2 REQUISITOS DE SOFTWARE

Prof. Me. Marcelo dos Santos Moreira

Engenharia de Software I

1. INTRODUÇÃO

Os requisitos de um sistema são descrições dos serviços fornecidos pelo sistema e as suas restrições operacionais. Esses requisitos refletem as necessidades dos clientes de um sistema que ajuda a resolver algum problema, por exemplo, controlar um dispositivo, enviar um pedido ou encontrar informações. O processo de descobrir, analisar, documentar e verificar esses serviços e restrições é chamado de **Engenharia de Requisitos** (RE - *Requirements Engineering*). Neste módulo, serão abordados os requisitos propriamente ditos e em como descrevê-los.

O termo requisito não é usado pela indústria de software de maneira consistente. Em alguns casos, um requisito é simplesmente uma declaração abstrata de alto nível de um serviço que o sistema deve fornecer ou uma restrição do sistema. No outro extremo, é uma definição formal e detalhada de uma função do sistema. Eis alguns motivos para essas diferenças:

> *Se uma empresa deseja estabelecer um contrato para o desenvolvimento de um grande projeto de software, ela precisa definir as suas necessidades de maneira suficientemente abstrata, para que uma solução não seja predefinida. Os requisitos devem ser redigidos de modo que os diversos fornecedores possam apresentar propostas, oferecendo, talvez, diferentes maneiras de atender às necessidades organizacionais do cliente. Após a aprovação do contrato, o fornecedor deve redigir uma definição mais detalhada do sistema para o cliente, de modo que o cliente possa compreender e validar o que o software fará. Esses dois documentos podem ser chamados de documentos de requisitos do sistema.*

Alguns dos problemas que surgem durante o processo de engenharia de requisitos são resultantes da falta de uma clara separação entre esses diferentes níveis de descrição. Essa distinção entre eles deve ser feita usando o termo ***requisitos de usuário*** para requisitos abstratos de alto nível e ***requisitos de sistema*** para a descrição detalhada do que o sistema deve fazer. Os requisitos de usuário e de sistema podem ser definidos da seguinte maneira:

1. ***Requisitos de usuário*** são declarações, em uma linguagem natural com diagramas, de quais serviços são esperados do sistema e as restrições sob as quais ele deve operar.

2. ***Requisitos de sistema*** definem, detalhadamente, as funções, os serviços e as restrições operacionais do sistema. O documento de requisitos de sistema (também chamado de *especificação funcional*) deve ser preciso.

Ele deve definir exatamente o que será implementado. Pode ser parte do contrato entre o comprador do sistema e os desenvolvedores de software.

Diferentes níveis de especificação de sistema são úteis porque eles comunicam informações sobre o sistema a diferentes tipos de leitores. O Quadro 1.1 ilustra a diferença entre os requisitos de usuário e de sistema. Esse exemplo, baseado em um sistema de biblioteca, mostra como um requisito de usuário pode ser dividido em diversos requisitos de sistema.

Quadro 1.1 – Requisitos de usuários e de sistema

Definição de requisitos de usuário
1. LIBSYS deve manter o acompanhamento de todos os dados exigidos pelas agências de licenciamento de direitos autorais no Reino Unido e em outros lugares.
Especificação dos requisitos de sistema
1.1. Ao solicitar um documento ao LIBSYS, deve ser apresentado ao solicitante um formulário que registra os detalhes do usuário e da solicitação feita. 1.2. Os formulários de solicitação do LIBSYS devem ser armazenados no sistema durante cinco anos, a partir da data da solicitação. 1.3. Todos os formulários do LIBSYS devem ser indexados por usuário, nome do material solicitado e fornecedor da solicitação. 1.4. O LIBSYS deve manter um registro de todas as solicitações feitas ao sistema. 1.5. Para materiais aos quais se aplicam os direitos de empréstimo dos autores, os detalhes do empréstimo devem ser enviados mensalmente às agências de licenciamento de direitos autorais que se registraram no LIBSYS.

Pode-se notar no Quadro 1.1 que o requisito de usuário é mais abstrato e os requisitos de sistema acrescentam detalhes, explicando os serviços e as funções que devem ser fornecidos pelo sistema a ser desenvolvido.

É necessário escrever os requisitos em diferentes níveis de detalhes, pois diferentes tipos de leitores usam os requisitos de diferentes maneiras. A Figura 1.1 mostra os tipos de leitores para os requisitos de usuário e de sistema. Os leitores dos requisitos de usuário geralmente não estão preocupados com o modo como o sistema será implementado e podem ser gerentes que não estejam interessados nos recursos detalhados do sistema. Os leitores dos requisitos de sistema necessitam conhecer mais precisamente o que o sistema fará, pois eles estão preocupados com o modo como o sistema apoiará os processos de negócio ou porque estão envolvidos na implementação do sistema.

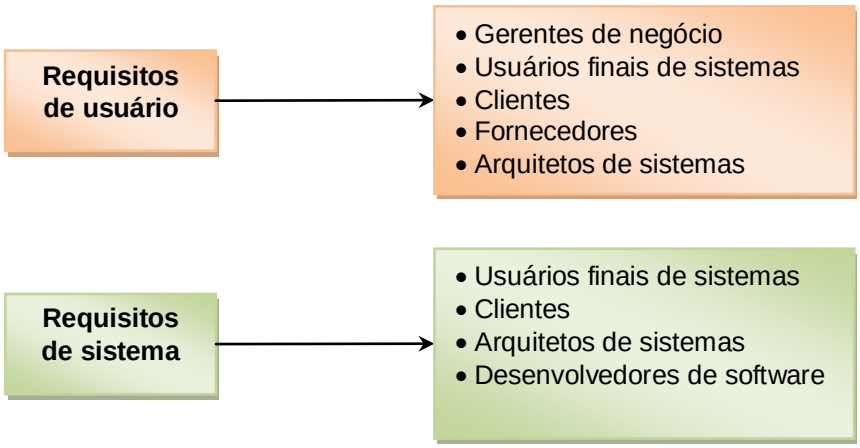

Figura 1.1 – leitores de diferentes tipos de especificação

2. REQUISITOS FUNCIONAIS E NÃO FUNCIONAIS

Os **requisitos de sistema** de software são, freqüentemente, classificados em requisitos funcionais, requisitos não funcionais ou requisitos de domínio:

1. ***Requisitos funcionais***: são as declarações de serviços que o sistema deve fornecer, como o sistema deve reagir a entradas específicas e como o sistema deve se comportar em determinadas situações. Em alguns casos, os requisitos funcionais podem também estabelecer explicitamente o que o sistema não deve fazer.

