PONTIFÍCIA UNIVERSIDADE CATÓLICA DE CAMPINAS

MARCIO COLEPICOLO

METODOLOGIAS PARA ANÁLISE DE RISCO E PRIORIZAÇÃO DO TRATAMENTO DE FALHAS DAS FUNCIONALIDADES DE SOFTWARE

CAMPINAS

2023

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE CAMPINAS
ESCOLA POLITÉCNICA
PROGRAMA DE PÓS-GRADUAÇÃO EM GESTÃO DE REDES DE TELECOMUNICAÇÕES
MARCIO COLEPICOLO

METODOLOGIAS PARA ANÁLISE DE RISCO E PRIORIZAÇÃO DO TRATAMENTO DE FALHAS DAS FUNCIONALIDADES DE SOFTWARE

Dissertação apresentada ao Programa de Pós-Graduação *Stricto Sensu* em Gestão de Redes de Telecomunicações da Escola Politécnica, da Pontifícia Universidade Católica de Campinas, como exigência para obtenção do título de Mestre em Gestão de Redes de Telecomunicações.

Orientador: Prof. Dr. Marcius Fabius Henriques de Carvalho

CAMPINAS
2023

Ficha catalográfica elaborada por Adriane Elane Borges de Carvalho CRB 8/9313
Sistema de Bibliotecas e Informação - SBI - PUC-Campinas

001.6425 Colepicolo, Marcio
C692m

Metodologias para análise de risco e priorização do tratamento de falhas das funcionalidades de software / Marcio Colepicolo. - Campinas: PUC-Campinas, 2023.

112 f.

Orientador: Marcius Fabius Henriques de Carvalho.

Dissertação (Mestrado em Gestão de Redes de Telecomunicações) - Programa de Pós-Graduação em Gestão de Redes de Telecomunicações , Escola Politécnica, Pontifícia Universidade Católica de Campinas, Campinas, 2023.
Inclui bibliografia.

1. Software. 2. Engenharia de sistemas - Veículos a motor. 3. Segmento automotivo - Processos de fabricação. I. Carvalho, Marcius Fabius Henriques de. II. Pontifícia Universidade Católica de Campinas. Escola Politécnica. Programa de Pós-Graduação em Gestão de Redes de Telecomunicações . III. Título.

23. ed. CDD 001.6425

MARCIO COLEPICOLO

METODOLOGIAS PARA ANÁLISE DE RISCO E PRIORIZAÇÃO DO TRATAMENTO DE FALHAS DAS FUNCIONALIDADES DE SOFTWARE

Dissertação apresentada como exigência para obtenção do título de Mestre em Gestão de Redes de Telecomunicações ao Programa de Pós-Graduação em Gestão de Redes de Telecomunicações da Escola Politécnica.
Área de Concentração: Gestão de Redes e Serviços.
Orientador (a): Prof. Dr. Marcius Fabius de Carvalho.

Dissertação defendida e aprovada em 14 de dezembro de 2023 pela Comissão Examinadora constituída dos seguintes professores:

Prof. Dr. Marcius Fabius Henriques de Carvalho
Orientador da Dissertação e Presidente da Comissão Examinadora
Pontifícia Universidade Católica de Campinas

Profa. Dra. – Lia Toledo Moreira Mota
Pontifícia Universidade Católica de Campinas

Dr. Prof. Dr. Jarbas Lopes Cardoso Junior
Centro de Tecnologia da Informação Renato Archer (CTI)

É impossível evitar todas as falhas

É claro que continua sendo nossa tarefa evitar falhas, se possível.

Karl R. Popper

(1902-1994)

AGRADECIMENTOS

Primeiramente à Deus por me conceder saúde e livrai-me de todo o mal.

À Pontifícia Universidade Católica de Campinas, juntamente com o Programa de Pós-Graduação pelo apoio da bolsa de estudos, disponibilização do acervo acadêmico e infraestrutura da universidade.

Ao Prof. Dr. Marcius Fabius Henriques de Carvalho
Orientador e incentivador dos meus trabalhos de pós-graduação na Faculdade de Engenharia Elétrica da Pontifícia Universidade Católica de Campinas, pelo apoio, atenção e amizade.

Aos meus pais, Sra. Aeko Tanaka Colepicolo e Sr. Hildo Colepicolo, por sempre me ensinarem a trilhar o caminho do bem, da dedicação e persuasão diante dos obstáculos da vida.

A minha querida e amada esposa, Walquiria Colepicolo, pela compreensão, amor e carinho durante o tempo que dediquei na elaboração deste trabalho.

Por fim, aos meus colegas de turma e professores da Faculdade Politécnica pelo suporte e companheirismo durante a jornada acadêmica.

RESUMO

COLEPICOLO, Marcio. Metodologias para Análise de Risco e Priorização do Tratamento de Falhas das Funcionalidades de *Software*. Dissertação de Mestrado. Programa de Pós-Graduação em Gestão de Redes de Telecomunicações, Pontifícia Universidade Católica de Campinas, Campinas, 2023.

O projeto de engenharia de sistemas automotivos com aplicações de *software* embarcado vem exigindo o estabelecimento de requisitos e funcionalidades de *software* complexas, especificados de forma colaborativa por sistemistas e montadoras automotivas, que geram custos no desenvolvimento e manutenção das funcionalidades. O fator da qualidade de *software* é um diferencial competitivo a ser considerado pelas empresas, uma vez que a qualidade inadequada pode causar mau funcionamento do produto, custos adicionais de reparo e prejuízos incalculáveis de imagem para a montadora automotiva. No sentido de identificar inadequabilidades das funcionalidades, métodos de classificação e priorização de tratamento de riscos são utilizados como mecanismos de prevenção de problemas e melhoria contínua dos processos e produtos. Este trabalho discute vantagens e desvantagens da aplicação do método *Failure Mode and Effect Analysis* (FMEA), amplamente empregado no segmento automotivo e o FMEA estendido como ferramenta para identificação e mitigação das falhas no desenvolvimento de funcionalidades de software. A seguir o método de análise de envoltória de dados *Data Envelopment Analysis* (DEA) é utilizado como uma forma de avaliação mais completa por permitir a hierarquização das funcionalidades por eficiência e indicar em que e o quanto uma funcionalidade ineficiente deve evoluir para se tornar eficiente. Os resultados apresentados demonstram que a aplicação dos métodos FMEA e DEA contribuem para a análise de risco, tratamento das falhas e priorização das funcionalidades de *software*, se tornando essenciais na proposição de ações de melhoria no desenvolvimento em aplicações de *software* embarcado.

Palavras-chave: FMEA. DEA. Análise de Risco. Tomada de decisão. Priorização de Funcionalidades.

ABSTRACT

COLEPICOLO, Marcio. Methodologies for Risk Analysis and Prioritization of Failures Treatment of *Software* Functionalities. Masters Dissertation. Postgraduate Program in Telecommunications Network Management, Pontifical Catholic University of Campinas, Campinas, 2023.

The engineering project of automotive systems with embedded *software* applications has required the establishment of complex *software* requirements and functionalities, specified collaboratively by systemists and automotive manufacturers, which generate costs in the development and maintenance of functionalities. The *software* quality factor is a competitive differentiator to be considered by companies, since inadequate quality can cause product malfunction, additional repair costs and incalculable image losses for the automotive manufacturer. In order to identify inadequacies in functionalities, risk classification and prioritization methods are used as mechanisms for preventing problems and continuously improving processes and products. This work discusses advantages and disadvantages of applying the Failure Mode and Effect Analysis (FMEA) method, widely used in the automotive segment, and the extended FMEA as a tool for identifying and mitigating failures in the development of *software* features. Additionally, the method Data Envelopment Analysis (DEA) is used as a more complete form of evaluation as it allows the ranking of functionalities by efficiency and indicates by what and how much an inefficient functionality must evolve to become efficient. The results presented demonstrate that the application of FMEA and DEA methods contribute to risk analysis, treatment of failures and prioritization of *software* functionalities, becoming essential in proposing improvement actions in the development of embedded *software* applications.

Keywords: FMEA. DEA. Risk Analysis. Making Decision. Funcionalities Prioritization.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

LISTA DE ABREVIAÇÕES E SIGLAS

AC	Apprraisal Cost
ADAS	Advanced Driver Assistance System
AHP	Analytic Hierarchy Process
AIAG	Automotive Industry Action Group
ASQ	American Society for Quality
ASPICE	Automotive Software Process Improvement Capability Determination
AP	Action Priority
APAC	Asia Pacif
B2B	Business to Business
BCC	Banker Charnes Cooper
CAGR	Compound Annual Growth Rate
CCR	Charnes Cooper Rhodes
CMMI	Capability Maturity Model Integration
CoQ	Cost of Quality
CoGQ	Cost of Good Quality
CoPQ	Cost of Poor Quality
CRS	Constant Returns Scale
CEATEC	Centro de Ciências Exatas, Ambientais e de Tecnologia
DBSCAN	Density Based Spatial Clustering of Applications with Noise
DEA	Data Envelopment Analysis
DFA	Dependent Failure Analysis
DFMEA	Design Failure Mode and Effect Analysis
DMU	Decision Making Unit
DMUs	Decision Making Units
DoD	Department of Defense
ECU	Eletronic Control Unit
EFC	External Failure Cost
EFMEA	Extended Failure Mode Effect Analysis
EMEA	Europe, Middle East and Africa
EPA	Environmental Protection Agency
EPB	Eletronic Parking Brake
EV	Eletric Vehicle
FEI	FMEA Effectiveness Index
FM	Failure Mode
FMEA	Failure Mode and Effect Analysis
FMECA	Failure Mode and Effect and Criticality Analysis
FS	Functionality of Software
FTA	Fault Tree Analysis
IATF	International Automotive Task Force
IEC	International Eletrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IFC	Internal Failure Cost
IoT	Internet of Things

ISO	International Organization for Standardization
JQL	Jira Query Language
LATAM	Latin America
LP	Linear Programming
MISRA	Motor Industry Software Reliability Association
MPS.BR	Melhoria do Processo de Software Brasileiro
NAFTA	North American Free Trade Agreement
NASA	National Aeronautics and Space Administration
NHTSA	National Highway Traffic Safety Administration
OEM	Original Equipment Manufacturer
PC	Prevention Cost
PFMEA	Process Failure Mode and Effect Analysis
PMBOK	Project Management Body of Knowledge
PMI	Project Management Institute
PROCONVE	Programa de Controle da Poluição do Ar por Veículos Automotores
QMS	Quality Management System
RDEA	Robustness Data Envelopment Analysis
RPN	Risk Priority Number
RUP	Rational Unified Process
SAE	Society for Automotive Engineering
S-AHP	Stratified Analytic Hierarchy Process
SBM	Slacks Based Measure
SOD	Severity Occurrence Detection
SPICE	Software Process Improvement Capability Determination
SQuaRE	System and Software Quality Requirements and Evaluation
TS	Technical Specification
VBA	Visual Basic for Applications
VDA	Verband der Automobilindrustrie
VRS	Variable Return to Scale

INTRODUÇÃO

A constante evolução tecnológica incentiva o desenvolvimento de sistemas complexos utilizando recursos de *hardware*, *software* que venham fornecer cada vez mais serviços ao usuário. Organizações ambientais, entidades de segurança cibernética, conectividade e eletrificação são algumas das áreas que potencializam a complexidade do desenvolvimento de sistemas. No segmento automotivo a organização internacional para padronização (do inglês *International Organization for Standardization* – ISO) e a força tarefa automotiva internacional (do inglês *International Automotive Task Force* - IATF) especificam requisitos normativos para a padronização e inclusão de fatores chave para o setor: segurança do produto, gerenciamento de risco, requisitos para *software* embarcado, gestão de mudanças e garantia da qualidade. O atendimento aos processos de qualidade de *software* estão estabelecidos no que tange a melhoria contínua nos processos de *software* requeridos na ISO 15504, também conhecida como (do inglês *Automotive Software Process Improvement Capability Determination* - ASPICE). Desta forma, o produto e os processos são avaliados e auditados no desenvolvimento de um *software* embarcado.

Um conjunto de requisitos funcionais bem definidos, elaborados antes do desenvolvimento de *software*, minimiza as falhas. No segmento automotivo, falhas identificadas após a entrega do produto podem gerar prejuízos financeiros incalculáveis e com grande influência na reputação da montadora automotiva. Nesse sentido, a aplicação de técnicas e métodos para apoio à tomada de decisão podem contribuir para a melhoria contínua da qualidade. No contexto de *software* embarcado, neste trabalho uma funcionalidade consiste em requisitos de sistema solicitados por uma montadora automotiva através de uma especificação. No desenvolvimento dessas funcionalidades, falhas podem ser detectadas em todas as fases do projeto, requerendo ações de melhoria para garantir a qualidade. As organizações utilizam o gerenciamento de riscos a fim de identificar e prevenir falhas, propondo alternativas para melhoria dos processos e produtos. Diante deste cenário, métodos para avaliação de risco em funcionalidades de *software* e tratamento de falhas são utilizados durante o desenvolvimento de processos e produtos. Este trabalho discute o método amplamente utilizado no segmento automotivo para avaliação de riscos denominado análise de modos de falha e seus efeitos (do inglês *Failure Mode and Effects Analysis* – FMEA). A abordagem do FMEA apresenta desvantagens inerentes. Desta forma, propõe uma extensão deste método considerando valores adicionais ao FMEA tradicional. Em seguida o método da análise de envoltória de dados (do inglês *Data Envelopment Analysis* – DEA) é utilizado para

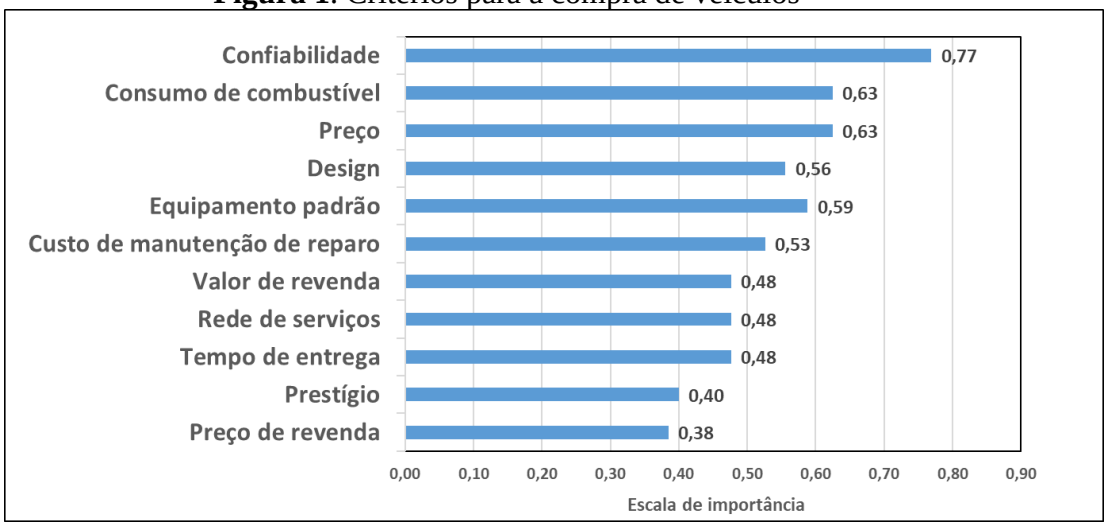
a pontuação da eficiência das funcionalidades de *software*. Por ele, a prioridade para o tratamento do risco e a proposição de ações corretivas nas funcionalidades de *software* são identificadas.

JUSTIFICATIVAS

Com o avanço da tecnologia automotiva, o desenvolvimento de funcionalidades de *software* torna-se cada vez mais complexo. Neste contexto, a confiabilidade de *software* destaca-se como um dos critérios de maior importância para um produto ou serviço. Esta definição de confiabilidade está associada com o conceito de probabilidade. Desta forma, entende-se por confiabilidade a probabilidade que um produto não falha sob dadas condições ambientais e funcionais durante um definido período de tempo (BERTSCHE, 2008). Na perspectiva de (BAZOVSKY, 1961), confiabilidade é definida como a probabilidade de um dispositivo executar sua finalidade adequada durante o período de tempo pretendido sob as condições operacionais encontradas. Na visão de (KAPUR, et al., 2014) a confiabilidade é a capacidade de um produto ou sistema de executar como pretendido (ou seja, sem falhas e dentro dos limites de desempenho especificado) por um tempo determinado, em sua condição de ciclo de vida.

A pesquisa referenciada no livro *Reliability in Automotive and Mechanical Engineering: Determination of Component and System Reliability* (BERTSCHE, 2008) destacou os critérios de avaliação para a compra de um veículo. A Figura 1 ilustra os critérios avaliados na compra de um veículo novo. Em onze critérios avaliados, sendo que a escala de avaliação compreende os valores de 0 (menor importância) e 1 (maior importância) destacou a confiabilidade (0,77) como de maior relevância.

Figura 1. Critérios para a compra de veículos



Fonte: Adaptado de (BERTSCHE, 2008)

Matematicamente, confiabilidade é expressa por R(t) e representa a probabilidade de um sistema operar com sucesso durante um determinado período t (BAUER, et al., 2009). A Eq. 1.1 representa este conceito.

$$R(t) = P(T > t) \qquad t \geq 0; \qquad (1.1)$$

Onde T é uma variável randômica que denota o tempo da falha. Desta forma, confiabilidade é a probabilidade que o valor da variável randômica seja maior do que o tempo t da missão. Por outro lado, define-se F(t) a probabilidade que o sistema falhará durante um determinado período t (BAUER, et al., 2009). A Eq. 1.2 representa este conceito.

$$F(t) = P(T \leq t) \qquad t \geq 0; \qquad (1.2)$$

F(t) representa a função de distribuição de falhas, que é comumente chamada de função de distribuição de falhas cumulativas (BAUER, et al., 2009). A função de confiabilidade é conhecida também como função de sobrevivência e expressa conforme representado na Eq. 1.3.

$$R(t) = 1 - F(t) \qquad (1.3)$$

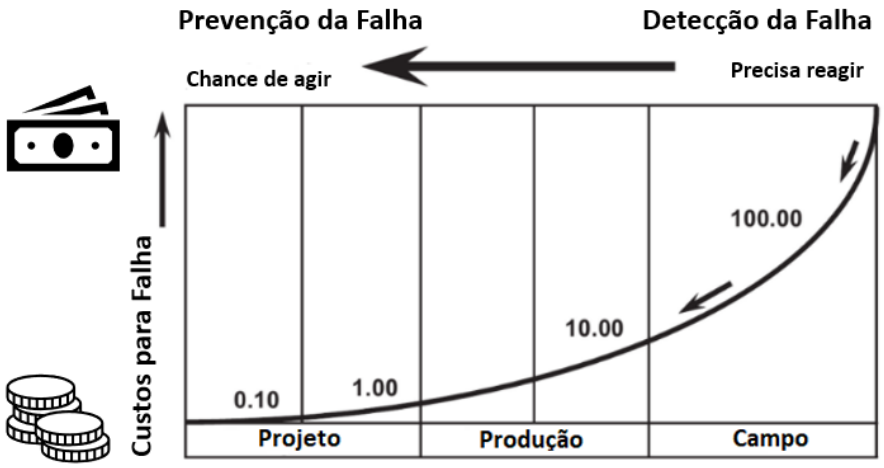
Existe sempre um risco de um produto ou serviço falhar em campo. O custo da prevenção das falhas bem como o custo da avaliação são inevitáveis nos projetos. Quando as

falhas são detectadas nas fases iniciais do projeto, as consequências financeiras são de baixo valor monetário.

As organizações que buscam a melhoria contínua e a excelência implementam estratégias de investir nos custos de prevenção e avaliação, cujo objetivo é reduzir custos evitáveis causador por falhas internas e externas.

Falhas geram prejuízos financeiros e afetam a qualidade e a confiabilidade do produto. A Figura 2 apresenta um fator do custo da identificação de falhas em diferentes fases de um produto.

Figura 2. Relação entre os custos de falhas e o ciclo de vida do produto



Fonte: Adaptado de (BERTSCHE, 2008)

Nos últimos anos, novas funcionalidades de *software* embarcadas no veículo estão sendo integradas para atender exigências de segurança estabelecidas na (ISO 26262, 2018) requisitos de conforto, exigências ambientais do Programa de Controle da Poluição do Ar por Veículos Automotores - PROCONVE (MILLER, et al., 2019), funcionalidades de entretenimento e características de segurança cibernética aderentes a (ISO 21434, 2021).

A cada inserção de novo *software* a indústria automotiva enfrenta desafios especialmente na questão da qualidade. Falhas de *software* geram perdas de reputação da marca bem como prejuízos financeiros com campanhas conhecidas como *recalls*. A Tabela 1 apresenta um resumo dos principais problemas de qualidade destacados no segmento automotivo.

Tabela 1. *Recalls* no segmento automotivo

Organização	Descrição
Stellantis	**60413** carros falhas de pressão alta na bomba de combustível (MOTORSAFETY, 2022)
Nissan	Sistema de ar-condicionado eletrônico com vulnerabilidades (ENGADGET, 2019)
Tesla	**53000** carros com falhas na funcionalidade *Eletronic Parking Brake* (EPB) (BBC, 2017)
FCA	**1.4** milhões de veículos equipados com sistema de entretenimento com vulnerabilidades (WIRED, 2015)
Takata	**33.8** milhões de veículos equipados com a funcionalidade do airbag (HUNTER, 2015)
Toyota	**7.5** milhões de veículos equipados com sistema de aceleração indesejada (VLASIC e APUZZO,2014)

Fonte: O autor.

Nota-se que os casos que afetam a qualidade estão presentes em diversas organizações com prejuízos financeiros significativos. As ocorrências variam entre 53 mil a 33,8 milhões, conforme os registros confirmados por Tesla e Takata, respectivamente.

No artigo de (PANCIK, et al., 2018), são destacados os prejuízos financeiros causados pelas falhas de *software* que resultam em *recalls*. O estudo de (PALIOTTA, 2015) destaca a importância da qualidade do código e testes de *software*. Além disso, enfatiza que a melhoria na qualidade de *software* tem o poder de transformar a reputação das marcas automotivas da mesma forma que a pesquisa de W. Edwards Deming (WILEY, 2012) transformou a manufatura automotiva.

