INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SÃO PAULO

CAMPUS SÃO PAULO

FABIO TOZETTO TAVEIRO

# ANÁLISE DO IMPACTO DE UM REQUISITO NÃO FUNCIONAL RELACIONADO A USABILIDADE

SÃO PAULO

2016

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SÃO PAULO

Fábio Tozetto Taveiro

# ANÁLISE DO IMPACTO DE UM REQUISITO NÃO FUNCIONAL RELACIONADO A USABILIDADE

Monografia apresentada como requisito para aprovação no Curso de Pós-graduação - Especialização em Gestão da Tecnologia da Informação do Instituto Federal de Educação, Ciência e Tecnologia de São Paulo, Campus São Paulo (IFSP-SP).

Orientador: Prof. Ms. José Oscar Machado Alexandre

SÃO PAULO

2016

T234a Taveiro, Fabio Tozetto.
Análise do impacto de um requisito não funcional relacionado a Usabilidade / Fabio Tozetto Taveiro. São Paulo: [s.n.], 2016.
103 f. : il.

Orientador: Prof. Me. José Oscar Machado Alexandre.

Monografia (Pós-graduação Lato Sensu em Gestão de Tecnologia da Informação) - Instituto Federal de Educação, Ciência e Tecnologia de São Paulo, IFSP, 2016.

1. Requisito não funcional 2. RNF 3. Usabilidade 4. Qualidade 5. Operabilidade I. Instituto Federal de Educação, Ciência e Tecnologia de São Paulo II. Título

CDU 004

## Dedicatória

Dedico este trabalho à minha esposa Elisangela,

aos meus filhos Gustavo e Gabriel e aos meus pais Pedro e Sônia.

# Agradecimentos

Agradeço principalmente a Deus, por ter me dado força e proteção nos momentos difíceis, sabedoria para escolher os caminhos corretos e também pelos momentos de alegria que tive durante esta caminhada.

Com atenção ao meu orientador Oscar, pela confiança, dedicação e profissionalismo durante o tempo que trabalhamos juntos. Por ter me oferecido a oportunidade de crescer pessoal e profissionalmente.

Especialmente à minha esposa Elisangela e aos meus filhos Gustavo e Gabriel pelo completo apoio e compreensão de ficarmos distantes durante o tempo que estive dedicando a este trabalho.

Pelos meus pais Pedro e Sonia que sempre me apoiaram e acreditaram em mim.

Aos mais que colegas de trabalho Rodrigo Peroni e Mauricio Lima que me apoiaram e que colaboraram na árdua tarefa de levantamento bibliográfico.

Aos professores e funcionários do IFSP.

Aos colegas de turma.

# Resumo

As características dos sistemas de software precisam ser conhecidas para que o produto resultante satisfaça as reais necessidades das partes interessadas. Os requisitos de sistema refletem essas necessidades, sendo estes classificados como requisitos funcionais e requisitos não funcionais. Os requisitos funcionais especificam o que o sistema de software deve ser capaz de realizar e o que não irá realizar. Já os requisitos não funcionais especificam atributos de qualidade desejados ou restringem o comportamento do software.

Para avaliar as características ou sub características de qualidade de forma quantitativa em projetos de software, pode-se utilizar modelos analíticos de qualidade para a medição de dados. Tais modelos ajudam a obter uma avaliação consistente da qualidade atual do produto. Foi adotado o modelo de qualidade da ISO/IEC definido pela série de normas chamada SQuaRe. As características de qualidade relacionadas a usabilidade foram avaliadas antes e após a execução de um dado projeto, no qual um dos principais requisitos do era a melhoria da usabilidade do produto através da reconstrução das telas operacionais.

Esse estudo de caso compara as métricas relacionadas a usabilidade com o objetivo de confirmar se a priorização do requisito não funcional de usabilidade no projeto em estudo influencia na melhora das características de qualidade chamada “Operabilidade” e “Usabilidade em Uso”. Ao realizar essa comparação, pretende-se realizar uma Prova de Conceito com o objetivo de comprovar que um requisito não funcional devidamente definido traz uma melhora direta na característica não funcional em questão.



**Palavras-chave**: Requisito, Não Funcional, RNF, Usabilidade, Qualidade, Operabilidade, Prova de Conceito.

# Abstract

The system software characteristics need to be known in the beginning of the development in order to satisfy the real stakeholder needs. The system requirements reflect these needs, being classified as functional requirements and non-functional requirements. Functional requirements define exactly what the software system is able to do and also what it will not. Non-functional requirements define the desired quality attributes and the restrictions on the software behavior.

In order to evaluate quantitatively the quality characteristics and sub characteristics for the software projects, quality analytic models can be used to measure the data. Those models help reaching to a consistent quality evaluation of the actual product. The ISO/IEC quality model defined by the SQuaRE standards was adopted. The quality characteristics related to the usability were evaluated before and after a project execution in which one of the main requirements was the usability improvement through the reconstructions all operational screens.

This case study compares the metrics related to the usability with the goal of confirming if a usability non-functional requirement prioritization influences in the quality characteristics called “Operability” and “Usability in Use”. Through this comparison, it is intended to perform a Proof of Concept with the objective of confirm that a non-functional requirement when properly defined will directly bring an improvement in the non-functional characteristic.



**Keywords**: Requirement, Non Functional, NFR, Usability, Quality, Operability, Proof of Concept.

# Lista de Ilustrações

# Lista de Abreviaturas e Siglas

| | |
|---|---|
| COTS | *Commercial off-the-shelf* |
| IEEE | *Institute of Electrical and Electronic Engineers* |
| IEC | *The International Electrotechnical Comission* |
| ISO | *International Organization for Standardization* |
| NFR | *Non functional requirement* |
| PoC | *Proof of concept* |
| PRS | *Product Requirements Specification* |
| RF | Requisito funcional |
| RNF | Requisito não funcional |
| SLA | *Service Level Agreement* |
| SQuaRE | *Systems and software Quality Requirements and Evaluation* |
| SRS | *Software Requirements Specification* |

# Sumário

# 1 INTRODUÇÃO

Desde o início do processo de desenvolvimento, as características dos sistemas de software devem ser conhecidas para que o produto resultante satisfaça as reais necessidades das partes interessadas (CHUNG, NIXON, *et al.*, 2000).

Os requisitos de um sistema é o conjunto de descrições do que o sistema deve fazer, dos serviços que sistema deve prover e das restrições de sua operação. Esses requisitos refletem as necessidades dos clientes para o sistema que serve um dado propósito como por exemplo controlar um dispositivo, fazer um pedido ou encontrar uma informação (SOMMERVILLE, 2011).

Diferentes níveis de requisitos são úteis para comunicar a informação sobre o sistema para diferentes tipos de interessados. Alguns dos problemas do processo de engenharia de requisitos são causados porque não existe uma separação clara entre os diferentes níveis de descrição, como por exemplo os “requisitos de usuário” e “requisitos de sistema” (SOMMERVILLE, 2011).

Existem dois tipos de requisitos: os requisitos funcionais e os requisitos não funcionais. Os requisitos funcionais são aqueles que o sistema, o sistema de software ou o componente de sistema deve ser capaz de executar (CHUNG, NIXON, *et al.*, 2000). Em alguns casos os requisitos funcionais podem também estabelecer explicitamente o que o sistema não irá fazer (SOMMERVILLE, 2011). Os requisitos funcionais são algumas vezes chamados de requisitos comportamentais ou operacionais porque eles especificam as entradas (estímulos) do sistema, as saídas (respostas) do sistema, e as relações de comportamento entre eles. Eles provêm uma análise detalhada dos dados que o sistema irá manipular, e pode incluir uma definição detalhada das interfaces do usuário para o sistema (YOUNG, 2004).

Os requisitos não funcionais são restrições nos serviços e nas funções oferecidas pelo sistema e, por exemplo, podem incluir restrições de tempo, restrições no processo de desenvolvimento, e restrições impostas por normas (SOMMERVILLE, 2011). Os requisitos não funcionais de software em geral se relacionam com padrões de qualidade e são importantes pois definem se o software será eficiente e adequado para a tarefa que se propõe a fazer (XAVIER, 2009),.

A qualidade pode ser definida a partir de duas visões principais que se complementam. A primeira é “Atender as especificações”, e significa produzir

produtos ou entregar serviços que tenham características mensuráveis e que atendam um conjunto fixo de especificações definidas numericamente. A segunda é “Satisfazer o cliente”, e significa satisfazer as expectativas de uso e consumo do cliente, independentemente de qualquer característica mensurável (HOYER e HOYER, 2001). Já a qualidade de um software pode ser definida como a conformidade com os requisitos funcionais e de performance explicitamente definidos e documentados pelos padrões de desenvolvimento, e as características implícitas que são esperadas por todos os profissionais de desenvolvimento de software (PRESSMAN, 2014).

A qualidade de um sistema é resultado da qualidade individual de cada elemento pertencente ao sistema e a suas interações entre si. A qualidade de software é o grau de satisfação de um produto de software em relação às necessidades declaradas e implícitas dentro de condições específicas de utilização por um usuário (ISO/IEC25010, 2011).

Os modelos de qualidade permitem uma avaliação quantitativa de características ou sub características de qualidade baseados na medição de dados em projetos de software. Esses modelos podem ajudar a obter a avaliação da qualidade atual do produto, em contraste com o modo subjetivo de avaliação baseado no julgamento e na avaliação qualitativa imprecisa (SAMASHIYA e WANG, 2010).

Um dos modelos de qualidade com maior relevância atualmente é o modelo SQuaRE criado pela ISO/IEC. Este modelo é composto dos seguintes elementos: Modelo de Qualidade do Produto de Software (Qualidade interna do software e Qualidade externa do software), Modelo de Qualidade dos Dados e Modelo de Qualidade do Sistema Em Uso (ISO/IEC25010, 2011).

A usabilidade do software é uma característica comum dos modelos de qualidade e se tornou material de estudo e de interesse do governo e de organizações de grande e pequeno porte. Com o crescimento considerável das aplicações distribuídas, aumentou-se a dificuldade para que os desenvolvedores de software acessem diretamente os usuários finais. A usabilidade do software se tornou então um meio determinante para a produtividade e para a aceitação do produto de software pelo usuário final (ABRAN, KHELIFI e SURYN, 2003).

Será utilizada como base de pesquisa uma empresa multinacional de grade porte no setor de automação, e que possui unidades distribuídas em diversos países com foco no desenvolvimento de hardware e de software, na produção fabril e na manutenção de equipamentos. O produto analisado é um sistema de monitoração de uma rede de equipamentos e é utilizado atualmente por algumas unidades da Organização que fornecem o serviço de manutenção de equipamentos para clientes baseados em SLA.

Será analisado a característica de qualidade chamada usabilidade, para as situações anterior e posterior ao Projeto A. O Projeto A tem como objetivo adicionar novas funcionalidades e corrigir problemas encontrados nas versões anteriores do produto. A principal nova funcionalidade do Projeto A é a reconstrução das telas operacionais do produto com o objetivo de fornecer uma melhor usabilidade para os usuários.

## 1.1 QUESTÃO DE PESQUISA

Considerando que nos projetos relacionados ao produto em estudo normalmente são priorizados somente requisitos funcionais, este estudo pretende responder a seguinte questão:

Em um produto aonde normalmente são priorizados somente requisitos funcionais, a priorização de um requisito não funcional relacionado a usabilidade reflete alguma melhoria prática no produto final?

## 1.2 OBJETIVOS DO TRABALHO

O principal objetivo deste estudo é responder à questão de pesquisa, comparando a usabilidade do produto nas situações anterior e posterior a execução do projeto sob análise.

Como complemento ao objetivo principal, com este estudo espera-se atingir os seguintes objetivos secundários:

- Realizar uma Prova de Conceito, confirmando que um requisito não funcional de usabilidade, quando claramente especificado, leva a uma melhora direta na usabilidade percebida pelos usuários.
- Realizar um levantamento bibliográfico partindo de requisitos e indo até as métricas de usabilidade;
- Medir o valor percentual efetivo de melhoria para a usabilidade após a execução do projeto em análise;
- Propor uma maneira alternativa para a análise da usabilidade do produto baseado na percepção dos usuários.

## 1.3 JUSTIFICATIVA

De acordo com uma pesquisa realizada por (IEEE, 2005), aproximadamente um trilhão de dólares é gasto mundialmente por ano com TI (Tecnologia da Informação) em hardware, software e serviços. Dos projetos de TI, até 15% dos projetos que são iniciados são abandonados antecipadamente devido a serem considerados inadequados. Ainda, até 50% do tempo de desenvolvedores de software é gasto com retrabalhos que poderiam ser evitados se as tarefas fossem feitas corretamente na primeira vez. Nesse estudo são apresentados os 12 maiores motivos de falha, sendo que 3 podem de alguma forma ter relação com fatores que influenciam na usabilidade: requisitos de software mal definidos; pobre comunicação entre clientes, desenvolvedores e usuários; e política entre as partes interessadas.

Ainda, em um estudo de caso feito dentro da IBM, (KAPLAN, CLARK e TANG, 1995) demonstraram o custo comparativo de se encontrar um dado erro em cada uma das fases de um projeto de software (Análise e definição de requisitos, Projeto (*Design*), Codificação, Testes de Desenvolvimento, Testes de Sistema e Operação em Campo). O resultado desta pesquisa mostra que o custo da correção aumenta de forma exponencial caso um erro “vaze” para as próximas etapas do desenvolvimento.

Segundo o (PMI, 2013), os benefícios primários em atender os requisitos de qualidade incluem menos retrabalhos, maior produtividade, custos menores aumento da satisfação das partes interessadas e aumento na lucratividade. A análise de custo-benefício para cada atividade de qualidade compara o custo do passo da

qualidade com o benefício esperado. O custo da qualidade inclui todos os custos ocorridos durante a vida do produto através do investimento em prevenir uma não conformidade com os requisitos, em avaliar se o produto tem conformidade com os requisitos e em falhar ao atender os requisitos (o que na prática significa retrabalho).

Os requisitos não funcionais normalmente não são definidos nos projetos, o que compromete diretamente a qualidade dos produtos. Este trabalho visa analisar o impacto da priorização de um requisito não funcional relacionado a usabilidade em um produto aonde os requisitos não funcionais normalmente não são nem elicitados e muito menos priorizados.

## 1.4 ESTRUTURA DO TRABALHO

No segundo capítulo são apresentados os elementos teóricos sobre os assuntos tratados nesse estudo e que foram considerados relevantes. No terceiro capítulo, são apresentados os métodos de pesquisa utilizados, como foi realizada a coleta dos dados e como os dados foram analisados. No quarto capítulo são apresentados os resultados da pesquisa e uma discussão sobre cada um dos aspectos analisados. O sexto capítulo apresenta as considerações finais, através das conclusões, das contribuições deste trabalho e dos trabalhos futuros que podem ser desenvolvidos.

# 2 REVISÃO BIBLIOGRÁFICA

Este capítulo tem como objetivo apresentar os elementos teóricos sobre os assuntos tratados nesse estudo, ou seja, sistemas de informação, requisitos de software, qualidade, normas ISO/IEC para a qualidade e usabilidade. Cada assunto foi separado em tópicos específica, sendo no total 5 tópicos.

## 2.1 SISTEMAS DE INFORMAÇÃO

As organizações empresariais são compostas de maneira hierárquica distribuídos em três níveis principais, como representado na Figura 2.1. Essa divisão é detalhada a seguir e é baseada nas atividades administrativas aonde as tarefas são realizadas (LAUDON e LAUDON, 2007):

- No nível operacional, a gerência operacional é responsável pelo monitoramento das atividades diárias. A decisão no nível operacional é um processo que deve assegurar que as atividades operacionais sejam bem desenvolvidas, e geralmente resulta em uma resposta imediata.
- No nível tático, a gerência média conduz taticamente os planos e programas definidos pela gerência sênior. As decisões no nível tático são normalmente relacionadas com o controle administrativo e são utilizadas para decidir sobre a designação de recursos, sobre as operações de controle e formular novas regras de decisão que serão aplicadas na operação e na designação de recursos.
- No nível estratégico, a gerência sênior toma as decisões estratégicas de longo prazo a respeito de produtos e serviços, além de garantir o desempenho financeiro da empresa


**Figura 2.1 - Níveis hierarquicos das organizações empresáriais**

Um sistema é um conjunto de partes que se interagem e dependem entre si, e que conjuntamente satisfazem um dado objetivo e através de uma determinada função. Um sistema de informação gerencial é um processo de transformação de dados em informações que são utilizadas na estrutura decisória da empresa, proporcionando uma sustentação administrativa para otimizar os resultados esperados (OLIVEIRA, 2012).

Existem diversos sistemas de informação gerenciais e estes são utilizados em cada um dos níveis citados. Quando esses sistemas são bem construídos e integrados, eles permitem com que os dados tenham a qualidade necessária e que as informações estejam disponíveis de acordo com o perfil de interesse e a necessidade dos executivos da empresa (OLIVEIRA, 2012).

O sistema utilizado nesse estudo de caso é utilizado no nível operacional da organização.

## 2.2 REQUISITOS DE SOFTWARE

### 2.2.1 Engenharia de Requisitos

Os requisitos para um sistema é o conjunto de descrições do que o sistema deve fazer, dos serviços que sistema deve prover e das restrições de sua operação. Esses requisitos refletem as necessidades dos clientes para o sistema que serve um dado propósito como por exemplo controlar um dispositivo, fazer um pedido ou encontrar uma informação. O processo de descobrir, analisar documentar e verificar esses serviços e restrições é chamado de Engenharia de Requisitos (SOMMERVILLE, 2011).

> Requisito é uma declaração que identifica um produto ou um processo operacional ou funcional, ou característica de projeto, ou uma restrição, que é inequívoca, testável ou mensurável, e necessário para o produto ou processo de aceitação (por consumidores ou pelas diretrizes internas de garantia de qualidade) (IEEE, 1998).

Segundo (SOMMERVILLE, 2011), diferentes níveis de requisitos são úteis por que eles comunicam a informação sobre o sistema para diferentes tipos de interessados. Alguns dos problemas do processo de engenharia de requisitos são causados porque não existe uma separação clara entre os diferentes níveis de descrição. O autor define dois níveis utilizando os termos “requisitos de usuário” e “requisitos de sistema”. Os requisitos de usuário são declarações das características esperadas que o sistema provenha para os usuários e as restrições na qual o sistema deve operar. Essas declarações podem ser feitas em uma linguagem natural e/ou através de diagramas. Os requisitos de sistema são descrições mais detalhadas das funções, dos serviços e das restrições operacionais do sistema de software. Os requisitos de sistema devem definir exatamente o que deve ser implementado, e pode ser considerado como parte do contrato entre o comprador do sistema e os desenvolvedores de software.

De acordo com (POHL e RUPP, 2015), existem quatro atividades principais na Engenharia de Requisitos. Estes estão detalhados abaixo e representados na Figura 2.2.

- Elicitação: Durante a elicitação de requisitos, diferentes técnicas são utilizadas para obter os requisitos das partes interessadas (e outras fontes) e para refinar os requisitos com um maior detalhamento;
- Documentação: Durante a documentação, os requisitos elicitados são descritos adequadamente. Diferentes técnicas são utilizadas para documentar os requisitos através do uso de linguagens naturais ou de modelos conceituais;
- Validação e Negociação: Com o objetivo e garantir que os critérios de qualidade sejam atendidos, os requisitos documentados devem ser validados e negociados o mais cedo possível;
- Gerenciamento: O gerenciamento dos requisitos é uma atividade ortogonal às outras atividades, e compreende qualquer providência que é necessária para a estruturação dos requisitos, a preparação para ser utilizado por diferentes funções, mantê-los consistentes após as mudanças, e garantir a sua implementação.


**Figura 2.2 - Principais atividades da Engenharia de Requisitos (POHL e RUPP, 2015)**

Os possíveis interessados para os requisitos de usuário e de sistema estão representados na Figura 2.3. Os interessados dos requisitos de usuário normalmente não estão preocupados em como o sistema será implementado. Os interessados dos requisitos de sistema precisam saber precisamente o que o sistema irá fazer porque eles estão preocupados em como eles irão apoiar o processo de negócio ou porque eles estão envolvidos na implementação do sistema. (SOMMERVILLE, 2011).

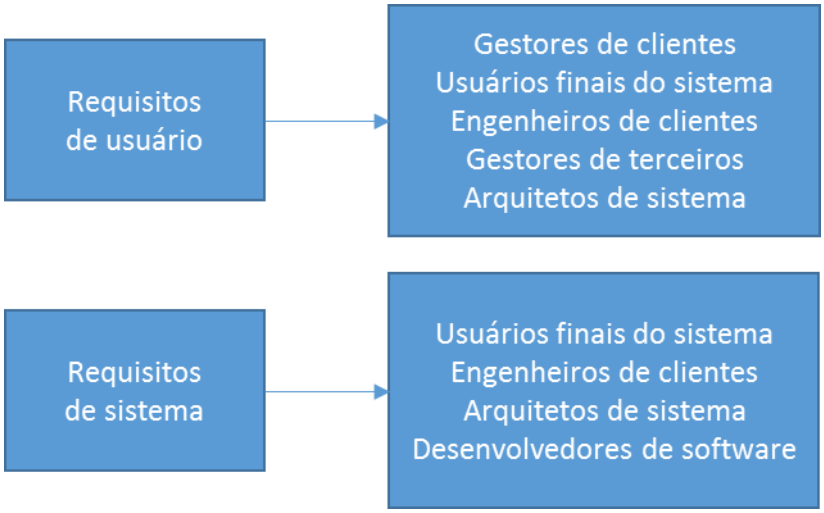


**Figura 2.3 - Interessados dos diferentes tipos de requisitos (SOMMERVILLE, 2011)**

### 2.2.2 Requisitos Funcionais e Requisitos Não Funcionais

Os requisitos de software são frequentemente classificados como requisitos funcionais (RFs) e como requisitos não funcionais (RNFs). Os requisitos funcionais são aqueles que descrevem os serviços que o sistema deve ter, como o sistema deve reagir a entradas específicas, e como o sistema deve se comportar em situações particulares. Em alguns casos os RFs podem também estabelecer explicitamente o que o sistema não irá fazer (SOMMERVILLE, 2011).