2. ***Requisitos não funcionais***: são restrições sobre os serviços ou as funções oferecidos pelo sistema. Eles incluem restrições de tempo, restrições sobre o processo de desenvolvimento e padrões. Os requisitos não funcionais aplicam-se, frequentemente, ao sistema como um todo. Em geral, eles não se aplicam às características ou serviços individuais de sistema.

3. ***Requisitos de domínio***: são requisitos provenientes do domínio da aplicação do sistema e que refletem as características e as restrições desse domínio. Podem ser requisitos funcionais ou não funcionais.

Na realidade, a distinção entre os diferentes tipos de requisitos não é tão clara como sugerem essas definições. Um requisito de usuário relacionado à proteção, por exemplo, pode parecer um requisito não funcional. No entanto, quando desenvolvido mais detalhadamente, esse requisito pode gerar outros requisitos claramente funcionais, como a necessidade de incluir recursos de autenticação de usuário no sistema.

2.1. Requisitos funcionais

Os requisitos funcionais de um sistema descrevem o que o sistema deve fazer. Esses requisitos dependem do tipo do software que está sendo desenvolvido, dos usuários a que o software se destina e da abordagem geral considerada pela organização ao redigir os requisitos. Quando expressos como requisitos de usuário, eles são geralmente descritos de forma bastante abstrata. No entanto, os requisitos funcionais descrevem a função do sistema detalhadamente, suas entradas e saídas, exceções etc.

Os requisitos funcionais de um sistema de software podem ser expressos de várias formas. Por exemplo, nesta seção, são apresentados os requisitos funcionais de um sistema de biblioteca de uma universidade chamado LIBSYS, usado por estudantes e pela faculdade para solicitar livros e documentos de outras bibliotecas:

1. O usuário deve ser capaz de fazer uma busca de todos os títulos disponíveis.

2. O sistema deve fornecer telas apropriadas para o usuário ler os documentos no repositório de documentos.

3. Para cada pedido, deve ser atribuído um identificador único (ORDER_ID), o qual o usuário deve ser capaz de copiar para a área de armazenamento permanente da conta.

Esses requisitos funcionais de usuário definem recursos específicos a serem fornecidos pelo sistema. Eles foram tirados do documento de requisitos de usuário e ilustram que os requisitos funcionais podem ser escritos em níveis diferentes de detalhes (comparar os requisitos 1 e 3).

O sistema LIBSYS é uma interface única com uma série de bancos de dados de artigos. Ele permite que os usuários baixem cópias de artigos publicados em revistas, jornais e periódicos científicos. .

A imprecisão na especificação de requisitos é o motivo de muitos problemas de engenharia de software. É natural que um desenvolvedor de sistema interprete um requisito ambíguo de modo a simplificar a sua implementação. Frequentemente, no entanto, isso não é o que o cliente quer. Novos requisitos precisam ser definidos e mudanças devem ser feitas no sistema. Naturalmente, isso atrasa a entrega do sistema e aumenta os custos.

Considere o segundo requisito do exemplo do sistema de biblioteca, referente a 'telas apropriadas' fornecidas pelo sistema. O sistema de biblioteca pode entregar documentos em uma série de formatos; a intenção desse requisito é que as telas de todos

esses formatos estejam disponíveis. No entanto, o requisito está redigido de forma ambígua, sem deixar claro que as telas de cada formato de documento devem ser fornecidas. Um desenvolvedor, sob pressão do cronograma, pode simplesmente fornecer uma tela de texto e argumentar que o requisito foi atendido.

Em princípio, a especificação de requisitos funcionais de um sistema deve ser completa e consistente. Completeza significa que todos os serviços exigidos pelo usuário devem ser definidos. Consistência significa que os requisitos não devem ter definições contraditórias. Na prática, em sistemas grandes e complexos, é praticamente impossível atingir a consistência e a completeza de requisitos.

Uma razão para isso é que é fácil cometer erros e omissões quando se redigem especificações para sistemas grandes e complexos. Outra razão é que diferentes *stakeholders* no sistema têm diferentes – e freqüentemente inconsistentes – necessidades. Essas inconsistências podem não ser óbvias quando os requisitos são especificados inicialmente, de modo que requisitos inconsistentes sejam incluídos na especificação. Os problemas podem aparecer apenas depois de uma análise mais profunda ou, às vezes, depois que o desenvolvimento estiver concluído e o sistema tiver sido entregue para o cliente.

2.2. Requisitos não funcionais

Os requisitos não funcionais, como o nome sugere, são aqueles não diretamente relacionados às funções específicas fornecidas pelo sistema. Eles podem estar relacionados às propriedades emergentes do sistema, como confiabilidade, tempo de resposta e espaço de armazenamento. Como alternativa, eles podem definir restrições, como a capacidade dos dispositivos de E/S (entrada/saída) e as representações de dados usadas nas interfaces do sistema.

Os requisitos não funcionais estão raramente associados às características individuais do sistema. Pelo contrário, esses requisitos especificam ou restringem as propriedades emergentes de sistema. Portanto, podem especificar desempenho, proteção, disponibilidade e outras propriedades emergentes do sistema. Isso significa que eles geralmente são mais importantes do que os requisitos funcionais individuais. Os usuários do sistema em geral encontram meios de contornar uma função do sistema que não atenda às suas necessidades. Contudo, uma falha no atendimento de um requisito não funcional pode significar que todo o sistema é inútil. Por exemplo, se uma aeronave não atende aos requisitos de confiabilidade, ela não será certificada como segura para operação; se um sistema de controle de tempo real falhar em atender aos requisitos de desempenho, as funções de controle não operarão corretamente.