Em função dos inúmeros casos de falhas de *software* no segmento automotivo, a organização americana de administração nacional de segurança de trafego rodoviário (do inglês *National Highway Traffic Safety Administration* - NHTSA) elaborou um *website* cujo objetivo é auxiliar os proprietários a pesquisar se existe um *recall* necessário para seu veículo (NHTSA, 2023).

Este estudo propõe como contribuição implementar métodos científicos para auxiliar na análise de riscos durante o desenvolvimento de *software*, reduzindo os custos das correções de não conformidades. Como benefícios, destacam-se a melhoria da qualidade e confiabilidade do produto como fatores impactantes em aplicações no desenvolvimento de *software* para o setor automotivo.

OBJETIVO

O objetivo deste trabalho é propor uma metodologia para apoio à tomada de decisão no desenvolvimento e correção de falhas de *software* embarcado automotivo.

1.1.1 Objetivos específicos

Os objetivos específicos deste trabalho são apresentados a seguir:

1) Desenvolver uma metodologia no tratamento e coleta de dados relativos as funcionalidades de *software* automotivo;
2) Aplicar o método FMEA tradicional na avaliação de funcionalidades de *software*;
3) Aplicar o método FMEA estendido na avaliação de funcionalidades de *software*;
4) Aplicar o método DEA para avaliação das eficiências e ineficiências das funcionalidades de *software*;
5) Analisar e comparar os resultados dos diferentes métodos;
6) Propor melhorias no processo de desenvolvimento das funcionalidades de *software*.

ORGANIZAÇÃO DO TRABALHO

A introdução dessa dissertação encontra-se no Capítulo 1, a qual apresenta uma visão geral das justificativas para a realização desse trabalho e lista o objetivo geral e específicos a serem atingidos.

O Capítulo 2 apresenta a fundamentação teórica embasada sobre as pesquisas, discussões e aplicações de autores referente aos métodos de análise de risco utilizados na literatura.

O Capítulo 3 apresenta a proposta de pesquisa para o segmento automotivo referente aos desafios da garantia da qualidade de funcionalidades de *software*.

O Capítulo 4 discorre os conceitos do FMEA tradicional e apresenta a extensão deste método pela inclusão de dois valores adicionais na análise. Como proposta complementar, o método DEA é implementado para a avaliação da eficiência nas funcionalidades de *software*.

O Capítulo 5 apresenta o estudo de caso da aplicação dos métodos no segmento automotivo para o desenvolvimento de *software*. Os resultados da implementação dos métodos aplicados são apresentados no Capítulo 6.

Por fim, a conclusão, contribuições e sugestões para extensão deste trabalho estão no Capítulo 7.

FUNDAMENTAÇÃO TEÓRICA

Este trabalho considera o método amplamente utilizado no segmento automotivo denominado FMEA como base para seu desenvolvimento. Embora este método seja extremante utilizado na indústria nas mais variadas aplicações, alguns autores destacam desvantagens na sua implementação. Desta forma, este capítulo apresenta uma visão da aplicação do método FMEA, uma proposta de extensão deste método e adicionalmente o método nomeado de DEA. Estes três métodos são apresentados e implementados no estudo de caso do desenvolvimento de *software* automotivo visando o tratamento de falhas, análise de risco e melhoria na qualidade do *software* neste setor.

Com a complexidade dos requisitos automotivos e a constante evolução tecnológica, a implementação de mecanismos para classificação, priorização e mitigação de riscos no desenvolvimento dos sistemas de *software* embarcados são requeridos para a garantia da qualidade e prevenção dos custos adicionais em projetos. Um sistema é considerado complexo se for composto por vários componentes que interagem entre si e em que a tarefa final não pode ser obtida pela soma de atividades de componentes individuais (BLOKUS-ROSZKOWSKA, et al., 2014).

Uma das alternativas para identificação e classificação de modos de falhas é o uso de FMEA. Esta técnica utiliza por definição a identificação de conhecidas e potenciais falhas, problemas e erros no sistema, projeto ou processo. Desta forma, o resultado possibilita que sejam realizadas correções antes que alcancem o cliente (STAMATIS, 2003). No trabalho de (COOPER, 2015), foi proposto a utilização de uma simples tabela de decomposição que captura todos os potenciais modos de falha e pela aplicação do FMEA prioriza-os pela criticidade de cada um, sugerindo ao decisor que as falhas sejam tratadas segundo esta prioridade. Os resultados desta proposta destacaram uma estimativa de economia de tempo de trabalho da equipe na elaboração do FMEA. Em (QIAN, et al., 2017) é aplicada uma nova avaliação da metodologia FMEA no segmento de fundição, considerando um índice de eficácia FMEA (do inglês FMEA *Effectiveness Index* - FEI), a fim de melhorar a eficiência dos processos de manufatura. No estudo publicado por (DUCOTE, et al., 2022) foi apresentado uma análise de FMEA em um sistema de controle de motor a fim de evitar uma condição de descontrole devido a falha de qualquer componente. Em (YOUSEFI, et al., 2017) o FMEA é estudado como um dos métodos usados na avaliação de risco. No entanto, considera-o como incompleto pois utiliza apenas o número de prioridade de risco (do inglês *Risk Priority Number* - RPN) como fator de priorização nas ações corretivas dos riscos. Desta forma, os

autores propõem uma análise robusta de envoltória de dados (do inglês *Robustness Data Envelopment Analysis* - RDEA) com integração do FMEA. No estudo, a priorização dos riscos adicionou dois valores na análise, incluindo custo e duração (saídas) além dos três valores de severidade, ocorrência e detecção (entradas) consideradas pelo método FMEA.

Na literatura o método FMEA é integrado com outros métodos, a fim de fornecer uma análise de risco mais robusta. A integração do FMEA com o método DEA é destacada no trabalho de (JOMTHANACHAI, et al., 2021). Neste estudo, os valores de risco considerados no FMEA são configurados para avaliação no método de eficiência DEA, cujo objetivo é propor um gerenciamento de riscos. Para o tratamento e monitoramento do risco, um mecanismo de aprendizado de máquina (do inglês *machine learning*) é utilizado a fim de prever o grau de risco remanescente em função de dados simulados.

Uma perspectiva de análise de funcionalidades de *software* é considerada no estudo de (BAGHERI, et al., 2010) utilizando o método conhecido como processo estratificado de hierarquia analítica (do inglês *Stratified Analytic Hierarchy Process* - S-AHP). O objetivo neste estudo é implementar o método a fim de priorização (ranqueamento) e filtro das funcionalidades de uma família de produto para melhorar e agilizar o recurso de seleção das funcionalidades e configuração do produto.

DESENVOLVIMENTO DE SOFTWARE AUTOMOTIVO

O desenvolvimento de *software* automotivo consiste em projetar, criar, testar e manter sistemas de *software* utilizados em veículos. Esse sistema de *software* pode ser desenvolvido para diversas aplicações, tais como: controle de motor, sistema de entretenimento, funcionalidades de conectividade, sistema de conforto e segurança, dentre outros. A indústria automotiva evoluiu rapidamente na última década, e o desenvolvimento de *software* assumiu um papel crucial nessa evolução. Os veículos modernos tornaram-se cada vez mais complexos e conectados, e o *software* tornou-se uma parte essencial nas aplicações de funcionalidades automotivas. Algumas das áreas em que o desenvolvimento de *software* teve um impacto significativo na indústria automotiva incluem:

1. Condução autônoma: as funcionalidades de *software* que permitem que os veículos operem de forma autônoma representam uma das áreas de foco mais importante da indústria automotiva. Isso envolve o desenvolvimento de sensores avançados, algoritmos de aprendizado de máquina e outras tecnologias que contribuem para que os carros percebam e reajam ao ambiente em tempo real.

2. Conectividade: os veículos modernos estão cada vez mais conectados, e o *software* desempenha um papel crucial para possibilitar essa conectividade. Os sistemas de infoentretenimento, por exemplo, fornecem acesso a música, notícias, navegação e outros recursos que exigem conexão com a internet ou outros dispositivos.

3. Segurança: as funcionalidades de *software* têm melhorado a segurança nos veículos através do desenvolvimento de sistemas avançados de assistência ao motorista (do inglês *Advanced Driver Assistance System* - ADAS), como avisos de saída de faixa, frenagem automática de emergência e controle de cruzeiro adaptativo, que ajudam os motoristas a evitar acidentes.

4. Manutenção e diagnóstico: funcionalidades de *software* também são usadas para monitorar e diagnosticar problemas com os sistemas de um veículo. Essas funcionalidades podem auxiliar na identificação de possíveis problemas antes que se tornem sérios, permitindo reparos e manutenção em tempo hábil.

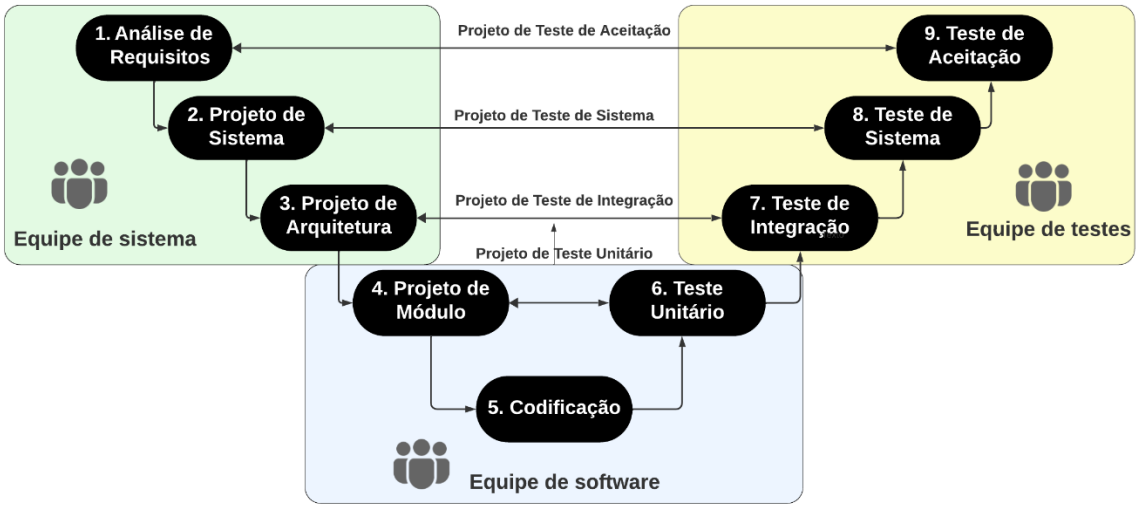
Além das funcionalidades destacadas acima, as perspectivas do mercado global na China, Estados Unidos e Europa convergem para o desenvolvimento de soluções de sistemas embarcados para atender o veículo elétrico *(*do inglês *Eletric Vehicle* - EV) (JULIUSSEN, 2022). Em contrapartida, no Brasil existem obstáculos para o crescimento deste mercado, em específico pela falta de uma política federal para veículos elétricos leves, limitando o crescimento do setor nacional (FAPESP, 2023). No geral, o desenvolvimento de funcionalidades de *software* no segmento automotivo tem contribuído para a evolução do sistema veicular e continuará a desempenhar um papel crucial na formação do futuro do transporte.

PROCESSO DE DESENVOLVIMENTO DE SOFTWARE

No segmento automotivo, o desenvolvimento de *software* embarcado nos veículos apresenta milhares de linhas de código e funcionalidades complexas que integram o sistema da arquitetura veicular. Para a implementação destas funcionalidades de *software* com qualidade, equipes multiculturais e multidisciplinares exercem um papel importante no desenvolvimento. Diante destes desafios, o processo especificado na ISO 15504 conhecido como *Automotive Software Process Improvement Capability Determination* (ASPICE, 2017) é definido como as melhores práticas para o desenvolvimento de *software* embarcado para veículos.

ASPICE é baseado no modelo V, conhecido como cascata. Neste modelo existe para cada fase do processo uma avaliação específica de testes. A Figura 3 apresenta as fases do processo de desenvolvimento de *software*.

Figura 3. Representação do processo de desenvolvimento de *software* – Modelo V



Fonte: Adaptado de (ASPICE, 2017)

O lado esquerdo do modelo V consiste das fases do desenvolvimento de *software*. O lado direito representa as fases de testes que interagem com cada fase do desenvolvimento. O detalhamento de cada fase é descrito conforme apresentado abaixo:

1. Análise de Requisitos: Coleta, identificação e priorização dos requisitos do cliente

2. Projeto de Sistema: Mapeamento das necessidades do cliente e criação de requisitos de sistema

3. Projeto de Arquitetura: Organização dos requisitos dentro de operações lógicas

4. Projeto de Módulo: Criação de requisitos de *software* que alinham com os requisitos de sistema e desenvolvimento de unidades de serviço

5. Codificação: Implementação do código das unidades de *software*

6. Teste Unitário: Teste para detecção se o código e o projeto da unidade de *software* corresponde aos padrões e requisitos dos módulos

7. Teste de Integração: Teste para avaliação da arquitetura de *software* e unidades de serviço

8. Teste de Sistema: Teste de integração completa do sistema

9. Teste de Aceitação: Testes finais de aceitação

CONCEITO DE QUALIDADE

A palavra qualidade vem do Latim *qualis*, ou seja, como é constituído (KAPUR, et al., 2014). O conceito de qualidade é definido por alguns estudiosos com perspectivas distintas.

Segundo Philip Crosby (CROSBY, 1985), qualidade está associada a conformidade com os requisitos. Os custos da qualidade estão relacionados com a definição de requisitos, ou seja, quanto maior a complexidade do produto ou serviço, maiores serão os custos. Assim, se a qualidade pode ser associada à conformidade, deduz-se que as falhas e as medidas que visem evitá-las acarretam um custo. Na visão de Edward Deming, qualidade é um grau previsível de uniformidade e confiabilidade, de baixo custo e adequado ao mercado (DEMING, 1990). A sociedade americana para a qualidade (do inglês *American Society for Quality – ASQ)*, define qualidade como a totalidade de recursos e características de um produto ou serviço que influenciam a sua capacidade de satisfazer as necessidades de um dado usuário (ASQ, 2023) (SOMMERVILLE, 2015). Na perspectiva de Joseph Juran (JURAN, 1951), qualidade é adequação ao uso e finalidade. Os custos da qualidade são aqueles que não deveriam existir se o produto saísse perfeito da primeira vez. O autor associa custos da qualidade com as falhas na produção que levam a retrabalho, desperdício e perda de produtividade. Para Armand Feigenbaum (FEIGENBAUM, 1990), os custos operacionais da qualidade estão associados à definição e planejamento, criação de controle da qualidade, assim como à avaliação e realimentação da conformidade com exigência em requisitos de desempenho, confiabilidade, segurança e também custos associados às consequências provenientes de falhas, em atendimento a essas exigências, tanto internamente nas organizações quanto nas mãos dos clientes. Um importante conceito que conecta dinheiro e qualidade é o termo custo da qualidade (do inglês *Cost of Quality* - COQ). Ele é destacado por Joseph Juran no livro Manual de Controle de Qualidade (JURAN, 1951). A extensão deste estudo foi elaborado por Armand Feigenbaum com o Controle de Qualidade Total (1956). O conceito inclui três tipos de custos: custo de prevenção, custo de avaliação e custo de falha composto de duas parcelas (interna e externa). O custo da boa qualidade (do inglês *Cost of Good Quality* - CoGQ) contempla custos de prevenção e custos de avaliação, que são atividades destinadas a reduzir as falhas e manter níveis aceitáveis de qualidade do produto. Em contrapartida, o custo da má qualidade (do inglês *Cost of Poor Quality* - CoPQ) abrange custos de falhas internas e externas, que estão associados a defeitos encontrados antes do produto chegar ao cliente e após o cliente receber o produto. A equação para cálculo do custo da qualidade é apresentada a seguir:

$$COQ = (PC + AC) + (IFC + EFC) \quad (2.1)$$

Onde:

1. COQ representa o custo da qualidade (do inglês *Cost of Quality*)
2. *PC* representa o custo de prevenção *(*do inglês *Prevention Cost*)
3. AC representa custo de avaliação (do inglês *Appraisal Cost*)

4. IFC representa o custo da falha interna (do inglês *Internal Failure Cost*)

5. EFC representa o custo da falha externa (do inglês *External Failure Cost*)

Os custos de prevenção são associados as atividades de projeto, implementação e operação do sistema de gestão da qualidade, ou seja, gastos ocasionados com o propósito de se evitar defeitos. Os custos de avaliação são as taxas que uma organização paga para detectar defeitos em seus produtos antes de entregá-los aos clientes. Este custo representa uma forma de controle de qualidade. Para a maioria das organizações, o dinheiro que seria perdido como resultado da venda de produtos ou serviços defeituosos supera o custo de avaliação. Os custos das falhas internas estão associados as falhas ocorridas e identificadas internamente, antes do produto ou serviço deixar a organização. Os custos das falhas externas podem acarretar perda de negócios, reputação no mercado para as organizações e valores monetários que não se pode mensurar. Portanto, métodos e gerenciamento de riscos são de extrema relevância para que as organizações se esforcem para manter altos padrões de qualidade a fim de evitar tais custos.

PLANEJAMENTO DE QUALIDADE

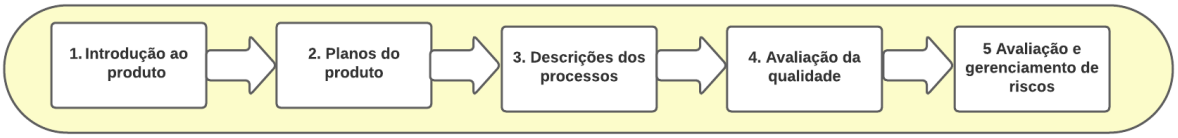
O gerenciamento da qualidade de *software* é um tema de extrema relevância nas organizações devido a constante preocupação na melhoria da qualidade dos produtos que são industrializados. O uso de técnicas de gerenciamento da qualidade, alinhado a métodos de priorização e tratamento de falhas, ferramentas de testes, gerenciamento de riscos bem como o engajamento e colaboração de equipes multidisciplinares, contribuem para melhorias significativas no nível da qualidade de *software*. A indústria desenvolveu um conceito de qualidade baseado na especificação detalhada do produto. A idéia é especificar um produto completo e estabelecer os procedimentos para conferir se os requisitos atendem a especificação. No desenvolvimento de *software* naturalmente são identificadas falhas que não satisfazem aos requisitos de uma especificação. Segundo (SOMMERVILLE, 2015) é impossível chegar a uma conclusão objetiva sobre se um sistema de *software* cumpre ou não a sua especificação pelas seguintes razões:

- Dificuldade na escrita de requisitos de *software* completos e inequívocos;
- Especificações integram requisitos de várias classes de *stakeholders*.

Esses requisitos estabelecem um acordo e podem não incluir os requisitos de todos os grupos de stakeholders. Desta forma, os que foram excluídos podem avaliar o sistema de baixa qualidade, mesmo que implemente e atenda aos requisitos estabelecidos.

• Determinadas características e atributos da qualidade são impossíveis de serem medidos, como exemplo a manutenibilidade. Na perspectiva de (HUMPHREY, 1989) são apresentados os tópicos para agregar o plano de gerenciamento da qualidade conforme representado na Figura 4.

Figura 4. Plano de gerenciamento da qualidade



Fonte: Adaptado de (HUMPHREY, 1989)

1. Introdução ao produto: Descrição do produto, mercado pretendido e expectativas da qualidade para o produto.

2. Planos do produto: Datas críticas para lançamento do produto, responsabilidades do produto e planos de manutenção.

3. Descrições dos processos: Definição dos processos e padrões que serão utilizados no desenvolvimento do produto.

4. Avaliação da qualidade: Elaboração do plano da qualidade e métricas a serem avaliadas.

5. Avaliação e gerenciamento de riscos: Identificação dos riscos que podem afetar a qualidade do produto e as ações para resolvê-los.

No contexto apresentado, o processo é oposto ao manifesto ágil, no qual o objetivo é gastar o mínimo tempo possível em documentos escritos e formalização. O gerenciamento ágil da qualidade não costuma depender de uma equipe da qualidade separada. Por outro lado, baseia-se em estabelecer uma cultura da qualidade na qual indivíduos e interações são mais relevantes do que processos e ferramentas (ATLASSIAN, 2023). O gerenciamento da qualidade de *software* está preocupado em garantir um número baixo de defeitos e que atinja os padrões dos processos e produtos exigidos pela organização.

Os planos de qualidade diferem entre as organizações dependendo da complexidade e do tipo de sistema em desenvolvimento. Há uma grande variedade de atributos de qualidade de *software* potenciais que podem ser considerados no processo de planejamento da qualidade. No plano de qualidade as organizações definem os atributos mais relevantes que serão avaliados e considerados no gerenciamento de riscos. Pode ocorrer de um dos atributos ser mais importante do que o outro no desenvolvimento de um sistema complexo, como exemplo

sistemas embarcados com estratégias de segurança cibernética. A Tabela 2 apresenta os potenciais atributos de qualidade de *software*.

Tabela 2. Atributos de qualidade de *software*

Atributos		
Segurança	Facilidade de compreensão	Portabilidade
Proteção	Facilidade de testes	Facilidade de uso
Confiabilidade	Adaptabilidade	Facilidade de reuso
Facilidade de recuperação	Modularidade	Eficiência
Robustez	Complexidade	Facilidade de aprendizado

Fonte: Adaptado de (SOMMERVILLE, 2015)

QUALIDADE DE PRODUTO DE SOFTWARE

A (ISO 9126, 1991) foi criada em 1991 com o objetivo de enfatizar a garantia de qualidade de *software*. Nesse modelo existe uma estrutura de hierarquia em árvore composta de características, sub características e atributos da qualidade. A (ISO 25010, 2011) substituiu a norma ISO 9126, que era amplamente utilizada para avaliar a qualidade de *software*. Ela foi revisada e atualizada para melhor se adequar às necessidades em constante evolução do desenvolvimento de *software* e sistemas. Esta norma define as seguintes características de qualidade do produto de *software*:

• **Funcionalidade**: Refere-se às capacidades funcionais do *software*, como suas funções, desempenho, compatibilidade, segurança e conformidade com padrões.

• **Confiabilidade**: Avalia a capacidade do *software* de manter um desempenho consistente e livre de erros ao longo do tempo. Inclui critérios como tolerância a falhas e facilidade de recuperação.

• **Usabilidade**: Está relacionada à facilidade com que os usuários podem interagir com o *software* e realizar suas tarefas de maneira eficiente e satisfatória.

• **Eficiência**: Avalia o desempenho do *software* em termos de velocidade, consumo de recursos e utilização de recursos do sistema.