Segundo (YOUNG, 2004), os RFs descrevem o que o sistema ou software deve fazer. Uma função é uma capacidade útil provida por um ou mais componentes do sistema. Os RFs são algumas vezes chamados de requisitos comportamentais ou operacionais porque eles especificam as entradas (estímulos) do sistema, as saídas (respostas) do sistema, e as relações de comportamento entre eles. Os RFs é uma coleção compreensiva das características do sistema e as capacidades que estão disponíveis para os usuários. Ele provê uma análise detalhada dos dados que o sistema irá manipular, e pode incluir uma definição detalhada das interfaces do usuário para o sistema.

Os RNFs são restrições nos serviços e nas funções oferecidas pelo sistema e, por exemplo, podem incluir restrições de tempo, restrições no processo de desenvolvimento, e restrições impostas por normas (SOMMERVILLE, 2011).

De acordo com (XAVIER, 2009), os RNFs de software em geral se relacionam com padrões de qualidade e são importantes pois definem se o software será eficiente e adequado para a tarefa que se propõe a fazer.

Durante o levantamento dos RFs e dos RNFs, normalmente se tem oportunidade de analisar somente os impactos entre os requisitos do mesmo tipo. Após a definição dos requisitos, (CYSNEIROS, LEITE e NETO, 2001) recomenda que haja uma análise para a integração dos RFs e dos RNFs. Durante esse processo, alguns impactos poderão aparecer, e será necessário analisar cada um deles, porém ao final tem-se os requisitos mais completos e consolidados como representa a Figura 2.4.

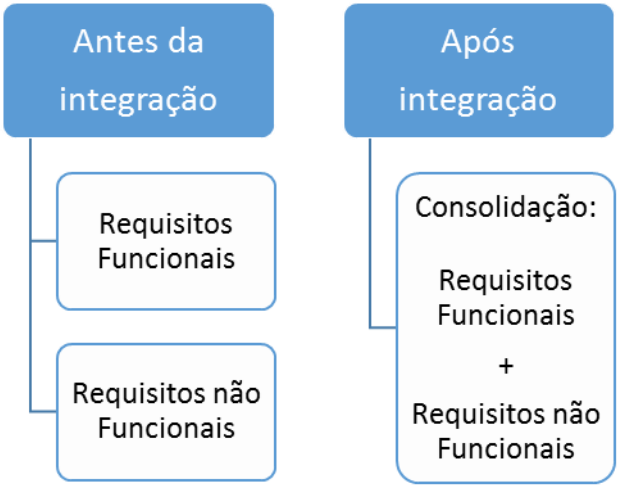


**Figura 2.4 - Integração entre RFs e RNFs (CYSNEIROS, LEITE e NETO, 2001)**

A complexidade se um sistema de software é determinada parcialmente por suas funcionalidades e parcialmente por seus requisitos não funcionais, tais como custos operacionais, desempenho, confiabilidade, manutenibilidade, portabilidade, robustez, entre outros. Os requisitos não funcionais possuem um papel crítico durante o desenvolvimento de sistemas pois ajudam na seleção de critérios para a escolha da maneira mais indicada de projeto e de implementação (CHUNG, NIXON, *et al.*, 2000).

A parte mais difícil em se construir um sistema de software é decidir precisamente o que deve ser construído. Nenhuma parte do trabalho conceitual é mais complexa do que estabelecer os requisitos detalhados incluindo todas as interfaces com as pessoas, máquinas, e outros sistemas de software. Os erros de requisitos têm grande impacto quando eles são identificados tardiamente, e a sua correção são as mais difíceis de se fazer (BROOKS, 1987).

Um problema comum com RNFs é que frequentemente os usuários ou os clientes propões esses requisitos como metas genéricas, como por exemplo a facilidade de uso, a habilidade do sistema em se recuperar de falhas ou uma rápida resposta para o usuário. Essas metas são definidas com boa intenção, mas na prática elas causam problemas para os desenvolvedores de sistema uma vez que elas deixam o escopo aberto para a interpretação e para uma possível controvérsia quando o sistema é entregue (SOMMERVILLE, 2011).

Durante o projeto de um sistema de software, quando se decide satisfazer um RNF, isso pode trazer conflitos com outros RFs e RNFs. De acordo com (CYSNEIROS, LEITE e NETO, 2001), se esses conflitos não forem analisados e tratados adequadamente, isto pode resultar em uma série de problemas durante a implementação. A identificação e a declaração apropriada dos RNFs são essenciais para o entendimento e a análise do impacto nas decisões de projeto.

Sempre que possível, os RNFs devem ser escritos de maneira a expressar requisitos não funcionais testáveis, ou seja, de maneira quantitativa, o que permitirá com que o requisito seja objetivamente verificado e testado (SOMMERVILLE, 2011). A Figura 2.5 apresenta algumas métricas que podem ser utilizadas para especificar as propriedades não funcionais do sistema. As medidas dessas características podem ser medidas quando o sistema é testado de maneira a confirmar se o sistema atende ou não os RNFs originalmente definidos.

| Propriedade | Medida |
|---|---|
| **Velocidade** | Transações processadas por segundo<br>Tempo de resposta<br>Tempo de atualização de tela |
| **Tamanho** | Número de Megabytes |
| **Facilidade de Uso** | Tempo de treinamento<br>Número de telas de ajuda |
| **Confiabilidade** | Tempo média entre falhas<br>Probabilidade de indisponibilidade<br>Taxa de ocorrência de falhas<br>Disponibilidade |
| **Robustez** | Tempo para reiniciar após falha<br>Porcentagem de eventos causando falhas<br>Probabilidade de corromper dados quando ocorre uma falha |
| **Portabilidade** | Número de declarações dependentes do alvo<br>Número de sistemas alvo |

**Figura 2.5 - Exemplo de métricas para especificar RNFs (SOMMERVILLE, 2011)**

### 2.2.3 RNFs segundo Chung

Segundo (CHUNG, NIXON, *et al.*, 2000) , os RNFs endereçam pontos importantes da qualidade de sistemas de software e são vitais para o seu sucesso. Quando os RNFs não são considerados pode-se ter maus resultados, como por exemplo: software inconsistente e de baixa qualidade; usuários, clientes e desenvolvedores insatisfeitos; e muito retrabalho, o que resulta em maior tempo de projeto e maior custo. Os RNFs têm as seguintes características (Figura 2.6):

- Subjetividade – Os RNFs podem ser vistos, interpretados e avaliados de forma diferente por diferentes pessoas. Isto é impulsionado pelo fato dos NFRs serem normalmente definidos de forma superficial e vaga.
- Relatividade – A interpretação e a importância dos RNFs podem variar dependendo do sistema a ser considerado. Não existe uma única solução que satisfaça todos os casos.
- Interatividade – Ao atender um RNF pode-se ferir ou beneficiar o atendimento de outros RNFs. Soluções localizadas podem não ser suficientes, uma vez que os RNFs têm impactos em todos os sistemas.

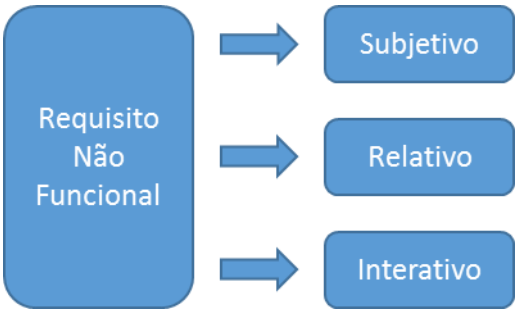


**Figura 2.6 - Características dos RNFs (CHUNG, NIXON, *et al.*, 2000)**

Como os RNFs são subjetivos, relativos e interativos, não é possível definir de maneira clara e óbvia se um RNF foi atendido ou satisfeito. Portanto o conceito de meta ou *goal* não é adequado para representar um RNF. Neste contexto, foi introduzido o conceito de *softgoal* para representar um RNFs, o qual pode ser suficientemente atendido (satisfeito até um certo nível), mas de forma não absoluta.

Não existe uma definição formal ou lista completa de RNFs, assim como não existe um esquema universal de classificação que acomode todas as diferentes necessidades e diferentes situações (CHUNG, NIXON, *et al.*, 2000).

RNFs cobrem uma grande variedade de questões da qualidade de um software. A Figura 2.7 lista, de maneira não categorizada, uma série de áreas possíveis de serem tratadas pelos RNFs. O autor não se preocupa em criar uma taxonomia para os RNFs pois não considera este o foco do seu trabalho, e por que existem uma série de outras literaturas que tratam dessa questão.

(CHUNG, NIXON, *et al.*, 2000) focam em definir uma maneira sistemática e racional de se lidar com os RNFs denominado NFR Framework. A análise de um RNF utilizando este framework aborda um refinamento, um levantamento de correlação e uma maneira de operacionalizar. Este processo ajuda a decompor os RNFs e lida com ambiguidades e outros problemas.

*Accessibility, accountability, accuracy, adaptability, additivity, adjustability, affordability, agility, auditability, availability, buffer space performance, capability, capacity, clarity, code-space performance, cohesiveness, commonality, communication cost, communication time, compatibility, completeness, component integration cost, component integration time, composability, comprehensibility, conceptuality, conciseness, confidentiality, configurability, consistency, controllability, coordination cost, coordination time, correctness, cost, coupling, customer evaluation time, customer loyalty, customizability, data-space performance, decomposability, degradation of service, dependability, development cost, development time, distributivity, diversity, domain analysis cost, domain analysis time, efficiency, elasticity, enhanceability, evolvability, execution cost, extensibility, external consistency, fault-tolerance, feasibility, flexibility, formality, generality, guidance, hardware cost, impact analyzability, independence, informativeness, inspection cost, inspection time, integrity, internal consistency, inter-operability, intuitiveness, learn ability, main-memory performance, maintainability, maintenance cost, maintenance time, maturity, mean performance, measurability, mobility, modifiability, modularity, naturalness, nomadicity, observability, off-peak-period performance, operability, operating cost, peak-period performance, perform ability, performance, planning cost, planning time, plasticity, portability, precision, predictability, process management time, productivity, project stability, project tracking cost, promptness, prototyping cost, prototyping time, reconfigurability, recoverability, recovery, reengineering cost, reliability, repeatability, replaceability, replicability, response time, responsiveness, retirement cost, reusability, risk analysis cost, risk analysis time, robustness, safety, scalability, secondary-storage performance, security, sensitivity, similarity, simplicity, software cost, software production time, space boundedness, space performance, specificity, stability, standardizability, subjectivity, supportability, surety, survivability, susceptibility, sustainability, testability, testing time, throughput, time performance ,timeliness, tolerance, traceability, trainability, transferability, transparency, understandability, uniform performance, uniformity, usability, user-friendliness, validity, variability, verifiability, versatility, visibility, wrappability*

**Figura 2.7 - Lista de RNF segundo (CHUNG, NIXON, et al., 2000)**

### 2.2.4 RNFs segundo a norma IEEE 830-1998

A norma IEEE 830-1998 tem como objetivo descrever uma abordagem prática para a especificação de requisitos de software. Ela é baseada em um modelo no qual o resultado esperado do processo de especificação de requisitos de software é um documento inequívocos e completo (IEEE, 1998).

Não existe uma definição do termo RNF bem como não existe uma taxonomia claramente especificada. No decorrer da norma são citadas quatro áreas relacionadas com os RNFs: interfaces externas, performance, atributos e restrições de projeto impostos pela implementação. Na Figura 2.8 pode-se observar essas quatro áreas um pouco mais detalhadas.

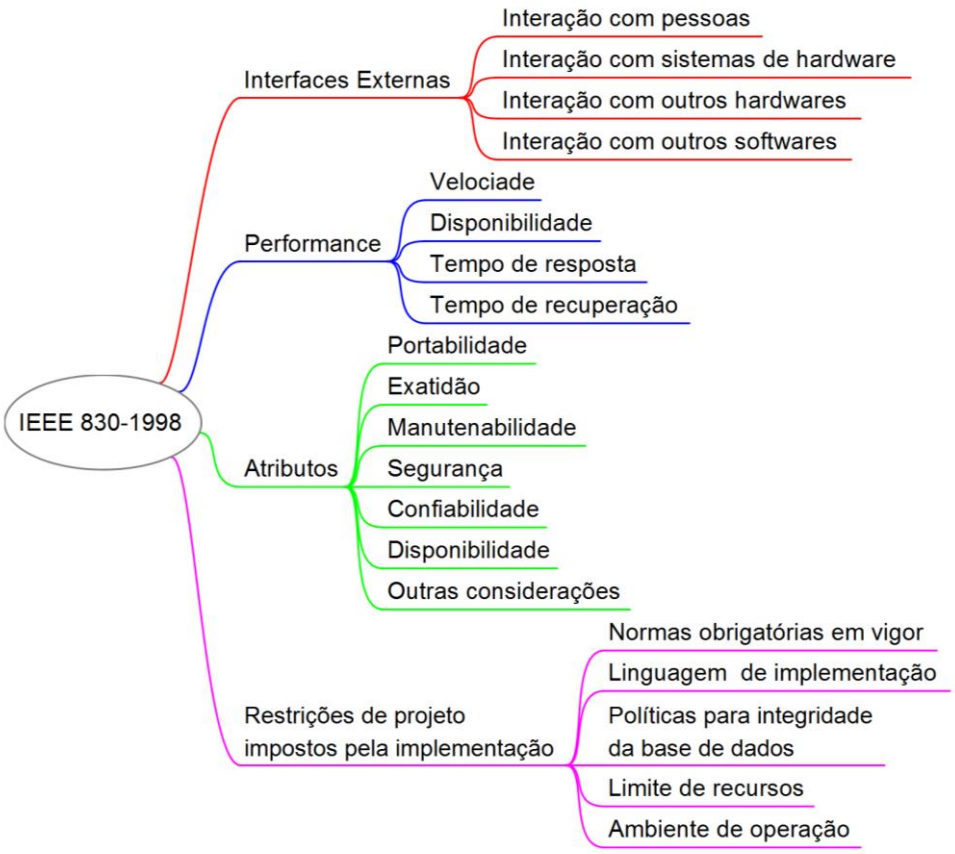


**Figura 2.8 - Requisitos não funcionais exemplificados por (IEEE, 1998)**

Em relação ao documento de especificação, a norma IEEE recomenda explicitamente especificar somente o produto de software e não especificar qualquer item relacionado ao processo de produzir o software. Os requisitos de projeto representam um acordo entre o cliente e o fornecedor sobre assuntos contratuais da produção de software, e estes devem ser especificados em outros documentos com por exemplo o plano de desenvolvimento de software e o plano de garantia de qualidade. Os requisitos de projeto normalmente incluem:

- Custo
- Prazos de entrega
- Procedimentos de report
- Métodos de desenvolvimento de software
- Garantia de qualidade
- Critérios de validação e de verificação
- Procedimentos de aceitação

### 2.2.5 RNFs segundo Sommerville

Segundo (SOMMERVILLE, 2011), os requisitos não funcionais surgem das necessidades dos usuários, devido a restrições de orçamento, politicas organizacionais, necessidade de interoperabilidade com outros sistemas de software ou hardware, ou fatores como normas de segurança ou legislação. A Figura 2.9 representa a taxonomia para requisitos não funcionais definida por (SOMMERVILLE, 2011) e que em um primeiro nível de classificação é dividida nas características necessárias para o software (requisitos de produto), na organização do desenvolvimento do software (requisitos organizacionais) ou de origem de fontes externas.

Os requisitos de produto são requisitos que especificam ou restringem o comportamento do software. Essa categoria inclui itens como por exemplo:

- Requisitos de performance - quão rápido o sistema deve executar e de quanta memória é necessária;
- Requisitos de confiabilidade – estabelecem uma taxa aceitável de falha;
- Requisitos de segurança;

- Requisitos de usabilidade.

Os requisitos organizacionais são requisitos que são derivados dos requisitos de sistema a partir de políticas e procedimentos do cliente e da organização de desenvolvimento. Essa categoria inclui itens como por exemplo:

- Requisitos operacionais de processo – definem como o sistema irá ser usado;
- Requisitos de processo de desenvolvimento – especificam as linguagens de desenvolvimento, o ambiente de desenvolvimento ou padrões de desenvolvimento a serem utilizados;
- Requisitos de ambiente – especificam o ambiente de operação do sistema.

Os requisitos externos têm ampla amplitude. Essa categoria abrange todos os requisitos que são derivados de fatores externos ao sistema e ao seu processo de desenvolvimento. Isso pode incluir:

- Requisitos regulatórios - estabelecem o que deve ser feito para o sistema ser aprovado para uso por um órgão regulatório (como por exemplo o Banco Central);
- Requisitos legislativos - devem ser seguidos para garantir que o sistema funcione dentro da lei;
- Requisitos éticos - garantem que o sistema seja aceitável para seus usuários e para o público em geral.

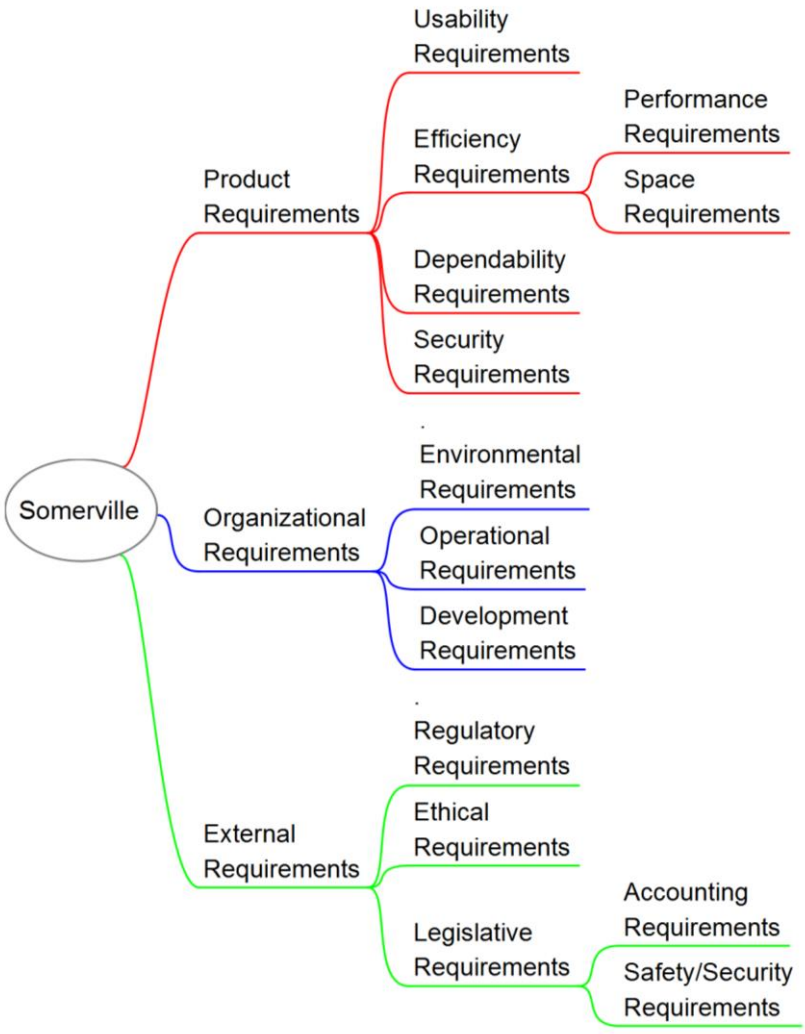


**Figura 2.9 - Tipos de requisitos não funcional (SOMMERVILLE, 2011)**

### 2.2.6 RNFs segundo Volere

Volere é um processo genérico para tratar requisitos e um modelo para a especificação de requisitos desenvolvido por (ROBERTSON e ROBERTSON, 1995-2016). A partir da experiência prática de vários anos trabalhando em projetos e dando consultoria para os seus clientes, os autores criaram um processo de levantamento e especificação de requisitos que contém princípios que podem ser aplicados em praticamente todos os tipos de aplicação e em todos os ambientes de desenvolvimento.

A taxonomia dos RNFs proposta por Volere (ROBERTSON e ROBERTSON, 2012) pode ser verificado com mais detalhes na Figura 2.10, e em resumos considera as seguintes áreas:

- Aparência: trata da alma da aparência do produto;
- Usabilidade e Humanidade: trata da facilidade de uso do produto e qualquer consideração especial de uso necessária para uma melhor experiência de usuário;
- Performance: trata o quão rápido, o quão protegido, quantas vez, o quão disponível, o quão preciso uma funcionalidade precisa ser;
- Operacionalidade: trata o ambiente operacional do produto, e qualquer consideração que deve se ter por conta do ambiente;
- Manutenção e Suporte: trata das mudanças necessárias, e do tempo necessário para realizar essas mudanças. Também especifica o suporte a ser dado para o produto;
- Segurança: trata da segurança (estar livre de perigo), da confidencialidade e da recuperabilidade do produto;
- Cultural e Política: trata de requisitos especiais que podem surgir por causa da cultura e customizações para as pessoas envolvidas na operação do produto;
- Legal: trata das leis e normas que são aplicadas ao produto.