Os requisitos não funcionais não estão relacionados apenas com o sistema de software a ser desenvolvido. Alguns deles podem restringir o processo que deve ser usado para desenvolver o sistema. Exemplos de requisitos de processo incluem uma especificação dos padrões de qualidade que devem ser usados no processo, uma especificação de que o projeto seja produzido com um determinado conjunto de ferramentas CASE e uma descrição do processo que deve ser seguido.

Os requisitos não funcionais surgem devido às necessidades do usuário, às restrições de orçamento, às políticas organizacionais, à necessidade de interoperabilidade com outros sistemas de software ou hardware ou a fatores externos, regulamentos de segurança ou legislação a respeito da privacidade. A Figura 2.1 apresenta uma classificação de requisitos não funcionais. Pode-se observar no diagrama que os requisitos não funcionais podem vir de características necessárias do software (requisitos de produto), da organização que desenvolve o software (requisitos organizacionais) ou de fontes externas.

Os tipos de requisitos não funcionais são:

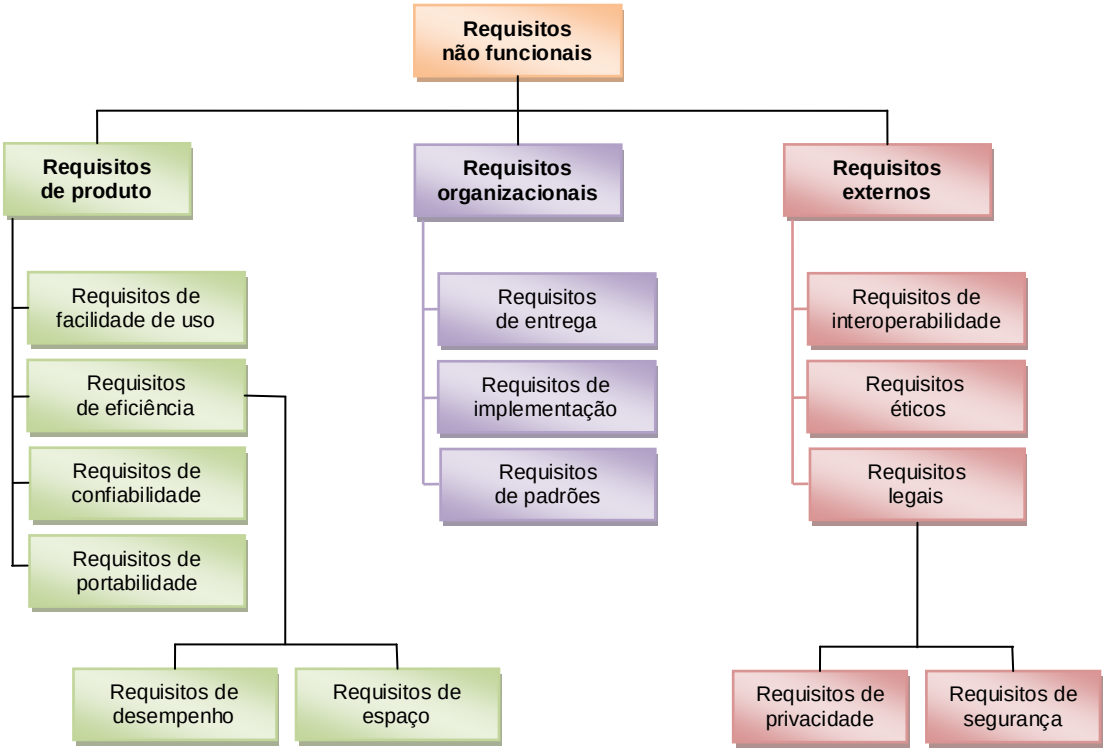

Figura 2.1 – Tipos de requisitos não funcionais

1. **Requisitos de produto**: Estes requisitos especificam o comportamento do produto. Entre os exemplos estão requisitos de desempenho quanto à rapidez com que o sistema deve operar e quanto de memória ele

requer, requisitos de confiabilidade que definem a taxa aceitável de falhas, requisitos de portabilidade e requisitos de usabilidade.

2. **Requisitos organizacionais**: Estes requisitos são derivados de políticas e procedimentos da organização do cliente e do desenvolvedor. Alguns exemplos incluem padrões de processo que devem ser usados, requisitos de implementação, como a linguagem de programação ou o método de projeto usado, e requisitos de entrega que especificam quando o produto e a sua documentação devem ser entregues.

3. **Requisitos externos**: Este título amplo abrange todos os requisitos derivados de fatores externos ao sistema e seu processo de desenvolvimento. Podem incluir requisitos de interoperabilidade que definem como o sistema interage com os sistemas em outras organizações, requisitos legais que devem ser seguidos para assegurar que o sistema funciona dentro da lei e requisitos éticos. Os requisitos éticos são aqueles incluídos no sistema para assegurar que ele será aceito por seus usuários e pelo público em geral.

O Quadro 2.1 mostra exemplos de requisitos de produto, organizacionais e externos, tirados do sistema LIBSYS, cujos requisitos de usuário foram explicados na Seção 2.1. Um requisito de produto restringe a liberdade dos projetistas do LIBSYS na implementação da interface com o usuário de sistema. Nada é dito sobre a funcionalidade do LIBSYS e isso identifica claramente uma restrição de sistema e não de função. Esse requisito foi incluído porque simplifica o problema de assegurar que o sistema funcione com diferentes navegadores.

Um requisito organizacional especifica que o sistema deve ser desenvolvido de acordo com um processo padrão da empresa, definido como XYZCo-SP-STAN-95. O requisito externo é derivado da necessidade de o sistema estar em conformidade com a legislação de privacidade. Ele especifica que o pessoal da biblioteca não deve ter acesso a dados como endereços dos usuários do sistema, os quais não são necessários para execução do trabalho.