• **Compatibilidade**: Diz respeito à capacidade do *software* de funcionar adequadamente em diferentes ambientes e com diferentes sistemas e dispositivos.

• **Segurança**: Avalia a capacidade do *software* de proteger os dados e garantir a privacidade dos usuários, bem como sua resistência a ameaças e ataques.

• **Manutenibilidade**: Refere-se à facilidade com que o *software* pode ser modificado, atualizado e corrigido ao longo do tempo.

• **Portabilidade**: Avalia a capacidade do *software* de ser adaptado e executado em diferentes plataformas e ambientes. A ISO 25010 oferece um conjunto de atributos mais precisos e abrangentes para avaliar a qualidade de *software*. Ela é útil para os *stakeholders* de desenvolvimento de *software* que desejam garantir a qualidade de produtos de *software* e sistemas.

PADRONIZAÇÃO NO DESENVOLVIMENTO

Um tópico importante para a garantia de qualidade refere-se a padronização do processo e produto no desenvolvimento de *software*. As organizações podem também, em função do tipo de segmento, aplicações e limitações financeiras, escolher técnicas, ferramentas e métodos para apoiar o uso desses padrões. Segundo (SOMMERVILLE, 2015), padronização é importante pelas seguintes razões:

1. Base de conhecimento que tem valor para a organização.

2. Decidir se um nível de qualidade exigido foi alcançado.

3. Garantir que todos os engenheiros dentro de uma organização adotem as mesmas práticas no desenvolvimento. Consequentemente, o esforço de aprendizagem ao iniciar novos projetos é reduzido.

No contexto de padronização, dois tipos podem ser implementados no plano de gerenciamento de qualidade de *software* segundo (SOMMERVILLE, 2015):

• **Padrões de Produto**: Define os padrões de documentos de requisitos, codificação, linguagem de programação e validação.

• **Padrões de Processo**: Define os processos e as boas práticas que serão integrados ao desenvolvimento de *software*.

As organizações direcionam a implementação e boas práticas no desenvolvimento de *software* adaptadas aos padrões nacionais e internacionais, tais como: (ISO 9001, 2015), Melhoria do Processo de *Software* Brasileiro (MPS.BR, 2003), *Institute of Electrical and Electronics Engineers* (IEEE, 2023), *Automotive Spice* (ASPICE, 2017), *Capability Maturity Model Integration* (CMMI, 2023), *Rational Unified Process - RUP* (ANWAR, 2014). A Tabela 3 contém exemplos tanto de padrões de produto quanto de processo no ambiente de desenvolvimento de *software*.

Tabela 3. Padronização de produto e processo

Padrões	
Produto	**Processo**
Formulários de revisão	Conduta para revisão
Estrutura de documento de requisitos	Submissão de código para construção do sistema
Formato de cabeçalho de método	Processo de lançamento de nova versão
Estilo de programação	Processo de aprovação do plano de projeto
Formato de plano de projeto	Processo de controle de mudança
Formulário de requisição de mudança	Processo de registro de teste

Fonte: Adaptado de (SOMMERVILLE, 2015)

Em alguns sistemas complexos são desenvolvidos padrões para atendimento a requisição de características de segurança cibernética (ISO 21434, 2021) e também requisitos de segurança funcional (ISO 26262, 2018). Os processos estão disponíveis e padronizados por entidades regulamentadoras tanto no âmbito nacional quanto internacional. Desta forma, os processos podem ser adaptáveis e definidos para o tipo de *software* que está sendo desenvolvido. A realidade e o segmento das organizações deve ser considerado em sinergia com os *stakeholders* de gerenciamento do projeto e qualidade para sistemas de *software* críticos e complexos na fase de iniciação.

CONTROLE DE QUALIDADE

O controle de qualidade e a verificação da confiabilidade são essenciais para garantir a qualidade geral de um produto. Na fase de projeto a base para a qualidade do produto é estabelecida e depois construída e refinada ao longo do processo de manufatura. Nessa fase são envolvidas diversas áreas para garantir a qualidade e confiabilidade. Um dos fatores mais importantes é garantir que o projeto atenda a todos os requisitos regulatórios. Isto inclui garantir que o produto seja seguro e esteja em conformidade com os padrões estabelecidos. Outro ponto importante é garantir que o projeto seja robusto e confiável. Isto envolve projetar o produto para suportar as condições ambientais do uso pretendido. Isto inclui identificar quaisquer potenciais pontos fracos e resolvê-los antes que o produto seja liberado para a manufatura. Para garantir o controle de qualidade e a verificação da confiabilidade no projeto, as organizações implementam diversas medidas. Uma das principais é o sistema de gestão da qualidade (do inglês *Quality Management System* - QMS) que abrange todos os aspectos do processo do projeto, desde o conceito até a manufatura (ISO 9001, 2015). Este sistema tem como função garantir que todos os produtos atendam aos padrões exigidos de qualidade e confiabilidade e que quaisquer problemas que surjam durante o desenvolvimento sejam resolvidos prontamente. Inclui medidas como testes de produtos, análise de falhas e ações corretivas para garantir que quaisquer problemas sejam resolvidos de forma rápida e eficaz.

Além disso, destacam-se a inclusão do uso de ferramentas avançadas de simulação para modelar o desempenho do produto e identificar possíveis problemas de projeto, juntamente com testes e validação rigorosos de todos os componentes e subsistemas. Ao implementar estas medidas, as organizações conseguem garantir que todos os produtos são concebidos de acordo com os mais elevados padrões de qualidade e confiabilidade, aumentando não só a satisfação dos clientes, mas também contribuindo para a segurança geral e o bem-estar da sociedade. A análise de riscos pode ser realizada qualitativa, quantitativa ou híbrida, com um nível de granularidade, de acordo com as informações coletadas e recursos disponíveis.

GERENCIAMENTO DE RISCOS

Os processos de gerenciamento de riscos estão presentes em diversas organizações como requisitos mandatórios nos sistemas de gerenciamento da qualidade. Há muitas perspectivas sobre a definição de riscos, conforme apresentadas abaixo:

• Risco é a possibilidade de sofrer danos (HAUPTMANNS, et al., 2012);

• Probabilidades associadas com as consequências da atividade, vista em relação a severidade dessas consequências (AVEN, 2015);

• Exposição a chance de lesão ou perda (MORGAN, et al., 1990);

• Um evento com uma probabilidade de ocorrer no futuro impactando o projeto de forma negativa (ameaça) ou positiva (oportunidade) (PMBOK, 2021);

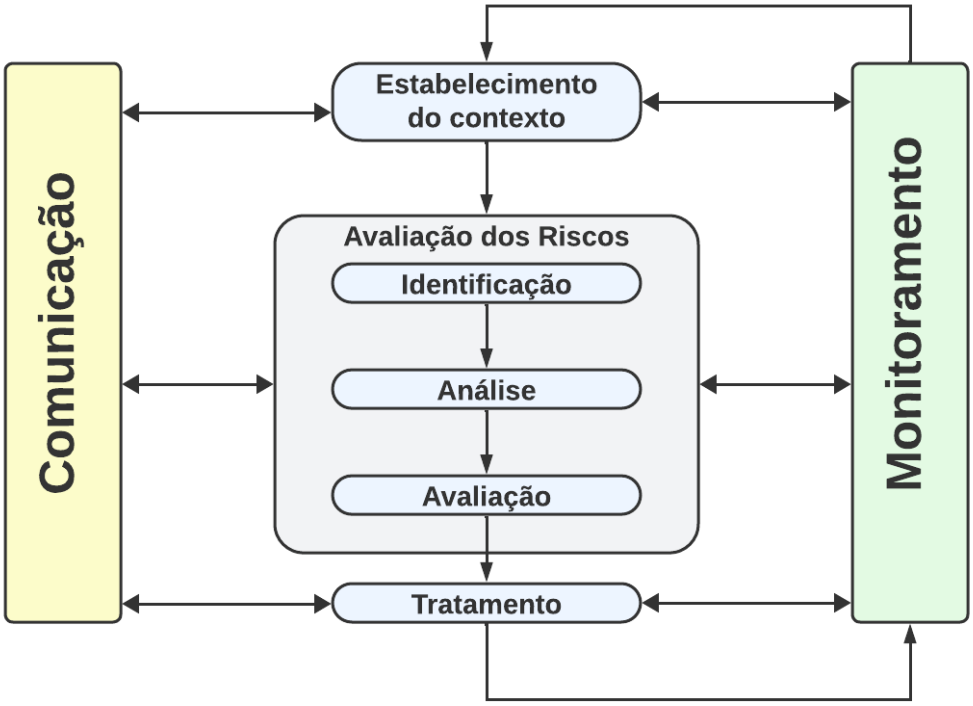
As definições de risco abrangem dois elementos básicos:

1. **Chance**: Incertezas e probabilidades;
2. **Consequências**: Impacto no custo, dano, perigo

No estudo de (KMENTA, et al., 2000) propõem que o risco deveria ser avaliado separadamente para cada cenário considerando a probabilidade e consequência. Diversos especialistas em riscos se uniram em um comitê para estabelecer os aspectos do gerenciamento de riscos definidos na (ISO 31000, 2018). A avaliação de risco nesta perspectiva compreende três etapas: identificação de riscos, análise de riscos e avaliação de risco. A identificação requer a aplicação de um processo sistemático para entender o que poderia acontecer, como, quando e por quê. Nesta norma, a análise de risco preocupa-se em desenvolver uma compreensão de cada risco, suas consequências e as probabilidades envolvidas. Um nível de granularidade pode ser realizado na análise de risco, dependendo das informações e recursos disponíveis. A avaliação de risco envolve a tomada de decisão sobre o nível de risco e a

prioridade, através da aplicação dos critérios desenvolvidos quando o contexto foi estabelecido. O tratamento de riscos é o processo pelo qual os mecanismos de controle existentes são melhorados ou novos controles são desenvolvidos e implementados. Esta parte envolve avaliação e seleção de opiniões, custos e benefícios, sendo que novos riscos podem ser identificados. A comunicação e o monitoramento atuam de forma transversal em todo o processo. A Figura 5 ilustra o gerenciamento de risco da ISO 31000.

Figura 5. Representação do gerenciamento de riscos



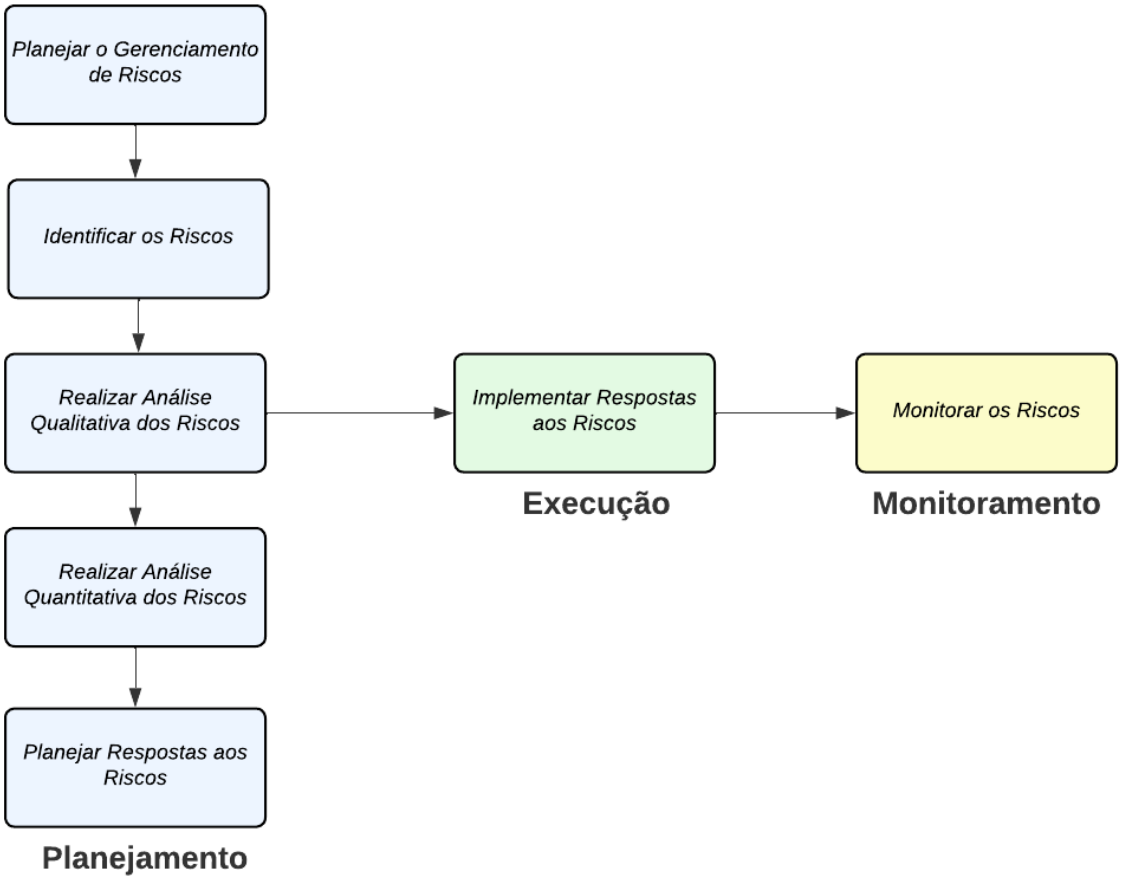
Fonte: Adaptado de (ISO 31000,2018)

No livro sobre gerenciamento de *software* de (HUMPHREY, 1989), é proposto que os riscos mais importantes que podem afetar a qualidade do produto devem ser priorizados com as ações a serem tomadas ao lidar com eles.

Em contrapartida, no desenvolvimento ágil não é apresentado técnicas explícitas de gerenciamento de risco conforme estudos de (TAVARES, et al., 2019). A razão principal é que os ciclos curtos de desenvolvimento iterativo minimizam qualquer impacto segundo os autores. Na visão do (PMBOK, 2021), risco é um evento ou condição incerta que, se ocorrer, terá um efeito positivo ou negativo sobre pelo menos um objetivo do projeto, como tempo, custo, escopo ou qualidade. As organizações percebem os riscos quando estão relacionados a ameaças ao sucesso do projeto. O gerenciamento de riscos de acordo com o Instituto de

Gerenciamento de Projetos (do inglês *Project Management Institute* - PMI) é composto pelos seguintes processos ilustrados na Figura 6.

Figura 6. Representação do gerenciamento de risco do projeto



Fonte: Adaptado de (PMBOK, 2021)

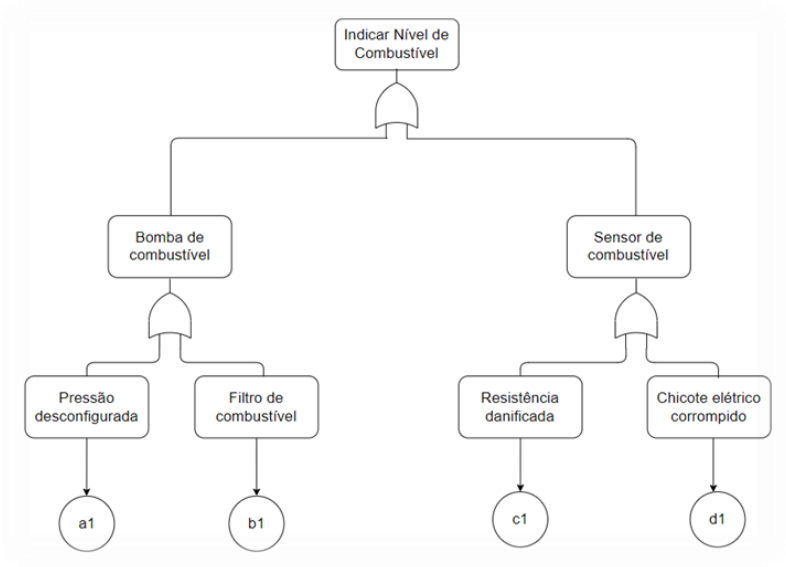
No planejamento do gerenciamento de riscos é decidido como abordar as atividades de gerenciamento de risco do projeto. Este processo é relevante para fornecer visibilidade se os recursos e o tempo são suficientes para atender as necessidades do projeto. A identificação de riscos determina quais fatores podem afetar o projeto. É um processo iterativo, pois em todo o ciclo de vida, novos riscos poderão ser identificados. A equipe do projeto deve ser motivada a participar continuamente neste processo. A análise qualitativa de risco abrange métodos de priorização dos riscos identificados para proposição de ações. As organizações apresentam recursos humanos finitos, e desta forma, concentram os recursos para mitigar os riscos de alta prioridade. A análise quantitativa utiliza os riscos que foram priorizados e analisa o efeito desses eventos de risco atribuindo uma classificação numérica. Neste processo, a tomada de decisão destaca-se como relevante, utilizando de algumas técnicas tais como a árvore de decisão e simulação de Monte Carlo. No planejamento de respostas aos riscos são

desenvolvidas as proposições de ações para aumentar as chances de sucesso e reduzir as ameaças aos objetivos do projeto. A implementação destas ações, devem ser adequadas à realidade do projeto e acordadas com as partes interessadas para que sejam efetivas na execução e mitigação do risco. No processo de monitoramento de riscos é realizado o acompanhamento, ou seja, reanálise dos riscos existentes, monitoramento de riscos residuais, revisão das ações de proposição aos riscos e eficácia da implementação das respostas aos riscos. O monitoramento é contínuo e é fundamental que seja seguido em todo o ciclo de vida do projeto. Na próxima seção serão apresentadas algumas técnicas e métodos para contribuir na análise de falhas.

ANÁLISE DE ÁRVORE DE FALHAS

Uma das técnicas empregadas para identificação e correção das falhas nas fases de desenvolvimento e utilização do *software* é a análise de árvore de falhas. Segundo (HAN, et al., 2013) o conceito de análise de árvore de falha (do inglês *Fault Tree Analysis* - FTA) é uma técnica dedutiva para identificação, avaliação e modelamento do inter-relacionamento entre eventos conduzindo para uma falha ou estado indesejado. Para exemplificar este conceito, considera-se uma funcionalidade de gerenciamento do nível de combustível (CRISTEA, et al., 2017). Os elementos estruturados neste sistema são: a bomba de combustível e o sensor de combustível. Os modos de falha (do inglês *Failure Mode* - FM) que podem ocorrer nos elementos são mapeados através dos ramos. Para o caso da bomba de combustível tem-se: pressão desconfigurada (a1) e filtro de combustível (b1). No caso do sensor de combustível: resistência danificada (c1) e chicote elétrico corrompido (d1). A Figura 7 ilustra esta análise para a funcionalidade da indicação do nível de combustível.

Figura 7. Representação de uma árvore de falhas



Fonte: O autor.

Esta técnica apresenta vantagens e desvantagens para a avaliação e identificação de falhas no sistema. O trabalho de (CRISTEA, et al., 2017) cita como vantagens:

1. Permite que partes potenciais de falhas do processo sejam vistas em detalhes;

2. Ajuda a identificar falhas dedutivamente;

3. Habilita uma análise qualitativa e quantitativa do processo a ser feito;

4. O método pode focar em partes individuais do processo, e pode extrair falhas específicas;

5. Representa de forma clara o comportamento do processo;

Como desvantagens são citadas:

1. Implementação do método requer consideráveis entradas e os números de eventos que influenciam aumentam consideravelmente;

2. A árvore de falha apresenta apenas dois estados: ativado e falha;

3. Dificuldade na estimativa do estado parcial de falha das partes do processo;

4. Requer um profundo conhecimento do processo por parte do especialista para que a análise seja confiável.

Embora a análise de árvore de falhas empregue lógica booleana para combinação de eventos que causam um estado indesejado no sistema, a técnica é considerada estática e incapaz de examinar múltiplas falhas (SINNAMON, et al., 1997). Além disso, é uma técnica limitada na modelagem para sistemas complexos, pois não é intuitiva e carece da capacidade de contabilizar adequadamente as interações dinâmicas entre os componentes.

1.1.2 A evolução da FMEA

FMEA foi um dos primeiros métodos sistemáticos para análise de falhas. O Exército dos Estados Unidos aplicou FMEA com uso da referência militar MIL-P-1629 (MIL-P1629, 1949) no setor da aeronáutica inicialmente em 1949. O objetivo era identificar potenciais efeitos em sistemas e falhas em equipamentos. Na indústria foi aplicado ao sistema de controle de vôo de aeronaves da marinha americana pela *Grumman Aircraft Corporation*. No início dos anos 1960, a Administração Nacional Aeronáutica e Espacial (do inglês *National Aeronautics and Space Administration* - NASA) aplicou o FMEA nos programas como *Apollo, Viking e Voyager*, tendo sido desenvolvido pela primeira vez como uma metodologia formal de aplicação em 1963. Em 1967 a organização Sociedade dos Engenheiros Automotivos (do inglês *Society of Automotive Engineers* – SAE) publicou a norma ARP 926 considerando uma extensão da FMEA nomeada de modo de falha e efeito e análise de criticidade (do inglês *Failure Mode and Effect and Criticality Analysis* - FMECA) (SAE,1967). Ela inclui uma análise de criticidade com o objetivo de avaliar a probabilidade dos modos de falha e a gravidade de suas consequências. Em 1972, a NASA preparou um relatório recomendando o uso da FMEA na avaliação da exploração de petróleo (DYER et al., 1972). Em fevereiro de 1973, um relatório da Agência de Proteção Ambiental dos Estados Unidos (do inglês *Environmental Protection Agency* - EPA) descreveu a aplicação do método em estações de tratamento de águas residuais (MALLORY, et al., 1973). O procedimento militar MIL-P-1629 foi revisado pelo Departamento de Defesa (do inglês *Department of Defense* - DoD) dos Estados Unidos em 24 de novembro de 1980, surgindo a primeira versão da norma MIL-STD-1629A. Esta norma teve atualizações em 1983 e 1984, até ser cancelada em 1998 (MIL-STD1629A, 1998). Na Europa a Comissão Eletrotécnica Internacional (do inglês *International Eletrotechnical Commission* - IEC) publicou em 1985 a norma IEC 812 (IEC, 1985) abordando a FMEA e a FMECA para uso geral. Esta norma foi substituída pela IEC 60812 (IEC60812, 2018) sendo a última revisão válida de 2018. Na década de 1980, a FMEA ganhou amplo uso na indústria automotiva para avaliar falhas relacionadas ao projeto e processo. Em 1993 o Grupo de Ação da Indústria Automotiva (do inglês *Automotive Industry Action Group* - AIAG) e a Sociedade Americana para Qualidade (do inglês *American Society for Quality* – ASQ) uniram-se com as montadoras automotivas e criaram um manual de referência para a FMEA. A organização SAE publicou o primeiro padrão relacionado a norma J1739 referente a análise do modo de falha potencial e dos efeitos no projeto (do inglês *Potential Failure Mode and Effects Analysis in Design – DFMEA)* e análise de modo de falha

potencial e efeitos em processos de fabricação e montagem (do inglês *Potential Failure Mode and Effects Analysis in Manufacturing and Assembly Processes - PFMEA*) que também teve atualizações, sendo a última disponibilizada em 2021 (SAE, 2021). Em 1996 a Associação da Indústria Automotiva Alemã (do alemão *Verband der Automobilindrustie* - VDA) publicou a primeira e única versão do guia VDA 4.2 – Garantia da qualidade antes da produção em série – Sistema FMEA. Na década de 2000, não houve o surgimento de novos procedimentos ou normas específicas, apenas atualizações de requisitos normativos existentes. Um considerável aumento na quantidade de organizações que passaram a aplicar FMEA no desenvolvimento de seus produtos e processos foram associados pela requisição da ISO Especificação Técnica (do inglês *Technical Specification* - TS) 16949 em 2002. Esta norma foi obrigatória a aplicação de FMEA em toda cadeia produtiva do segmento automotivo. Ela tornou-se obsoleta em 2016 sendo substituída pela norma IATF 16949, com os mesmos requisitos mandatórios (IATF, 2016). Em 2019, ocorreu a harmonização dos requisitos entre AIAG e VDA com a publicação do Manual de FMEA AIAG e VDA a ser usado como referência para os fornecedores da indústria automotiva no desenvolvimento do projeto (DFMEA), processo (PFMEA) e também da operação através do FMEA Suplementar para Monitoramento e Resposta do Sistema (do inglês *Monitoring and System Response* - MSR) (AIAG/VDA, 2019). Embora inicialmente a metodologia tenha sido desenvolvida pelos militares, FMEA tem sido amplamente aplicado em uma variedade de indústrias ao longo das décadas. No entanto, foi no ramo automotivo que houve maior disseminação de sua utilização, não apenas por ser um dos requisitos obrigatórios para os sistemas de gestão da qualidade, mas também pela evolução do número de organizações pertencentes ao segmento automotivo.

