCIAMENTO DE REQUISITOS** temos:

* Rastreamento dos requisitos
* Controle dos requisitos
* Identificação dos requisitos
* Modificação dos requisitos
* Compreensão das mudanças dos requisitos
* Controle das mudanças dos requisitos

**Estágios do gerenciamento de requisitos - Sommerville:**

* Identificação de requisitos
* Processo de gerenciamento de mudanças
* Políticas de Rastreabilidade
* Ferramenta de apoio

**Validação de Requisitos - Tipos de Verificação**

* **Verificação de Validade** – O sistema provê as funcionalidades que melhor atendem a necessidade dos clientes?
* **Verificação de Consistência – Existem requisitos conflitantes?**
* **Verificação de Completude** – Todas as funções requeridas pelo cliente estão inclusas?
* **Verificação de Realismo** – Os requisitos podem ser implementados de acordo com a tecnologia atual? Dadas às restrições do projeto (tecnológicas, financeiras e temporais) o sistema especificado tem de ser implementável (considera o **orçamento** e o cronograma).
* **Verificabilidade** - Os requisitos podem ser conferidos? Forma a evitar futuras discordâncias quanto à concretização dos requisitos especificados, estes devem ser descritos de forma que seja possível verificar se foram ou não implementados.

**Compreensibilidade / Ambiguidade**: os requisitos devem ser compreendidos de forma inequívoca pelas partes interessadas.

* **Adaptabilidade**: capacidade de sofrer alterações sem produzir efeitos em outros requisitos.
* **Conformidade com normas:** para além dos aspectos funcionais dos requisitos, a sua especificação deve obedecer às normas técnicas usadas para o desenvolvimento do sistema.

Durante o processo de validação de requisitos, diferentes tipos de verificação devem ser efetuados com os requisitos no documento de requisitos. Essas verificações incluem:

RESUMO:

* **Correção:**requisitos necessários devem coincidir com requisitos identificados.
* **Não ambiguidade:** requisitos são entendidos por todos da mesma maneira.
* **Completude:** descreve todas demandas necessárias.
* **Consistências:**não existe conflito entre requisitos.
* **Verificabilidade:**existe forma de verificar que sistema cumpre requisitos.
* **Modificabilidade:**é modificável de forma consistente com demais requisitos.

Parte superior do formulário

Parte inferior do formulário

Exemplos:

**Verificações de Validade -** identificar, por meio de análise mais aprofundada, outras funções necessárias, adicionais ou diferentes, além daquelas que um usuário pensava que fossem as necessárias para o sistema executar determinadas funções.

**Verificações de Consistência** - evitar que requisitos, no documento, entrem em conflito uns com outros, ou seja, não deve haver restrições contraditórias ou descrições diferentes para mesma função do sistema.

**Verificações de Completude -** garantir que o documento de requisitos contenha os requisitos que definem todas as funções e as restrições pretendidas pelos usuários do sistema.

**Verificações de Realismo -** assegurar, usando o conhecimento das tecnologias existentes, que os requisitos verificados possam ser realmente implementados, considerando o orçamento e o cronograma para o desenvolvimento do sistema.

**Verificabilidade** - reduzir o potencial de conflito entre o cliente e o contratante por meio de um conjunto de testes que demonstre que o sistema entregue atende a cada requisito especificado.

**Técnicas de Levantamento de Requisitos**

As técnicas de levantamento de requisitos têm por objetivo superar as dificuldades relativas a esta fase. Todas as técnicas possuem um conceito próprio e suas respectivas vantagens e desvantagens, que podem ser utilizadas em conjunto pelo analista. Algumas técnicas:

* **Joint Application Development** — **JAD** ou **Joint Application Design** é uma metodologia criada pela IBM do Canadá em 1977 e adaptada para o Brasil em 1982 por Hugo Gattoni para **moderação de discussões** de ***brainstorming***acelerando e consolidando o desenvolvimento de aplicações de Sistemas de Informação.

Parte superior do formulário

Parte inferior do formulário

Os **requisitos não funcionais** surgem de acordo com as necessidades dos usuários, em razão de restrições orçamentárias, de politicas organizacionais, pela necessidade de interoperabilidade com outros sistemas de software ou hardware e até mesmo em função de fatores externos. Esses últimos podem ser, por exemplo, de natureza legal ou de segurança.

* **Entrevistas:** em entrevista formal ou informal, a equipe de Engenharia de Requisitos **formula questões** para os stakeholders sobre os sistemas que eles usam e o sistema a ser desenvolvido. As entrevistas são úteis para obter um entendimento geral sobre o que os stakeholders fazem como eles podem interagir com o sistema e as dificuldades que enfrentam com os sistemas atuais. **Existem dois tipos de entrevistas:**

**Entrevistas fechadas**: um conjunto de questões **predefinidas** é respondido. (Observem que na questão existe a palavra chave: **predefinidas**)

**Entrevistas abertas**:  **não há um roteiro predefinido**, grande variedade de assuntos são explorados com os stakeholders.

A estrutura de uma entrevista diz respeito à organização das questões em uma sequência lógica. De acordo com (KENDALL; KENDALL, 2010), há três formas básicas de se organizar as questões de uma entrevista:  
  
**• Estrutura de Pirâmide (abordagem indutiva):** inicia com questões detalhadas e objetivas e, à medida que a entrevista progride, questões mais gerais, subjetivas, são colocadas. Útil para situações em que o entrevistado necessita de um “aquecimento” para falar no assunto ou quando o analista deseja obter uma finalização sobre o assunto.