Um problema comum com os requisitos não funcionais é que eles podem ser difíceis de verificar. Os usuários ou os clientes freqüentemente definem esses requisitos como metas gerais, como usabilidade, capacidade do sistema de se recuperar de falhas ou resposta rápida ao usuário. Essas metas vagas causam problemas para os desenvolvedores de sistema, pois eles deixam o escopo para interpretação e discussão subseqüentes, depois que o sistema é entregue. Como ilustração desse problema, considere o Quadro 2.2. Ele mostra uma meta de sistema relacionada à usabilidade de um sistema de controle de uma forma típica de como um usuário pode expressar os requisitos de facilidade de uso. Esse requisito foi reescrito para

demonstrar como a meta pode ser expressa como um requisito não funcional verificável por teste. Mesmo que seja impossível verificar objetivamente a meta do sistema, você pode projetar testes de sistema para contar os erros cometidos pelos controladores que usam um simulador de sistema.

Quadro 2.1 – Exemplos de requisitos não funcionais

Requisito de produto
A interface de usuário para o LIBSYS deve ser implementada como simples HTML, sem *frames* ou *applets* de Java.
Requisito organizacional
O processo de desenvolvimento do sistema e os documentos a serem entregues devem estar em conformidade com o processo e produtos a serem entregues definidos em XYZCo-SP-STAN-95.
Requisito externo
O sistema não deve revelar quaisquer informações pessoais sobre os usuários do sistema ao pessoal da biblioteca que usa o sistema, com exceção do nome e número de referência da biblioteca.

Quadro 2.2 – Metas do sistema e requisitos verificáveis

Meta do sistema
O sistema deve ser fácil de ser usado pelos controladores experientes e ser organizado de modo que os erros dos usuários sejam minimizados.
Requisito não funcional verificável
Os controladores experientes devem ser capazes de usar todas as funções do sistema depois de um **treinamento** no total de **duas horas**. Após esse treinamento, o **número médio de erros** cometidos pelos usuários experientes não deve exceder **dois por dia**.

Sempre que possível, deve-se escrever os requisitos não funcionais quantitativamente, de modo que eles possam ser testados objetivamente. A Tabela 2.1 mostra uma série de métricas possíveis de serem usadas para especificar as propriedades não funcionais do sistema. Podem-se medir essas características quando o sistema estiver sendo testado para verificar se ele atendeu ou não aos requisitos não funcionais.

Tabela 2.1 – Métricas para especificar requisitos não funcionais

Velocidade	Transações processadas/segundo Tempo de resposta de usuário/evento Tempo de atualização da tela
Tamanho	Kbytes Número de chips de RAM
Facilidade de uso	Tempo de treinamento Número de frames de ajuda
Confiabilidade	Tempo médio de falha Probabilidade de indisponibilidade Taxa de ocorrência de falhas Disponibilidade
Robustez	Tempo para reiniciar após falha Porcentagem de eventos que causam falhas Probabilidade de corrupção de dados por falhas
Portabilidade	Porcentagem de declarações dependentes do sistema-alvo Número de sistemas-alvo

Na prática, no entanto, os clientes de um sistema podem considerar praticamente impossível traduzir suas metas em requisitos quantitativos. Para algumas metas, como facilidade de manutenção, não existem métricas. Em outros casos, mesmo quando uma especificação quantitativa seja possível, os clientes podem não ser capazes de relacionar as suas necessidades com essas especificações. Eles não compreendem o que um número que define a confiabilidade necessária, por exemplo, significa em termos de sua experiência diária com sistemas de computador. Além disso, o custo da verificação objetiva dos requisitos não funcionais quantitativos pode ser muito alto e os clientes que pagam pelo sistema podem não considerar que esses custos sejam justificados.

Portanto, os documentos de requisitos freqüentemente incluem declarações de metas combinadas com requisitos. Essas metas podem ser úteis para os desenvolvedores, pois fornecem indicações das prioridades do cliente. Contudo, deve-se sempre informar aos clientes que elas estão abertas à má interpretação e não podem ser verificadas objetivamente.

Os requisitos não funcionais freqüentemente entram em conflito e interagem com outros requisitos funcionais e não funcionais. Por exemplo, pode haver um requisito que

limite a memória máxima usada por um sistema a 4 Mbytes. As restrições de memória são comuns em sistemas embutidos, no qual o espaço ou o peso é limitado e o número de chips ROM que armazenam o software do sistema deve ser mínimo. Outro requisito pode ser que o sistema seja escrito em Ada, uma linguagem de programação para desenvolvimento de software crítico de tempo real. Entretanto, pode não ser possível compilar um programa em Ada com a funcionalidade necessária em menos que 4 Mbytes. Portanto, precisa haver um compromisso entre esses requisitos: uma linguagem de programação alternativa ou mais memória adicionada ao sistema.

Será útil se puder diferenciar os requisitos funcionais e não funcionais no documento de requisitos. Na prática, isso é difícil de ser feito. Se os requisitos não funcionais forem definidos separadamente dos funcionais, às vezes será difícil verificar os relacionamentos entre eles. Se eles forem definidos com os requisitos funcionais, poderá haver dificuldade em separar as considerações funcionais e não funcionais e identificar os requisitos relacionados ao sistema como um todo. Contudo, deve-se destacar explicitamente os requisitos claramente relacionados com as propriedades emergentes de sistema, como o desempenho e a confiabilidade. Pode-se fazer isso colocando-os em uma seção separada do documento de requisitos ou diferenciando-os, de alguma forma, dos demais requisitos de sistema. Os requisitos não funcionais, como segurança e proteção, são particularmente importantes para sistemas críticos.

2.3. Requisitos de domínio

Os requisitos de domínio são derivados do domínio de aplicação do sistema, em vez das necessidades específicas dos usuários do sistema. Geralmente incluem uma terminologia específica de domínio ou fazem referência a conceitos do domínio. Podem ser novos requisitos funcionais em si mesmos, podem restringir os requisitos funcionais existentes ou estabelecer como cálculos específicos devem ser realizados. Como esses requisitos são especializados, os engenheiros de software freqüentemente encontram dificuldade em compreender como eles estão relacionados a outros requisitos do sistema.

Os requisitos de domínio são importantes porque, com freqüência, refletem os fundamentos do domínio da aplicação. Se esses requisitos não forem satisfeitos, pode ser impossível fazer o sistema funcionar satisfatoriamente. O sistema LIBSYS inclui um conjunto de requisitos de domínio:

1. Deve existir uma interface com o usuário-padrão para todos os bancos de dados e que deverá ser baseada no padrão Z39.50.