1.1.3 O conceito FMEA

FMEA tem como objetivo analisar falhas, efeitos e causas de um sistema (BERTSCHE, 2008). A análise inicia-se do nível de componente onde os modos de falha são identificados e suas consequências examinadas. No segmento automotivo, esta metodologia analítica é aceita por várias montadoras automotivas tais como: Stellantis, Ford, Honda, General Motors, BMW, Volkswagen, Mercedes (AIAG_VDA_FMEA, 2023). A metodologia FMEA abrange dois conceitos de falha e defeitos:

- **Falha**: Falta de capacidade de um item em atender a sua função;
- **Defeito**: Refere-se a uma não conformidade do produto em relação aos requisitos do cliente;

As falhas de uma função são derivadas das descrições da função. Há vários tipos de falhas potenciais conforme referenciado no manual do FMEA (AIAG/VDA, 2019):

- **Perda de função**: refere-se a função inoperante.
- **Degradação da função**: refere-se a função com perda de desempenho ao longo do tempo.
- **Função intermitente**: refere-se a função randômica (estado de funcionamento variável entre funciona e não funciona).
- **Função parcial**: refere-se a função com parte de requisitos sendo atendidos e outros inoperantes.
- **Função não intencional**: refere-se a função que opera de forma errônea e um determinado tempo.
- **Função extrapolada**: refere-se a função que opera acima dos limites aceitáveis.
- **Função atrasada**: refere-se a função que opera após um intervalo de tempo não intencional.

Na execução do FMEA, tipicamente é envolvido um time composto de pessoas multiculturais e multidisciplinares com diferentes perspectivas (exemplo: mecânica, *software*, manutenção, produção, *hardware*) a fim de aumentar a probabilidade de que todas as falhas sejam identificadas e seus efeitos estimados corretamente (TINGA, 2013). Este método parte da identificação da falha e determinação da estatística dos valores de severidade (S), ocorrência (O) e detecção (D) de cada componente para obter a criticidade de cada modo de falha. O resultado do produto destes valores constitui-se em um indicador da prioridade de risco (do inglês *Risk Priority Number* – RPN) e está associado ao modo de falha de cada componente de *software* que coloca em risco determinada funcionalidade. O RPN considera o produto entre três valores para representação do risco. Este método contribui para a tomada de decisão e ações para a melhoria na qualidade durante o desenvolvimento do produto. Na Eq. 2.2 é apresentado a formulação do RPN.

$$RPN = S * O * D \tag{2.2}$$

Mecanismos de controle nas organizações podem auxiliar na identificação de falhas potenciais, tais como: revisão do projeto, inspeções, testes de verificação e validação e estimar a probabilidade de uma falha. A Tabela 4 apresenta uma visão geral da pontuação utilizada no método FMEA.

Tabela 4. Critério de pontuação para severidade, ocorrência e detecção

Rank	Severidade	Ocorrência	Detecção
10	Perigoso sem alerta	Muita alta: falha é quase inevitável	Absoluta incerta
9	Perigoso com alerta	Muita alta: falha é quase inevitável	Muito remota
8	Muito alta	Alta: repetidas falhas	Remota
7	Alta	Alta: repetidas falhas	Muito baixa
6	Moderada	Moderada: falhas ocasionais	Baixa
5	Baixa	Moderada: falhas ocasionais	Moderada
4	Muito baixa	Baixa: relativamente algumas falhas	Moderadamente alta
3	Menor	Baixa: relativamente algumas falhas	Alta
2	Muito menor	Remota: falha é improvável	Muito alta
1	Nenhuma	Remota: falha é improvável	Quase certa

Fonte: Adaptado de (YOUSEFI, et al., 2017)

O estudo de (GUEORGUIEV, et al., 2020) apresentaram as recentes tendências na metodologia FMEA com aplicação na indústria automotiva. Enfatizam que o último guia do FMEA revisado (AIAG/VDA, 2019) utiliza 7 passos para desenvolvimento do FMEA:

1. Planejamento e Preparação: Nesta etapa o objetivo é definir o tipo de FMEA e o escopo que será implementado com base na análise sendo desenvolvida (sistema, subsistema ou componente).

2. Análise Estrutural: Identificar e dividir o escopo FMEA em sistema, subsistema e componentes para análise técnica de risco.

3. Análise Funcional: Garantir que as funções especificadas por requisitos estão apropriadamente alocadas aos elementos do sistema.

4. Análise de Falha: Identificar causa das falhas, modos, efeitos e mostrar suas relações para permitir a avaliação de risco.

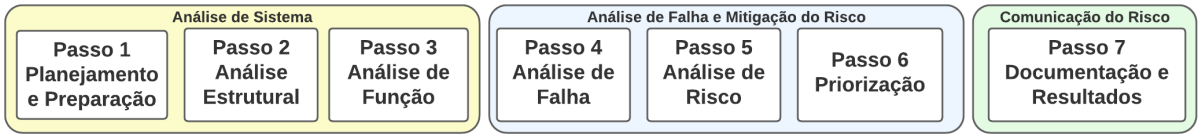
5. Análise de Risco: Estimar o risco avaliando a severidade, ocorrência e detecção, e priorizando as ações.

6. Priorização: Determinar ações para mitigar riscos e avaliar a eficácia dessas ações.

7. Documentação e Resultados: Resumir e comunicar os resultados da atividade do FMEA.

A Figura 8 ilustra os passos para o método FMEA.

Figura 8. Representação dos passos do FMEA



Fonte: Adaptado de (AIAG/VDA, 2019)

O FMEA permite que as organizações identifiquem e resolvam proativamente falhas potenciais antes que elas ocorram, resultando em um ambiente de trabalho colaborativo e mais seguro, com maior qualidade do produto, maior eficiência e maior satisfação do cliente. No guia de referência FMEA (AIAG/VDA, 2019) é apresentado o método de prioridade de ação (do inglês *Action Priority* - AP) que oferece uma abordagem de prioridade alta, média e baixa para cada ação. Assim, as organizações podem utilizar um único sistema para avaliar as ações em vez de vários sistemas específicos requeridos pelos clientes. A aplicação do método AP pode ser encontrada em (ANACKOVSKI, et al., 2021) e (BARSALOU, 2022). A Tabela 5 representa os critérios para a ação recomendada referente a prioridade.

Tabela 5. Representação dos critérios de prioridade da ação

Prioridade	Ação Recomendada
Alta	A equipe precisa identificar uma ação para melhorar a prevenção e/ou controles de detecção
Média	A equipe deveria identificar ações para melhorar a prevenção e/ou controles de detecção
Baixa	A equipe poderia identificar ações para melhorar controles de prevenção ou detecção

Fonte: (AIAG/VDA, 2019)

Considerando apenas dois valores na análise do FMEA, ou seja, severidade (S) e ocorrência (O), o método FMECA foi apresentado. Este método usa probabilidade para medir a chance de uma falha em relação as categorias de severidade. Destaca-se em comparação ao uso do número de prioridade de risco (RPN) por vários motivos conforme destacado no estudo de (KMENTA, et al., 2000):

- a frequência de falhas é medida com probabilidade;
- o índice de detecção é eliminado;
- medidas ordinais não são multiplicadas.

1.1.4 FMEA Suplementar

Na atualização do manual do FMEA (AIAG/VDA, 2019) é destacado o método nomeado de FMEA-MSR. Ele fornece um meio para a análise de detecção de diagnóstico e mitigação das falhas durante a operação do veículo, cuja finalidade é manter um estado seguro ou estado de conformidade regulatória. Neste método, a prioridade da ação é baseada em combinações de severidade (S), frequência (F) e monitoramento (M) para priorizar as ações e reduzir os riscos. Os três elementos utilizados neste método são detalhados abaixo:

• **Severidade (S):** gravidade do dano, perda da funcionalidade ou descumprimento regulatório.

• **Frequência (F):** estimativa de uma causa de falha no contexto de uma situação operacional.

• **Monitoramento (M):** possibilidade de técnicas para evitar ou limitar o efeito da falha através da detecção de diagnóstico e resposta automatizada, combinado com possibilidades humanas para evitar ou limitar o efeito de falha. A combinação de frequência (F) e monitoramento (M) refere-se a uma estimativa da probabilidade de ocorrência do efeito de falha devido à causa da falha e comportamento de mau funcionamento (modo de falha). FMEA-MSR é aplicado para garantir que os objetivos de segurança funcional requeridos na ISO 26262 são integralmente considerados e atendidos no ciclo de vida de sistemas elétricos e eletrônicos. Embora a ISO 26262 seja um padrão da indústria automotiva, complementar a aplicação do FMEA com conceitos de segurança funcional é uma boa prática para o desenvolvimento de sistemas eletrônicos na indústria.

1.1.5 FMEA Estendido

O método FMEA é aplicado no segmento automotivo segundo os requisitos estabelecidos na IATF 16949. No entanto, na comunidade científica este método é questionado por apresentar algumas desvantagens enfatizadas no estudo de (YOUSEFI, et al., 2017) e relatadas a seguir:

- Ênfase em riscos com maior RPN mas com baixa severidade;
- Falta de consideração de outros importantes indicadores na avaliação de risco;
- Falta de consideração de incertezas em alguns indicadores;
- Baixo poder de discernimento;

Nos estudos de (CHIN, et al., 2009) são destacadas as seguintes desvantagens do FMEA:

- Diferentes combinações de ocorrência, severidade e detecção podem produzir exatamente o mesmo valor de RPN;
- A importância relativa entre ocorrência, severidade e detecção não é levada em consideração.
- Os três valores são assumidos serem igualmente importantes;
- A fórmula matemática para cálculo do RPN é questionável e debatível;
- Não há uma razão específica para a multiplicação da severidade (S), ocorrência (O) e detecção (D) produzir o RPN;

Na perspectiva de (KMENTA, et al., 2000) são destacadas estratégias de implementação para melhorar a eficácia do FMEA tradicional:

1. Organizar o FMEA por meio de cenário de falha ao invés de modos de falha;
2. Avaliar o risco usando probabilidade e custo;

O método pode ajudar a atribuir custos a cenários de falha e orientar decisões de projeto com mais precisão. Além disso, os autores acreditam que o FMEA baseado em cenários ajudará nos desafios de projeto e suporte de longo prazo para produtos complexos.

ANÁLISE DE ENVOLTÓRIA DE DADOS (DEA)

DEA é uma técnica não paramétrica usada para avaliar a eficiência relativa das unidades de tomada de decisão (do inglês *Decision Making Unit* - DMU) que convertem entradas e saídas. A técnica foi inicialmente proposta em 1978 para avaliação das organizações de ensino e determinação da eficiência (CHARNES, et al., 1978). DEA tem sido implementado em uma variedade de aplicações conforme listado na Tabela 6.

Tabela 6. Aplicações do método DEA

Aplicação	Autores
Sistema de eletricidade	(COSTA, et al., 2023)
Aeroporto	(MONTOYA-QUINTERO, et al., 2022)
Agricultura	(ÁLVAREZ, et al., 2020)
Cooperativas de crédito	(TEXEIRA, et al., 2020)
Logística	(YI, et al., 2019)
Riscos no setor automotivo	(YOUSEFI, et al., 2017)

Recursos públicos na educação	(FLACH, et al., 2017)
Clubes de futebol	(FREITAS, et al., 2017)
Produção nas províncias da China	(WANG, et al., 2015)
Modelo de estimativa	(BANKER, et al., 1984)
Educação	(CHARNES, et al., 1978)

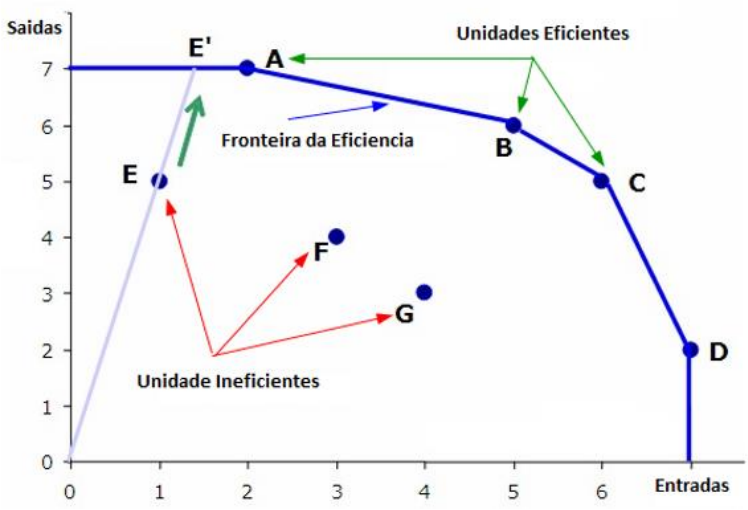
Fonte: O autor.

No estudo de (EMROUZNEJAD, et al., 2018) é apresentado uma revisão da literatura do uso do método DEA desde sua origem em 1978 até 2016. Os setores que destacam-se na aplicação deste método são: agricultura, bancário, logística, transporte e políticas públicas. Segundo (PEÑA, 2008) o conceito de eficiência é a combinação ótima dos insumos e métodos necessários (entradas) no processo produtivo de maneira que resulte no máximo de produtos possíveis (saídas), isto é, eficiência é a capacidade de fazer certo as coisas, de minimizar a relação insumo e produto, e desse modo, maximizar a utilização de recursos. A eficiência é representada pela letra grega θ e pode ser formulada conforme Eq. 2.3.

$$\theta = \frac{Saída}{Entrada} \qquad (2.3)$$

Uma vantagem do DEA é que ele pode lidar com várias entradas e saídas simultaneamente, o que o torna útil para analisar ambientes complexos de tomada de decisão. DEA é baseada em programação linear para a medição da eficiência relativa entre unidades alternativas considerando várias entradas e saídas. Identifica as unidades eficientes segundo critérios pré-estabelecidos podendo também servir como um elemento identificador de unidade ineficiente, ou como um enfoque para o estabelecimento de metas eficientes para cada unidade produtiva (CHARNES, et al., 1978). O uso deste método para medir a eficiência relativa de organizações e unidades produtivas tem-se mostrado bastante atrativo em diversos setores de aplicação, além de apoiar decisões de agentes públicos e empresas privadas, bem como indicar as fontes de ineficiência e as unidades eficientes que podem servir de referência às práticas adotadas. A Figura 9 apresenta unidades em análise eficientes e não eficientes.

Figura 9. Fronteira de eficiência das DMUs



Fonte: Adaptado de (CHARNES, et al., 1978)

As unidades eficientes (A,B,C,D) definem a fronteira de eficiência. Enquanto as unidades (E,F,G) estão dentro de uma região de ineficiência. A unidade E, não eficiente, deveria caminhar até o ponto E' no sentido de tornar-se eficiente.

O método DEA calcula a distância de cada unidade com relação a fronteira de eficiência que é um limite hipotético de eficiência das DMUs. Todas as unidades que se encontram neste limite são consideradas eficientes. Quanto mais distante da fronteira de eficiência, menos eficiente é a unidade. Assumindo como exemplo um conjunto com n unidades de decisão DMUs (j = 1,...n) cada qual usando x_{ij} entradas (i = 1, ...m) e gerando as saídas y_{rj}(r = 1, ..s), que são conhecidas e não negativas. A eficiência (θ_j) de uma DMU_j é definida conforme a Eq. 2.4.

$$\theta_j = \frac{\sum_{r=1}^{s} u_r y_{rj}}{\sum_{i=1}^{m} v_i x_{ij}} \qquad (2.4)$$

Os multiplicadores u_r e v_i representam os pesos, conhecidos como variáveis de decisão associados às saídas e entradas respectivamente. A Eq. 2.4 tem como objetivo encontrar o conjunto de coeficientes (u) associados a cada saída, e de coeficientes (v) associados a cada entrada que resultará na máxima eficiência para a unidade de tomada de decisão (DMU) avaliada. Nesta equação (j) representa o índice da unidade que está sendo avaliada.

Para determinar a eficiência da DMU_j relativa para outras DMUs, (CHARNES, et al., 1978) desenvolveram o modelo nomeado de CCR (*Charnes, Cooper e Rhodes*) para medir as melhores eficiências relativas de DMUs conforme apresentado na Eq. 2.5. O zero subscrito representa a DMU em avaliação.

$$Max\ \theta_j = \frac{\sum_{r=1}^{s} u_r y_{r0}}{\sum_{i=1}^{m} v_i x_{i0}} \qquad (2.5)$$

Sujeito as seguintes restrições:

$$\theta_j = \frac{\sum_{r=1}^{s} u_r y_{rj}}{\sum_{i=1}^{m} v_i x_{ij}} \leq 1 \qquad (2.6)$$

$$j = 1, \ldots., n; \quad r = 1, \ldots. s; \quad i = 1, \ldots. m; \quad ur, vi \geq 0 \quad (2.7)$$

A Eq. 2.5 é conhecida como programação fracionária. É possível linearizar este modelo transformando em um modelo equivalente de programação linear (do inglês *Linear Programming* - LP) conforme a Eq. 2.8.

$$Max\ \theta_0 = \sum_{r=1}^{s} u_r y_{r0} \qquad (2.8)$$

Sujeito as seguintes restrições:

$$\sum_{r=1}^{s} u_r y_{rj} - \sum_{i=1}^{m} v_i x_{ij} \leq 0 \quad j = 1, \ldots. . n \qquad (2.9)$$

$$\sum_{r=1}^{m} v_i x_{i0} = 1 \qquad (2.10)$$

$$ur, vi \geq 0; \quad r = 1, \ldots. s; \quad i = 1, \ldots. m \qquad (2.11)$$

Quando o mesmo conjunto de coeficientes de entrada e saída for aplicado a todas as outras unidades de tomada de decisão (DMUs) que estão sendo comparadas, nenhuma unidade excederá 100% de eficiência ou uma razão de 1. Onde y_{rj} e x_{ij} são saídas e entradas conhecidas. Diz-se que a unidade j é eficaz se $\theta = 1$. Caso contrário, se θ é menor que 1 a unidade é considerada ineficiente. O método FMEA se distingue do método DEA por delegar a um time multidisciplinar a tarefa de estabelecer o peso para cada um dos modos de falha entre as unidades avaliadas, ao passo que o DEA, método não paramétrico, propositivo, atribui pesos aos modos de falha de cada unidade por meio de um problema de otimização. A modelagem DEA apresenta duas propostas de análise. A primeira destina-se a retornos constantes de escala (do inglês *Constant Returns Scale* - CRS). A segunda aplica-se retornos variáveis de escala (do inglês *Variable Return to Scale* - VRS). Estas análises são detalhadas nas seções 2.11.2 e 2.11.3, respectivamente. A escolha da modelagem DEA pode ser associada ao planejamento estratégico nas organizações, maximizando a eficiência nas seguintes perspectivas:

1. Orientação ao insumo: reduzir o consumo de insumos (entradas) e mantendo o nível de produtos (saídas);

2. Orientação ao produto: aumentar os produtos (saídas) mantendo o nível de insumos (entradas);

Na orientação ao insumo (entrada), a saída (produto) ficará constante e a meta será a redução dos insumos. Por outro lado, na orientação ao produto (saída), o insumo (entrada) ficará constante e a meta será o aumento dos produtos. A aplicação do método DEA apresenta uma sequência para realizar a modelagem. Primeiramente inicia-se selecionando as unidades de tomada de decisão DMUs. Em seguida, descreve-se o processo das unidades que estão sob análise a fim de identificar os atributos que são considerados como insumos (entradas) e produtos (saídas). Por fim, o método é executado utilizado com o suporte de *softwares* disponíveis, tais como: *Frontier Analyst (*BANXIA SOFTWARE, 2023), suplemento do *Excel* (SOLVER, 2023) ou linguagem R com instalação das bibliotecas que implementam o DEA.

1.1.6 Benchmarking

As melhorias em produtos e serviços ocorrem quando relacionadas com padrões estabelecidos. Estas melhorias são incorporadas em novos padrões, com o objetivo de alcançar a excelência.