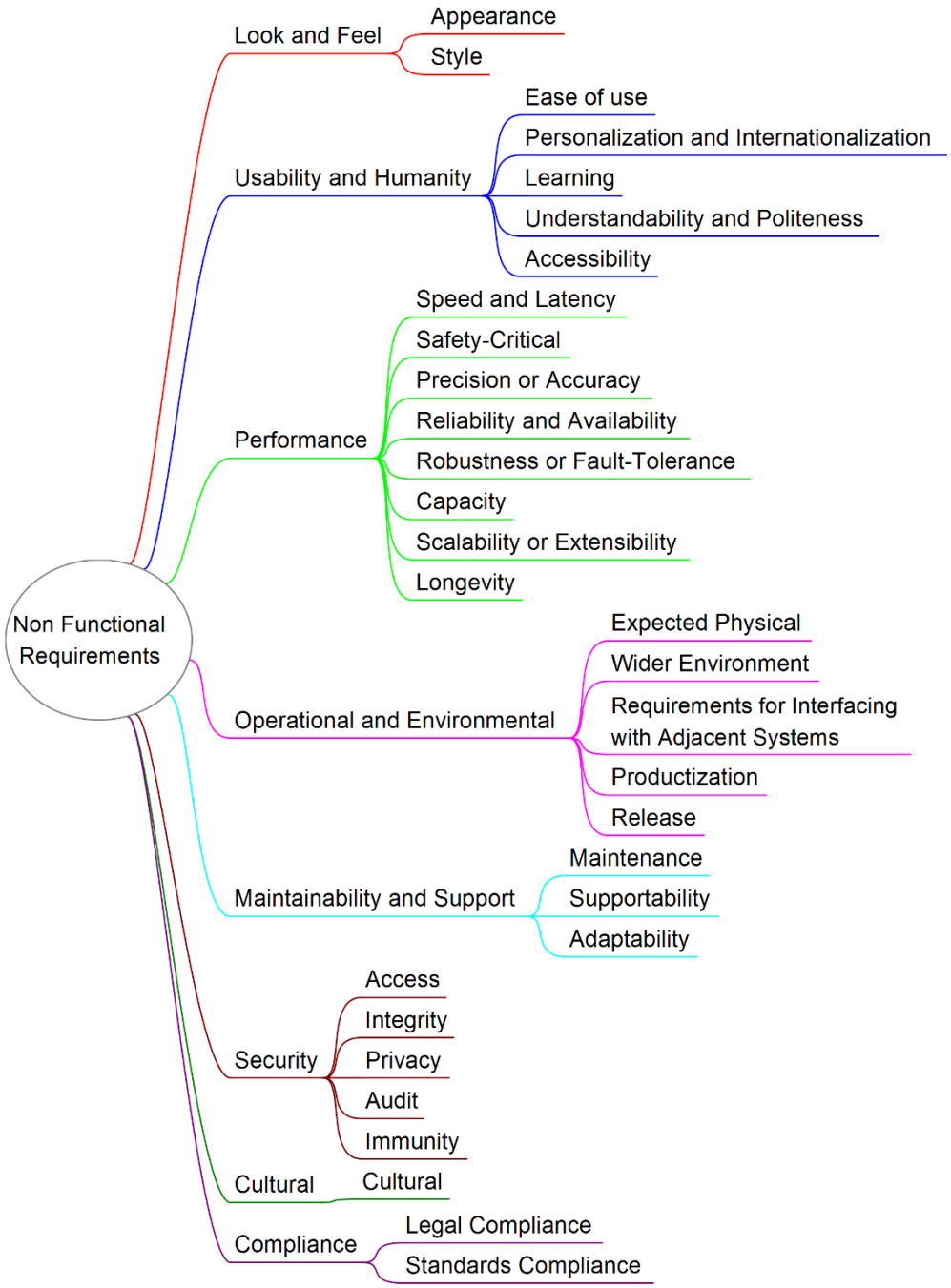


**Figura 2.10 - Taxonomia dos RNFs segundo Volere**

### 2.2.7 Pesquisa sobre o uso dos RNFs na especificação de produtos

Em uma pesquisa realizada em 2012 na Espanha chegou-se à conclusão que na prática a qualidade relacionada aos RNFs não é considerada como fonte primária de requisitos na arquitetura de um software (BUSCHMANN, AMELLER, *et al.*).

O método utilizado foi a pesquisa qualitativa e exploratória com entrevistas estruturadas e com foco em um projeto por entrevistado. Foram entrevistados treze arquitetos de software que trabalham na Espanha e cobrindo diferentes tipos de aplicações. Os projetos tinham diferenças consideráveis em funcionalidades, nos tamanhos e nas equipes envolvidas. As questões e os resultados obtidos podem ser verificados na Figura 2.11.

| Questões | Resultados |
|---|---|
| Quais são os tipos de RNFs relevantes para a arquitetura de software? | A Figura 2.12 mostra importância de diferentes tipos de RNFs percebida pelos entrevistados. A figura mostra o número de vezes que cada RNF foi citado. Cada item foi classificado como restrição ou como um atributo de qualidade. |
| Como os RNFs são levantados? | Com a ajuda do arquiteto de software, os clientes participaram do levantamento dos RNFs em somente três projetos. Nos outros dez projetos, o próprio arquiteto fez todo o levantamento dos RNFs. |
| Como os RNFs são documentados? | Nove arquitetos não documentaram os RNFs. Outros três utilizam algum tipo de template como por exemplo o Volere. Em um caso, o arquiteto documentou em texto livre. |
| Como os RNFs são validados? | Onze arquitetos afirmaram que os RNFs foram atendidos no final do projeto. A validação foi restrita a poucos tipos, principalmente a confiabilidade, a eficiência, a usabilidade e a acuracidade. Cinco arquitetos não mencionaram a validação dos RNFs. |

**Figura 2.11 - Perguntas e resultados obtidos na pesquisa (BUSCHMANN, AMELLER, et al.)**

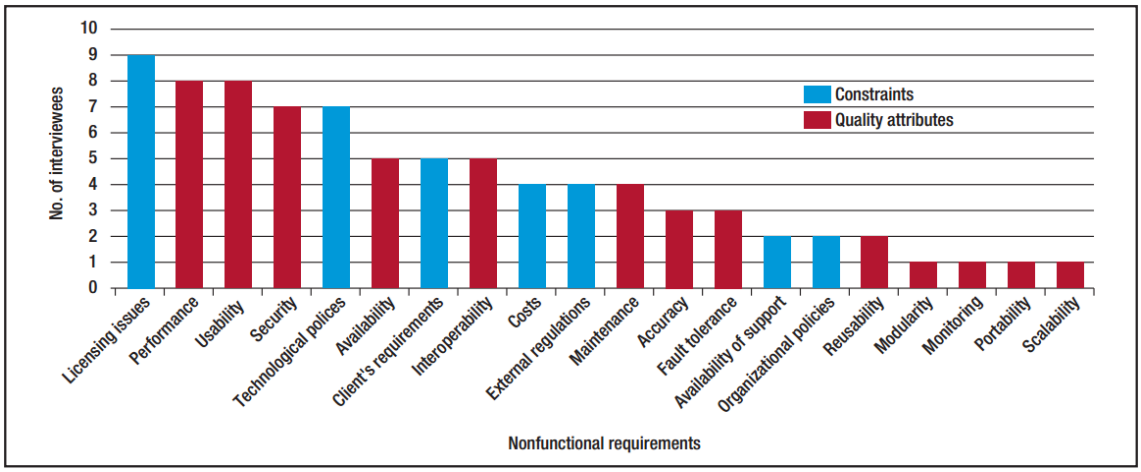


**Figura 2.12 - Relevância dos tipos de RNFs a partir da perspectiva de arquitetos de software (BUSCHMANN, AMELLER, *et al.*)**

Restrições como licenças, provedores de tecnologia e custo apareceram ser os maiores motivadores de decisões de arquitetura, pelo menos de maneira proeminente que os requisitos de qualidade como a performance, a usabilidade e a segurança.

O estudo revelou que o cliente é a principal fonte para os requisitos funcionais. Quando se trata de RNFs, o arquiteto toma o papel mais ativo. Alguns entrevistados declararam que, como arquitetos de software, se consideram como especialistas em definir eficientemente alguns aspectos de qualidade

O estudo também revelou que os RNFs nem sempre são documentados, e que quando documentados nem sempre são precisos. Alguns entrevistados consideram a documentação necessária somente o cliente solicita ou quando existe alguma natureza crítica envolvida.

Parece existir uma falta de alinhamento entre a percepção dos arquitetos em relação a satisfação dos RNFs e a sua validação. Quando ocorreu a validação, ela foi na maior parte das vezes de maneira informal e limitada a um ou dois tipos de RNFs e que podiam ser facilmente verificados por sua natureza quantitativa.

O tratamento dos RNFs deve estar alinha com o valor do negócio. A decisão de qual qualidade não funcional deve ser tratada é de responsabilidade do arquiteto de software, que deve encontrar um balanceamento entre o custo e as qualidades não funcionais que ajudam os usuários e os desenvolvedores. Se a qualidade for muito baixa, isto poderá resultar na redução da aceitação do usuário ou no aumento do

custo do ciclo de vida do software. Se a qualidade for muito alta, isto poderá demandar muito investimento não justificável para o software (BUSCHMANN, AMELLER, *et al.*). O autor sugere o uso modernos middleware COTS (software comercial intermediário de terceiros e disponíveis para compra), o que permite focar mais em aspectos funcionais uma vez que vários dos aspectos de qualidade não funcionais já foram tratados por esses softwares. O autor também defende a importância em se desenvolver uma cultura de sempre se considerar os RNFs nos projetos de maneira adequada. Em uma cultura madura de desenvolvimento, a análise dos RNFs acontece nas primeiras fases de projeto, porque as partes interessadas das áreas técnica e de negócio sabem que uma decisão errada em um RNF de valor pode causar uma grande não conformidade de custo.

## 2.3 QUALIDADE

Embora a demanda para a qualidade seja parte da natureza humana há muito tempo, a quantificação da qualidade e a criação de padrões formais para a qualidade são algo que começou a ganhar foco no século XX. A atenção desordenada para a qualidade durante a últimas décadas criou um mercado de consumidores focados na qualidade. Esses consumidores acreditam que sabem o que significa a qualidade, eles acreditam que reconhecem a qualidade quando eles a veem, e eles demandam qualidade em cada produto e serviço que é comprado (HOYER e HOYER, 2001).

### 2.3.1 Definição de Qualidade

Com o aumento no foco nas últimas décadas sobre o assunto qualidade, diversos trabalhos e definições de qualidade foram publicados. (HOYER e HOYER, 2001) fizeram uma análise detalhada em relação as perspectivas dos principais autores que definem qualidade. Foram analisados 8 autores chamados como “gurus da qualidade” pelos autores, sendo estes:

- Philip B. Crosby (1979)
- W. Edwards Deming (1988)
- Armand. V. Feigenbaum (1983)
- Kaoru Ishikawa (1985)
- Joseph M. Juran (1988)
- Robert M. Pirsig (1974)
- Walter A. Shewhart (1931)
- Genichi Taguchi (1979)

A Figura 2.13 e Figura 2.14 apresentam um resumo da visão de cada um dos gurus do qualidade, sendo que a definição de Shewhart pode ser considerada a mais completa.

**Philip B. Crosby (1979)**

- É necessário definir qualidade, caso contrário não se sabe o suficiente para gerenciá-la.
- De alguma maneira, alguém precisa saber quais são os requisitos e ser capaz de traduzi-los em características mensuráveis.
- Com os requisitos definidos numericamente, pode se medir as características do produto/serviço e saber se ele tem alta qualidade.

**W. Edwards Deming (1988)**

- Qualidade é definida em termos da satisfação do cliente.
- Qualidade é multidimensional.
- Existem definitivamente diferentes graus de qualidade, pois a qualidade é essencialmente equacionada com a satisfação do cliente. Dois clientes podem ter diferentes perspectivas.

**Armand. V. Feigenbaum (1983)**

- Qualidade deve ser definida em termos da satisfação do cliente.
- Qualidade é multidimensional.
- Qualidade é dinâmica, pois as necessidades e expectativas dos clientes vem mudando.
- O mercado avalia o nível da qualidade que ele está disposto a pagar.

**Kaoru Ishikawa (1985)**

- Qualidade é equivalente a satisfação do cliente e precisa ser definida de forma compreensiva.
- Definição de qualidade está sempre mudando (devido à mudança das necessidades e requisitos dos clientes).
- O preço de um produto/serviço é uma parte importante da qualidade. Não se define qualidade sem considerar preço.
-

**Figura 2.13 - Definição de qualidade - Parte I (HOYER e HOYER, 2001)**

**Joseph M. Juran (1988)**

- A definição prática da qualidade é provavelmente não possível de ser feita.
- Inicialmente definiu como satisfação do cliente. Depois viu que era necessário ter características definidas e mensuráveis. Achou ambas inconsistentes em conjunto e preferiu definir como "*fitness for use*" (aptidão de algo ser utilizado).

**Robert M. Pirsig (1974)**

- Não é possível definir qualidade (todos são ignorantes em relação a qualidade). Se não se consegue definir um conceito, é impossível então saber se ele realmente existe.
- Não é possível definir qualidade, porém (quase sempre) somos capazes de reconhecer quando a vemos.

**Walter A. Shewhart (1931)**

- Existem 2 faces quando se fala de qualidade: Subjetivo (o que o cliente deseja) e Objetivo (propriedades do produto, independente do cliente).
- Uma dimensão importante da qualidade é o valor recebido em troca do preço.
- Padrões de qualidade precisam ser expressos em termos físicos (características do produto medidos quantativamente).
- Estatística precisa ser utilizada para se obter informação sobre produtos e serviços desejados a partir de um número grande de potenciais clientes. Deve-se então traduzir em características mensuráveis de um produto/serviço que satisfarão as necessidades da sociedade.

**Genichi Taguchi (1979)**

- Qualidade é a perda que um produto causa para a sociedade após ser entregue para o cliente (perda não relacionado diretamente com a função intrínseca).
- Clientes diferentes têm visões diferente sobre a qualidade, que pode se manifestar em 2 níveis: Qualidade aceitável (Cliente satisfeito) e Qualidade inaceitável (Cliente insatisfeito).

**Figura 2.14 - Definição de qualidade - Parte II (HOYER e HOYER, 2001)**

Após a análise pode-se resumir que existem duas visões principais sobre o tema: “Atender as especificações” e “Satisfazer o cliente”. A Figura 2.15 apresenta como cada guru pensa em relação a estas duas visões.

- Atender as especificações - Significa produzir produtos ou entregar serviços que tenham características mensuráveis e que atendam um conjunto fixo de especificações definidas numericamente.
- Satisfazer o cliente - Significa satisfazer as expectativas de uso e consumo do cliente, independentemente de qualquer característica mensurável.

| | Atender as especificações | Satisfazer o cliente |
|---|---|---|
| **Philip B. Crosby (1979)** | X | |
| **W. Edwards Deming (1988)** | | X |
| **Armand. V. Feigenbaum (1983)** | | X |
| **Kaoru Ishikawa (1985)** | | X |
| **Joseph M. Juran (1988)** | X | X |
| **Robert M. Pirsig (1974)** | | X |
| **Walter A. Shewhart (1931)** | X | X |
| **Genichi Taguchi (1979)** | | X |

**Figura 2.15 - Visão dos gurus sobre a qualidade**

Em relação a qualidade de um software, (PRESSMAN, 2014), define que é a conformidade com os requisitos funcionais e de performance explicitamente definidos e documentados pelos padrões de desenvolvimento, e as características implícitas que são esperadas por todos os profissionais de desenvolvimento de software.

Já a ISO define qualidade de um software como o grau no qual o produto de software satisfaz as necessidades definidas explicitamente e existentes implicitamente quando este é utilizado em condições específicas (ISO/IEC25010, 2011).

### 2.3.2 Modelos de Qualidade

Modelos de Qualidade são modelos analíticos que proveem uma avaliação quantitativa de características ou sub características de qualidade baseados na medição de dados em projetos de software. Esses modelos podem ajudar a obter a avaliação da qualidade atual do produto, em contraste com o modo subjetivo de avaliação baseado no julgamento e na avaliação qualitativa imprecisa (SAMASHIYA e WANG, 2010).

A Figura 2.16 apresenta uma visão temporal dos modelos de qualidade publicados desde o primeiro modelo de Mc Call em 1977 até 2013. Os modelos de qualidade básicos foram publicados entre 1977 e 2001, e tinham o objetivo de avaliar o produto como um todo de maneira compreensiva. Os modelos de qualidade adaptados foram publicados a partir de 2001 e tem como objetivo de avaliar componentes (MIGUEL, MAURICIO e RODRIGUEZ, 2014). O modelo de qualidade da ISO/IEC 25010 é considerado básico devido a sua natureza.

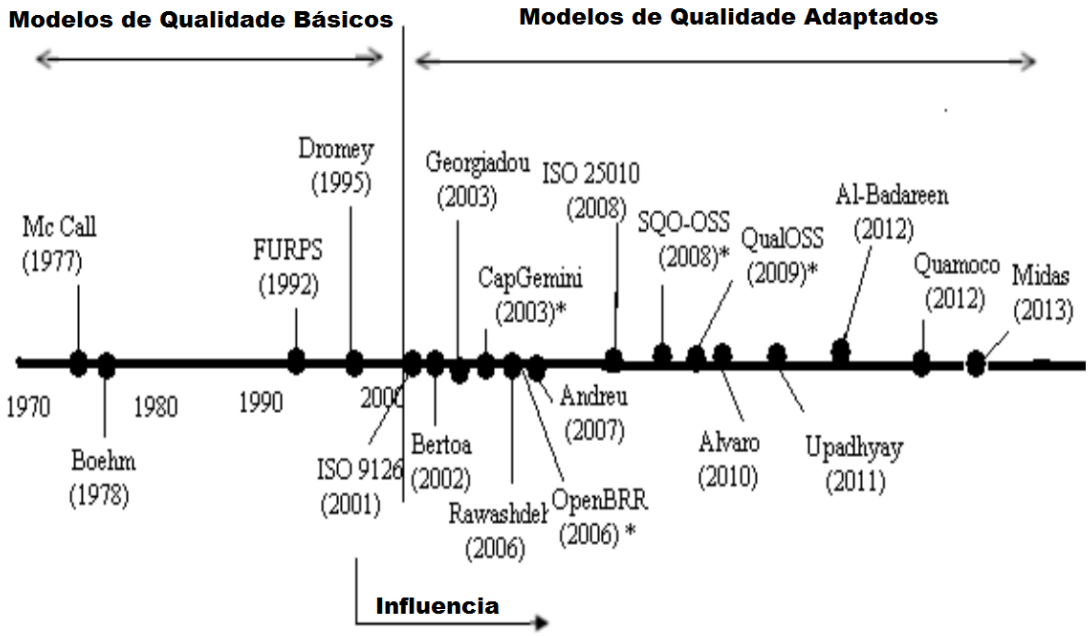


**Figura 2.16 - Visão temporal da criação dos Modelos e Qualidade (MIGUEL, MAURICIO e RODRIGUEZ, 2014)**

### 2.3.3 Comparação entre Modelos de Qualidade

Existem diversos trabalhos que comparam os modelos de qualidade publicados, como por exemplo (SAMASHIYA e WANG, 2010) e (MIGUEL, MAURICIO e RODRIGUEZ, 2014). Esses trabalhos apresentam em detalhes cada um dos modelos de qualidade considerados relevantes e ao fim realizam uma comparação direta entre eles.

A análise de (SAMASHIYA e WANG, 2010) em relação aos modelos básicos de qualidade mostra que a norma ISO/IEC 25010, que na verdade é uma revisão da norma ISO/IEC 9126, possui maior completude em relação as características e sub características de qualidade. Essa comparação pode ser observada na Figura 2.17.

| Characteristic | McCall | Boehm | FURPS | Dromey | ISO-9126 | ISO-25010 |
|---|---|---|---|---|---|---|
| Accuracy | | | | | X | X |
| Adaptability | | | X | | | X |
| Analyzability | | | | | X | X |
| Attractiveness | | | | | X | X |
| Changeability | | | | | X | X |
| Correctness | X | | | | | X |
| Efficiency | X | X | | X | X | X |
| Flexibility | X | | | | | |
| Functionality | | | X | X | X | X |
| Human Engineering | | X | | | | |
| Installability | | | | | X | X |
| Integrity | X | | | | | X |
| Interoperability | X | | | | | X |
| Maintainability | X | | | X | X | X |
| Maturity | | | | | X | X |
| Modifiability | | | | | | X |
| Operability | | | | | X | X |
| Performance | | | X | | X | X |
| Portability | X | X | | X | X | X |
| Reliability | X | X | X | X | X | X |
| Resource utilization | | | | | X | X |
| Reusability | X | | | X | | X |
| Stability | | | | | X | X |
| Suitability | | | | | X | X |
| Supportability | | | X | | X | X |
| Testability | X | X | | | X | X |
| Transferability | | | | | | X |
| Understandability | | X | | | X | X |
| Usability | X | | X | X | X | X |

**Figura 2.17 - Análise comparativa entre os modelos de qualidade básicos (SAMASHIYA e WANG, 2010)**

### 2.3.4 Custo da Qualidade

Segundo o (PMI, 2013), os benefícios primários em atender os requisitos de qualidade incluem menos retrabalhos, maior produtividade, custos menores aumento da satisfação das partes interessadas e aumento na lucratividade. A análise de custo-benefício para cada atividade de qualidade compara o custo do passo da qualidade com o benefício esperado. O custo da qualidade inclui todos os custos ocorridos durante a vida do produto através do investimento em:

- Prevenir uma não conformidade com os requisitos
- Avaliar se o produto tem conformidade com os requisitos
- Falhar em atender os requisitos (o que na prática significa retrabalho)

A Figura 2.18 mostra alguns exemplos de cada uma dessas áreas.