2. Devido às restrições de direitos autorais, alguns documentos devem ser excluídos imediatamente na chegada. Dependendo dos requisitos

de usuário, esses documentos serão impressos localmente no servidor do sistema para serem encaminhados manualmente para o usuário ou direcionados a uma impressora de rede.

O **primeiro** requisito é uma restrição de projeto. Ele especifica que a interface com o usuário do banco de dados deve ser implementada de acordo com um padrão específico de biblioteca. Os desenvolvedores, portanto, devem conhecer o padrão antes de iniciar o projeto de interface. O **segundo** requisito foi introduzido devido às leis de direitos autorais aplicadas ao material usado em bibliotecas. Ele especifica que o sistema deve incluir um recurso automático '*excluir após impressão*' para algumas classes de documento. Isso significa que os usuários do sistema de biblioteca não podem ter suas próprias cópias eletrônicas do documento.

Para ilustrar os requisitos de domínio que especificam como um cálculo deve ser realizado, considere o Quadro 2.3, cujos dados são da especificação de requisitos do sistema de proteção de um trem automatizado. Esse sistema pára automaticamente um trem caso ele ultrapasse um sinal vermelho. O requisito define como a desaceleração do trem é calculada pelo sistema. Ele usa a terminologia específica do domínio. Para compreendê-lo, é necessário algum conhecimento sobre operação de sistemas ferroviários e das características do trem.

Quadro 2.3 – Requisito de domínio de um sistema de proteção de trens

A desaceleração do trem deve ser calculada como:

$D_{trem} = D_{controle} + D_{gradiente}$

onde $D_{gradiente}$ é 9,81 ms^2 * gradiente compensado/alfa e onde os valores de 9,81 ms^2/alfa são conhecidos para diferentes tipos de trens.

O requisito do sistema de trem ilustra um grande problema com os requisitos de domínio. Eles são redigidos na linguagem do domínio da aplicação (equações matemáticas, neste caso) e, geralmente, os engenheiros de software têm dificuldade em compreendê-los. Os especialistas de domínio podem deixar determinadas informações fora de um requisito, simplesmente por serem muito óbvias para eles. Contudo, pode não ser óbvia para os desenvolvedores do sistema e eles podem, portanto, implementar o requisito de forma equivocada.

3. REQUISITOS DE USUÁRIO

Os requisitos de usuário de um sistema devem descrever os requisitos funcionais e não funcionais, de modo que eles sejam compreensíveis pelos usuários do sistema que não possuem conhecimento técnico detalhado. Eles devem especificar apenas o comportamento externo do sistema e evitar, sempre que possível, características de projeto do sistema.

Conseqüentemente, se estiver sendo escrito requisitos de usuário, não se deve usar jargões de software, notações estruturadas ou formais ou descrever os requisitos por meio da implementação do sistema. Deve-se escrever os requisitos de usuário em linguagem simples, com tabelas e formulários simples e diagramas intuitivos.

No entanto, vários problemas podem surgir quando os requisitos são redigidos em um documento com texto em linguagem natural:

1. **Falta de clareza**: Às vezes, é difícil usar a linguagem de maneira precisa e não ambígua sem tomar o documento prolixo e difícil de ler.
2. **Confusão de requisitos**: Requisitos funcionais, requisitos não funcionais, metas de sistema e informações de projeto podem não estar claramente diferenciados.
3. **Fusão de requisitos**: Diversos requisitos diferentes podem ser expressos juntos como um único requisito.

Como ilustração de alguns desses problemas, considere um dos requisitos para a biblioteca apresentado no Quadro 3.1.

Quadro 3.1 – Requisito de usuário para um sistema de contabilidade no LIBSYS

O LIBSYS **deve** fornecer um sistema de contabilidade financeira que mantenha registros de todos os pagamentos realizados pelos usuários do sistema. Os gerentes **podem** configurar esse sistema de modo que os usuários freqüentes possam receber descontos.

Esse requisito inclui informações conceituais e detalhadas. Ele expressa o conceito de que deve haver um sistema de contabilidade como parte inerente de LIBSYS. Contudo, ele também diz que o sistema de contabilidade deve apoiar descontos aos usuários freqüentes de LIBSYS. Esse detalhe seria mais adequado na especificação de requisitos do sistema.

É uma boa prática separar os requisitos de usuário dos requisitos mais detalhados do sistema em um documento de requisitos. De outra forma, os leitores não técnicos dos requisitos de usuário podem ser sobrecarregados pelos detalhes que, na verdade, são relevantes apenas para os técnicos. O Quadro 3.2 ilustra essa confusão. Esse exemplo é um documento real de requisitos para uma ferramenta CASE de edição de modelos de projeto de software. O usuário pode especificar que uma grade deve ser apresentada, de modo que as entidades possam ser posicionadas precisamente em um diagrama.

Quadro 3.2 – Requisito de usuário para uma grade de editor

Recursos de grade: Para ajudar no posicionamento de entidades sobre um diagrama, o usuário pode ativar uma grade em centímetros ou em polegadas por meio de uma opção no painel de controle. Inicialmente, a grade não estará ativada. A grade poderá ser ativada ou desativada a qualquer instante durante uma sessão de edição e poderá ser alternada entre polegadas e centímetros, a qualquer instante. Uma opção de grade será fornecida na visão reduzida, mas o número de linhas de grade mostradas será reduzido para evitar o preenchimento do diagrama.

A primeira sentença mistura três tipos de requisitos:

1. O requisito funcional conceitual define que o sistema de edição deve fornecer uma grade. Ele apresenta uma justificativa lógica para isso.

2. O requisito não funcional fornece informações detalhadas sobre as unidades da grade (centímetros ou polegadas).

3. O requisito não funcional de interface com o usuário define como a grade é ativada e desativada pelo usuário.

O requisito no Quadro 3.2 também apresenta algumas, mas não todas, informações de iniciação. Ele define que a grade inicialmente está desativada. Contudo, não define as suas unidades quando é ativada. Ele fornece algumas informações detalhadas – isto é, que o usuário pode alternar entre as duas unidades – mas não o espaçamento entre as linhas da grade.