Benchmarking é o processo contínuo de medir produtos, serviços e processos, comparando-se com os líderes da indústria ou os mais fortes concorrentes. Isso resulta na procura de melhores práticas, que vão conduzir a um desempenho superior, implementando continuamente mudanças e praticando uma cultura de melhoria contínua para realizar o melhor (OAKLAND, 1994).

Quando combinada com a lucratividade, a análise de eficiência DEA é útil no planejamento estratégico das organizações. As unidades de tomada de decisão com desempenhos de excelência são encontradas, isto é, classificadas como *benchmark*. Essas unidades eficientes servem como exemplo para outras emularem quais valores devem alterar para atingir a eficiência (FITZSIMMONS, et al., 2014).

1.1.7 O modelo DEA-CCR

Inicialmente o modelo proposto por (CHARNES, et al., 1978) nomeado de CCR, foi modelado para uma análise com retornos constantes de escala (CRS).

No modelo CCR com CRS, a fronteira eficiente é dada por uma reta a partir da origem até a unidade produtiva que forma o maior raio com o eixo de insumo.

Nesta modelagem é possível estimar as metas e propor estratégias para tornar DMUs ineficientes em eficientes. O modelo DEA-CCR é conhecido como modelo do envelope e pode ser orientado a entrada (insumo) ou a saída (produto). O coeficiente de importância (λ) identifica os *benchmarks* para cada DMU ineficiente. A formulação matemática representada abaixo tem como objetivo maximizar os produtos (saídas).

$$x_{io} \geq \sum_{j=1}^{n} xij\ \lambda j \qquad i = 1,2,\ldots..m \qquad (2.12)$$

$$\theta y_{r0} \leq \sum_{j=1}^{n} yrj\ \lambda j \qquad r = 1,2,\ldots..s \qquad (2.13)$$

Sujeito a restrição da Eq.2.14, tem-se a modelagem DEA-CCR:

$$\lambda j \geq 0 \qquad j = 1,2,\ldots..n \qquad (2.14)$$

Para aumentar a eficiência, pode-se também utilizar o modelo DEA-CCR conhecido como envelope com orientação as entradas (insumos). A formulação matemática abaixo representa este conceito:

$$\theta x_{i0} \geq \sum_{j=1}^{n} xij\ \lambda j \qquad i = 1,2,\ldots..m \qquad (2.15)$$

$$y_{r0} \leq \sum_{j=1}^{n} yrj\ \lambda j \qquad r = 1,2,\ldots..s \qquad (2.16)$$

1.1.8 O modelo DEA-BCC

Em 1984, a modelagem DEA foi estendida por BCC para adicionar retornos variáveis de escala, VRS e passou a ser nomeada de BCC. Este modelo forma uma fronteira convexa eficiente com as melhores unidades, e desta forma, envelopa as unidades ineficientes para cada escala de produção. Portanto, o modelo BCC permite que a eficiência varie em função de economias de escala, assim como permite a comparação de unidades de diferentes tamanhos [91]. O apêndice C representa as fronteiras de eficiência dos modelos DEA.

Sujeito a restrição da Eq.2.17, tem-se a modelagem DEA-BCC:

$$\sum_{j=1}^{n} \lambda j\ = 1 \quad (2.17)$$

1.1.9 A regra de ouro

Na aplicação do método DEA, surge o questionamento de quantas DMUs são ideais para a análise da eficiência. No estudo de (PEÑA, 2008) destaca-se que quanto ao número de unidades, não existem normas definidas, porém quanto maior a quantidade, maior será a capacidade discriminatória do modelo.

Para a aplicação do método DEA, há recomendações denominadas de regras de ouro (do inglês *Golden Rules*) especialmente no que diz respeito a quantidade mínima de DMUs que devem ser consideradas para a aplicação do método. A literatura recomenda que o número de variáveis de entrada e de saída não exceda um terço do número de DMUs. Quando isto ocorre, os modelos DEA tradicionais (CCR e BCC) apresentam uma baixa discriminação dos dados. Em (TOLOO, et al., 2015) estudos permitem identificar a quantidade mínima de DMUs a fim da aplicação do método DEA clássico.

Considerando N o número de DMUs, I (quantidade de atributos de entrada) e S (quantidade de atributos de saída) a primeira regra a ser analisada é o produto entre a quantidade dos atributos de entrada com os atributos de saída conforme Eq. 2.18.

$$N_{min} = I * S \qquad (2.18)$$

A segunda regra é a soma entre a quantidade dos atributos de entrada com os atributos de saída. O resultado multiplica-se por um fator de 3 conforme a Eq. 2.19.

$$N_{min} = 3 * (I + S) \qquad (2.19)$$

No livro de (FITZSIMMONS, et al., 2014) destaca que o número de DMUs está baseado em pesquisas empíricas e na experiência de usuários da DEA. A Eq. 2.20 é proposta pelo autor na literatura.

$$N_{min} \geq 2 * (I + S) \qquad (2.20)$$

Como melhores práticas na aplicação do método DEA recomenda-se atentar a essas regras. O impacto da violação destas regras gera um problema conhecido como ponderação excessiva e dispersão irrealista de pesos. Em resumo, pode ocasionar um conjunto de DMUs falsamente eficientes, o que implica em uma análise completamente equivocada e uma tomada de decisão não assertiva. No estudo de (EMROUZNEJAD, et al., 2009) é destacado que modelos DEA podem resultar em pontuações de eficiência incorretas para os dados que são utilizados na modelagem do método.

1.1.10 Super eficiência

A super eficiência representa um artifício matemático que apresenta como objetivo remover uma restrição da modelagem. Desta forma, para uma específica DMU que era

considerada 100% eficiente, neste conceito poderá alcançar valores acima de 100% de eficiência. As DMUs que já eram ineficientes, no cálculo da super eficiência não terão alteração do seus valores iniciais. Matematicamente, para encontrar a super eficiência é suficiente remover a Eq. 2.9 para a DMU que estiver sob análise. A Eq. 2.21 apresenta a formulação para este conceito.

$$\sum_{r=1}^{s} u_r y_{rj} - \sum_{i=1}^{m} v_i x_{ij} \leq 0 \quad j = 1, \ldots . n; \quad j \neq 0 \qquad (2.21)$$

Sujeito as seguintes restrições:

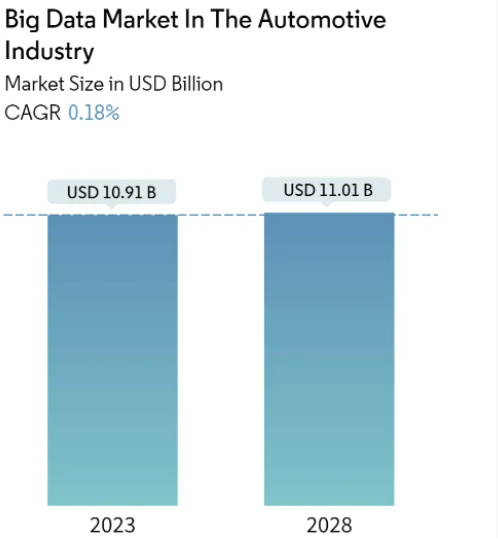
$$ur, vi \quad \geq 0; \quad r = 1, \ldots . s; \quad i = 1, \ldots . m \qquad (2.22)$$

AGRUPAMENTO DE DADOS

O mercado automotivo apresenta um crescimento expressivo na quantidade de dados armazenados motivado por novas tecnologias, aplicações e serviços que envolvem o desenvolvimento de sistemas de *software* complexos. Novas funcionalidades para veículo autônomo bem como uma crescente base de veículos conectados, impulsionam o crescimento de soluções e sistemas no segmento automotivo. Com o avanço da tecnologia e alinhado com o aumento de novas funcionalidades, a quantidade de dados armazenados gerou uma transformação para grandes bases de dados (do inglês *Big Data*). Desta forma, o conceito de agrupamento de dados (do inglês *Clustering*) é de relevância para analisar os dados segundo o grau de semelhança.

Segundo a pesquisa conduzida pela organização *Mordor Intelligence* (MORDOR, 2023) estima-se que o tamanho do mercado de *Big Data* na indústria automotiva cresça de US$ 5,30 bilhões em 2023 para US$ 11,21 bilhões até 2028, com uma taxa de crescimento anual composta (do inglês *Compound Annual Growth Rate* - CAGR) de 16,15% durante o período de previsão (2023-2028). O mercado de crescimento mais acelerado é representado pela Ásia-Pacífico (do inglês *Asia Pacific* - APAC) enquanto que o maior mercado é considerado a Europa. A Figura 10 ilustra esta tendência de crescimento do setor.

Figura 10. Tamanho do mercado de *Big Data*



Período de Estudo	2018 - 2028
Tamanho do mercado (2023)	USD 5,30 bilhões de dólares
Tamanho do mercado (2028)	USD 11,21 bilhões de dólares
CAGR(2023 - 2028)	16.15 %
Mercado de Crescimento Mais Rápido	Ásia-Pacífico
Maior Mercado	Europa

Fonte: (MORDOR, 2023)

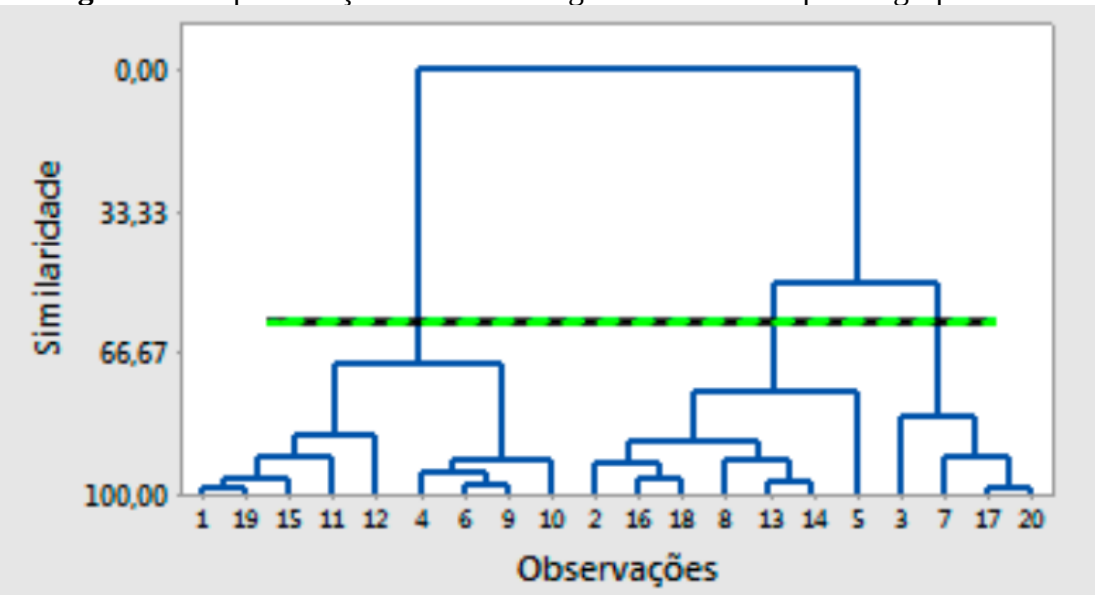
Como é uma realidade e existe uma previsão no aumento de grandes dados, o agrupamento visa encontrar uma estrutura de grupos (do inglês *clusters*) em que os objetos pertencentes a cada grupo compartilham alguma característica ou propriedade relevante para o domínio do problema em estudo.

Há na literatura um grande número de algoritmos de agrupamento de dados, como exemplos: hierárquico aglomerativo (BOEHMKE, 2023), *K-Means* (PIECH, 2013), particional (CELEBI, 2015) e o agrupamento espacial baseado em densidade de aplicativos com ruído (do inglês *Density-Based Spatial Clustering of Applications with Noise* - DBSCAN) (HAHSLER, et al., 2019).

1.1.11 Agrupamento hierárquico aglomerativo

Este conceito cria uma hierarquia de grupos no estilo de uma árvore, que é conhecido como dendrograma. Nesta representação, os dados individuais são as folhas da árvore e os nós do interior são aglomerados em grupos. O eixo vertical representa o nível de similaridade entre os grupos em análise. O eixo horizontal apresenta as diferentes características dos elementos. A Figura 11 representa um diagrama que ilustra o dendrograma.

Figura 11. Representação de um dendrograma com hierarquia e agrupamento



Fonte: (MINITAB, 2023)

Este agrupamento é uma abordagem alternativa para identificar grupos no conjunto de dados. Ele não exige um número de *clusters* a serem gerados, conforme requerido em outras abordagens tais como o agrupamento por *K-Means*. Com suporte da linguagem R, a distância euclidiana é usada para medir a dissimilaridade entre cada par de observações.

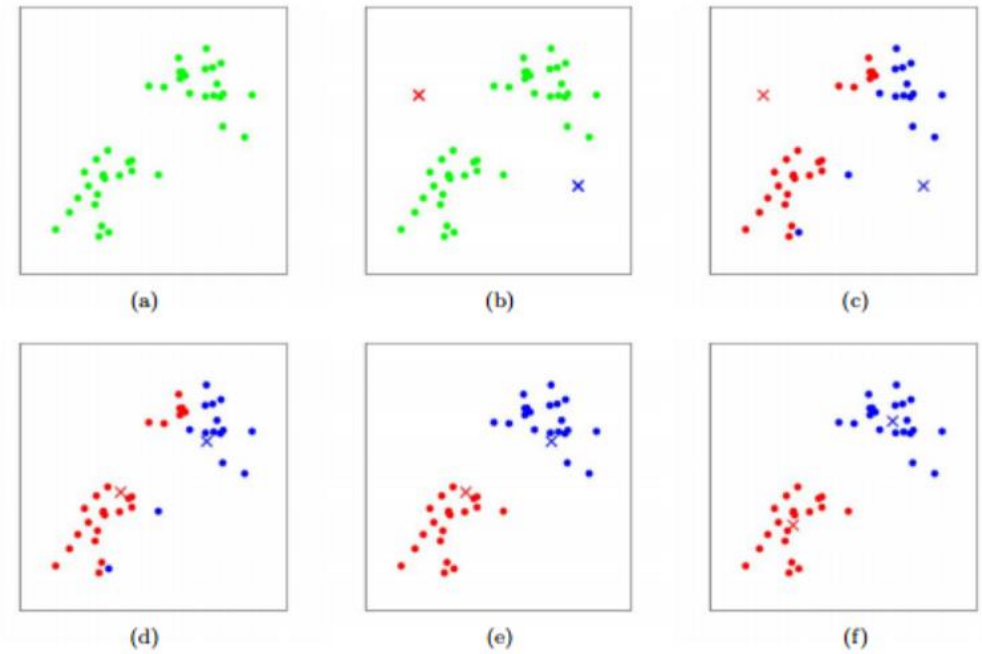
Os algoritmos que envolvem o agrupamento hierárquico podem ser encontrados em (MURTAGH e CONTRERAS, 2012) e (PASUPATHI, et al., 2021).

1.1.12 Agrupamento K-Means

Este conceito também nomeado de K-Médias apresenta um número de variações devido à sua simplicidade de implementação em linguagens computacionais (SCIKIT-LEARN, 2023). Ele tem como objetivo fornecer uma classificação de informações de acordo com os próprios dados. A classificação é baseada em análise e comparações entre os valores numéricos dos dados. Esta técnica de agrupamento implementa um conceito nomeado de centroide, correspondente a média dos padrões de um grupo.

A aplicação de agrupamento *K-Means* pode ser encontrada nos estudos de (ABDULLAH, et al., 2022), (SINAGA, et al., 2020) e (NIU, et al., 2021). A Figura 12 apresenta a representação do agrupamento por meio deste conceito.

Figura 12. Representação do agrupamento *K-Means*



Fonte: (PIECH, 2013)

PROPOSTA DE PESQUISA

Esta seção apresenta de maneira sistemática quais caminhos foram percorridos para se alcançar os objetivos propostos. Os métodos utilizados são apresentados a fim de contribuir com a priorização e o tratamento das falhas de funcionalidades de *software* no segmento automotivo.

APLICAÇÃO DO MÉTODO FMEA

No segmento automotivo a fabricante do equipamento original (do inglês *Original Equipment Manufacturer* - OEM), contrata sistemistas para desenvolver e validar *software* embarcado de uma unidade de controle eletrônico (do inglês *Eletronic Control Unit* - ECU). A especificação contendo os requisitos das funcionalidades de *software* é encaminhada da montadora para a sistemista. Durante o desenvolvimento do produto, conforme os processos definidos na ISO 15504 (ASPICE, 2017), diversas falhas são identificadas nas fases de desenvolvimento. Algumas funcionalidades de *software* não satisfazem os requisitos da especificação, gerando impactos na qualidade e confiabilidade do produto.

Este trabalho apresenta uma proposta de análise por integração do método FMEA com o DEA para avaliar e priorizar os riscos de funcionalidades de *software* no desenvolvimento de uma unidade eletrônica automotiva ECU. A aplicação do DEA visa contribuir com as lacunas do método FMEA na avaliação de riscos. Inicialmente as funcionalidades de *software* são descritas em uma especificação compartilhada pela montadora de veículos OEM para a sistemista, que apresenta o time de desenvolvimento composto de: analista de requisitos, desenvolvedores, analista da qualidade e validadores de *software*. A seguir os valores de severidade, ocorrência e detecção de falhas são levantados para cada funcionalidade de *software*.

O método FMEA é implementado por um time de desenvolvimento. Neste método a contribuição e avaliação da perspectiva de equipes multiculturais e multidisciplinares é relevante para a determinação dos valores de análise. A classificação das funcionalidades com relação ao risco é obtida através do cálculo do número de prioridade de risco (RPN). No método FMEA o RPN com maior valor representa a maior prioridade de correção e é interpretado como crítico. Com este resultado, os riscos das funcionalidades de *software* podem ser priorizados para o tratamento das falhas.

No desenvolvimento de *software* embarcado, as falhas são identificadas pelos validadores de *software* a fim de realizar a fase de validação das funcionalidades do componente. Desta forma, o levantamento da quantidade de falhas é reportado no sistema de

gerenciamento de defeitos e o tempo estimado de reparo são estimados pelo time de desenvolvimento. Os defeitos são conhecidos como *bugs*. Para um conjunto de *bugs* corrigidos, é emitido uma nova versão de *software* com as correções integradas de tal forma que sejam validadas.

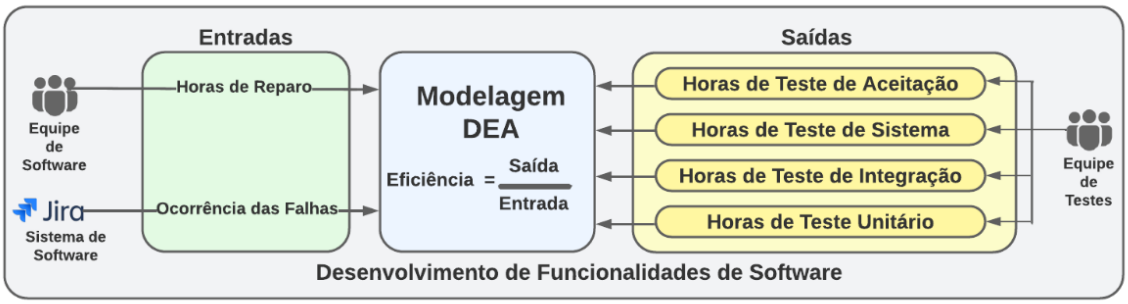
APLICAÇÃO DO MÉTODO FMEA ESTENDIDO

No FMEA tradicional, a prioridade da ação (AP) é obtida pela multiplicação dos 3 valores: severidade (S), ocorrência (O) e detecção (D). No método FMEA estendido, a proposta é utilizar 5 valores para priorização dos riscos: severidade (S), ocorrência (O), detecção (D), quantidade de falhas (F) e tempo estimado de reparo (T). A maior pontuação representará o maior risco, com alto impacto para a organização, ou seja, as equipes de desenvolvimento deverão tratar as falhas e conduzir a priorização dos riscos com este critério.

APLICAÇÃO DO MÉTODO DEA

O método DEA considera os valores das horas de tempo de aceitação, horas de teste de sistema, horas de teste de integração, horas de teste unitário (como saídas), quantidade de falhas e o tempo estimado de correção das falhas (como entradas) para a identificação da eficiência das funcionalidades como representado na Figura 13.

Figura 13. Modelagem do método DEA para a proposta de pesquisa



Fonte: O autor.

Com a aplicação dos métodos, os resultados são avaliados para priorização dos riscos das funcionalidades de *software*. Por fim, uma proposição de ações preventivas e corretivas do tratamento das falhas de *software* com maiores riscos são levantados pelo time de desenvolvimento, buscando a melhoria da qualidade do *software*.

As ocorrências das falhas detectadas no desenvolvimento de um produto geram esforços adicionais e retrabalhos. A quantidade de falhas e o tempo de reparo das correções

das funcionalidades de *software* são duas variáveis relevantes que influenciam na qualidade e confiabilidade.

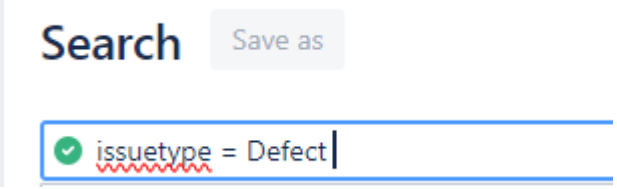
O Capítulo 4 apresenta a metodologia com o detalhamento da proposta de avaliação dos riscos utilizando os métodos FMEA, FMEA estendido e DEA neste estudo de caso.