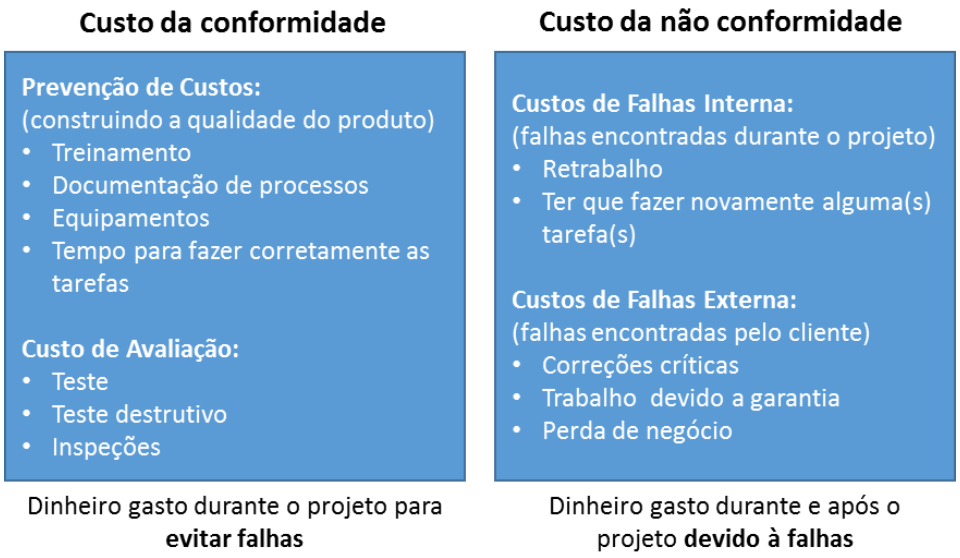


**Figura 2.18 - Exemplos do custo da qualidade (PMI, 2013)**

Em um estudo de caso feito dentro da IBM, (KAPLAN, CLARK e TANG, 1995) demonstraram o custo comparativo de se encontrar um dado erro em cada uma das fases de um projeto de software:

- Análise e definição de requisitos
- Projeto (*Design*)
- Codificação
- Testes de Desenvolvimento
- Testes de Sistema
- Operação em Campo

O resultado desta pesquisa está representado no gráfico da Figura 2.19, aonde pode-se observar que o custo da correção aumenta de forma exponencial caso um erro “vaze” para as próximas etapas do desenvolvimento.

De acordo com (PRESSMAN, 2014), a execução de testes é necessária, porém é muito cari encontrar os erros. Ao gastar tempo encontrando erros antecipadamente no processo, é possível reduzir significativamente os custos de depuração de código e de teste.

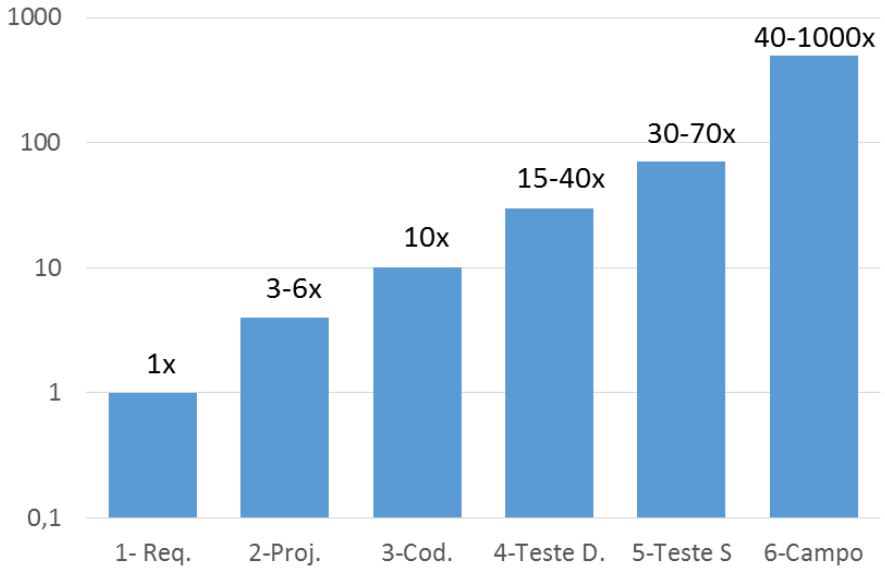


**Figura 2.19 - Custo relativo de corrigir um erro (KAPLAN, CLARK e TANG, 1995)**

## 2.4 NORMAS ISO/IEC PARA QUALIDADE DE SOFTWARE

A ISO (*International Organization for Standardization*) definiu uma série de normas relacionadas aos requisitos e à avaliação da qualidade de sistemas e de software. Essa série de normas é conhecida como SQuaRE (*Systems and software Quality Requirements and Evaluation*) e consiste em cinco grades divisões:

- ISO/IEC 2500n – Divisão sobre Gerenciamento da Qualidade
- ISO/IEC 2501n – Divisão sobre o Modelo de Qualidade
- ISO/IEC 2502n – Divisão sobre a Medida da Qualidade
- ISO/IEC 2503n – Divisão sobre os Requisitos da Qualidade
- ISO/IEC 2504n – Divisão sobre a Avaliação da Qualidade
- ISO/IEC 25050 à ISO/IEC 25099 – Divisão reservada para extensões

Segundo a (ISO/IEC25000, 2014), a série SQuaRE foi criada a partir das normas ISO/IEC 9126 (Qualidade de produtos de software) e ISO/IEC 14598 (Avaliação de produtos de software). O uso prático de ambas as normas impulsou a criação da série SQuaRE pelos seguintes motivos:

- Ambas as normas possuíam a mesma origem normatiza, referencial e funcional
- Ambas as normas formavam um conjunto complementar
- Cada norma possuía um ciclo de vida independente, e isso criou inconsistências entre elas

Até o momento da criação deste de trabalho, algumas normas da série SQuaRE ainda não foram disponibilizadas pois ainda estão em desenvolvimento. Por este motivo a série de normas SQuaRE notifica que ainda é necessário acessar as normas ISO/IEC 9126 e ISO/IEC 14598 para os pontos ainda não complementarmente abordados.

### 2.4.1 Modelo de Qualidade segundo a ISO/IEC

A qualidade de um sistema é resultado da qualidade individual de cada elemento pertencente ao sistema e a suas interações entre si. A qualidade de software é o grau de satisfação de um produto de software em relação às necessidades declaradas e implícitas dentro de condições específicas de utilização por um usuário (ISO/IEC25010, 2011).

Os modelos de qualidade definidos pela norma estão representados na Figura 2.20, e em resumo são os seguintes:

- Modelo de Qualidade do Produto de Software
  - Qualidade interna do software
  - Qualidade externa do software
- Modelo de Qualidade dos Dados
- Modelo de Qualidade do Sistema Em Uso

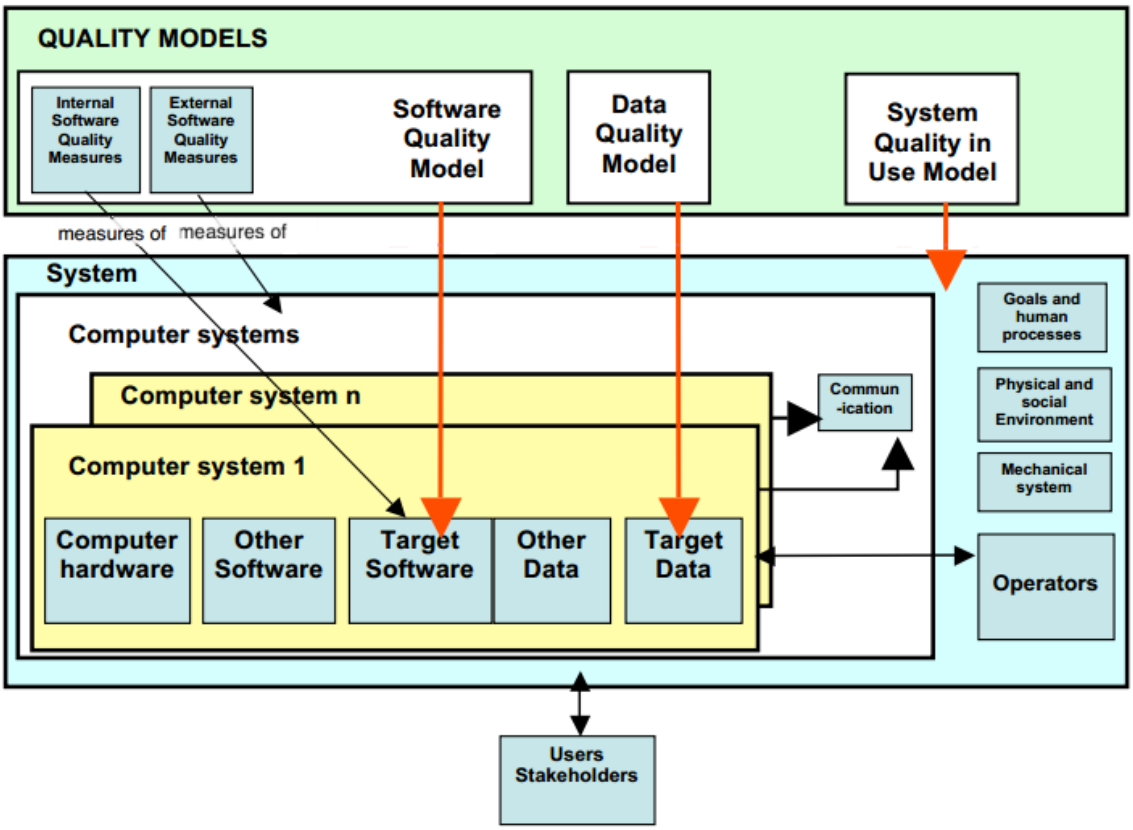


**Figura 2.20 - Diferentes tipos de medida de qualidade (ISO/IEC25010, 2011)**

Para cada um dos modelos citados acima, as normas (ISO/IEC25010, 2011) e (ISO/IEC25012, 2008) dividem as características de qualidade em sub características de forma hierárquica com o objetivo de tornar a cobertura da qualidade de forma mais compreensiva.

### 2.4.2 Modelo de Qualidade do Produto de Software

O Modelo de Qualidade do Produto de Software contém características e sub características que podem ser medidas internamente (qualidade interna) e externamente (qualidade externa).

A qualidade externa do software provê uma visão “caixa preta” do software e endereça propriedades relacionas à execução do software em um hardware computacional e um sistema operacional. A qualidade interna do software provê uma visão “caixa branca” do software e endereça propriedades do produto de software que estão disponíveis durante o desenvolvimento. A qualidade interna do software é, de modo geral, relacionada com propriedades estáticas do software. A qualidade interna de software possui um impacto na qualidade externa do software, que por sua vez possui impacto na qualidade em uso. (ISO/IEC25010, 2011)

O Modelo de Qualidade do Produto de Software é definido por oito características de qualidade:

- *Functional suitability*
- *Reliability*
- *Performance efficiency*
- *Operability*
- *Security*
- *Compatibility*
- *Maintainability*
- *Transferability*

A lista de todas as características e sub características de qualidade do Modelo de Qualidade do Produto de Software pode ser verificado na Figura 2.21.


**Figura 2.21 - Modelo de Qualidade do Produto de Software (ISO/IEC25010, 2011)**

### 2.4.3 Modelo de Qualidade dos Dados

O modelo da qualidade dos dados representa a região aonde o sistema, que acessa a qualidade dos dados, é construído. No modelo de qualidade dos dados, as principais características da qualidade dos dados precisam ser consideradas quando o acesso aos dados do produto é estabelecido. A qualidade de dados do produto pode ser entendida como o grau no qual os dados satisfazem os requisitos definidos pelo dono do produto na organização (ISO/IEC25012, 2008).

A qualidade inerente dos dados refere-se ao grau no qual as características de qualidade dos dados têm o potencial intrínseco para satisfazer as necessidades explícitas e implícitas quando os dados são utilizados em condições específicas. Do ponto de vista inerente, a qualidade dos dados refere-se aos dados em si, e em particular aos seguintes itens:

- Valores de domínio do dado e possíveis restrições (exemplo: regras de negócio que regulam a qualidade necessária para a característica de uma dada aplicação)
- Relacionamento dos valores dos dados (exemplo: consistência)
- Metadados

A qualidade dos dados dependente do sistema refere-se ao grau em que a qualidade dos dados é atingida e preservada dentro de um sistema computacional quando o dado é utilizado em condições específicas. Deste ponto de vista, a qualidade dos dados depende do domínio tecnológico aonde o dado é usado. Isso é alcançado através da capacidade dos componentes do sistema computacional como:

- Dispositivos de hardware (exemplo: para deixar o dado disponível ou para obter a precisão necessária)
- O software do sistema computacional (exemplo: o software de backup para atingir a capacidade de recuperação)
- Outros softwares (exemplo: ferramentas de migração para atingir a portabilidade)

O modelo de qualidade de dados definido pela norma (ISO/IEC25012, 2008) é composto por 15 características. As características da qualidade dos dados são classificadas em duas categorias principais: a qualidade inerente dos dados e a qualidade dos dados dependente do sistema. A lista de todas as características de qualidade do Modelo de Qualidade dos Dados pode ser verificada na Figura 2.22.

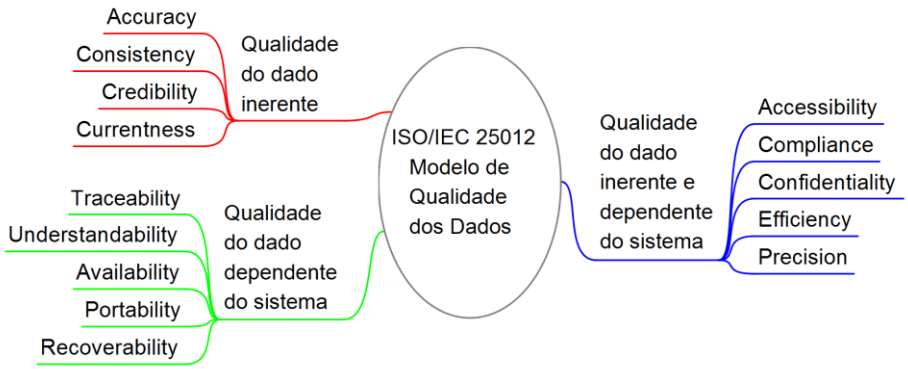


**Figura 2.22 - Modelo de Qualidade dos Dados (ISO/IEC25012, 2008)**

### 2.4.4 Modelo de Qualidade Em Uso

A qualidade em uso é uma medida da qualidade geral do sistema no seu ambiente operacional para usuários específicos executando tarefas específicas. (ISO/IEC25010, 2011)

As características do modelo de qualidade em uso podem ser utilizadas para especificar e avaliar os requisitos para o efeito da qualidade do software no contexto do seu uso. O modelo de qualidade em uso é definido por três características no nível sistêmico:

- *Usability in use*
- *Flexibility in use*
- *Safety*

A lista de todas as características e sub características de qualidade do Modelo de Qualidade em Uso pode ser verificado na Figura 2.23.

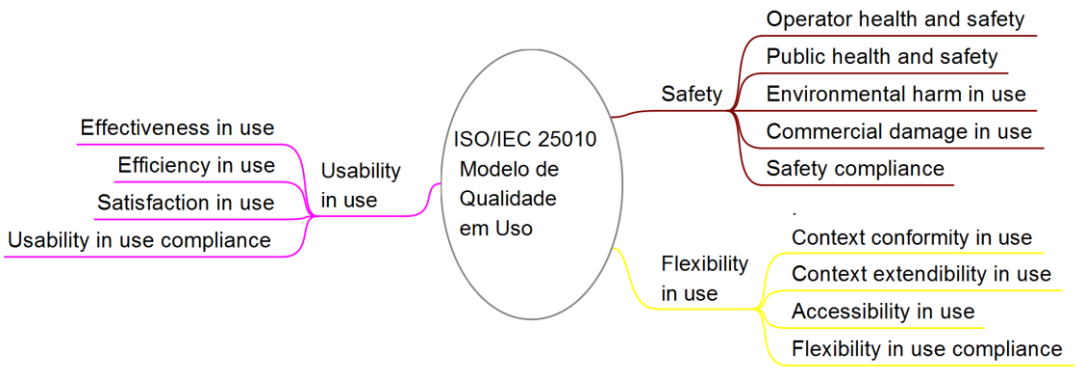


**Figura 2.23 - Modelo de Qualidade em Uso (ISO/IEC25010, 2011)**

### 2.4.5 Qualidade do Produto e o Ciclo de Vida

Um dos processos que ocorre dentro do ciclo de vida de desenvolvimento de software é a avaliação se o produto de software satisfaz a qualidade requerida. Como pode ser observado na Figura 2.24, essa avaliação pode ser realizada através das seguintes medições:

- Medição da qualidade interna - normalmente através de medições estáticas de produtos intermediários
- Medição da qualidade externa - normalmente através da medição do comportamento do código quando executado
- Medição da qualidade em uso

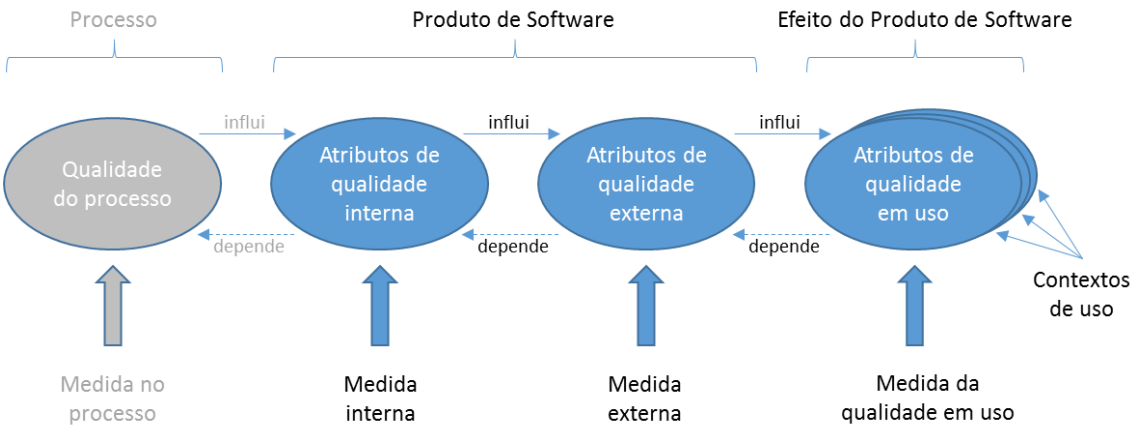


**Figura 2.24 - Qualidade no ciclo de vida (ISO/IEC25000, 2014)**

O Modelo do Ciclo de Vida da Qualidade do Produto de Software está representado na Figura 2.25. Nele, a qualidade do software é tratada em três fases do ciclo de vida do produto de software: durante o desenvolvimento, durante os testes e quando o produto está em uso.

A fase aonde o produto está em desenvolvimento cuida da qualidade interna do produto de software. A fase de testes do produto cuida qualidade externa do produto de software. A fase aonde o produto está em uso cuida da qualidade em uso.