• **Estrutura de Funil (abordagem dedutiva):** inicia com questões gerais subjetivas e, à medida que a entrevista avança, perguntas mais específicas, usando questões objetivas, são feitas. Essa estrutura provê um meio fácil e mais amigável para se começar uma bateria de entrevistas. Permite levantar informação detalhada, sendo desnecessárias longas sequências de questões objetivas e de aprofundamento.

• **Estrutura de Diamante:** é uma combinação das duas anteriores, começando com questões específicas, passando a questões gerais e fechando a entrevista novamente com questões específicas. É uma boa forma de se estruturar uma entrevista, já que mantém o interesse do entrevistado em uma variedade de questões. Contudo, tende a ser mais longa.

* **Brainstorm**: tradução literal seria "Temporal de ideias". Toda ideia deve ser levada em consideração, para isso é necessário que "atrasem" os julgamentos, para que todos possam participar e darem suas ideias, por mais diferentes que sejam. Essa é a ideia dessa técnica.
* **Etnografia:**é uma técnica de observação. O trabalho do dia a dia é observado e são feitas anotações sobre as tarefas reais em que os participantes estão envolvidos. A etnografia ajuda a descobrir requisitos implícitos do sistema que refletem as formas reais com que as pessoas trabalham.
* **Prototipagem**: na Engenharia de Software, **protótipo**é um modelo sem funcionalidades inteligentes,**podendo conter apenas funcionalidades gráficas**.**Utilizado para fins de ilustração e melhor entendimento**, geralmente em reuniões entre a equipe de Análise de Sistemas e o contratante
* Levantamento orientado a pontos de vista
* Workshops
* Questionários

**QFD**: No levantamento de requisitos existe uma técnica conhecida como "disponibilização da função de qualidade" (**quality function deployment - QFD**) que tenta traduzir as necessidades dos clientes em requisitos técnicos do software. Essa técnica identifica três tipos de necessidades:

* **Requisitos normais** - refletem os objetivos e metas do sistema. Se estiverem presentes no sistema, o cliente fica satisfeito.
* **Requisitos esperados** - requisitos implícitos do sistema. Por ser algo tão fundamental, o cliente pode não os declarar explicitamente. Sua ausência pode gerar grande insatisfação.
* **Requisitos fascinantes** - recursos que extrapolam as expectativas do cliente e que causa grande satisfação quando presentes.

**Uma Revisão Técnica Formal (RTF)** - é uma atividade de Garantia da Qualidade de Software realizada por engenheiros de software (e outros). A RTF é o filtro mais efetivo do ponto de vista de Garantia da Qualidade.

Os objetivos da RTF são:

* Descobrir erros na função, na lógica ou na implementação, para qualquer representação do software;
* **Verificar se o software sob revisão satisfaz seus requisitos;**
* **Garantir que o software tenha sido representado de acordo com padrões predefinidos;**
* Conseguir software que seja desenvolvido de modo uniforme;
* Tornar os projetos mais administráveis.

Para muitos profissionais da engenharia, conforme o **SWEBOK**, especificação refere-se à atribuição de valores numéricos ou limites para os objetivos do projeto. No entanto, a principal atividade desta fase (especificação) é a confecção da documentação do sistema, que pode ser sistematicamente revisada, validada e aprovada, especificando os componentes de software. Está é dividida em três subáreas:

* **Documentação de definição do sistema:** também conhecido como documento de requisitos do usuário, sendo descrito a definição dos requisitos do sistema de alto nível a partir da perspectiva de domínio. Seus leitores incluem os representantes dos usuários do sistema (clientes). O documento lista os requisitos de sistema, juntamente com informações básicas sobre os objetivos gerais para o sistema, seu ambiente de destino e um mapa de restrições, premissas e requisitos não funcionais.
* **Especificação dos requisitos do sistema:** Neste tópico os requisitos do sistema são especificados, O detalhamento deste tópico está fora do escopo do guia, sendo descrito como uma atividade da engenharia de sistemas.
* **Especificação dos requisitos do software:** estabelece as bases para um acordo entre os clientes e fornecedores para o produto de software em desenvolvimento, bem como o que não faz parte do seu escopo. Para o documento de especificação de requisitos de software é recomendado um documento de definição de requisitos de software, fornecendo uma base realista para estimar os custos dos produtos, riscos e cronogramas.

Estrutura da Entrevista

Diz respeito à organização das questões em uma sequência lógica. Há quatro formas básicas de se estabelecer a sequência das questões:

* **Estrutura de Pirâmide** (Abordagem Indutiva): inicia com questões bastante detalhadas, geralmente objetivas, e, à medida que a entrevista progride, questões mais gerais, subjetivas, são colocadas. Útil para situações onde o entrevistado parece relutante em abordar um assunto determinado ou se o engenheiro de software deseja obter uma finalização sobre o assunto.

* **Estrutura de Funil (Abordagem Dedutiva):** inicia com questões gerais subjetivas e à medida que a entrevista avança, perguntas mais específicas, usando questões objetivas, são feitas. Esta estrutura provê um meio fácil e não ameaçador para se começar uma bateria de entrevistas. Permite levantar bastante informação detalhada, sendo desnecessárias longas seqüências de questões objetivas e de aprofundamento.
* **Estrutura de Diamante:**Combinação das duas anteriores: começa com questões específicas, passa a questões gerais e fecha a entrevista novamente com questões específicas. Freqüentemente, é a melhor forma de se estruturar uma entrevista, já que mantém o interesse do entrevistado em uma variedade de questões. Contudo, tende a ser mais longa.
* **Entrevista Não Estruturada:** Não há uma definição da seqüência das questões. De acordo com o andar da entrevista, caminhos possíveis são avaliados e a seqüência é estabelecida. Requer mais tempo. Vale ressaltar que, ainda que a seqüência das questões não seja definida a priori, as questões devem ser definidas antecipadamente, ou seja, o planejamento é necessário.