METODOLOGIA

Os principais elementos de estudo neste trabalho foram as funcionalidades de *software* que apresentam falhas e impactam na qualidade do produto. O ambiente de desenvolvimento automotivo é composto por equipes multiculturais e multidisciplinares que participam ativamente nas diversas fases, atividades e tomadas de decisão do projeto. A montadora automotiva OEM atua como cliente, especificando as necessidades, requisitos funcionais e normativos para o desenvolvimento. A sistemista é a organização que participa no projeto, desenvolvimento, correção e suporte do tratamento das falhas identificadas nas funcionalidades de *software* durante todo do ciclo de vida do produto. O desenvolvimento deste trabalho é composto por procedimentos organizacionais, métodos de análise e priorização de riscos, bem como ferramentas de suporte. Os procedimentos organizacionais estão interligados com as políticas da qualidade. O desenvolvimento das funcionalidades de *software* embarcado automotivo exige a garantia da qualidade de *software* como passo inicial. Em seguida, a identificação das funcionalidades de *software* foram levantadas na especificação do produto utilizando ferramentas de documentação (*Microsoft Word e Google Docs*). O uso destes recursos facilita a criação e a manutenção de uma documentação organizada de funcionalidades de *software*. Este procedimento foi realizado por uma equipe de analistas de requisitos, cuja finalidade foi analisar, identificar e quantificar as principais funcionalidades requeridas pelo cliente (OEM) estabelecidas na especificação. O próximo passo foi a captura dos requisitos de sistema, também conduzido pelas equipes de analistas de requisitos. Este procedimento foi importante pois estabelece de forma clara o que o sistema deve ser capaz de fazer para satisfazer as necessidades e requisitos dos *stakeholders*. Neste procedimento foi utilizado como ferramenta de suporte o (*Microsoft Word*) com a finalidade de consolidar a captura dos requisitos de sistema. Posteriormente, os riscos foram identificados para cada uma das funcionalidades de *software*. As equipes de desenvolvimento analisaram, avaliaram e trataram os riscos através da ferramenta de suporte nomeada de *Jira* (JIRA, 2023). Ele combina técnicas de desenvolvimento com recursos ágeis, a fim de auxiliar as equipes na criação, manutenção e reparo de *software* de forma colaborativa. O gerenciamento de mudanças, identificação de falhas, rastreamento de problemas e melhorias nas funcionalidades são algumas das principais características destacadas nesta ferramenta. O Jira apresenta um recurso incorporado nomeado de linguagem de consulta Jira (do inglês *Jira Query Language* - JQL). Por meio dele, a consulta de falhas para uma específica funcionalidade de *software* pode ser identificada e analisada. A Figura 14 ilustra o mecanismo de consulta de falhas de

funcionalidades de *software*. Por esta consulta, foram listadas todas as falhas identificadas no *software* embarcado automotivo.

Figura 14. Captura de falhas em funcionalidades de *software* por meio do recurso JQL.



Fonte: (JIRA, 2023)

Para cada risco identificado nas funcionalidades de *software*, os valores de severidade, ocorrência e detecção foram pontuados por meio de uma reunião com a participação colaborativa das equipes do projeto. Estes valores foram estabelecidos com o suporte de uma planilha em formato *Microsoft Excel*. Este procedimento foi importante para capturar as perspectivas dos membros da equipe do projeto, bem como contribuir para o próximo passo que refere-se a implementação do método FMEA.

O primeiro método implementado como forma de análise de riscos e priorização foi o FMEA. Como o setor de desenvolvimento de *software* embarcado refere-se ao segmento automotivo, o FMEA foi escolhido por ser um dos requisitos mandatórios para a garantia da qualidade (QUALYTEAM, 2021). Com base nos valores de severidade, ocorrência e detecção, a priorização dos riscos de funcionalidades de *software* foi calculada através do número de prioridade de risco (RPN), conforme Eq. 2.2. Neste método a pontuação do maior RPN representa o risco crítico, ou seja, aquele a ser tratado prioritariamente. O segundo método proposto referiu-se a uma extensão do método FMEA, considerando cinco valores na análise: severidade, ocorrência, detecção, quantidade de falhas e tempo estimado de reparo das falhas. Na comunidade científica, o FMEA tradicional apresenta algumas desvantagens conforme detalhado na seção 2.10.4. Desta forma, a extensão do FMEA considerou a quantidade de falhas detectadas por cada funcionalidade por meio da ferramenta *Jira* e recurso de consulta *JQL*. O fator de tempo estimado de correção das falhas também foi considerado. Ele representa o esforço estimado (em horas), pontuado pela equipe de desenvolvimento de *software* para a correção das falhas identificadas pelas equipes de teste. A priorização dos riscos pelo FMEA estendido teve o tratamento das falhas pelo ranqueamento da maior pontuação para a menor. O terceiro método referiu-se a implementação da modelagem utilizando os conceitos de DEA para estabelecimento da eficiência das funcionalidades de *software* com o objetivo de identificar as de melhor desempenho, considerando-as como eficientes dentro de um conjunto

de funcionalidades com múltiplas entradas e saídas. O conceito da eficiência estabelece a razão entre o numerador (saídas, produtos) e o denominador (entradas, insumos). A identificação dos insumos (entradas) neste trabalho foi considerada pela quantidade de falhas e do tempo estimado de reparo das falhas (em horas). As saídas (produtos) foram estimadas pelas equipes de *software* e testes referente as horas de cada fase do processo de verificação e validação, conforme ilustrada pelas fases 6,7,8 e 9 da Figura 3. No que tange a automação do processo de cálculo da eficiência das funcionalidades de *software*, a programação VBA (do inglês *Visual Basic for Applications - VBA*) foi aplicada (VBA, 2023). O apêndice A apresenta o algoritmo elaborado para a automação deste processo. A linguagem R proporciona um ambiente remoto nomeado de *Posit Cloud* (POSIT, 2022) para desenvolvimento de soluções de programação utilizando o método DEA. Os pacotes integrados com a linguagem R para o método DEA estão descritos na Tabela 7.

Tabela 7. Pacotes em *R Studio*

Pacotes	Descrição
Benchmarking	Métodos para análise da fronteira, análise de envoltória de dados sob diferentes retornos de escala (CRS,VRS) e diferentes orientações (insumo, produto).
deaR	ConFiguração de funções para análise de envoltória de dados. Este pacote implementa o modelo clássico e fuzzy DEA.
readxl	Fornece a importação de arquivos em *Excel* para o R.
MultiplierDEA	Fornece funções para calcular a eficiência usando o método DEA das unidades de tomada de decisão.
rio	Fornece a importação e exportação de dados simplificados para o Excel.
Fastcluster	Fornece eficiente algoritmo para agrupamento hierárquico aglomerativo através da função *hclust*.
additiveDEA	Fornece a eficiência aditiva da análise de envoltória de dados através da medida baseada em folgas (SBM).

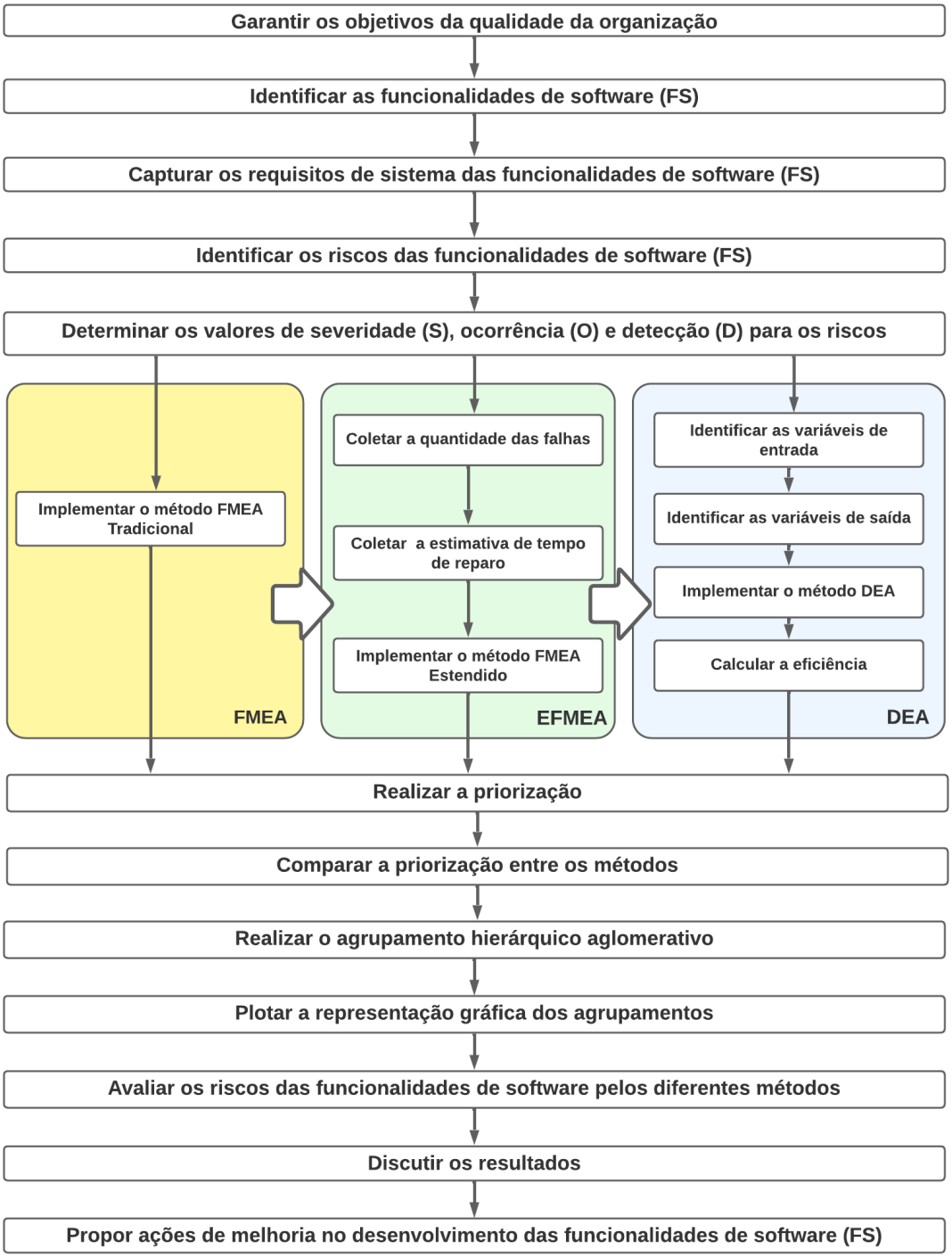
Fonte: (BOGETOFT e OTTO,2022) , (COLL-SERRANO et al., 2023), (WICKHAM et al, 2023), (PUTHANPURA, 2022),(BECKER et al., 2023) , (MÜLLNER,2013), (SOTERIADES, 2022)

Para o agrupamento dos riscos deste estudo de caso, a programação na linguagem R foi aplicada. Através da biblioteca e funções de plotagem de gráficos, um específico dendrograma com agrupamento de riscos foi desenvolvido. O dendrograma representa um diagrama que organiza os riscos das funcionalidades de *software*, permitindo analisar os níveis de distância

entre os riscos. As representações gráficas dos dendrogramas foram ilustrados na análise dos resultados. Com a implementação dos três métodos para avaliação dos riscos e tratamento das falhas de funcionalidades de *software*, foi implementado o agrupamento de dados hierárquico aglomerativo através da ferramenta de suporte *Posit Cloud* com o uso da linguagem R. A biblioteca instalada no ambiente *Posit Cloud* foi a *fastcluster* (MÜLLNER, 2013) que forneceu um eficiente algoritmo para implementar o agrupamento de dados hierárquico por meio da função *hclust*. As representações gráficas dos dendrogramas foram plotadas contendo 9 clusters para cada método (FMEA, *Extended Failure Mode and Effect Analysis* (EFMEA) e DEA) como forma de avaliar os grupos das funcionalidades de *software* com similaridade. Os códigos implementados por meio da linguagem R foram apresentados no apêndice B. Em seguida foi conduzido a discussão dos resultados e proposição de melhorias no desenvolvimento para correção das falhas das funcionalidades. Por fim, ações de prevenção e correção foram propostas pelas equipes de desenvolvimento da organização, no sentido de melhoria da qualidade do *software.*

A metodologia deste estudo está representada conforme a Figura 15.

Figura 15. Fluxograma dos passos e métodos deste estudo de caso



Fonte: O autor.

Os passos detalhados no fluxograma da metodologia são representados a seguir:

• **Passo 1 - Garantir os objetivos da qualidade da organização**: Este passo representa a garantia da qualidade, atender aos requisitos da especificação, satisfação do cliente e atendimento aos procedimentos organizacionais;

• **Passo 2 - Identificar as funcionalidades de *software* (FS)**: Este passo representa a identificação das funcionalidades de *software* da especificação fornecida pela montadora automotiva;

• **Passo 3 - Capturar os requisitos de sistema das funcionalidades de *software* (FS)**: Este passo representa a captura dos requisitos funcionais especificados para as funcionalidades de *software*;

• **Passo 4 - Identificar os riscos das funcionalidades de *software* (FS)**: Este passo representa a identificação dos riscos das funcionalidades de *software* como uma forma de análise preventiva das falhas;

• **Passo 5 - Determinar os valores de severidade, ocorrência e detecção para os riscos**: Este passo representa a classificação dos valores pontuados pelas equipe de desenvolvimento;

• **Passo 6 - Implementar o método FMEA Tradicional**: Este passo representa a implementação do método FMEA tradicional considerando o número de prioridade de risco (RPN) com três valores, conforme formulado através da Eq. 2.2;

• **Passo 7 - Coletar a quantidade das falhas**: Este passo representa a quantidade de falhas detectadas pelas equipes de *software* e testes nas fases 6,7,8 e 9 da Figura 3;

• **Passo 8 - Coletar a estimativa de tempo de reparo**: Este passo representa a estimativa de tempo (em horas) que a equipe de *software* levará para fornecer a correção das falhas detectadas em uma específica funcionalidade de *software*;

• **Passo 9 - Implementar o método FMEA Estendido**: Este passo representa a extensão do método FMEA tradicional, considerando a implementação do número de prioridade de risco (RPN) com cinco valores: severidade, ocorrência, detecção, quantidade de falhas e tempo estimado de reparo;

• **Passo 10 - Identificar as variáveis de entrada**: Este passo representa a identificação das variáveis de entrada (insumos) que foram utilizadas no denominador da Eq. 2.4. A quantidade de falhas e o tempo estimado de reparo (em horas) foram as variáveis estabelecidas como entradas (insumos);

• **Passo 11 - Identificar as variáveis de saída:** Este passo representa a identificação das variáveis de saída (produtos) que foram utilizadas no numerador da Eq. 2.4. As horas

estimadas de teste unitário, teste de integração, teste de sistema e teste de aceitação foram as variáveis estabelecidas com saídas (produtos);

• **Passo 12 - Implementar o método DEA**: Este passo representa a implementação do método DEA, considerando previamente as variáveis de entrada e saída estabelecidas, definição das restrições do modelo, identificação da função objetivo, orientação e tipo da modelagem;

• **Passo 13 - Calcular a eficiência**: Este passo representa o resultado do cálculo da eficiência para cada funcionalidade de *software* conforme Eq. 2.4;

• **Passo 14 - Realizar a priorização**: Este passo representa a priorização dos riscos das funcionalidades de *software*, ou seja, quais funcionalidades as equipes de desenvolvimento devem concentrar os esforços para o tratamento das falhas;

• **Passo 15 - Comparar a priorização entre os métodos**: Este passo representa o comparativo da priorização dos riscos das funcionalidades de *software* entre os métodos FMEA, EFMEA e DEA;

• **Passo 16 - Realizar o agrupamento hierárquico aglomerativo**: Este passo representa a implementação do agrupamento das funcionalidades de *software* em grupos, de tal forma a encontrar a similaridade dentre um conjunto de dados;

• **Passo 17 - Plotar a representação gráfica dos agrupamentos**: Este passo representa a plotagem dos gráficos para cada método (FMEA, EFMEA e DEA) através dos dendrogramas conforme apresentado na análise dos resultados;

• **Passo 18 - Avaliar os riscos das funcionalidades de *software* pelos diferentes métodos**: Este passo representa a avaliação dos riscos perante a priorização e tratamento das falhas utilizando os métodos estudados;

• **Passo 19 - Discutir os resultados**: Este passo representa a discussão dos resultados obtidos com a implementação dos métodos estudados;

• **Passo 20 - Propor ações de melhoria no desenvolvimento das funcionalidades de *software* (FS)**: Este passo representa a proposição de ações de melhoria para os riscos das funcionalidades de *software* a fim de diminuir as ocorrências das falhas e aumentar a confiabilidade do produto.

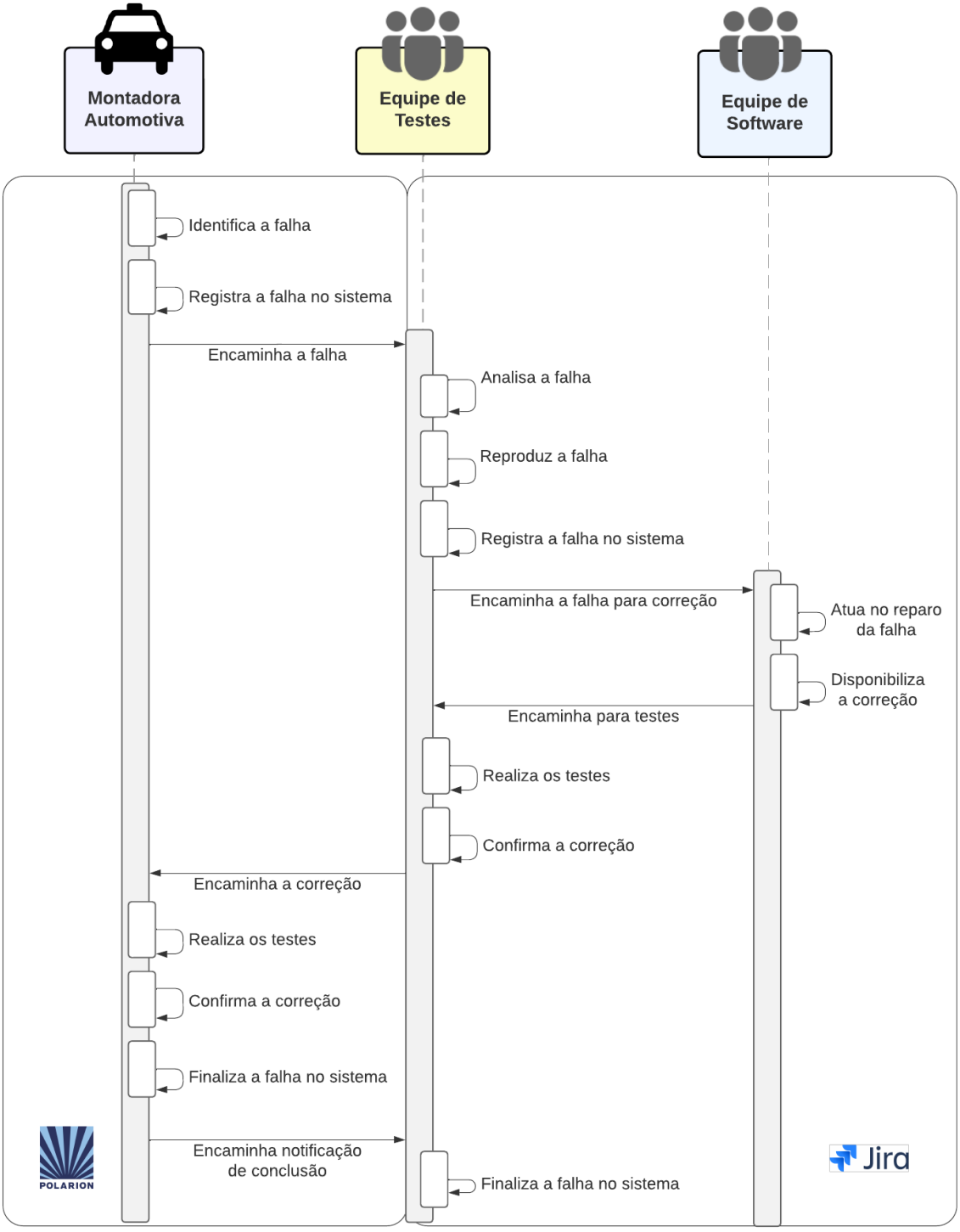
ESTUDO DE CASO

Este Capítulo apresenta o estudo de caso da aplicação dos métodos propostos para o desenvolvimento de funcionalidades de *software* no segmento automotivo com um modelo de negócios entre empresas (do inglês *Business to Business* - B2B). De um lado tem-se a montadora automotiva (OEM), contratante de serviços de desenvolvimento de funcionalidades de *software*. Por outro lado tem-se a sistemista, organização contratada cujo o objetivo é atender as expectativas da contratante referente ao desenvolvimento de *software* embarcado automotivo. A montadora automotiva define os requisitos do veículo e disponibiliza para a sistemista uma especificação contendo as características das funcionalidades de *software*. O desenvolvimento de soluções de sistemas embarcados automotivos requer equipes multidisciplinares que participam no ciclo de desenvolvimento (analistas de requisitos, desenvolvedores de *software* e validadores). No segmento automotivo, as boas práticas de desenvolvimento de *software* são padronizadas através da ISO 15504. Esta norma estabelece o processo de desenvolvimento para a garantia da qualidade de *software*. No desenvolvimento do *software*, as falhas são identificadas tanto pela montadora automotiva quanto pela sistemista. Para a avaliação, priorização e tratamento das falhas das funcionalidades de *software*, neste estudo de caso foram contextualizados:

- Análise de Modo de Falha e seus Efeitos (FMEA);
- Análise de Modo de Falha e seus Efeitos Estendido (EFMEA);
- Análise de Envoltória de Dados (DEA);

No que tange o método DEA, tanto o modelo com orientação as entradas (insumos) quanto saídas (produtos) foram analisados. Em relação ao tipo de retorno de escala, os retornos constantes de escala (CRS) e retornos variáveis de escala (VRS) foram implementados. O fluxograma de identificação de uma falha e a interação entre a montadora automotiva e equipes de teste de *software* é representado na Figura 16.

Figura 16. Diagrama de sequência de identificação das falhas



Fonte: O autor.

ANÁLISE DOS RESULTADOS

Este Capítulo apresenta a análise do resultados referente aos métodos para apoio no tratamento das falhas e ranqueamento das funcionalidades de *software* como forma de melhorar a qualidade. Nas seções abaixo são apresentados as vantagens e desvantagens referente a análise dos resultados dos métodos aplicados neste estudo de caso.

CENÁRIO 1: AVALIAÇÃO DOS RESULTADOS POR FMEA

O método FMEA proposto neste cenário consiste na priorização de funcionalidades de *software* (FS) a fim de identificar a prioridade da ação (AP) para que a organização concentre os esforços para o tratamento das falhas reportadas no sistema *Jira*. Na aplicação deste método, o produto dos três valores (severidade, ocorrência e detecção) resultou na prioridade da ação com um valor entre 2 e 480. Isto significa que o risco crítico correspondeu a FS22 com um valor de AP igual a 480. O valor da severidade para esta funcionalidade classificou-se como perigosa (10), com altas ocorrências (8) e baixa detecção (6). Com estes valores, esta funcionalidade de *software* caracterizou-se como primeira no ranqueamento e considera-se prioritária para o tratamento das falhas pelo time de desenvolvimento. Por outro lado, o menor risco correspondeu a FS11 que apresentou um valor de AP igual a 2. Neste caso não há uma alta severidade (1), a ocorrência improvável (1) e robustos mecanismos de detecção das falhas (2). Em função destes dados, a FS11 representou a de menor prioridade para o tratamento das falhas, assumindo a trigésima no ranqueamento.