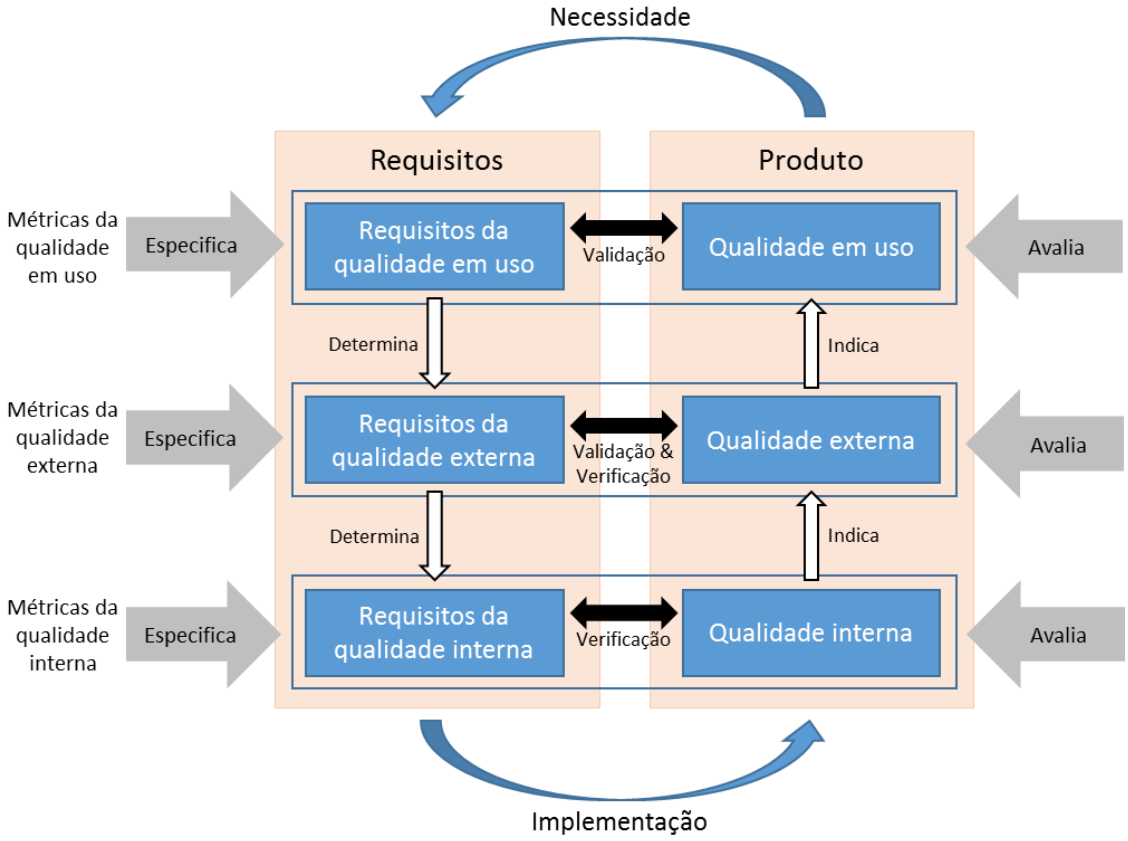


**Figura 2.25 - Modelo do Ciclo de Vida da Qualidade do Produto de Software (ISO/IEC25010, 2011)**

## 2.5 USABILIDADE

Com o crescimento considerável das aplicações distribuídas, aumentou-se a dificuldade para que os desenvolvedores de software acessem diretamente os usuários finais. A usabilidade do software não é mais um luxo, e ao invés disso se tornou um meio determinante para a produtividade e a aceitação do produto de software pelo usuário final (ABRAN, KHELIFI e SURYN, 2003).

A usabilidade se tornou material de estudo e de interesse do governo e de organizações de grande e pequeno porte. Diversos materiais e guias vêm sendo publicados nesta área como por exemplo em (U.S. DEPARTMENT OF HEALTH & HUMAN SERVICES, 2015), (GOOGLE, 2016), (GOOGLE, 2015), (KRUG, 2006).

Diversos autores definem o termo usabilidade de acordo com as suas visões e contexto, mas em essência essas definições não são divergentes e estão relacionadas a facilidade em que um usuário encontra ao utilizar um objeto.

A (ISO/IEC25010, 2011) define a usabilidade como a capacidade do produto de software ser entendido, ser aprendido, ser usado e ser atrativo para o usuário, quando utilizado sob uma determinada condição. Ela ainda complementa que também se refere ao grau no qual usuários específicos podem alcançar objetivos com eficiência, efetividade e satisfação em um dado contexto de uso.

Segundo (NIELSEN, 1993), a usabilidade não é uma propriedade simples da interface com o usuário. A usabilidade possui múltiplos componentes e tradicionalmente é associada com atributos: a facilidade do usuário aprender rapidamente; a eficiência no uso (produtividade); a facilidade do usuário em memorizar e lembrar quando torna a utilizar novamente o sistema; a taxa de problemas que ocorre no uso do sistema; e a satisfação que o usuário tem ao utilizar o sistema.

A partir da experiência prática, (KRUG, 2006) define usabilidade como ter certeza que algo funciona bem de modo que uma pessoa com habilidade e experiência média (ou até abaixo da média) pode utilizar uma coisa, independente se é um software, uma página web ou um avião de guerra.

### 2.5.1 Usabilidade segundo a ISO/IEC 25010

A (ISO/IEC25010, 2011) define a usabilidade dentro de dois aspectos distintos, a primeira considerando o produto de software por si só (Qualidade do Produto de Software) e a segunda considerando o software quando funcionando em um caso real em produção (Qualidade em Uso).

Em relação a Qualidade do Produto de Software, a característica definida pela norma relacionada a usabilidade é chamada de *Operability*, sendo que esta possui as seguintes sub características: *Appropriateness recognisability*, *Learnability*, *Ease of use*, *Helpfulness*, *Attractiveness*, *Technichal assessibility* e *Compliance*. A partir deste momento, será utilizada a tradução especificada na Figura 2.26 para os termos em inglês citados acima

| Termo original em Inglês<br>**Fonte: ISO 25010** | Tradução para Português<br>**Fonte: Autor** |
|---|---|
| *Operability* | Operabilidade |
| *Appropriateness recognisability* | Reconhecimento de Apropriação |
| *Learnability* | Facilidade de Aprender |
| *Ease of use* | Facilidade de Uso |
| *Helpfulness* | Utilidade |
| *Attractiveness* | Atratividade |
| *Technichal assessibility* | Acessibilidade Técnica |
| *Compliance* | Conformidade da Operabilidade |

**Figura 2.26 - Tradução dos termos relacionados à característica Operability**

Em relação a Qualidade Em Uso, a característica definida pela norma relacionada a usabilidade é chamada de *Usability In Use*, sendo que esta possui as seguintes sub características: *Effectiveness in use*, *Eficiency in use*, *Satisfaction in use* e *Compliance*. A partir deste momento, será utilizada a tradução especificada na Figura 2.27 para os termos em inglês citados acima

| Termo original em Inglês<br>**Fonte: ISO 25010** | Tradução para Português<br>**Fonte: Autor** |
|---|---|
| *Usability in use* | Usabilidade em Uso |
| *Effectiveness in use* | Eficácia em Uso |
| *Eficiency in use* | Eficiência em Uso |
| *Satisfaction in use* | Satisfação em Uso |
| *Compliance* | Conformidade da Usabilidade em Uso |

**Figura 2.27 - Tradução dos termos relacionados à característica Usability in Use**

### 2.5.2 Operabilidade

Segundo a norma (ISO/IEC25010, 2011), a Operabilidade é uma das características do software, e corresponde ao grau no qual o produto de software pode ser entendido, ser aprendido, ser usado e ser atrativo para o usuário, quando utilizado sob uma determinada condição. A Operabilidade é subdividida em sete sub características, como apresentado na Figura 2.28.


**Figura 2.28 - Sub características da Operabilidade (ISO/IEC25010, 2011)**

A seguir encontra-se a definição de cada uma das sub características da Operabilidade:

- **Reconhecimento de Apropriação** - É o grau no qual o produto de software permite com que os usuários reconheçam se o software é apropriado para as suas necessidades.
- **Facilidade de Aprender** - É o grau no qual o produto de software permite com que os usuários aprendam o seu uso.
- **Facilidade de Uso** - É o grau no qual o produto de software torna fácil para os usuários operarem e controlarem ele.
- **Utilidade**- É o grau no qual o produto de software provê ajuda quando os usuários precisam de assistência.
- **Atratividade** - É o grau no qual o produto de software é atrativo para os usuários.
- **Acessibilidade Técnica** - É o grau de operabilidade do produto de software para usuários com deficiências específicas.
- **Conformidade da Operabilidade** - É o grau no qual o produto de software é aderente a normas, convenções, guias de estilo ou regulamentos relacionados a operabilidade.

### 2.5.3 Usabilidade em Uso

Segundo a norma (ISO/IEC25010, 2011), Usabilidade em Uso é uma das características do software e corresponde ao grau no qual usuários específicos podem alcançar objetivos específicos com eficácia de uso, eficiência de uso e satisfação de uso em um contexto específico de uso. A Usabilidade em Uso é subdividida em quatro sub características, como apresentado na Figura 2.29.


**Figura 2.29 - Sub características da Usabilidade em Uso (ISO/IEC25010, 2011)**

A seguir encontra-se a definição de cada uma das sub características da Usabilidade em Uso:

- **Eficácia em Uso** - É o grau no qual usuários específicos podem alcançar objetivos específicos com precisão e plenitude em um contexto específico de uso.
- **Eficiência em Uso** - É o grau no qual usuários específicos gastam a quantidade apropriada de recursos em relação a eficácia atingida em um contexto específico de uso.
- **Satisfação em Uso** - É o grau no qual os usuários estão satisfeitos com um contexto específico de uso. A satisfação é dividida em quatro sub características:
    - Gosto (satisfação cognitiva)
    - Prazer (satisfação emocional)
    - Conforto (satisfação física)
    - Confiança
- **Conformidade da Usabilidade em Uso** - É a aderência a normas ou convenções relacionadas a Usabilidade em Uso.

### 2.5.4 Métricas para a usabilidade

Dentro da série de normas SQuaRE, a divisão ISO/IEC 2502n tem como objetivo a medição da qualidade. Essa divisão é compreendida das seguintes normas:

- ISO/IEC 25020 – Guia e modelo de referência de medida
- ISO/IEC 25021 – Elementos de medida de qualidade
- ISO/IEC 25022 – Medição da qualidade em uso (revisão da norma ISO/IEC TR 9126-4)
- ISO/IEC 25023 – Medição da qualidade do sistema e do produto de software (revisão das normas ISO/IEC TR 9126-2 e ISO/IEC TR 9126-3)
- ISO/IEC 25024 – Medição da qualidade dos dados

Como citado anteriormente, até o momento da criação deste de trabalho, algumas normas da série SQuaRE ainda não foram disponibilizadas pois ainda estão em desenvolvimento e por este motivo a série de normas SQuaRE recomenda utilizar as normas ISO/IEC 9126 e ISO/IEC 14598 para os pontos ainda não complementarmente abordados. As normas ISO/IEC 25022 e ISO/IEC 25023 ainda não foram disponibilizadas, portanto foram utilizadas as normas ISO/IEC TR 9126-2, ISO/IEC TR 9126-3 e ISO/IEC TR 9126-4 como referência das métricas a serem utilizadas na medição da qualidade relacionada a Operabilidade e para a Usabilidade em Uso.

Os nomes de algumas características de qualidade e de algumas sub características de qualidade foram renomeadas na revisão feita pela norma ISO/IEC 25010, e por esse motivo foi utilizado os quadros apresentados na Figura 2.30 – Revisão do nome da característica Operabilidade e de suas sub características Figura 2.30 e na Figura 2.31 para a Operabilidade e para a Usabilidade em Uso respectivamente.

Parte das métricas internas do Produto de Software propostas pela norma ISO/IEC TR 9126-3 são relacionadas a característica de qualidade Operabilidade e estas estão apresentadas no Anexo 8.1. Parte das métricas externas do Produto de Software propostas pela norma ISO/IEC TR 9126-2 são relacionadas a característica de qualidade Operabilidade e estas estão apresentadas no Anexo 8.2. Já na norma ISO/IEC TR 9126-4 apresenta algumas métricas para a característica de qualidade Usabilidade em Uso, e estas estão apresentadas no Anexo 8.3.

| Termo segundo a ISO 25010:2011 | Termo segundo a ISO 9126-1:2001 | Nota |
|---|---|---|
| *Operability* | *Usability* | Renomeado para evitar conflito |
| *Appropriateness recognisability* | *Understandability* | O novo nome é mais preciso |
| *Learnability* | *Learnability* | - |
| *Ease of use* | *Operability* | Renomeado |
| *Helpfulness* | - | Nova sub característica |
| *Attractiveness* | *Attractiveness* | - |
| *Technical accessibility* | - | Nova sub característica |

**Figura 2.30 – Revisão do nome da característica Operabilidade e de suas sub características (ISO/IEC25010, 2011)**

| Termo segundo a ISO 25010:2011 | Termo segundo a ISO 9126-1:2001 | Nota |
|---|---|---|
| *Usability in use* | - | - |
| *Effectiveness in use* | *Effectiveness* | Renomeado |
| *Efficiency in use* | *Productivity* | Alinhado com eficiência |
| *Satisfaction in use* | *Satisfaction* | Renomeado |

**Figura 2.31 - Revisão do nome da característica Usabilidade em Uso e de suas sub características (ISO/IEC25010, 2011)**

# 3 METODOLOGIA

Pesquisa é um procedimento formal, sistemático, controlado e crítico, com método de pensamento reflexivo e que trata dos passos necessários para se conhecer a realidade ou para descobrir verdades parciais (MARCONI e LAKATOS, 2003).

Ainda, segundo (GIL, 2010), uma pesquisa pode ser definida como um procedimento formal, racional e sistemático que tem como objetivo fundamental encontrar respostas para problemas. A pesquisa é necessária quando não existe informação suficiente para responder um dado problema, ou quando a informação existe, porém ela está de maneira desorganizada a ponto de ser relacionada ao problema de forma adequada.

Este capítulo aborda os aspectos relacionados a metodologia utilizada neste trabalho.

## 3.1 TIPO DE PESQUISA

Segundo (MARCONI e LAKATOS, 2003), no estudo exploratório são realizadas investigações de pesquisa empírica com o objetivo de formular questões ou apresentar um problema. Esse estudo tem como finalidade desenvolver hipóteses, aumentar a familiaridade do pesquisador com o fenômeno ou com o ambiente, e para modificar ou esclarecer conceitos ou facilitar uma pesquisa futura mais precisa. Normalmente são empregados procedimentos sistemáticos para a obtenção de observações empíricas e/ou para a análise dos dados. O investigador deve conceituar as inter-relações entre as propriedades do fenômeno ou do ambiente observado a partir de descrições qualitativas e/ou quantitativas. Os dados podem ser coletados de diversas maneiras, como por exemplo entrevista, questionário, observação, entre outros, e o estudo se baseia em um número pequeno de dados aonde normalmente não são aplicadas técnicas probabilísticas de amostragem. O investigador pode manipular uma variável independente com a finalidade de descobrir seus potenciais efeitos.

A pesquisa exploratória tem como objetivo proporcionar uma maior familiaridade com o problema estudado, deixando o mais explícito ou construindo hipóteses sobre ele. Este tipo de pesquisa foca no aprimoramento de ideias ou na descoberta de

intuições, e seu planejamento é bastante flexível de modo que possibilite a consideração dos mais variados aspectos relativos ao fato estudado. Normalmente essa pesquisa envolve o levantamento bibliográfico, entrevistas com pessoas que tiveram experiências práticas e diretas com o fato estudado, e por fim uma análise que estimule uma melhor compreensão do fato (GIL, 2010).

> O estudo de caso é apenas uma das muitas maneiras de se fazer pesquisa em ciências sociais. Experimentos, levantamentos, pesquisas históricas e análise de informações em arquivos (como em estudos de economia) são alguns exemplos de outras maneiras de se realizar pesquisa. Cada estratégia apresenta vantagens e desvantagens próprias, dependendo basicamente de três condições: a) o tipo de questão da pesquisa; b) o controle que o pesquisador possui sobre os eventos comportamentais efetivos; c) o foco em fenômenos históricos, em oposição a fenômenos contemporâneos (YIN, 2003, p. 19)

Os estudos de caso normalmente representam a estratégia preferida quando o pesquisador possui pouco controle sobre os eventos, quando o foco se encontra em fenômenos inseridos em algum contexto da vida real e quando são colocadas questões do tipo "como" e "por que" (YIN, 2003).

Segundo (GIL, 2010), não existe um consenso das etapas a serem seguidas durante uma pesquisa de estudo de caso. A partir da literatura, o autor sugere algumas etapas que podem ou nãos ser seguidas de acordo com o estudo de caso realizado. Essas etapas consistem na formulação do problema, na definição da unidade-caso, na determinação do número de casos, na elaboração do protocolo, na coleta de dados, na avaliação e análise dos dados, e na preparação do relatório.

Prova de Conceito é outra técnica utilizada nesta pesquisa. Esta técnica também é conhecida como PoC (sigla oriunda do Inglês *Proof of Concept*) e consiste na realização de um ensaio prático com o objetivo de comprovar um conceito teórico. O dicionário (OXFORD) define a Prova de Conceito como uma evidência que é tipicamente derivada de uma experiência ou de um projeto piloto e que demonstra a viabilidade e veracidade de uma ideia, um conceito ou uma proposta. Segundo (CARSTEN, 1989), o termo Protótipo de Prova de Conceito foi um termo criado por

ele em 1984 com o objetivo de definir um circuito construído através de maneira similar a engenharia de protótipos, mas com o objetivo único de demonstrar a viabilidade de um novo circuito e/ou uma nova técnica de fabricação (não tinha a intenção de ser uma versão preliminar de um produto).

Este trabalho consiste em uma pesquisa descritiva baseada na técnica de Estudo de Caso e de Prova de Conceito.

## 3.2 COLETA DE DADOS

Segundo (GIL, 2010), o processo de coleta de dados no estudo de caso é mais complexo do que em outras modalidades de pesquisa, pois sempre utiliza mais do que uma técnica de coleta de dados e a maioria das pesquisas utiliza somente uma técnica (embora isso não seja mandatório). O fato de se utilizar mais do que um procedimento é fundamental para garantir a qualidade dos dados obtidos. No estudo de caso, os dados devem ser provenientes da convergência ou da divergência das observações obtidas nos diferentes procedimentos. Dessa maneira é que se torna capaz de conferir a validade do estudo, evitando que ele fique subordinado à subjetividade do pesquisador.

A qualidade dos dados obtidos pode ser influenciada diretamente por características e habilidades que devem ser cuidadosamente tratadas durante a fase de planejamento e de coleta de dados. Um bom levantamento de dados é baseado nos seguintes itens: capacidade de se fazer boas perguntas e interpretar as respostas, ser bom ouvinte e não ser enganado por suas próprias ideologias e preconceitos, capacidade de ser adaptável e flexível, ter noção clara das questões estudadas, ser imparcial em relação a noções preconcebidas e ter sensibilidade a provas contraditórias (YIN, 2003).

Os dados utilizados nesse estudo de caso foram oriundos de diversas fontes dependendo do tópico estudado, como representado a seguir:

- Requisitos dos projetos
- Métricas de qualidade relacionadas a Operabilidade Interna
- Métricas de qualidade relacionadas a Operabilidade Externa
- Métrica de qualidade relacionadas a Usabilidade em Uso

Os dados utilizados para o tópico “Requisitos dos projetos” foram extraídos dos documentos PRS e SRS. Ambos os documentos são elaborados na fase inicial dos projetos e têm o objetivo ser um acordo entre os clientes e a organização de desenvolvimento em relação ao “o que” deve ser entregue. Estes documentos não têm como objetivo informar ou detalhar “como” a solução será ou deverá ser implementada. Foram analisados ambos os projetos.

PRS (*Product Requirements Specification* ou Especificação de Requisitos do Produto) é um documento que foca nos requisitos do produto considerados necessários para atender as necessidades do mercado alvo e em informações que permitem obter uma visão geral e compreensiva do produto.

SRS (*Software Requirements Specification* ou Especificação de Requisitos do Software) é um documento que foca nos requisitos dos softwares envolvidos no produto.

Os dados utilizados para o tópico “Métricas de qualidade relacionadas a Operabilidade Interna” foram extraídos do código fonte de software, de documentações internas e de documentações para usuário. Foi também utilizado o Questionário A que pode ser encontrado na sessão 8.1. Esse questionário foi aplicado em desenvolvedores de software do projeto em estudo. Foi seguido o a norma ISO/IEC TR 9126-3 para a definição da métricas e para o método de aquisição desses dados. Foram analisadas as situações anterior e posterior ao projeto.

Os dados utilizados para o tópico “Métricas de qualidade relacionadas a Operabilidade Externa” foram levantados através da aplicação do Questionário B que pode ser encontrado na sessão 8.2. Esse questionário foi aplicado em desenvolvedores de software e em usuários dos projetos em estudo. Foi seguido o a norma ISO/IEC TR 9126-2 para a definição da métricas, porém para o método de aquisição desses dados foi utilizado um questionário para obter a percepção dos participantes em relação as métricas definidas. Foram analisadas as situações anterior e posterior ao projeto.

Os dados utilizados para o tópico “Métrica de qualidade relacionadas a Usabilidade em Uso” foram levantados através da aplicação do Questionário C que pode ser

encontrado na sessão 8.3. Esse questionário foi aplicado em desenvolvedores de software e em usuários dos projetos em estudo. Foi seguido o a norma ISO/IEC TR 9126-4 para a definição da métricas, porém para o método de aquisição desses dados foi utilizado um questionário para obter a percepção dos participantes em relação as métricas definidas. Foi analisado somente a situação anterior ao projeto, uma que ele ainda não foi totalmente implantado para uso dos clientes.

Para os questionários foi seguida a sugestão de (GIL, 2008) que defende que depois que um questionário é redigido, este deve passar por uma prova preliminar antes de ser aplicado definitivamente. O objetivo deste pré-teste é evidenciar possíveis falhas na redação do questionário e assegurar validade e precisão. O pré-teste foi aplicado em um usuário do sistema com o objetivo de se melhorar a clareza das questões relacionadas as métricas de qualidade. Como resultado, as questões foram reformuladas para o formato de afirmações aonde o participante teria que dar a sua percepção a partir da escala Likert, aonde os participantes só informam se concordam ou se discordam da afirmação. Em sequência foi aplicado novamente o pré-teste em um segundo usuário, e como resultado pequenas alterações foram feitas e o questionário foi considerado apto a ser aplicado definitivamente.