Os requisitos de usuário que incluem muitas informações restringem a liberdade do desenvolvedor do sistema de apresentar soluções inovadoras para os problemas de usuário e são difíceis de serem compreendidos. O requisito de usuário deve simplesmente enfocar recursos principais a serem fornecidos. O requisito de grade de editor foi reescrito – Quadro 3.3 – para enfocar apenas as características essenciais do sistema.

Sempre que possível, deve-se tentar associar uma justificativa lógica a cada requisito do usuário. A justificativa deve esclarecer por que o requisito foi incluído, e é

particularmente útil quando os requisitos são mudados. Por exemplo, a justificativa lógica no Quadro 3.3 reconhece que uma grade ativa, na qual os objetos posicionados 'saltam' automaticamente para uma linha da grade pode ser útil. Entretanto, isso foi intencionalmente rejeitado em favor do posicionamento manual.

Quadro 3.3 – Definição de um recurso de grade de editor

Recursos de grade:

O editor deve fornecer um recurso de grade no qual uma matriz de linhas horizontais e verticais fornece um fundo para a janela do editor. Essa grade deve ser passiva e o alinhamento das entidades é de responsabilidade do usuário.

***Justificativa lógica**: Uma grade ajuda o usuário a criar um diagrama bem organizado com entidades bem espaçadas. Embora uma grade ativa, na qual as entidades 'saltam' as linhas de grade, possa ser útil, o posicionamento é impreciso. O usuário é a melhor pessoa para decidir onde as entidades devem ser posicionadas.*

Especificação: ECLIPSE/WS/Tools/DE/FS Seção 5.6

Fonte: Ray Wilson, escritório de Glasgow

Se uma mudança nesse requisito for proposta mais tarde, ficará claro que o uso de uma grade passiva foi intencional, e não uma decisão de implementação.

Para minimizar os mal-entendidos ao redigir requisitos de usuário, recomenda-se que sejam seguidas algumas diretrizes simples:

1. Invente um formato padrão e assegure-se de que todas as definições de requisitos aderiram a esse formato. A padronização de formato torna as omissões menos prováveis e os requisitos mais fáceis de serem verificados. O formato usado mostra o requisito inicial em negrito, inclui uma declaração da justificativa lógica de cada requisito de usuário e uma referência à especificação detalhada de requisitos do sistema. Pode-se, também, incluir informações sobre quem propôs o requisito (a fonte do requisito) de modo que se saiba a quem consultar se o requisito tiver de ser mudado.

2. Use a linguagem de forma consistente. Deve-se sempre fazer distinção entre requisitos obrigatórios e desejáveis. **Requisitos obrigatórios** são aqueles a que o sistema deve atender e são escritos com o uso da palavra '***deve***'. **Requisitos desejáveis** não são essenciais e são escritos com o uso da palavra '***pode***'.

3. Use destaque no texto (negrito, itálico ou cor) para ressaltar as partes principais do requisito.

4. Evitar, sempre que possível, o uso de jargões de informática. Contudo, inevitavelmente aparecerão termos técnicos detalhados nos requisitos de usuário.

O método VOLERE de engenharia de requisitos recomenda que os requisitos de usuário sejam inicialmente escritos em cartões, sendo um requisito por cartão. Sugere alguns campos, como as justificativas lógicas dos requisitos, as dependências com outros requisitos, a fonte dos requisitos, material de apoio etc. Isso amplia o formato que usado no Quadro 3.3 e pode ser usado para requisitos de usuário e de sistema.

4. REQUISITOS DE SISTEMA

Os requisitos de sistema são versões expandidas dos requisitos de usuário usados pelos engenheiros de software como ponto de partida para o projeto do sistema. Eles adicionam detalhes e explicam como os requisitos de usuário devem ser fornecidos pelo sistema. Podem ser usados como parte do contrato para a implementação do sistema e devem, portanto, ser uma especificação completa e consistente de todo o sistema.

De forma ideal, os requisitos de sistema devem simplesmente descrever o comportamento externo do sistema e as suas restrições operacionais. Eles não devem estar relacionados a como o sistema pode ser projetado ou implementado. Entretanto, para especificar completamente um sistema de software complexo no nível de detalhamento necessário, é impossível, na prática, excluir todas as informações de projeto. Existem várias razões para isso:

1. Pode-se ter de projetar uma arquitetura inicial do sistema para auxiliar na estruturação da especificação de requisitos. Os requisitos de sistema são organizados de acordo com os diferentes subsistemas que constituem o sistema. Essa definição de arquitetura é essencial caso se deseje reusar componentes de software na implementação do sistema.

2. Na maioria dos casos, os sistemas devem interoperar com outros sistemas existentes. Isso restringe o projeto e essas restrições impõem requisitos ao novo sistema.

3. O uso de uma arquitetura específica para satisfazer os requisitos não funcionais (tal como programação de N-versões para conseguir a confiabilidade) pode ser necessário. Um regulamentador externo que necessita certificar que o sistema é seguro pode especificar que um projeto de arquitetura que já tenha sido certificado seja usado.

A linguagem natural é freqüentemente usada para redigir especificações de requisitos de sistema bem como requisitos de usuário. Contudo, como os requisitos de sistema são mais detalhados que os requisitos de usuário, as especificações em linguagem natural podem ser confusas e difíceis de serem compreendidas:

1. A compreensão da linguagem natural depende do uso das mesmas palavras para os mesmos conceitos pelos leitores e pelos elaboradores da especificação. Isso leva a mal-entendidos devido à ambigüidade da linguagem natural. Um excelente exemplo disso é ao explicar os avisos apresentados em uma escada rolante: '*Sapatos devem ser usados*' e '*Cachorros devem ser carregados*'. Quanta interpretação conflitante nestas frases!!!

2. Uma especificação de requisitos em linguagem natural é flexível demais. É possível dizer a mesma coisa de maneiras completamente diferentes. Fica por conta do leitor descobrir quando os requisitos são os mesmos e quando são distintos.