A Tabela 8 apresenta os dados e o resultado da aplicação do método FMEA tradicional considerando o ranqueamento das funcionalidades de *software* (FS) analisadas neste estudo de caso.

Tabela 8. Avaliação dos riscos de funcionalidades de *software* por FMEA Tradicional

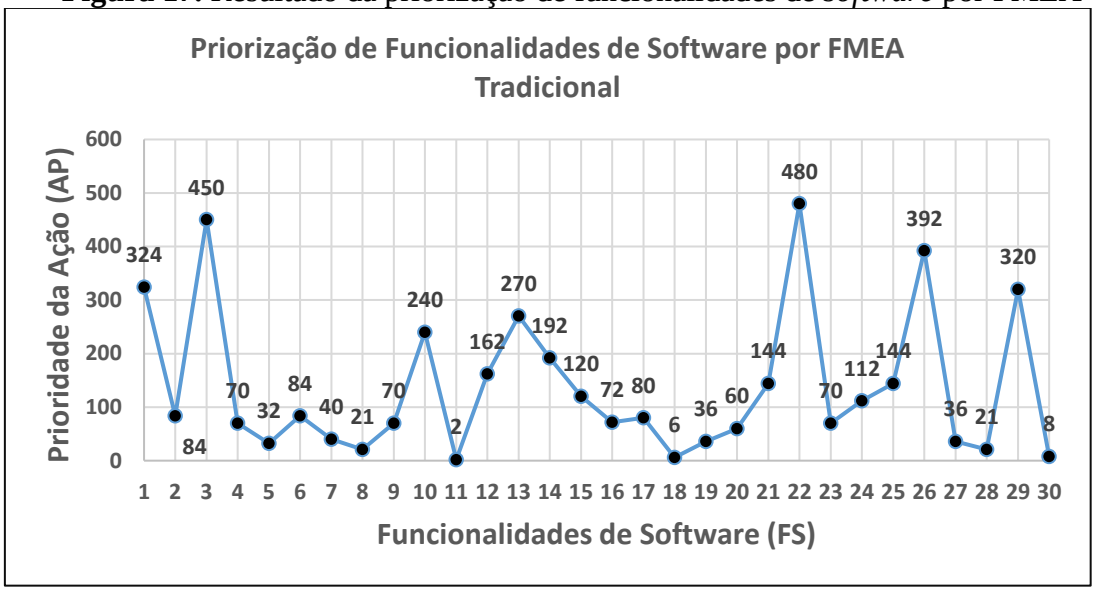
FMEA Tradicional

(FS)	Severidade	Ocorrência	Detecção	(AP)	Rank
1	6	9	6	324	4°
2	3	4	7	84	14°
3	10	5	9	450	2°
4	2	5	7	70	18°
5	8	1	4	32	25°
6	7	6	2	84	14°
7	1	10	4	40	22°
8	3	1	7	21	26°
9	1	10	7	70	18°
10	10	8	3	240	7°
11	1	1	2	2	30°
12	6	3	9	162	9°
13	10	3	9	270	6°
14	8	6	4	192	8°
15	8	3	5	120	12°
16	6	6	2	72	17°
17	10	1	8	80	16°
18	6	1	1	6	29°
19	6	1	6	36	23°
20	10	2	3	60	21°
21	2	8	9	144	10°
22	10	8	6	480	1°
23	1	10	7	70	18°
24	2	8	7	112	13°
25	2	9	8	144	10°
26	8	7	7	392	3°
27	2	3	6	36	23°
28	1	7	3	21	26°
29	8	8	5	320	5°
30	4	2	1	8	28°

Fonte: O autor.

A análise por meio do gráfico da Figura 17 representa a priorização das funcionalidades de *software* para o tratamento das falhas. Nota-se que a primeira no ranqueamento foi a FS22, seguida da FS3 e FS26. Para as três funcionalidades, observa-se uma severidade pontuada acima de alta (>7). Por outro lado, o gráfico reforçou que as funcionalidades FS11, FS18 e FS30 representaram as com menor prioridade no ranqueamento (30°, 29° e 28° respectivamente). Os robustos mecanismos de detecção, alinhado com baixas ocorrências de falhas, categorizam como risco baixo a priorização do tratamento de falhas destas funcionalidades.

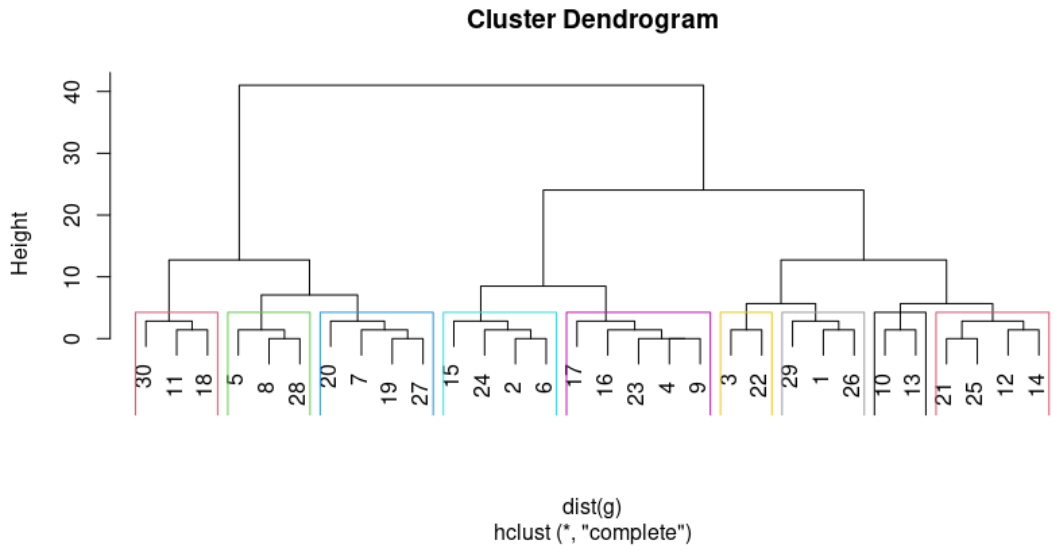
Figura 17. Resultado da priorização de funcionalidades de *software* por FMEA



Fonte: O autor.

Com o apoio do agrupamento hierárquico aglomerativo, permitiu-se visualizar o resultado do dendrograma representado na Figura 18. O nível de similaridade foi plotado no eixo vertical, representando o ranqueamento das funcionalidades de *software* (FS). No eixo horizontal foi destacado o conjunto composto pelas 30 FS. Este dendrograma representou 9 grupos (*clusters*) compostos de funcionalidades que apresentaram proximidade no ranqueamento da aplicação do método FMEA tradicional.

Figura 18. Dendrograma para método por FMEA



Fonte: O autor

Enumerando os *clusters* da esquerda para a direita de 1 até 9, os resultados analisados destacaram que o primeiro cluster agrupou as funcionalidades com risco baixo composto de: FS30, FS11 e FS18. A prioridade do tratamento das falhas de

funcionalidades de *software* foi estabelecida na seguinte ordem relativo ao agrupamento: conforme representado na Tabela 9.

Tabela 9. Ranqueamento de clusters para as funcionalidades de *software*

Cluster	FS	Rank
1	30,11,18	9°
2	5,8,28	8°
3	20,7,19,27	7°
4	15,24,2,6	5°
5	17,16,23,4,9	6°
6	3,22	1°
7	29,1,26	2°
8	10,13	3°
9	21,25,12,14	4°

Fonte: O autor

Com a solução de priorização do tratamento de falhas por FMEA tradicional, percebe-se que diferentes combinações da severidade, ocorrência e detecção resultam no mesmo valor da prioridade da ação, como exemplo o *cluster* 5 que contém a FS4, FS9 e FS23. As três funcionalidades apresentaram o AP igual a 70 para diferentes combinações de valores. Neste sentido, a desvantagem do uso do FMEA tradicional foi justificada pelo baixo poder de discernimento que o método é contextualizado.

CENÁRIO 2: AVALIAÇÃO DOS RESULTADOS POR EFMEA

Este cenário apresenta os resultados da aplicação do método FMEA estendido (EFMEA) que utiliza cinco valores na avaliação da prioridade da ação. Os valores de severidade, ocorrência e detecção do método antecessor foram mantidos. A extensão foi estabelecida pela adição do valores do tempo de reparo das falhas (em horas) estimado pela equipe de *software*, bem como a quantidade de falhas submetidas pela equipe de testes para cada funcionalidade de *software* armazenadas na ferramenta *Jira*.

A Tabela 10 apresenta os valores estabelecidos para este cenário e o resultado da aplicação do método FMEA Estendido, considerando o ranqueamento das funcionalidades de *software* (FS) analisadas neste estudo de caso.

Tabela 10. Avaliação dos riscos de funcionalidades de *software* por EFMEA

(FS)	FMEA Estendido						
	Severidade	**Ocorrência**	**Detecção**	**Falhas**	**Tempo de Reparo**	**(AP)**	**Rank**

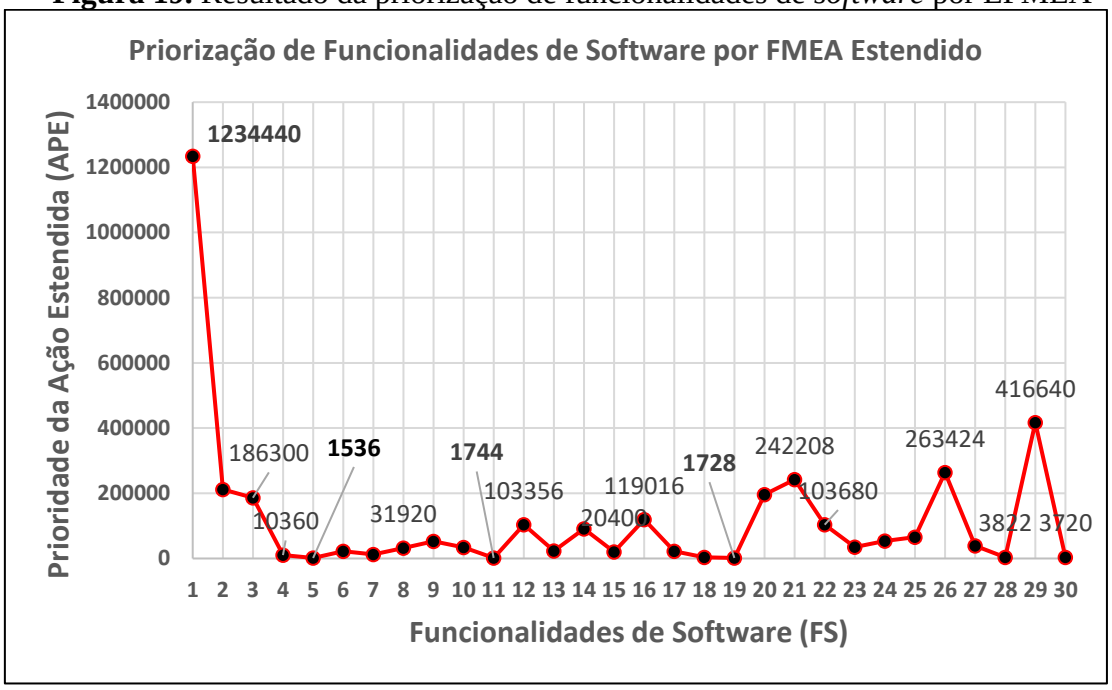
					(horas)		
1	6	9	6	30	127	1234440	1°
2	3	4	7	25	101	212100	5°
3	10	5	9	9	46	186300	7°
4	2	5	7	4	37	10360	24°
5	8	1	4	1	48	1536	30°
6	7	6	2	19	14	22344	20°
7	1	10	4	3	107	12840	23°
8	3	1	7	20	76	31920	18°
9	1	10	7	22	34	52360	14°
10	10	8	3	1	142	34080	17°
11	1	1	2	8	109	1744	28°
12	6	3	9	22	29	103356	10°
13	10	3	9	1	86	23220	19°
14	8	6	4	25	19	91200	11°
15	8	3	5	2	85	20400	22°
16	6	6	2	19	87	119016	8°
17	10	1	8	13	21	21840	21°
18	6	1	1	7	92	3864	25°
19	6	1	6	4	12	1728	29°
20	10	2	3	28	117	196560	6°
21	2	8	9	29	58	242208	4°
22	10	8	6	27	8	103680	9°
23	1	10	7	12	42	35280	16°
24	2	8	7	20	24	53760	13°
25	2	9	8	19	24	65664	12°
26	8	7	7	28	24	263424	3°
27	2	3	6	22	49	38808	15°
28	1	7	3	14	13	3822	26°
29	8	8	5	21	62	416640	2°
30	4	2	1	15	31	3720	27°

Fonte: O autor.

A análise por meio do gráfico da Figura 19 representa que a primeira no ranqueamento foi a FS1, seguida da FS29 e FS26. Para as três funcionalidades, observa-se uma ocorrência classificada como alta (>=7) e com falhas acima de 20. Por outro lado, o gráfico reforçou que as funcionalidades FS5, FS19 e FS11 representaram menor prioridade no ranqueamento (30°, 29° e 28° respectivamente). Nota-se que as funcionalidades com menor prioridade apresentaram registros de falhas abaixo de 9 e uma pontuação de ocorrência no valor de 1 (falha remota ou improvável). Isto significa que inicialmente as equipes de desenvolvimento avaliaram como baixa probabilidade a ocorrência de falhas dessas funcionalidades e foram comprovadas com os resultados apresentados pelos registros nas fases de testes. O custo da falha não está diretamente

relacionado com o tempo de reparo, mas sim com a natureza da falha. Podem ter falhas que sejam rapidamente reparadas, mas o componente a ser substituído é de alto valor.

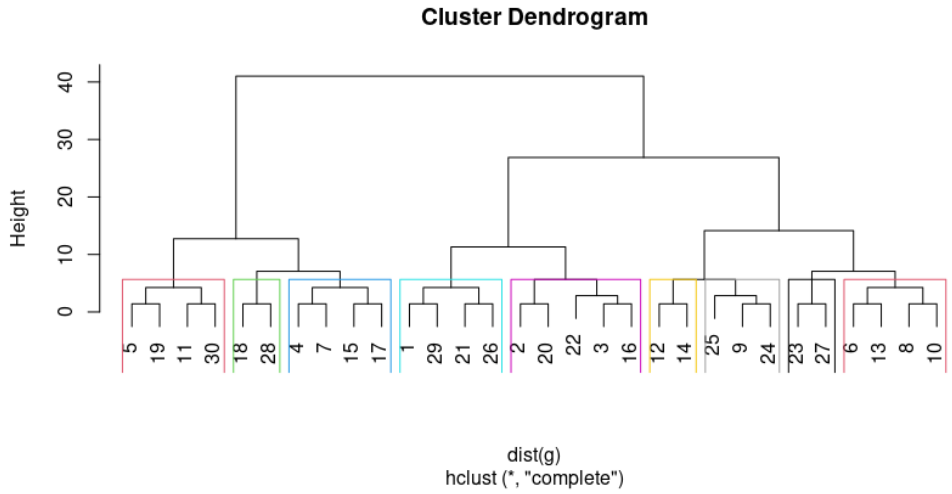
Figura 19. Resultado da priorização de funcionalidades de *software* por EFMEA



Fonte: O autor.

Com o apoio do agrupamento hierárquico aglomerativo, permitiu-se visualizar o resultado do dendrograma representado na Figura 20. O nível de similaridade foi plotado no eixo vertical, representando o ranqueamento das funcionalidades de *software* (FS). No eixo horizontal foi destacado o conjunto composto pelas 30 FS. Este dendrograma representou 9 grupos (*clusters*) compostos de funcionalidades que apresentaram proximidade no ranqueamento da aplicação do método FMEA Estendido.

Figura 20. Dendrograma para o método por EFMEA



Fonte: O autor.

Enumerando os *clusters* da esquerda para a direita de 1 até 9, os resultados analisados destacaram que o primeiro cluster agrupou as funcionalidades com risco baixo composto de: FS5, FS19, FS11 e FS30. A prioridade do tratamento das falhas de funcionalidades de *software* foi estabelecida na seguinte ordem de ranqueamento de *clusters* conforme representado na Tabela 11.

Tabela 11. Ranqueamento de clusters para as funcionalidades de *software*

Cluster	FS	Rank
1	5,19,11,30	9°
2	18,28	8°
3	4,7,15,17	7°
4	1,29,21,26	1°
5	2,20,22,3,16	2°
6	12,14	3°
7	25,9,24	4°
8	23,27	5°
9	6,13,8,10	6°

Fonte: O autor

Neste método observa-se que a prioridade da ação foi específica para cada valor avaliado no conjunto das funcionalidades de *software* analisadas, diferentemente do FMEA tradicional que resultou no mesmo valor de AP. A abordagem do FMEA Estendido apresenta maior poder de discernimento comparado ao FMEA tradicional.

CENÁRIO 3: AVALIAÇÃO DOS RESULTADOS POR DEA

Este cenário apresenta os resultados da aplicação do método DEA que utiliza variáveis de entrada (insumo) e saída (produto) para o cálculo da eficiência. Os resultados analisados foram modelados para redução dos insumos bem como para maximização dos produtos. A Tabela 12 apresenta os valores estabelecidos para este cenário de aplicação do método DEA, considerando o ranqueamento das funcionalidades de *software* (FS) analisadas neste estudo de caso.

Tabela 12. Variáveis de entrada/saída e dados para aplicação do método DEA

FS (DMU)	Ocorrências de Falhas (Quantidade)	Horas Estimadas de Reparo	Teste Unitário (Horas)	Teste de Integração (Horas)	Teste de Sistema (Horas)	Teste de Aceitação (Horas)
1	30	127	28	36	83	58
2	25	101	22	43	18	91
3	9	46	42	43	115	48
4	4	37	38	50	114	106
5	1	48	18	31	31	48
6	19	14	39	11	66	115
7	3	107	43	26	123	40
8	20	76	21	19	118	88
9	22	34	18	27	18	96
10	1	142	23	59	89	37
11	8	109	8	46	119	101
12	22	29	18	11	57	63
13	1	86	30	62	89	59
14	25	19	34	58	49	34
15	2	85	18	51	51	41
16	19	87	38	10	97	90
17	13	21	34	62	90	128
18	7	92	25	33	90	67
19	4	12	10	38	121	80
20	28	117	23	34	29	75
21	29	58	41	58	62	93
22	27	8	11	33	56	121
23	12	42	26	20	24	112
24	20	24	16	35	129	74
25	19	24	10	20	59	37
26	28	24	32	63	24	67
27	22	49	35	36	62	69
28	14	13	29	38	29	15
29	21	62	14	62	51	50
30	15	31	26	8	28	127

Fonte: O autor.

Este método DEA utiliza 2 variáveis de entrada, 4 variáveis de saída e a quantidade de DMUs são referenciadas por 30 funcionalidades de *software*. Nota-se que as regras de ouro (do inglês *Golden Rules*) apresentadas na literatura foram cumpridas, sendo elas:

- $N_{min} = I * S \ \therefore \ (2 * 4) = 8$
- $N_{min} = 3 * (I + S) \ \therefore \ 3* (2 +4) = 18$
- $N_{min} \geq 2 * (I + S) \ \therefore \ 2 * (2 + 4) = 12$

N_{min} representa a quantidade mínima de DMUs que são necessárias para a modelagem DEA.

1.1.13 *Modelo DEA orientado a entrada (CRS)*

O método DEA orientado a entrada (insumo) com retornos constantes de escala (CRS) estabeleceu como metas de redução duas variáveis: a quantidade de falhas e horas estimadas de reparo. Esta modelagem forneceu as melhores funcionalidades de *software* que foram classificadas como *benchmarks* em um conjunto de 30 FS. As melhores apresentaram a eficiência relativa de 100%. Isto é relevante a fim de identificar para as funcionalidades ineficientes quais variáveis de insumo (ocorrência de falhas e horas de reparo) deverão reduzir para atingir a eficiência. Como exemplo para a FS1 a eficiência relativa foi de 18,68%, uma ocorrência de 30 falhas e 127 horas de reparo. As metas estabelecidas foram de 5,60 falhas e 23,72 horas de reparo. Desta forma, se a FS1 alcançar as metas de redução, ela passará de uma funcionalidade ineficiente para eficiente (de 18,68% para 100%). A Tabela 13 apresenta os valores das metas de redução para este cenário de aplicação do método DEA orientado a entrada (insumo), com retornos constantes de escala, considerando o ranqueamento das funcionalidades de *software* (FS) analisadas neste estudo de caso.