Foram utilizadas 5 escalas no total, sendo que para cada opção foi determinado um valor numérico para análise futura conforme apresentado abaixo:

- 5 - Concordo plenamente
- 4 - Concordo parcialmente
- 3 - Não concordo e nem discordo
- 2 - Discordo parcialmente
- 1 - Discordo plenamente

Também foi dada uma opção para quem responde informar se não sabe sobre o assunto ou se não gostaria de responder. Essas repostas foram desconsideradas nas análises de dados.

- Não sei ou não quero responder

## 3.3 ANÁLISE DE DADOS

Após a coleta dos dados, estes foram analisados com o objetivo de comparar a as situações anterior e posterior ao projeto estudado. A análise de cada tópico apresentado no item anterior foi realizada individualmente e está apresentada no próximo capítulo.

Em uma pesquisa a fase seguinte da coleta dos dados é a análise e interpretação dos dados. Esses processos, apesar se serem conceitualmente distintos, são estreitamente relacionados. O objetivo da análise é organizar e resumir os dados de tal forma que seja possível fornecer ao problema proposto de investigação. O objetivo da interpretação é procurar um sentido mais amplo das respostas, o que é feito a partir da conexão com outros conhecimentos previamente obtidos (GIL, 2008).

Segundo (MARCONI e LAKATOS, 2003), o processo de análise dos dados pode ser realizado em três níveis. A interpretação trata da verificação das relações entre as variáveis. A explicação busca esclarecimentos sobre a origem da variável dependente. A especificação cuida da explicitação de até que ponto a relação entre as variáveis são validas.

> A análise de dados consiste em examinar, categorizar, classificar em tabelas ou, do contrário, recombinar as evidências tendo em vista proposições iniciais de um estudo. Analisar as evidências de um estudo de caso é uma atividade particularmente difícil, pois as estratégias e as técnicas não foram muito bem definidas no passado (YIN, 2003, p. 131).

Já o processo de interpretação dos dados possui dois aspectos importantes. A construção de tipos, modelos e esquemas pode ocorrer após procedimentos estatísticos realizados com as variáveis e da determinação de todas as relações permitidas e possíveis. A ligação com a teoria que é um problema que aparece desde o momento inicial de escolha do tema (MARCONI e LAKATOS, 2003).

Ainda segundo (MARCONI e LAKATOS, 2003), ao executar a análise e a interpretação dos dados, deve-se levar em consideração o planejamento bem elaborado para facilitar a análise e a interpretação, e a complexidade ou simplicidade das hipóteses ou dos problemas.

A validação dos dados ocorreu a partir da análise e interpretação combinada de dados obtidos da análise de documentações, de questionários e de eventos observados pelo pesquisador. Esse procedimento está alinhado com a proposta de (YIN, 2003) chamada de triangulação de dados.

# 4 RESULTADOS DA PESQUISA

Nesse capítulo serão apresentados a caracterização da empresa, a caracterização do projeto e os resultados obtidos nesta pesquisa. Para manter as condições de confidencialidade solicitada pela empresa, esta será referida ao longo desta pesquisa como Organização, o produto de software será referido como Produto X e o projeto analisado será referido como Projeto A.

## 4.1 CARACTERIZAÇÃO DA ORGANIZAÇÃO

A Organização é uma empresa multinacional de grade porte no setor de automação e a sua sede está localizada nos Estados Unidos da América. A Organização tem operação em diversos países da América do Norte, Europa, Oriente Médio, África, Ásia e América Latina, aonde possui em torno de 15.000 funcionários distribuídos por suas unidades. A Figura 4.1 apresenta um detalhamento do faturamento da Organização em relação a sua atuação.

A maior parte das unidades possui foco na manutenção dos equipamentos, porém existem unidades com centros de desenvolvimento de hardware e de software e outras com capacidade fabril.

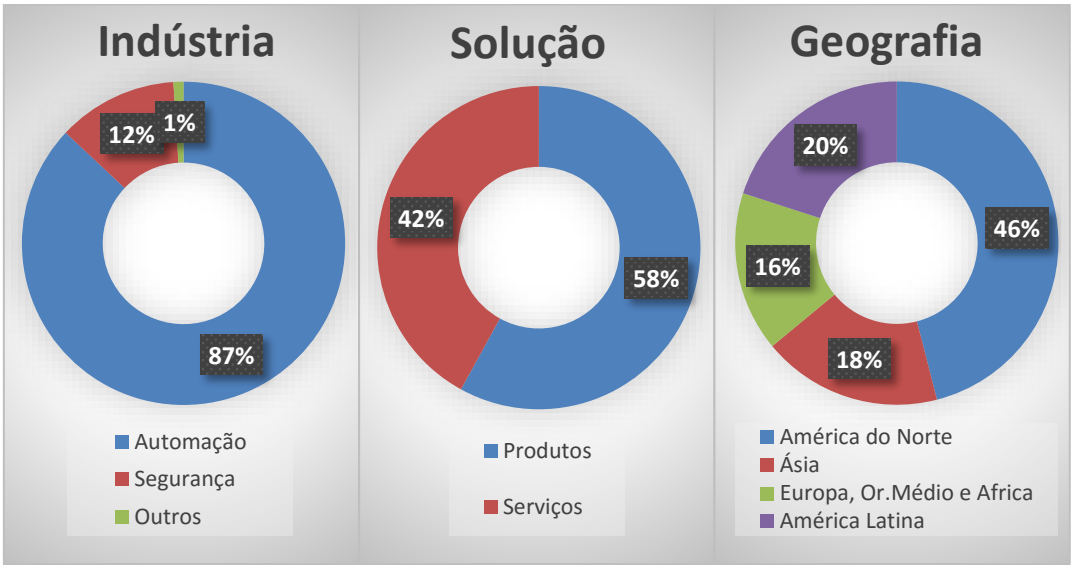


**Figura 4.1 - Detalhes do faturamento da Organização**

Os centros de desenvolvimento da Organização possuem projetos locais para atender a demandas do mercado regional e possuem projetos globais para atender a demandas estratégicas da corporação. Atualmente existem 58 projetos globais ativos, sendo que grande parte destes são projetos com gestão, desenvolvimento e testes distribuído entre os diversos centros de desenvolvimento.

Atualmente os principais centros de desenvolvimento estão localizados no Estados Unidos da América, no Brasil e na Índia, sendo que o principal foco da Índia tem sido na manutenção de produtos de software globais previamente desenvolvidos.

## 4.2 CARACTERIZAÇÃO DO PROJETO A

O Produto X é um sistema de monitoração de uma rede de equipamentos e é utilizado atualmente por algumas unidades da Organização que fornecem o serviço de manutenção de equipamentos para clientes baseados em SLA.

Dentre as principais características do Produto X podemos citar:

- Arquitetura baseada no modelo de três camadas: apresentação, negócio e dados
- Capacidade de monitorar diversos modelos e tipos de equipamentos, uma vez que ele possui a característica de fácil adaptação na entrada
- Permite uma alta personalização do tratamento dos problemas dos equipamentos, o que permite o tratamento totalmente automático, totalmente manual e parcialmente automatizado
- Automação no tratamento dos problemas dos equipamentos, uma vez que o sistema possui integração com uma série de sistemas terceiros
- Capacidade de emitir relatórios customizados
- Instalação distribuída e escalável

O Projeto A tem como objetivo adicionar novas funcionalidades e corrigir problemas encontrados nas versões anteriores do Produto X. A principal nova funcionalidade do Projeto A é a reconstrução das telas operacionais do Produto X com o objetivo de fornecer uma melhor usabilidade para os usuários.

Como pode ser observado na Figura 4.2, ao total foram alocadas 40 pessoas no Projeto A que tem duração planejada de 8 meses e atualmente está na fase final de desenvolvimento e de correções de defeitos.

| Função | Número de colaboradores |
|---|---|
| **Gerente de Produto** | 1 (tempo parcial) |
| **Gerente de Projeto** | 1 |
| **Especificação** | 3 |
| **Arquitetura e design** | 4 |
| **Desenvolvedores de Banco de Dados** | 4 |
| **Desenvolvedores de Aplicação e Interface Web** | 10 |
| **Desenvolvedores de Testes** | 10 |
| **Analistas de Teste** | 6 |
| **Analistas de Documentação** | 1 (tempo parcial) |

**Figura 4.2 - Alocação de recursos no Projeto A**

## 4.3 ELEMENTOS DE ANÁLISE

Para a análise dos dados coletados foram considerados as características de qualidade da ISO relacionadas a usabilidade, que englobam a Operabilidade Interna e Externa e a Usabilidade em Uso. Também foi feita uma análise em cima dos requisitos do Projeto A em relação a usabilidade.

### 4.3.1 Requisitos do Projeto A

Nos documentos PRS e SRS do Projeto A foram incluídos 6 requisitos ao total sendo todos estes classificados internamente como novas funcionalidades. A definição dos requisitos a serem implementados no Projeto A foi originado através de um levantamento envolvendo as partes interessadas e os clientes do Produto X.

Três destes requisitos são relacionadas a atualização da infraestrutura que o Produto X utiliza, e têm como objetivo de tornar o produto compatível com as novas

tecnologias de mercado. Nenhum desses três requisitos implementa realmente novas funções no sistema.

Um dos requisitos é relacionado à melhoria do processo de instalação que atualmente possui muitos passos manuais, o que torna o processo de instalação muito suscetível a falhas humana.

Outro requisito é relacionado a implementação de uma nova função no sistema que tem como objetivo a permitir com que o usuário do Produto X consiga efetivar alterações em múltiplos registros com apenas uma ação.

O último requisito, classificado internamente como o mais importante do Projeto A, é relacionado a revisão e *redesign* da camada de aplicação e da camada de interface com o usuário para todas as telas do sistema que são utilizadas para a operação dos usuários (excluem-se telas relacionadas à configuração e a geração de relatórios). Esse requisito tem como objetivo de melhorar a performance e a usabilidade das telas relacionadas, e declaradamente impõe as seguintes necessidades:

- Permitir redimensionamento nas telas
- Aumentar o tamanho da tela para facilitar a visualização
- Visualização em tela cheia das telas
- Revisão da Interface de Usuário para as telas operacionais
- Habilidade de permitir múltiplas sessões abertas ao mesmo tempo
- Melhorar o acesso aos menus
- Padronizar os controles da interface gráfica
- Melhorar a velocidade de realização de funções
- Filtros de dados mais flexíveis
- Permitir o uso de diferentes navegadores de internet

Este último requisito é bastante amplo e inclui uma parte funcional (RF), aonde são especificadas algumas funcionalidades que são necessárias adicionar ao Produto X, mas também inclui uma parte não funcional (RNF), que está ligada diretamente a fatores de qualidade também considerados importantes para o Produto X.

A parte não funcional deste último requisito não é completamente definida e deixa ambiguidades em aberto. Uma boa abordagem seria “quebrar” esse requisito em requisitos menores e promover um detalhamento mais consistente e objetivo de

todos os atributos de qualidade desejados. Essa abordagem sugerida é aderente a proposta de (SOMMERVILLE, 2011) que defende que o escopo dos requisitos não deve ser aberto para a interpretação dos desenvolvedores e das partes interessadas.

Em uma comparação rápida com os projetos anteriores, o Projeto A foi o primeiro a especificar um requisito amplo e importante relacionado a usabilidade. Isto permitiu com que toda a equipe envolvida no Projeto A discutisse e participasse da definição das melhores maneiras de implementar cada um dos pontos requisitados.

### 4.3.2 Operabilidade Interna

Para a análise da Operabilidade Interna, foram coletados os dados a partir de duas fontes primárias destacadas a seguir. Estes dados foram coletados em uma condição anterior e uma condição posterior ao Projeto A.

- Medições de métricas de qualidade através da análise do código fonte, de documentações internas e de documentações para usuário.
- Medições de métricas de qualidade através da aplicação do Questionário A em desenvolvedores que participaram do Projeto A e de projetos anteriores do Produto X.

As métricas utilizadas foram definidas pela ISO/IEC TR 9126-3 em relação ao que medir e como medir, e os valores resultantes deste levantamento estão apresentados na Figura 4.3.

| | | | Antes do projeto | | | | Após o projeto | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Atributo de Qualidade** | **Identificador** | **Nome da métrica segundo a ISO** | Valor | Média | Mediana | Moda | Valor | Média | Mediana | Moda |
| Reconhecimento de Apropriação | 1.1.1 | Completeness of description | 0,1 | | | | 0,1 | | | |
| | 1.1.2 | Demonstration accessibility | 0,9 | | | | 1,0 | | | |
| | 1.1.3 | Demonstration accessibility in use | 0,9 | | | | 0,9 | | | |
| | 1.1.4 | Demonstration effectiveness | 0,8 | | | | 0,8 | | | |
| Facilidade de Aprender | 2.2.1 | Completeness of user documentation and/or help facility | 1,0 | | | | 1,0 | | | |
| Facilidade de Uso | 1.3.1 | Input validity checking | 0,6 | | | | 0,9 | | | |
| | 1.3.2 | User operation cancellability | 1,0 | | | | 1,0 | | | |
| | 1.3.3 | User operation Undoability | 0,0 | | | | 0,0 | | | |
| | 1.3.4 | Customisability | 0,0 | | | | 0,0 | | | |
| | 1.3.5 | Physical accessibility | 0,0 | | | | 0,0 | | | |
| | 1.3.6 | Operation status monitoring capability | 1,0 | | | | 1,0 | | | |
| | 1.3.7 | Operational consistency | 1,0 | | | | 1,0 | | | |
| | 1.3.8 | Message Clarity | 0,6 | | | | 0,8 | | | |
| | 1.3.9 | Interface element clarity | 1,0 | | | | 1,0 | | | |
| | 1.3.10 | Operational error recoverability | 0,0 | | | | 0,0 | | | |
| Atratividade | 1.4.1 | Attractive interaction | | 3,5 | 3,5 | 4 | | 4,5 | 4,5 | 5 |
| | 1.4.1 | Attractive interaction | | 2,3 | 2 | 2 | | 4,3 | 4 | 4 |
| | 1.4.2 | User Interface appearance customisability | 0,0 | | | | 0,0 | | | |

**Figura 4.3 - Dados das métricas de Operabilidade Interna**

Os valores encontrados antes e após o Projeto A foram similares em muitos aspectos, porém pode se verificar melhora em algumas áreas:

- Reconhecimento de Apropriação - métrica 1.1.2 (relacionada a acessibilidade para demonstrações)
- Facilidade de Uso – métricas 1.3.1 (relacionada a verificação da validade dos parâmetros de entrada) e 1.3.8 (relacionada a clareza das mensagens do software)
- Atratividade – métricas 1.4.1 relacionadas a atratividade em relação a interação com o Produto X)

Existem muitos aspectos que obtiveram nota mínima ou nota próxima a mínima. Esses aspectos podem ser considerados oportunidades de melhoria em projetos futuros do Produto X.

A Figura 4.4 é uma representação gráfica dos dados levantados para a Operabilidade Interna. Para esta representação, os dados que representam valores percentuais foram normalizados para a escala de 0 a 5. Após essa normalização pode-se observar uma melhora de 12% na Operabilidade Interna após o desenvolvimento do Projeto A. Nesta figura, quanto mais distante do centro melhor é o valor da métrica analisada, e consequentemente melhor é a usabilidade.

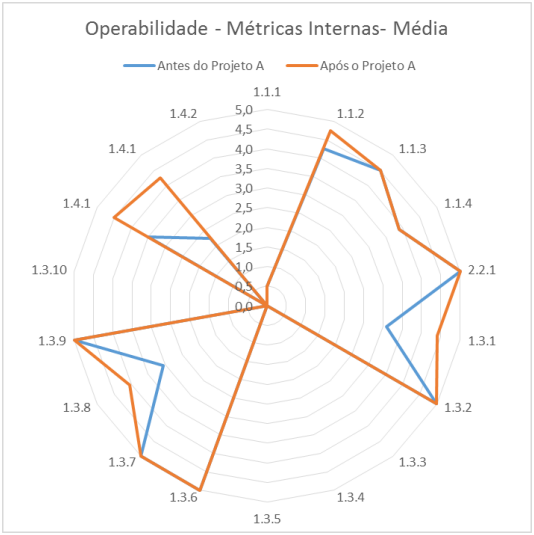


**Figura 4.4 - Representação gráfica dos dados da Operabilidade Interna**

### 4.3.3 Operabilidade Externa

Para a análise da Operabilidade Externa, os dados foram coletados a partir da aplicação do Questionário B em desenvolvedores e em usuários do Produto X. Os desenvolvedores selecionados participaram do Projeto A e de projetos anteriores do Produto X. Os usuários selecionados são usuários internos do Produto X e tiveram contato antes e depois do Projeto A

As métricas utilizadas foram definidas pela ISO/IEC TR 9126-2 em relação ao que medir, porém para o levantamento foi utilizado um questionário para obter a percepção dos participantes em relação as métricas definidas. Os valores resultantes deste levantamento estão apresentados na Figura 4.5.

| | | | Antes do projeto | | | Após o projeto | | |
|---|---|---|---|---|---|---|---|---|
| **Atributo de Qualidade** | **Identificador** | **Nome da métrica segundo a ISO** | Média | Mediana | Moda | Média | Mediana | Moda |
| Reconhecimento de Apropriação | 2.1.1 | Completeness of description | 3,8 | 4 | 4 | 3,7 | 4 | 4 |
| | 2.1.2 | Demonstration accessibility | 2,6 | 2,5 | 1 | 2,7 | 2,5 | 1 |
| | 2.1.3 | Demonstration accessibility in use | 2,5 | 2 | 1 | 2,5 | 2 | 1 |
| | 2.1.4 | Demonstration effectiveness | 3,4 | 4 | 4 | 3,9 | 4 | 4 |
| | 2.1.5 | Evident functions | 3,6 | 4 | 4 | 4,1 | 4 | 4 |
| | 2.1.6 | Function understand-ability | 4,1 | 4 | 4 | 4,1 | 4 | 4 |
| | 2.1.7 | Understandable input and output | 3,6 | 4 | 4 | 4,3 | 4 | 4 |
| Facilidade de Aprender | 2.2.1 | Ease of function learning | 2,6 | 2 | 2 | 3,6 | 4 | 4 |
| | 2.2.2 | Ease of learning to perform a task in use | 2,0 | 2 | 2 | 3,9 | 4 | 4 |
| | 2.2.3<br>2.2.4 | Effectiveness of the user documentation and/or help | 3,4 | 4 | 4 | 4,0 | 4 | 4 |
| | 2.2.5 | Help accessibility | 1,8 | 1,5 | 1 | 1,7 | 1 | 1 |
| | 2.2.6 | Help frequency | 1,8 | 2 | 2 | 2,3 | 2 | 3 |
| Facilidade de Uso | 2.3.1 | Operational consistency in use | 3,3 | 4 | 4 | 4,1 | 4 | 4 |
| | 2.3.2 | Error correction | 2,8 | 3 | 3 | 3,6 | 4 | 4 |
| | 2.3.3 | Error correction in use | 2,9 | 3 | 4 | 3,7 | 4 | 4 |
| | 2.3.4 | Default value availability in use | 3,0 | 3,5 | 4 | 4,4 | 5 | 5 |
| | 2.3.5 | Message understand-ability in use | 3,6 | 4 | 4 | 4,3 | 4 | 4 |
| | 2.3.5 | Message understand-ability in use | 3,1 | 3 | 2 | 3,8 | 3,5 | 3 |
| | 2.3.5 | Message understand-ability in use | 3,4 | 3 | 5 | 4,0 | 4 | 5 |
| | 2.3.6 | Self-explanatory error messages | 3,2 | 2,5 | 2 | 3,8 | 4 | 5 |
| | 2.3.7 | Operational error recoverability in use | 2,8 | 3 | 4 | 3,6 | 4 | 4 |
| | 2.3.8 | Time between human error operations in use | 3,1 | 3 | 5 | 4,2 | 4 | 4 |
| | 2.3.9 | Undoability (User error correction) | 3,9 | 4 | 4 | 4,2 | 4 | 4 |
| | 2.3.9 | Undoability (User error correction) | 2,7 | 2 | 4 | 3,3 | 3,5 | 4 |
| | 2.3.10 | Customisability | 1,9 | 1 | 1 | 4,7 | 5 | 5 |
| | 2.3.10 | Customisability | 2,0 | 1 | 1 | 4,2 | 4 | 4 |
| | 2.3.10 | Customisability | 2,3 | 1 | 1 | 4,4 | 5 | 5 |
| | 2.3.11 | Operation procedure reduction | 1,6 | 1 | 1 | 4,0 | 4 | 3 |
| Acessibilidade Técnica | 2.3.12 | Physical accessibility | 1,1 | 1 | 1 | 1,7 | 1 | 1 |
| Atratividade | 2.4.1 | Attractive interaction | 1,6 | 1 | 1 | 4,1 | 4 | 4 |
| | 2.4.2 | Interface appearance customisability | 1,0 | 1 | 1 | 4,7 | 5 | 5 |
| Utilidade | P1 | Criada pelo pesquisador | 4,8 | 5 | 5 | 4,9 | 5 | 5 |
| | P2 | Criada pelo pesquisador | 4,6 | 5 | 5 | 4,7 | 5 | 5 |

**Figura 4.5 - Dados das métricas de Operabilidade Externa**

Considerando para a análise o cálculo do valor médio das respostas dadas pelos participantes, pode-se observar que:

- 6 métricas permaneceram praticamente idênticas comparando-se antes e depois do Projeto A. Foram contabilizadas as métricas com diferença menor ou igual a 0,1 entre os valores médios de cada situação.
- 18 métricas obtiveram uma melhora comparando-se antes e depois do Projeto A. Foram contabilizadas as métricas com diferença maior que 0,1 e menor 1,0 entre os valores médios de cada situação.