3. Não existe uma maneira fácil de padronizar os requisitos em linguagem natural. Pode ser difícil encontrar todos os requisitos relacionados. Para descobrir as conseqüências de uma mudança, pode ser necessário verificar todos os requisitos em vez de apenas um grupo de requisitos relacionados.

Devido a problemas, as especificações de requisitos escritas em linguagem natural são propensas a mal-entendidos. Eles frequentemente não são descobertos até as fases finais do processo de software e, portanto, a solução deles pode ser muito onerosa.

É essencial escrever os requisitos de usuário em uma linguagem que os não-especialistas possam compreender. Contudo, pode-se escrever os requisitos de sistema usando notações mais especializadas – Tabela 4.1. Essas notações incluem linguagem natural estruturada e estilizada, modelos gráficos dos requisitos, como casos de uso até especificações matemáticas formais.

Tabela 4.1 – Notações para especificação de requisitos

Linguagem natural estruturada	Esta abordagem depende da definição de formulários ou *templates*-padrão para expressar a especificação de requisitos.
Linguagens de descrição de projeto	Esta abordagem usa uma linguagem semelhante à linguagem de programação, porém com mais características abstratas, para especificar os requisitos por meio da definição de um modelo operacional do sistema. Essa abordagem não é amplamente usada hoje em dia, embora possa ser útil para especificações de interfaces.
Notações gráficas	Uma linguagem gráfica, complementada com anotações de texto é usada para definir os requisitos funcionais do sistema. Um antigo exemplo dessa linguagem gráfica é SADT. Atualmente, as descrições de casos de uso e os diagramas de seqüência são comumente usados (UML).
Especificações matemáticas	São notações baseadas em conceitos matemáticos, como máquinas de estados finitos ou conjuntos. Essas especificações não ambíguas reduzem discussões entre cliente e fornecedor. No entanto, a maioria dos clientes não compreende as especificações formais e são relutantes em aceitá-las no momento da contratação.

5. DOCUMENTO DE REQUISITOS DE SOFTWARE

O documento de requisitos de software (algumas vezes chamado de especificação de requisitos de software ou SRS - *Software Requirements Specification*) é a declaração oficial do que os desenvolvedores de sistema devem implementar. Deve incluir os requisitos de usuário de um sistema e uma especificação detalhada dos requisitos de sistema. Em alguns casos, os requisitos de usuário e de sistema podem estar integrados em uma única descrição. Em outros casos, os requisitos de usuário estão definidos em uma introdução à especificação dos requisitos de sistema. Se houver um grande número de requisitos, os requisitos detalhados de sistema podem ser apresentados em um documento separado.

O documento de requisitos possui um conjunto diversificado de usuários, desde a gerência sênior da organização, que paga pelo sistema, até os engenheiros responsáveis pelo desenvolvimento do software. A Figura 5.1 ilustra os possíveis usuários do documento e como eles o utilizam.

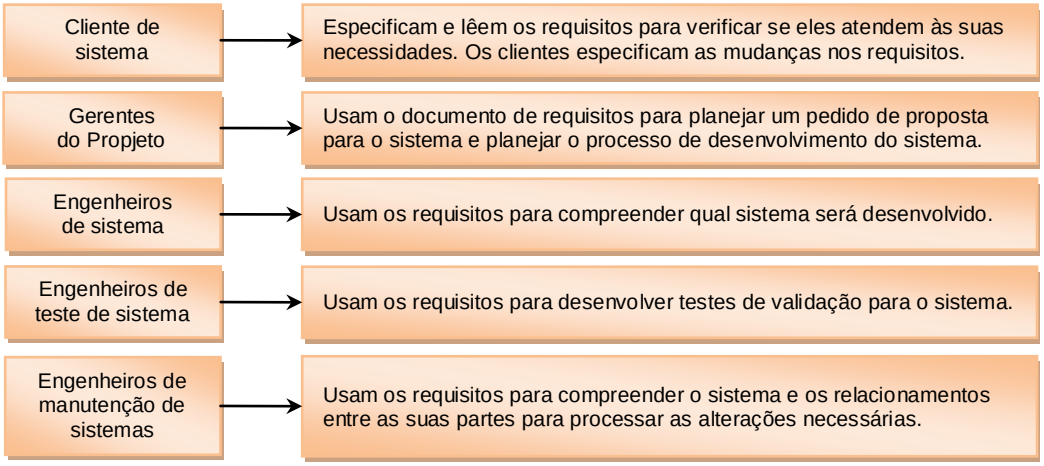

Figura 5.1 – Usuários de um documento de requisitos

A diversidade de possíveis usuários significa que o documento de requisitos precisa ser um compromisso entre a comunicação dos requisitos para os clientes, a definição dos requisitos em detalhes precisos para os desenvolvedores e testadores e a inclusão de informações sobre uma possível evolução do sistema. As informações sobre mudanças previstas podem auxiliar os projetistas a evitar decisões restritivas de projeto e auxiliar os engenheiros de manutenção de sistema que devem adaptar o sistema a novos requisitos. O nível de detalhamento a ser incluído em um documento de requisitos depende do tipo de sistema que está sendo desenvolvido e do processo de desenvolvimento usado. Quando o sistema for desenvolvido por um fornecedor externo, as especificações de sistema crítico devem ser precisas e muito detalhadas. Quando houver maior flexibilidade nos requisitos e quando um processo de desenvolvimento interno e iterativo for usado, o documento de requisitos pode ser muito menos detalhado e qualquer ambigüidade será resolvida durante o desenvolvimento do sistema.