- 8 métricas obtiveram uma melhora significativa comparando-se antes e depois do Projeto A. Foram contabilizadas as métricas com diferença maior ou igual a 1,0 entre os valores médios de cada situação. Isso significa que na média, essas métricas obtiveram uma melhora de pelo menos 1 ponto na escala Likert.

Dentre as métricas com melhora significativa podemos citar:

- Facilidade de Aprender – métrica 2.2.2 (relacionada a facilidade de aprender a fazer uma tarefa durante o uso do sistema)
- Facilidade de Uso – métrica 2.3.4 (relacionada aos valores padrão apresentados pela interface do sistema)
- Facilidade de Uso – métrica 2.3.10 (3 métricas relacionadas a capacidade de customização do software pelo usuário)
- Atratividade – métrica 2.4.1 (relacionada a atratividade do sistema original)
- Atratividade – métrica 2.42 (relacionado a atratividade do sistema através da customização da aparência)

A Figura 4.6 é uma representação gráfica dos dados levantados para a Operabilidade Externa. Nesta figura, são apresentados os gráficos que representam a média, a mediana e a moda dos dados coletados para a situação anterior e posterior ao Projeto A. Quanto mais distante do centro melhor é o valor da métrica analisada, e consequentemente melhor é a usabilidade.

Nos três gráficos é possível verificar que a usabilidade foi melhorada após o Projeto A. Em relação à média, os valores melhoraram em torno de 32%. Comparando a mediana, os valores melhoraram 37%. Já para a moda, houve uma melhora de 29%.

Foram adicionadas duas métricas não previstas pela ISO/IEC TR 9126-2 com o objetivo de se medir a sub característica de Utilidade.


**Figura 4.6 - Representação gráfica dos dados da Operabilidade Externa**

### 4.3.4 Usabilidade em Uso

Para a análise da Usabilidade em Uso, os dados foram coletados a partir da aplicação do Questionário C em desenvolvedores e em usuários do Produto X. Os desenvolvedores selecionados participaram de projetos anteriores do Produto X. Os usuários selecionados são usuários internos do Produto X. Essa métrica é baseada no uso do sistema de software no ambiente real de produção. Por esse motivo foi coletado somente os dados referentes a versão do Produto X anterior ao Projeto A, uma vez que o Projeto A não foi totalmente implantado em ambiente real de produção.

As métricas utilizadas foram definidas pela ISO/IEC TR 9126-4 em relação ao que medir, porém para o levantamento foi utilizado um questionário para obter a percepção dos participantes em relação as métricas definidas. Os valores resultantes deste levantamento estão apresentados na Figura 4.7.

| | | | Antes do projeto | | |
|---|---|---|---|---|---|
| **Atributo de Qualidade** | **Identificador** | **Nome da métrica segundo a ISO** | Média | Mediana | Moda |
| Eficácia em Uso | 3.1.1 | Task effectiveness | 4,5 | 4,5 | 4 |
| | 3.1.2 | Task completion | 4,4 | 4,5 | 5 |
| | 3.1.3 | Error frequency | 2,9 | 3 | 2 |
| Eficiência em Uso | 3.2.1 | Task time | 2,4 | 2 | 1 |
| | 3.2.2 | Task efficiency | 3,4 | 4 | 4 |
| | 3.2.3 | Economic productivity | 4,0 | 4 | 4 |
| | 3.2.4 | Productive proportion | 3,7 | 4 | 4 |
| | 3.2.5 | Relative user efficiency | 2,9 | 3 | 3 |
| Satisfação em Uso | 3.3.1 | Satisfaction scale | 3,0 | 3 | 3 |
| | 3.3.2 | Satisfaction questionnaire | 3,5 | 4 | 4 |
| | 3.3.3 | Discretionary usage | 3,2 | 3 | 3 |
| | P3 | Criada pelo pesquisador | 3,9 | 4 | 4 |
| | P4 | Criada pelo pesquisador | 3,0 | 3 | 3 |
| | P5 | Criada pelo pesquisador | 3,2 | 3 | 3 |

**Figura 4.7 - Dados das métricas de Usabilidade em Uso**

Considerando os valores médios, podemos classificar as métricas em três categorias:

- 3 métricas obtiveram resultados maior ou igual a 4. Essas métricas representam qualidades a serem mantidas.
- 8 métricas obtiveram resultados maior ou igual a 3 e menor que 4. Essas métricas representam qualidades a serem melhoradas.
- 3 métricas obtiveram resultados inferior a 3. Essas métricas representam qualidades a serem analisadas e melhoradas imediatamente.

Foram adicionadas três métricas não previstas pela ISO/IEC TR 9126-4 com o objetivo de se medir a sub característica de Satisfação Em Uso. Os aspectos que estavam faltando na norma e foram adicionados no questionário são relacionados a confiança, a satisfação emocional e a satisfação física.

A Figura 4.8 é uma representação gráfica dos dados levantados para a Usabilidade em Uso. Nesta figura, são apresentados os gráficos que representam a média, a mediana e a moda dos dados coletados para a situação anterior ao Projeto A. Quanto mais distante do centro melhor é o valor da métrica analisada, e consequentemente melhor é a usabilidade.

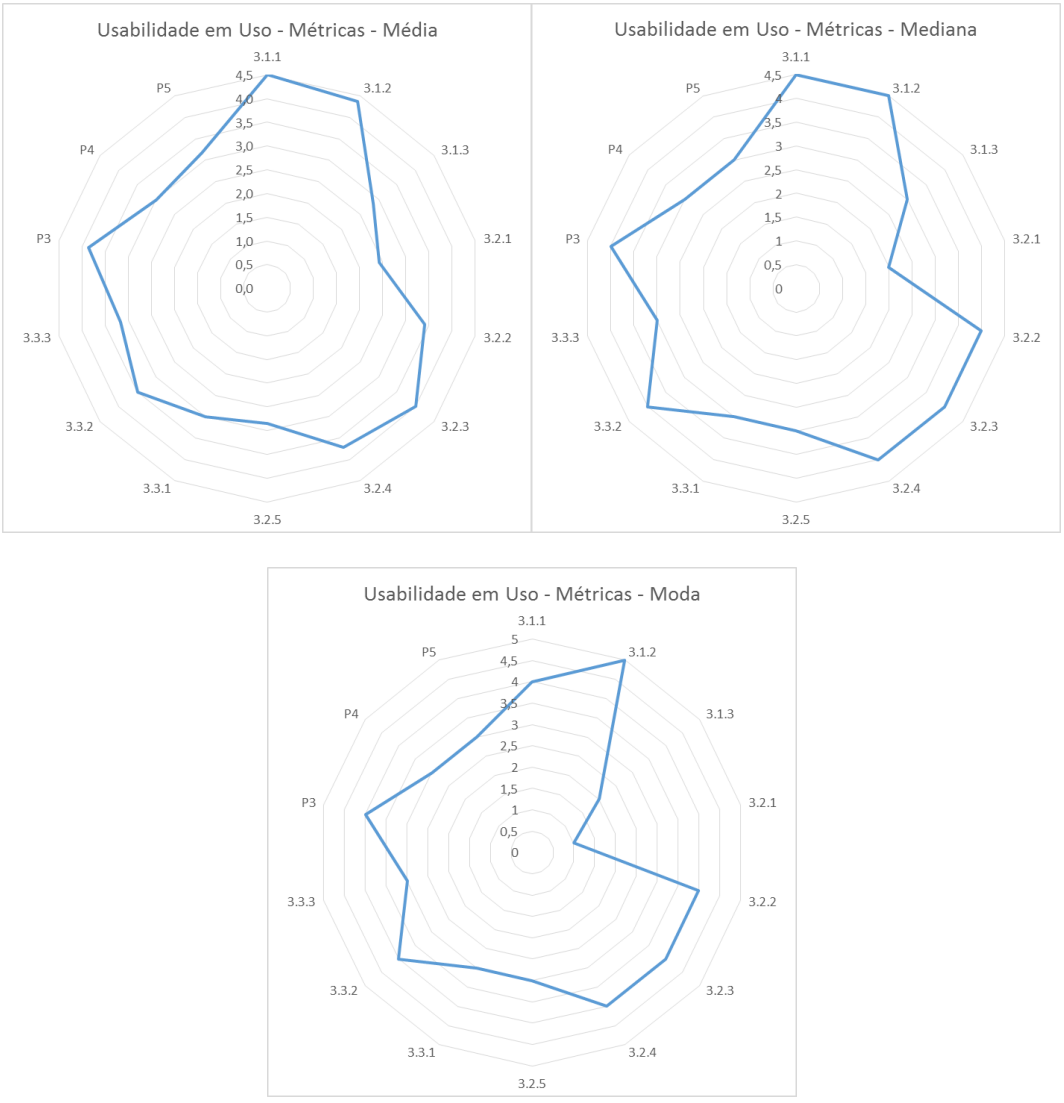


**Figura 4.8 - Representação gráfica dos dados da Usabilidade em Uso**

# 5 CONSIDERAÇÕES FINAIS

Neste capítulo são apresentadas as considerações finais, através das conclusões, a resposta à questão da pesquisa e os trabalhos futuros que podem ser desenvolvidos.

## 5.1 CONCLUSÕES

Este trabalho apresentou a importância de uma boa definição inicial dos requisitos que podem ser desdobrados em requisitos funcionais e requisitos não funcionais. Os requisitos não funcionais especificam atributos de qualidade desejados ou restringem o comportamento do software.

Para avaliar as características ou sub características de qualidade de forma quantitativa em projetos de software, pode-se utilizar modelos analíticos de qualidade para a medição de dados. Tais modelos ajudam a obter uma avaliação consistente da qualidade atual do produto.

A falta da adoção de um modelo de qualidade pode levar a um julgamento fraco, superficial e muitas vezes errado do verdadeiro estado da qualidade.

A participação dos usuários na avaliação da qualidade é fundamental para que os resultados sejam consistentes. Essa avaliação pode ser feita através da observação do usuário utilizando o sistema em condições reais e também através da aplicação de questionários;

As normas da série SQuaRE da ISO/IEC fornecem um modelo de qualidade com visão abrangente para os mais diversos atributos de qualidade. Como nem todas as normas foram revisadas e publicadas para a série 25000, ainda é necessário acessar as normas ISO/IEC 9126.

Mesmo utilizando um modelo de qualidade, a tarefa de avaliar as métricas da característica de qualidade Operabilidade e da característica de qualidade Usabilidade em Uso não foi simples. Para essas 2 características de qualidade foi analisado individualmente cada uma das 57 métricas propostas pela ISO.

Durante a análise das métricas, observou-se que a ISO/IEC 9126 não considera métricas para a sub característica de utilidade e para os aspectos emocional, físico e de confiança da sub característica de satisfação.

Mesmo após o Projeto A, cinco métricas relacionadas a característica de Operabilidade Interna permaneceram com a nota mínima. Isso significa que 31% das métricas tem espaço para melhoria.

Após o Projeto A, verificou-se uma melhora de 12% na característica de Operabilidade Interna. Isso demonstra que a usabilidade foi melhorada após o Projeto A.

Mesmo após o Projeto A, quinze métricas relacionadas a característica de Operabilidade Externa permaneceram com média inferior a nota 4 que corresponde à opção Concordo Parcialmente. Isso significa que 45% das métricas tem espaço para melhoria.

Após o Projeto A, verificou-se uma melhora de 32% na característica de Operabilidade Externa. Isso demonstra que a usabilidade foi melhorada após o Projeto A.

A Usabilidade em Uso não pôde ser verificada após o Projeto A, pois este ainda não foi totalmente liberado para uso em produção. Como a Operabilidade Interna e a Operabilidade Externa influenciam diretamente na Usabilidade em Uso, na Figura 5.1 é apresentada uma projeção de como ficariam as métricas da Usabilidade em Uso considerando uma melhora de 30%.

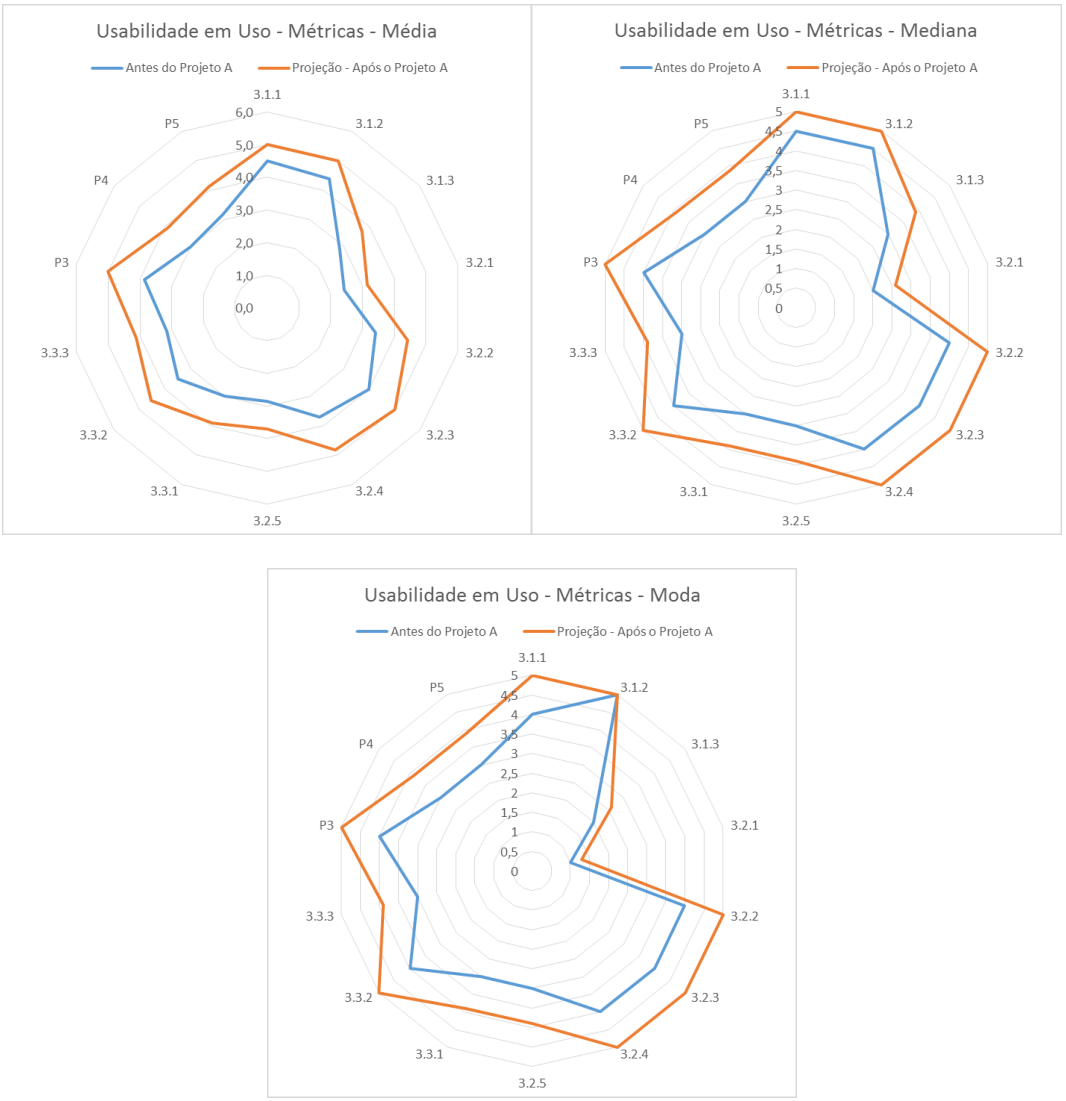


**Figura 5.1 - Previsão para a Usabilidade em Uso após o Projeto A**

## 5.2 RESPOSTA À QUESTÃO DA PESQUISA

O principal objetivo deste trabalho é responder a seguinte Questão de Pesquisa: “Em um produto aonde normalmente são priorizados somente requisitos funcionais, a priorização de um requisito não funcional relacionado a usabilidade reflete alguma melhoria prática no produto final?”.

Como pôde se verificar nas conclusões, a característica de Operabilidade Interna teve uma melhora média de 12% e a característica de Operabilidade Externa teve uma melhora média de 32%. No caso da característica de Usabilidade em Uso não foi possível constatar, pois essa só pode ser avaliada quando o produto está em uso real em produção. Especula-se que a Usabilidade em Uso terá uma melhora em torno de 30%;

Portanto a resposta para a Questão de Pesquisa é sim. A priorização de um requisito não funcional relacionado a usabilidade refletiu em uma melhoria direta no produto final.

Ao responder à Questão de Pesquisa, podemos concluir a Prova de Conceito citada nos objetivos deste trabalho. No produto analisado, os envolvidos relataram que a usabilidade é um assunto considerado diariamente, e que não veem valor adicional em criar um requisito específico para a usabilidade nos projetos. No primeiro projeto aonde um requisito deste tipo foi elicitado verificou-se uma melhora considerável. Provou-se que a usabilidade percebida pelos usuários melhorou consideravelmente após a execução de um projeto aonde um requisito não funcional de usabilidade foi declado explicitamente.

## 5.3 TRABALHOS FUTUROS

Os trabalhos relacionados a seguir são sugeridos como continuação deste trabalho:

- Testar a validade do questionário em outros produtos similares;
- Confirmar se a previsão feita para a característica de Usabilidade em Uso realmente ocorre após o lançamento do Projeto A em produção;
- Reavaliar as métricas propostas pela ISO, e propor novas métricas e novas maneiras de medir;

- Confrontar os dados levantados por questionário com a maneira que a ISO propõe, e com isso avaliar qual ferramenta é mais adequada;
- Elaborar uma maneira de medição contínua da qualidade relacionada as métricas internas do produto de software;
- Avaliar as outras características do Produto X para conhecer a situação atual dele.

# 6 REFERÊNCIAS BIBLIOGRÁFICAS

ABRAN, A.; KHELIFI, A.; SURYN, W. Usability Meanings and Interpretations in ISO Standards. **Software Quality Journal**, n. 11, nov. 2003. 235-338.

BROOKS, F. P. No Silver Bullet: Essence and Accidents of Software Engineering. **IEEE Computer**, Vol 20, Abril 1987. 10-19.

BUSCHMANN, F. et al. Architecture Quality Revisited. **IEEE Computer Society**, n. 0740-7459/12, p. 22-24.

CARSTEN, B. Carsten's Corner. **Power Conversion and Intelligent Motion**, n. 38, nov. 1989.

CHUNG, L. et al. **Non-Functional Requirements in Software Engineering**. Boston, Dordrecht, London: Kluwer Academic Publishers, 2000.

CYSNEIROS, L. M.; LEITE, J. C. S. D. P.; NETO, J. D. M. S. A framework for integrating non-functional requirements into conceptual models. **Requirements Engineering Springer-Verlag**, London, 2001. 97-115.

GIL, A. C. **Métodos e Técnicas de Pesquisa Social**. 6. ed. São Paulo: Editora Atlas S.A., 2008.

GIL, A. C. **Como elaborar projetos de pesquisa**. 5. ed. São Paulo: Editora Atlas S.A., 2010.

GOOGLE. Material Design, 2015. Disponivel em: <https://material.google.com/>. Acesso em: 04 jun. 2016.

GOOGLE. Google Design, 2016. Disponivel em: <https://design.google.com/>. Acesso em: 04 jun. 2016.

HOYER, R. W.; HOYER, B. B. Y. What Is Quality? **Quality Progress**, n. 34, jul. 2001.

IEEE. **IEEE Std.830-1998: IEEE Recommended Practice for Software Requirements Specifications**. Software Engineering Standards Committee of the IEEE Computer Society. New York. 1998.

IEEE. IEEE Spectrum. **Why Software Fails**, 2005. Disponivel em: <http://spectrum.ieee.org/computing/software/why-software-fails>. Acesso em: 04 jun. 2016.

ISO/IEC25000. **ISO/IEC Std.25000:2014: Systems and software engineering - Systems and software quality requirements and evaluation (SQuaRE) - Guide to SQuaRE**. International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC). [S.l.]. 2014.

ISO/IEC25010. **ISO/IEC Std.25010:2011: Systems and software engineering - Systems and software quality requirements and evaluation (SQuaRE) - System and software quality models**. International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC). [S.l.]. 2011.

ISO/IEC25012. **ISO/IEC Std.25012: 2008:Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Data quality model**. International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC). [S.l.]. 2008.

KAPLAN, C.; CLARK, R.; TANG, V. **Secrets of software quality:** 40 innovations from IBM. New York: McGraw-Hill, Inc, 1995.

KRUG, S. **Dont Make Me Think:** A Common Sense Approach to Web Usability. 2. ed. Berkeley: New Riders, 2006.

LAUDON, K. C.; LAUDON, J. P. **Sistemas de Informação Gerenciais**. 7. ed. São Paulo: Pearson Prentice Hall, 2007.

MARCONI, M. D. A.; LAKATOS, E. M. **Fundamentos de Metodologia Científica**. 5. ed. São Paulo: Editora Atlas S.A., 2003.

MIGUEL, J. P.; MAURICIO, D.; RODRIGUEZ, G. A Review of Software Quality Models for the Evaluation of Software Products. **International Journal of Software Engineering & Applications (IJSEA)**, v. 5, n. 6, nov. 2014.

NIELSEN, J. **Usability Engineering**. Mountain View: Academic Press Professional, 1993.

OLIVEIRA, D. D. P. R. D. **Sistemas de Informações Gerenciais**. 15. ed. São Paulo: Atlas, 2012.

OXFORD. **Oxford Dictionaries**. Disponivel em: <http://www.oxforddictionaries.com/us/definition/american_english/proof-of-concept>. Acesso em: 01 jun. 2016.

PMI. **A guide to the project management body of knowledge**. 5. ed. Pennsylvania: Project Management Institute, 2013.

POHL, K.; RUPP, C. **Requirements Engineering Fundamentals:** A study guide for certified professional for requirements engineering exam. 2. ed. Santa Barbara, California: Rockynook, 2015.

PRESSMAN, S. R. **Software Engineering:** A practitioner's approach. 8. ed. Boston: McGraw-Hill, 2014.

ROBERTSON, S.; ROBERTSON, J. **Volere Requirements Resource**, 1995-2016. Disponivel em: <http://www.volere.co.uk/>. Acesso em: 23 mar. 2016.

ROBERTSON, S.; ROBERTSON, J. **Mastering the Requirements Process:** Getting Requirements Right. 3. ed. Upper Saddle River, NJ: Addison-Wesley Professional, 2012.

SAMASHIYA, D.; WANG, S.-H. Quality Models: Role and Value in Software Engineering. **International Conference on Software Technology and Engineering (ICSTE)**, v. 1, n. 2, p. 320-324, 2010.

SOMMERVILLE, I. **Software Engineeting**. 9. ed. Boston: Pearson Education, 2011.

U.S. DEPARTMENT OF HEALTH & HUMAN SERVICES. Usability.gov Improving the User Experience, 2015. Disponivel em: <http://www.usability.gov/>. Acesso em: 04 jun. 2016.

XAVIER, L. **Integração de requisitos não-funcionais a processos de negócio: integrando BPMN e RNF**. Dissertação de Mestrado - Universidade Federal de Pernambuco. Recife. 2009.

YIN, R. K. **Estudo de caso:** Planejamento e Métodos. 2. ed. Porto Alegre: Bookman, 2003.

YOUNG, R. R. **The requirements engineering handbook**. 1. ed. Norwood: Artech House, 2004.

# 7 APÊNDICE

## 7.1 QUESTIONÁRIO A

Leia cada afirmação abaixo com atenção e marque um X a opção que você acredita que mais se adequa para o produto de software em análise.

Parte 1 – Atratividade

| | | | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|
| 1 | 1.4.1 | A maneira que as telas são construídas tornam o sistema atrativo para o usuário. | | | | | | |
| 2 | 1.4.2 | Nas telas do sistema, os itens a seguir contribuem para a atratividade:<br>• Alinhamento dos itens (vertical e horizontal)<br>• Agrupamentos<br>• Uso de cores<br>• Tamanhos de gráficos adequado<br>• Uso de espaços em branco, separadores, bordas<br>• Animações<br>• Estilos de letra<br>• Interface 3D | | | | | | |

Legenda:

| | | | |
|---|---|---|---|
| A | Concordo Plenamente | B | Concordo Parcialmente |
| C | Não concordo e nem discordo | D | Discordo Parcialmente |
| E | Discordo Plenamente | F | Não sei ou não quero responder |

## 7.2 QUESTIONÁRIO B

Leia cada afirmação abaixo com atenção e marque um X a opção que você acredita que mais se adequa para o produto de software em análise.

Parte 1 – Reconhecimento de Apropriação

| | | | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|
| 1 | 2.1.1 | As funções do sistema são entendidas após ler a descrição do produto. | | | | | | |
| 2 | 2.1.2 | As demonstrações/tutoriais podem ser acessadas pelo usuário. | | | | | | |
| 3 | 2.1.3 | As demonstrações/tutoriais estão disponíveis quando o usuário realmente precisa durante uma operação. | | | | | | |
| 4 | 2.1.4 | As funções do produto podem ser operadas com sucesso após uma demonstração ou após o uso de tutorial. | | | | | | |
| 5 | 2.1.5 | Em condições iniciais de uso, as funções podem ser identificadas pelo usuário. | | | | | | |
| 6 | 2.1.6 | As funções do produto são capazes de serem entendidas corretamente pelo usuário. | | | | | | |
| 7 | 2.1.7 | Os usuários entendem o que é necessário como entrada de dados e entendem o que é provido como saída do sistema. | | | | | | |

Parte 2 – Facilidade de aprender

| | | | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|
| 8 | 2.2.1 | Os usuários demoram pouco tempo para aprender a utilizar as funções. | | | | | | |
| 9 | 2.2.2 | Os usuários demoram pouco tempo para aprender a realizar as tarefas de forma eficiente. | | | | | | |
| 10 | 2.2.3<br>2.2.4 | As tarefas/funções podem ser completas corretamente após utilizar a documentação do sistema e/ou ajuda do sistema. | | | | | | |
| 11 | 2.2.5 | Os tópicos de ajuda podem ser localizados pelo usuário. | | | | | | |
| 12 | 2.2.6 | Os usuários não precisam acessar ajuda para aprender a operar e a completar as suas tarefas. | | | | | | |

Legenda:

| | | | |
|---|---|---|---|
| A | Concordo Plenamente | B | Concordo Parcialmente |
| C | Não concordo e nem discordo | D | Discordo Parcialmente |
| E | Discordo Plenamente | F | Não sei ou não quero responder |

Parte 3 – Facilidade de Uso

| | | | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|
| 13 | 2.3.1 | Os componentes da interface gráfica são consistentes. | | | | | | |
| 14 | 2.3.2 | Os usuários demoram pouco tempo para corrigir erros nas tarefas. | | | | | | |
| 15 | 2.3.3 | Os usuários podem facilmente corrigir erros nas tarefas. | | | | | | |
| 16 | 2.3.4 | Os usuários podem facilmente selecionar valores nos parâmetros de forma conveniente para a operação. | | | | | | |
| 17 | 2.3.5 | Os usuários podem facilmente entender as mensagens do sistema de software. | | | | | | |
| 18 | 2.3.5 | As mensagens do sistema não causam atraso no entendimento da próxima ação a ser realizada. | | | | | | |
| 19 | 2.3.5 | Os usuários podem facilmente memorizar as mensagens importantes. | | | | | | |
| 20 | 2.3.6 | Os usuários normalmente fazem as ações corretas para recuperar das condições de erro. | | | | | | |
| 21 | 2.3.7 | Os usuários podem se recuperar facilmente de situações complicadas. | | | | | | |
| 22 | 2.3.8 | Os usuários podem operar o software pelo tempo necessário sem erro humano. | | | | | | |
| 23 | 2.3.9 | Os usuários conseguem frequentemente corrigir erros de entrada de dados. | | | | | | |
| 24 | 2.3.9 | Os usuários conseguem frequentemente desfazer (*undo*) erros. | | | | | | |
| 25 | 2.3.10 | Os usuários conseguem customizar os procedimentos de operação para sua conveniência. | | | | | | |
| 26 | 2.3.10 | Os usuários quem instruem os usuários finais podem facilmente customizar procedimentos padrão de operação para prevenir erros de operação. | | | | | | |
| 27 | 2.3.10 | As funções do sistema podem ser customizadas. | | | | | | |
| 28 | 2.3.11 | Os usuários conseguem facilmente reduzir os procedimentos de operação para sua conveniência. | | | | | | |

Legenda:

| | | | |
|---|---|---|---|
| A | Concordo Plenamente | B | Concordo Parcialmente |
| C | Não concordo e nem discordo | D | Discordo Parcialmente |
| E | Discordo Plenamente | F | Não sei ou não quero responder |

## Parte 4 – Acessibilidade Técnica

| | | | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|
| 29 | 2.3.12 | As funções do sistema podem acessadas por usuários com limitações físicas (ex: visão, audição, controle motor, etc) | | | | | | |

## Parte 5 – Atratividade

| | | | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|
| 30 | 2.4.1 | A interface é atrativa para os usuários. | | | | | | |
| 31 | 2.4.2 | Os elementos da interface podem ser customizados para a satisfação dos usuários. | | | | | | |

## Parte 6 – Utilidade

| | | | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|
| 32 | P1 | O sistema é útil para os usuários. | | | | | | |
| 33 | P2 | O sistema tem todas as funções que são necessárias para a operação | | | | | | |

Legenda:

| | | | |
|---|---|---|---|
| A | Concordo Plenamente | B | Concordo Parcialmente |
| C | Não concordo e nem discordo | D | Discordo Parcialmente |
| E | Discordo Plenamente | F | Não sei ou não quero responder |

## 7.3 QUESTIONÁRIO C

Leia cada afirmação abaixo com atenção e marque um X a opção que você acredita que mais se adequa para o produto de software em análise.

Parte 1 – Eficácia em Uso

| | | | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|
| 1 | 3.1.1 | Os objetivos das tarefas são atingidos adequadamente. | | | | | | |
| 2 | 3.1.2 | Todas as tarefas são completadas. | | | | | | |
| 3 | 3.1.3 | Os erros acontecem com baixa frequência. | | | | | | |

Parte 2 – Eficiência em Uso

| | | | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|
| 4 | 3.2.1 | As tarefas são completas em um tempo adequado. | | | | | | |
| 5 | 3.2.2 | Os usuários são eficientes. | | | | | | |
| 6 | 3.2.3 | Os usuários são produtivos comparados com os seus custos. | | | | | | |
| 7 | 3.2.4 | Os usuários gastam a maior parte do tempo executando ações produtivas. | | | | | | |
| 8 | 3.2.5 | Os usuários são tão eficientes quanto os especialistas do sistema. | | | | | | |

Parte 3 – Satisfação em Uso

| | | | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|
| 9 | 3.3.1 | Os usuários estão satisfeitos. | | | | | | |
| 10 | 3.3.2 | Os usuários estão satisfeitos com as funcionalidades específicas do software. | | | | | | |
| 11 | 3.3.3 | Os usuários escolhem utilizar o sistema. | | | | | | |
| 12 | P3 | O usuário confia no sistema (confiança). | | | | | | |
| 13 | P4 | O usuário tem prazer em utilizar o sistema (satisfação emocional). | | | | | | |
| 14 | P5 | O usuário se sente fisicamente confortável ao utilizar o sistema (satisfação física). | | | | | | |

Legenda:

| | | | |
|---|---|---|---|
| A | Concordo Plenamente | B | Concordo Parcialmente |
| C | Não concordo e nem discordo | D | Discordo Parcialmente |
| E | Discordo Plenamente | F | Não sei ou não quero responder |

# 8 ANEXOS

O conteúdo do tópico 8.1 foi extraído da norma ISO/IEC TR 9126-3.

O conteúdo do tópico 8.2 foi extraído da norma ISO/IEC TR 9126-2.

O conteúdo do tópico 8.3 foi extraído da norma ISO/IEC TR 9126-4.

## 8.1 OPERABILIDADE - MÉTRICAS INTERNA

| | | Metric name | Purpose of the metrics | Method of application |
|---|---|---|---|---|
| Understandability | 1.1.1 | Completeness of description | What proportion of functions (or types of function) are described in the product description? | Count the number of functions which are adequately described and compare with the total number of functions in the product. |
| | 1.1.2 | Demonstration capablity | What proportion of functions requiring demonstration have demonstration capability? | Count the number of functions that are adequately demonstrable and compare with the total number of functions requiring demonstration capability |
| | 1.1.3 | Evident functions | What proportion of the product functions are evident to the user? | Count the number of functions that are evident to the user and compare with the total number of functions |
| | 1.1.4 | Function understandability | What proportion of the product functions will the user be able to understand correctly. | Count the number of user interface functions where purposes is understood by the user and compare with the number of user interface functions. |
| Learnability | 1.2.1 | Completeness of user documentation and/or help facility | What proportion of functions are described in the user documentation and/or help facility? | Count the number of functions implemented with help facility and/or documentation and compare with the total number of functions in product. |

| | | Metric name | Purpose of the metrics | Method of application |
|---|---|---|---|---|
| Operability | 1.3.1 | Input validity checking | What proportion of input items provide check for valid data | Count the number of input items, which check for valid data and compare with the number of input items, which could check for valid data |
| | 1.3.2 | User operation cancellability | What proportion of functions can be cancelled prior to completion? | Count the number of implemented functions, which can be cancelled by the user prior to completion and compare it with the number of functions requiring the precancellation capability |
| | 1.3.3 | User operation Undoability | What proportion of functions can be undone? | Count the number of implemented functions, which can be undone by the user after completion and compare it with the number of functions |
| | 1.3.4 | Customisability | What proportion of functions can be customised during operation? | Count the number of implemented functions, which can be customized by the user during operation and compare it with the number of functions requiring the customization capability |
| | 1.3.5 | Physical accessibility | What proportion of functions can be customised for access by users with physical handicaps | Count the number of implemented functions, which can be customised and compare it with the number of functions |
| | 1.3.6 | Operation status monitoring capability | What proportion of functions have operations status monitoring capability? | Count the number of implemented functions, which status can be monitored and compare it with the number of functions requiring the monitoring capability. |
| | 1.3.7 | Operational consistency | What proportion of operations behave the same way to similar operations in other parts of the system? | Count the number of instances of operations with inconsistent behaviour and compare it with the total number of operations |
| | 1.3.8 | Message Clarity | What proportion of messages are self-explanatory? | Count the numbers of implemented messages with clear explanations and compare it with the total number of messages implemented. |
| | 1.3.9 | Interface element clarity | What proportion of interface elements are self-explanatory? | Count the number of interface elements which are self explanatory and compare it with the total number of interface elements |
| | 1.3.10 | Operational error recoverability | What proportion of functions can tolerate user error? | Count the number of functions implemented with user error tolerance and compare it to the total number of functions requiring the tolerance capability |

| | | Metric name | Purpose of the metrics | Method of application |
|---|---|---|---|---|
| Attractiveness | 1.4.1 | Attractive interaction | How attractive is the interface to the user? | Questionnaire to users |
| | 1.4.2 | User Interface appearance customisability | What proportion of user interface elements can be customised in appearance. | Inspection (by expert) |
| Usability compliance | 1.5.1 | Usability compliance | How compliant is the product to applicable regulations, standards and conventions for usability | Count the number of items requiring compliance that have been met and compare with the number of items requiring compliance as in the specification.. |

## 8.2 OPERABILIDADE - MÉTRICAS EXTERNA

| | | Metric name | Purpose of the metrics | Method of application |
|---|---|---|---|---|
| Understandability | 2.1.1 | Completeness of description | What proportion of functions (or types of functions) is understood after reading the product description? | Conduct user test and interview user with questionnaires or observe user behaviour. Count the number of functions which are adequately understood and compare with the total number of functions in the product. |
| | 2.1.2 | Demonstration accessibility | What proportion of the demonstrations/ tutorials can the user access? | Conduct user test and observe user behaviour. Count the number of functions that are adequately demonstrable and compare with the total number of functions requiring demonstration capability |
| | 2.1.3 | Demonstration accessibility in use | What proportion of the demonstrations / tutorials can the user access whenever user actually needs to do during operation? | Observe the behaviour of the user who is trying to see demonstration/tutorial. Observation may employ human cognitive action monitoring approach with video camera. |
| | 2.1.4 | Demonstration effectiveness | What proportion of functions can the user operate successfully after a demonstration or tutorial? | Observe the behaviour of the user who is trying to see demonstration/tutorial. Observation may employ human cognitive action monitoring approach with video camera. |
| | 2.1.5 | Evident functions | What proportion of functions (or types of function) can be identified by the user based upon start up conditions? | Conduct user test and interview user with questionnaires or observe user behaviour. Count the number of functions that are evident to the user and compare with the total number of functions. |
| | 2.1.6 | Function understand-ability | What proportion of the product functions will the user be able to understand correctly? | Conduct user test and interview user with questionnaires. Count the number of user interface functions where purposes are easily understood by the user and compare with the number of functions available for user. |
| | 2.1.7 | Understandable input and output | Can users understand what is required as input data and what is provided as output by software system? | Conduct user test and interview user with questionnaires or observe user behaviour. Count the number of input and output data items understood by the user and compare with the total number of them available for user. |

| | | Metric name | Purpose of the metrics | Method of application |
|---|---|---|---|---|
| Learnability | 2.2.1 | Ease of function learning | How long does the user take to learn to use a function? | Conduct user test and observe user behaviour. |
| | 2.2.2 | Ease of learning to perform a task in use | How long does the user take to learn how to perform the specified task efficiently? | Observe user behaviour from when they start to learn until they begin to operate efficiently. |
| | 2.2.3 | Effectiveness of the user documentation and/or help system | What proportion of tasks can be completed correctly after using the user documentation and/or help system? | Conduct user test and observe user behaviour. Count the number of tasks successfully completed after accessing online help and/or documentation and compare with the total number of tasks tested. |
| | 2.2.4 | Effectiveness of user documentation and/or help systems in use | What proportion of functions can be used correctly after reading the documentation or using help systems? | Observe user behaviour. Count the number of functions used correctly after reading the documentation or using help systems and compare with the total number of functions. |
| | 2.2.5 | Help accessibility | What proportion of the help topics can the user locate? | Conduct user test and observe user behaviour. Count the number of tasks for which correct online help is located and compare with the total number of tasks tested. |
| | 2.2.6 | Help frequency | How frequently does a user have to access help to learn operation to complete his/her work task? | Conduct user test and observe user behaviour. Count the number of cases that a user accesses help to complete his/her task. |

| | | | Metric name | Purpose of the metrics | Method of application |
|---|---|---|---|---|---|
| Operability | Conforms with operational user expectations | 2.3.1 | Operational consistency in use | How consistent are the component of the user interface? | Observe the behaviour of the user and ask the opinion. |
| | Controllable | 2.3.2 | Error correction | Can user easily correct error on tasks? | Conduct user test and observe user behaviour. |
| | | 2.3.3 | Error correction in use | Can user easily recover his/her error or retry tasks? | Observe the behaviour of the user who is operating software |
| | | | | Can user easily recover his/her input? | Observe the behaviour of the user who is operating software |
| | Suitable for the task operation | 2.3.4 | Default value availability in use | Can user easily select parameter values for his/her convenient operation? | Observe the behaviour of the user who is operating software.<br>Count how many times user attempts to establish or to select parameter values and fails, (because user can not use default values provided by the software). |
| | Self descriptive (Guiding) | 2.3.5 | Message understand-ability in use | Can user easily understand messages from software system?<br>Is there any message which caused the user a delay in understanding before starting the next action?<br>Can user easily memorise important message? | Observe user behaviour who is operating software |
| | | 2.3.6 | Self-explanatory error messages | In what proportion of error conditions does the user propose the correct recovery action? | Conduct user test and observe user behaviour. |
| | Operational error tolerant (Human error free) | 2.3.7 | Operational error recoverability in use | Can user easily recover his/her worse situation? | Observe the behaviour of the user who is operating software. |
| | | 2.3.8 | Time between human error operations in use | Can user operate the software long enough without human error? | Observe the behaviour of the user who is operating software |
| | | 2.3.9 | Undoability (User error correction) | How frequently does the user successfully correct input errors? | Conduct user test and observe user behaviour. |
| | | | | How frequently does the user correctly undo errors? | Conduct user test and observe user behaviour. |
| | Suitable for individualisation | 2.3.10 | Customisability | Can user easily customise operation procedures for his/her convenience?<br>Can a user, who instructs end users, easily set customised operation procedure templates for preventing their errors?<br>What proportion of functions can be customised? | Conduct user test and observe user behaviour. |
| | | 2.3.11 | Operation procedure reduction | Can user easily reduce operation procedures for his/her convenience? | Count user's strokes for specified operation and compare them between before and after customising operation. |
| | | 2.3.12 | Physical accessibility | What proportion of functions can be accessed by users with physical handicaps? | Conduct user test and observe user behaviour. |

| | | Metric name | Purpose of the metrics | Method of application |
|---|---|---|---|---|
| Attractiveness | 2.4.1 | Attractive interaction | How attractive is the interface to the user? | Questionnaire to users |
| | 2.4.2 | Interface appearance customisability | What proportion of interface elements can be customised in appearance to the user's satisfaction? | Conduct user test and observe user behaviour. |
| Usability compliance | 2.5.1 | Usability compliance | How completely does the software adhere to the standards, conventions, style guides or regulations relating to usability? | Specify required compliance items based on standards, conventions, style guides or regulations relating to usability.<br>Design test cases in accordance with compliance items.<br>Conduct functional testing for these test cases. |

## 8.3 USABILIDADE EM USO - MÉTRICAS

| | | Metric name | Purpose of the metrics | Method of application |
|---|---|---|---|---|
| Effectivenes | 3.1.1 | Task effectiveness | What proportion of the goals of the task is achieved correctly? | User test |
| | 3.1.2 | Task completion | What proportion of the tasks are completed? | User test |
| | 3.1.3 | Error frequency | What is the frequency of errors? | User test |
| Productivity | 3.2.1 | Task time | How long does it take to complete a task? | User test |
| | 3.2.2 | Task efficiency | How efficient are the users? | User test |
| | 3.2.3 | Economic productivity | How cost-effective is the user? | User test |
| | 3.2.4 | Productive proportion | What proportion of the time is the user performing productive actions? | User test |
| | 3.2.5 | Relative user efficiency | How efficient is a user compared to an expert? | User test |
| Satisfaction | 3.3.1 | Satisfaction scale | How satisfied is the user? | User test |
| | 3.3.2 | Satisfaction questionnaire | How satisfied is the user with specific software features? | User test |
| | 3.3.3 | Discretionary usage | What proportion of potential users choose to use the system? | Observation of usage |