Uma série de grandes organizações, como o Departamento de Defesa dos EUA e o IEEE[1], definiram padrões para documentos de requisitos. O padrão mais amplamente conhecido é IEEE/ANSI 830-1998. O padrão do IEEE sugere a seguinte estrutura para os documentos de requisitos:

[1] *Institute of Electrical and Electronics Engineers*

1. **Introdução**
   - 1.1. Propósito do documento de requisitos
   - 1.2. Escopo do produto
   - 1.3. Definições, acrônimos e abreviaturas
   - 1.4. Referências
   - 1.5. Visão geral do restante do documento
2. **Descrição geral**
   - 2.1. Perspectiva do produto
   - 2.2. Funções do produto
   - 2.3. Características dos usuários
   - 2.4. Restrições gerais
   - 2.5. Suposições e dependências
3. **Requisitos específicos**

   abrangem requisitos funcionais e não funcionais. Esta é, obviamente, a parte mais substancial no documento, mas, devido à grande variação na prática organizacional, não é apropriado definir uma estrutura-padrão para esta seção. Os requisitos devem documentar interfaces externas, descrever a funcionalidade e o desempenho do sistema, especificar requisitos lógicos de banco de dados, restrições de projeto, propriedades emergentes do sistema e características de qualidade.
4. **Apêndices**
5. **Índice**

Embora o padrão do IEEE não seja ideal, ele contém uma grande quantidade de boas recomendações de como redigir requisitos e evitar problemas. Ele é muito geral para funcionar como padrão de uma organização. É um *framework* geral que pode ser configurado e adaptado para definir um padrão dirigido às necessidades de determinada organização. A Tabela 5.1 ilustra uma possível organização para um documento de requisitos com base no padrão do IEEE. No entanto, o padrão foi estendido para incluir informações sobre a evolução prevista do sistema. Isso auxilia o pessoal de manutenção do sistema e permite que os projetistas incluam as bases para futuras características do sistema.

Naturalmente, as informações incluídas em um documento de requisitos devem depender do tipo de software que está sendo desenvolvido e da abordagem usada para o desenvolvimento. Se uma abordagem evolucionária for adotada para um produto de software, por exemplo, o documento de requisitos deixará de fora várias destas sugestões. O foco será a definição dos requisitos de usuário e dos requisitos não funcionais de alto nível do sistema.

Nesse caso, os projetistas e os programadores usam as suas avaliações para decidir como atender aos requisitos de usuários delineados para o sistema.

Por outro lado, quando o software é parte de um grande projeto de engenharia que inclui a interação de sistemas de hardware e de software, frequentemente é indispensável definir os requisitos em um nível refinado de detalhes. Isso significa que os documentos de requisitos serão, provavelmente, muito extensos e devem incluir a maior parte dos, se não todos, capítulos mostrados na Tabela 5.1. Para documentos extensos, é particularmente importante incluir uma tabela abrangente de conteúdo e índice do documento, de modo que os leitores possam encontrar as informações de que precisam.

Tabela 5.1 – Estrutura de um documento de requisitos

Capítulo	Descrição
Prefácio	Deve definir o público-alvo do documento e descrever o seu histórico de versões, incluindo uma justificativa lógica para a criação da nova versão e um resumo das mudanças feitas em cada versão.
Introdução	Deve descrever a necessidade do sistema. Deve descrever brevemente as suas funções e explicar como o sistema irá funcionar com outros sistemas. Deve descrever como o sistema atende aos objetivos gerais de negócios e estratégicos da organização que encomendou o software.
Glossário	Deve definir os termos técnicos usados no documento. Não se deve fazer suposições sobre a experiência ou as habilidades do leitor.
Definição de requisitos de usuário	Os serviços fornecidos ao usuário e os requisitos não funcionais do sistema devem ser descritos nesta seção. Essa descrição pode usar linguagem natural, diagramas e outras notações compreensíveis pelos clientes. Padrões de produto e de processo a serem seguidos devem ser especificados.
Arquitetura de sistema	Este capítulo deve apresentar uma visão geral de alto nível da arquitetura prevista do sistema, mostrando a distribuição das funções nos módulos do sistema. Os componentes de arquitetura reusados devem ser destacados.
Especificação de requisitos de sistema	Deve descrever os requisitos funcionais e não funcionais mais detalhadamente. Caso necessário, mais detalhes podem também ser adicionados aos requisitos não funcionais; por exemplo, interfaces com outros sistemas devem ser definidas.
Modelos de sistema	Deve estabelecer um ou mais modelos de sistema, mostrando os relacionamentos entre os componentes e o sistema e o seu ambiente. Podem ser modelos de objetos, modelos de fluxos de dados e modelos semânticos de dados.
Evolução de sistema	Deve descrever as hipóteses fundamentais sobre as quais o sistema está baseado, além de mudanças previstas devido à evolução do hardware, mudança das necessidades do usuário etc.
Apêndices	Deve fornecer informações detalhadas e específicas relacionadas à aplicação que está sendo desenvolvida. Exemplos de apêndices que podem ser incluídos são descrições de hardware e de banco de dados. Os requisitos de hardware definem as configurações mínima e ideal para o sistema. Os requisitos de banco de dados definem a organização lógica dos dados usados pelo sistema e os relacionamentos entre os dados.
Índice	Podem ser incluídos diversos índices para o documento. Assim como um índice alfabético normal, pode haver um índice dos diagramas, índice das funções etc.

Os documentos de requisitos são indispensáveis quando um fornecedor externo está desenvolvendo o sistema de software. No entanto, os métodos ágeis de desenvolvimento argumentam que os requisitos mudam tão rapidamente que um documento de requisitos fica desatualizado tão logo seja redigido, por isso o esforço é desperdiçado. Em vez de um documento formal, as abordagens como *extreme programming* propõem que os requisitos de usuário sejam coletados de maneira incremental e escritos em cartões. O usuário, então, prioriza os requisitos a serem implementados no próximo incremento do sistema. Para sistemas de negócios, nos quais os requisitos são instáveis, talvez esta seja uma boa abordagem. Contudo, é ainda útil escrever um documento breve de apoio que defina os requisitos de negócio e de confiança do sistema. É fácil se esquecer os requisitos que se aplicam ao sistema como um todo quando se enfoca os requisitos funcionais para a próxima versão do sistema.

Prof. Me. Marcelo dos Santos Moreira

- **Empresário**
- **Consultor de Empresas**
- **Professor Universitário**
- **Mestre em Informática**
  - **Gestão de Sistemas de Informação**
- **Especialista em Sistemas de Informatização Empresarial**
- **Bacharel em Administração**
- **Tecnólogo em Processamento de Dados**

marcelo.moreira@fatec.sp.gov.br