Tabela 13. Metas para redução das entradas (CRS)

Metas (*Benchmarking*)

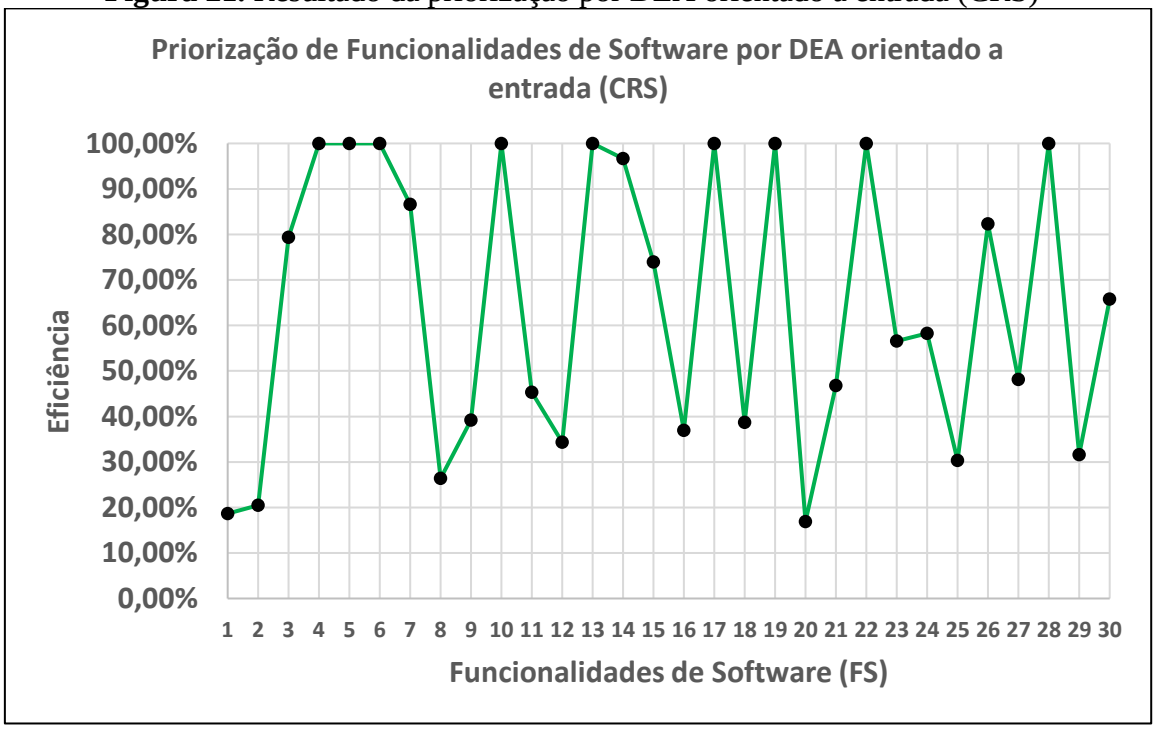
FS (DMU)	Ocorrências de Falhas (Quantidade)	Horas Estimadas de Reparo	Eficiência (θ)	Rank
1	5,60	23,72	18,68%	2°
2	5,12	20,68	20,48%	3°
3	7,14	36,51	79,37%	18°
4	4,00	37,00	100,00%	22°
5	1,00	48,00	100,00%	22°
6	19,00	14,00	100,00%	22°
7	2,60	92,64	86,58%	20°
8	5,28	20,05	26,38%	4°
9	8,63	13,33	39,21%	10°
10	1,00	142,00	100,00%	22°
11	3,63	49,41	45,33%	11°
12	7,56	9,96	34,35%	7°
13	1,00	86,00	100,00%	22°
14	24,17	18,37	96,67%	21°
15	1,48	62,85	73,95%	17°
16	7,02	32,14	36,94%	8°
17	13,00	21,00	100,00%	22°
18	2,71	35,61	38,71%	9°
19	4,00	12,00	100,00%	22°
20	4,74	19,80	16,93%	1°
21	13,56	27,12	46,76%	12°
22	27,00	8,00	100,00%	22°
23	6,79	23,76	56,58%	14°
24	6,94	13,98	58,25%	15°
25	5,76	7,28	30,31%	5°
26	23,05	19,76	82,33%	19°
27	10,59	23,58	48,13%	13°
28	14,00	13,00	100,00%	22°
29	6,63	19,57	31,56%	6°
30	9,87	20,40	65,82%	16°

Fonte: O autor.

A análise por meio do gráfico da Figura 21 representa que a primeira no ranqueamento foi a FS20, seguida da FS1 e FS2. Para as três funcionalidades, observa-se uma eficiência abaixo de 25%, sendo que neste método as com menor eficiência são prioritárias para o tratamento das falhas. Por outro lado, o gráfico reforçou que 9 funcionalidades apresentaram uma eficiência relativa de 100% e foram classificadas como eficientes (FS4, FS5, FS6, FS10, FS13, FS17, FS19, FS22 e FS28). Desta forma, é possível avaliar e distinguir as funcionalidades eficientes e ineficientes, bem como

direcionar as equipes de desenvolvimento na solução do tratamento das falhas de funcionalidades ineficientes.

Figura 21. Resultado da priorização por DEA orientado a entrada (CRS)



Fonte: O autor.

1.1.14 Modelo DEA orientado a saída (CRS)

O método DEA orientado a saída (produto) com retornos constantes de escala (CRS) estabeleceu como metas de aumento quatro variáveis: horas de teste unitário, horas de teste de integração, horas de teste de sistema e horas de teste de aceitação. Esta modelagem forneceu as melhores funcionalidades de *software* que foram classificadas como *benchmarks* em um conjunto de 30 FS. As melhores apresentaram a eficiência relativa de 100%. Isto é relevante a fim de identificar para as funcionalidades ineficientes quais variáveis de produto deverão aumentar para atingir a eficiência. Como exemplo para a FS1 a eficiência relativa foi de 18,68%, 28 horas de teste unitário, 36 horas de teste de integração, 83 horas de teste de sistema e 58 horas de teste de aceitação. As metas de aumento estabelecidas foram de 149,92 horas para teste unitário, 192,75 horas para teste de integração, 440,40 horas de teste de sistema e 456,40 horas de teste de aceitação. Desta forma, se a FS1 alcançar as metas de aumento, ela passará de uma funcionalidade ineficiente para eficiente (de 18,68% para 100%). Para cada funcionalidade há metas específicas de aumento dos testes na fase de validação. A

Tabela 14 apresenta os valores das metas de aumento para este cenário de aplicação do método DEA orientado a saída (produto) com retornos constantes de escala (CRS), considerando o ranqueamento das funcionalidades de *software* (FS) analisadas neste estudo de caso.

Tabela 14. Metas para aumento das saídas (CRS)

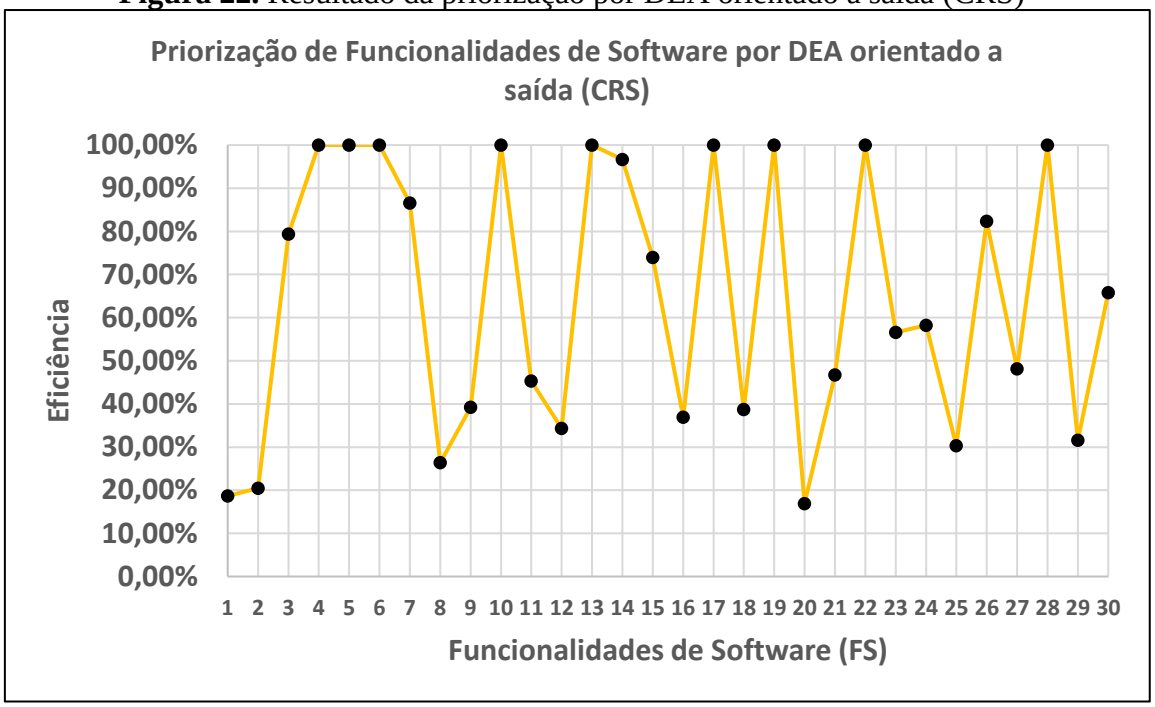
	Metas (*Benchmarking*)					
FS (DMU)	**Teste Unitário (Horas)**	**Teste de Integração (Horas)**	**Teste de Sistema (Horas)**	**Teste de Aceitação (Horas)**	**Eficiência (θ)**	**Rank**
1	149,92	192,75	444,40	456,40	18,68%	2°
2	107,43	211,40	523,97	444,36	20,48%	3°
3	52,91	60,34	147,00	149,03	79,37%	18°
4	38,00	50,00	114,00	106,00	100,00%	22°
5	18,00	31,00	31,00	48,00	100,00%	22°
6	39,00	11,00	66,00	115,00	100,00%	22°
7	49,67	88,61	147,97	113,06	86,58%	20°
8	79,59	153,87	447,24	351,81	26,38%	4°
9	45,90	87,09	293,67	244,82	39,21%	10°
10	23,00	59,00	89,00	37,00	100,00%	22°
11	79,81	123,64	262,53	222,82	45,33%	11°
12	52,39	45,48	165,91	185,83	34,35%	7°
13	30,00	62,00	89,00	59,00	100,00%	22°
14	35,17	60,00	82,80	82,61	96,67%	21°
15	31,17	68,97	116,42	77,12	73,95%	17°
16	102,86	113,22	280,60	290,34	36,94%	8°
17	34,00	62,00	90,00	128,00	100,00%	22°
18	64,58	106,85	232,50	190,39	38,71%	9°
19	10,00	38,00	121,00	80,00	100,00%	22°
20	135,88	211,27	426,16	443,08	16,93%	1°
21	87,68	124,04	224,81	295,33	46,76%	12°
22	11,00	33,00	56,00	121,00	100,00%	22°
23	45,95	94,35	229,90	197,95	56,58%	14°
24	27,47	66,89	221,46	165,92	58,25%	15°
25	32,99	65,98	194,63	167,15	30,31%	5°
26	38,87	76,52	135,84	123,90	82,33%	19°
27	72,72	74,80	177,54	221,10	48,13%	13°
28	29,00	38,00	29,00	15,00	100,00%	22°
29	51,73	196,44	624,83	414,26	31,56%	6°
30	39,50	92,63	205,98	192,96	65,82%	16°

Fonte: O autor.

A análise por meio do gráfico da Figura 22 representa que a primeira no ranqueamento foi a FS20, seguida da FS1 e FS2. Nota-se que o ranqueamento e a

quantidade de funcionalidades eficientes tanto pela modelagem orientada a entrada quanto saída apresentaram os mesmos resultados. Os resultados apresentados identificaram quais testes deverão ser aumentados e quantas horas serão essenciais a fim de direcionar as equipes de desenvolvimento na solução do tratamento das falhas de funcionalidades ineficientes.

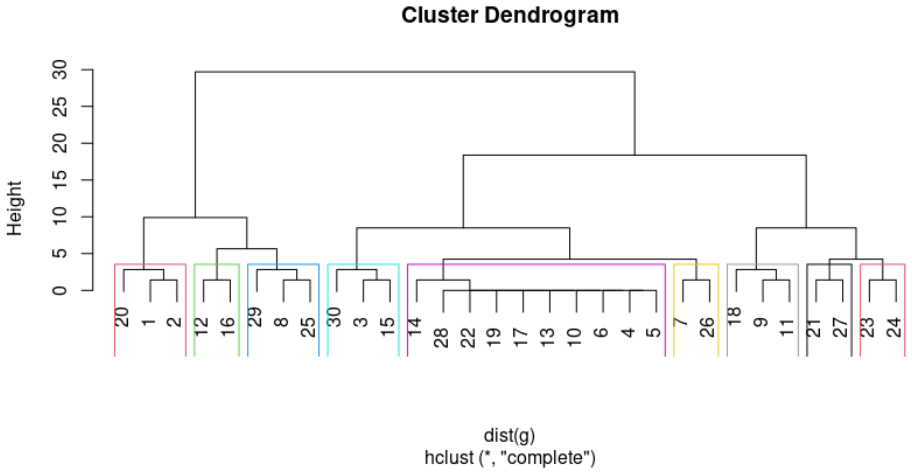
Figura 22. Resultado da priorização por DEA orientado a saída (CRS)



Fonte: O autor.

Com o apoio do agrupamento hierárquico aglomerativo, permitiu-se visualizar o resultado do dendrograma representado na Figura 23. Assim como nos métodos antecessores, o nível de similaridade foi plotado no eixo vertical, representando o ranqueamento das funcionalidades de *software* (FS). No eixo horizontal foi destacado o conjunto composto pelas 30 FS. Este dendrograma representou 9 grupos (*clusters*) compostos de funcionalidades que apresentaram proximidade da eficiência relativa no ranqueamento da aplicação do método DEA com orientação a entrada (insumo), saída (produto) e retornos constantes de escala (CRS).

Figura 23. Dendrograma para o método por DEA (CRS)



Fonte: O autor.

Enumerando os *clusters* da esquerda para a direita de 1 até 9, os resultados analisados destacaram que o primeiro cluster agrupou as funcionalidades com eficiência relativa inferior a 25%, composta de: FS20, FS1 e FS2. O melhor desempenho foi percebido pelo quinto cluster, com eficiência relativa acima de 99%. Este *cluster* é considerado como o *benchmark* e representa as melhores práticas para que os demais busquem a eficiência relativa de 100%. A prioridade do tratamento das falhas de funcionalidades de *software* foi estabelecida na seguinte ordem de ranqueamento de *clusters* conforme representado na Tabela 15.

Tabela 15. Agrupamento por método DEA(CRS)

Clusters	FS	FS Eficientes	Eficiência Média	Rank
1	20,1,2	0	18,66%	1°
2	12,16	0	35,5%	3°
3	29,8,25	0	29,33%	2°
4	30,3,15	0	73%	7°
5	14,28,22, 19,17,13, 10,6,4,5	9	99,7%	9°
6	7,26	0	84,5%	8°
7	18,9,11	0	41%	4°
8	21,27	0	47,5%	5°
9	23,24	0	57,5%	6°

Fonte: O autor.

1.1.15 Modelo DEA orientado a entrada (VRS)

O método DEA orientado a entrada (insumo), com retornos variáveis de escala (VRS) estabeleceu como metas de redução duas variáveis: a quantidade de falhas e horas estimadas de reparo. Esta modelagem forneceu as melhores funcionalidades de *software* que foram classificadas como *benchmarks* em um conjunto de 30 FS. As melhores apresentaram a eficiência relativa de 100%. Isto é relevante a fim de identificar para as funcionalidades ineficientes quais variáveis de insumo (ocorrência de falhas e horas de reparo) deverão reduzir para atingir a eficiência. Como exemplo para a FS1 a eficiência relativa foi de 19,70%, uma ocorrência de 30 falhas e 127 horas de reparo. As metas estabelecidas foram de 5,91 falhas e 25,02 horas de reparo. Desta forma, se a FS1 alcançar as metas de redução, ela passará de uma funcionalidade ineficiente para eficiente (de 19,70% para 100%). A Tabela 16 apresenta os valores das metas de redução para este cenário de aplicação do método DEA orientado a entrada (insumo), com retornos variáveis de escala, considerando o ranqueamento das funcionalidades de *software* (FS) analisadas neste estudo de caso.

Tabela 16. Metas para redução das entradas (VRS)

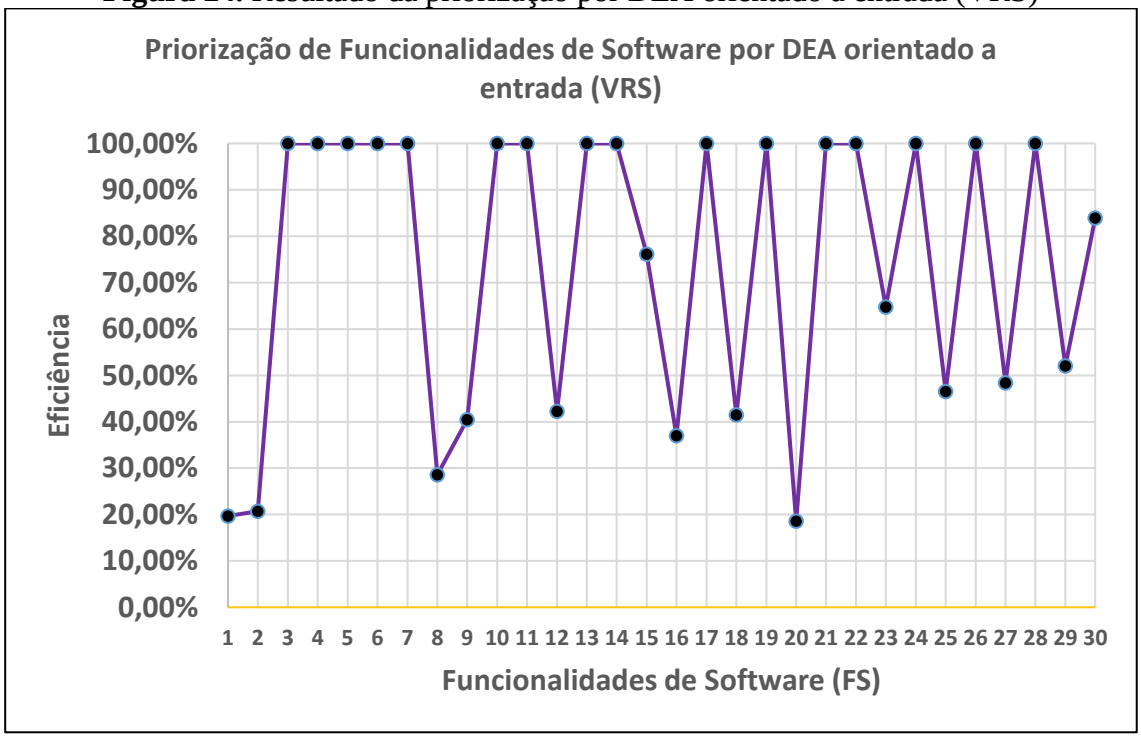
Metas (*Benchmarking*)				
FS (DMU)	**Ocorrências de Falhas (Quantidade)**	**Horas Estimadas de Reparo**	**Eficiência (θ)**	**Rank**
1	5,91	25,02	19,70%	2°
2	5,19	20,95	20,75%	3°
3	9,00	46,00	100,00%	15°
4	4,00	37,00	100,00%	15°
5	1,00	48,00	100,00%	15°
6	19,00	14,00	100,00%	15°
7	3,00	107,00	100,00%	15°
8	4,08	21,70	28,55%	4°
9	8,90	13,75	40,45%	6°
10	1,00	86,00	100,00%	15°
11	8,00	109,00	100,00%	15°
12	9,29	12,25	42,25%	8°
13	1,00	86,00	100,00%	15°
14	25,00	19,00	100,00%	15°
15	1,52	64,74	76,16%	13°
16	7,03	32,18	36,99%	5°
17	13,00	21,00	100,00%	15°
18	2,90	38,15	41,47%	7°
19	4,00	12,00	100,00%	15°
20	5,19	21,70	18,55%	1°
21	29,00	58,00	100,00%	15°
22	27,00	8,00	100,00%	15°
23	7,77	27,20	64,76%	12°
24	20,00	24,00	100,00%	15°
25	8,83	11,16	46,50%	9°
26	28,00	24,00	100,00%	15°
27	10,65	23,72	48,40%	10°
28	14,00	13,00	100,00%	15°
29	10,92	32,25	52,02%	11°
30	12,59	21,73	83,94%	14°

Fonte: O autor.

A análise por meio do gráfico da Figura 24 representa que a primeira no ranqueamento foi a FS20, seguida da FS1 e FS2. Para as três funcionalidades, observa-se uma eficiência abaixo de 25%, sendo que neste método as com menor eficiência são prioritárias para o tratamento das falhas. Por outro lado, o gráfico reforçou que 16 funcionalidades apresentaram uma eficiência relativa de 100% e foram classificadas como eficientes (FS3, FS4, FS5, FS6, FS7, FS10, FS11, FS13, FS14, FS17, FS19, FS21,

FS22, FS24, FS26 e FS28). Desta forma, é possível avaliar e distinguir as funcionalidades eficientes e ineficientes, bem como direcionar as equipes de desenvolvimento na solução do tratamento das falhas de funcionalidades ineficientes.

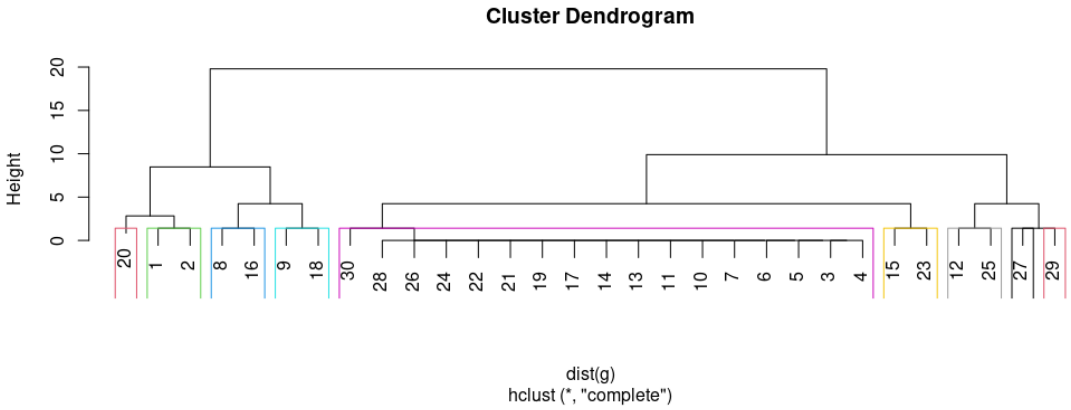
Figura 24. Resultado da priorização por DEA orientado a entrada (VRS)



Fonte: O autor.

Com o apoio do agrupamento hierárquico aglomerativo, permitiu-se visualizar o resultado do dendrograma representado na Figura 25. Assim como nos métodos antecessores, o nível de similaridade foi plotado no eixo vertical, representando o ranqueamento das funcionalidades de *software* (FS). No eixo horizontal foi destacado o conjunto composto pelas 30 FS. Este dendrograma representou 9 grupos (*clusters*) compostos de funcionalidades que apresentaram proximidade da eficiência relativa no ranqueamento da aplicação do método DEA com orientação a entrada (insumo) e retornos variáveis de escala (VRS).

Figura 25. Dendrograma para o método por DEA orientado a entrada (VRS)



Fonte: O autor.

Enumerando os *clusters* da esquerda para a direita de 1 até 9, os resultados analisados destacaram que o primeiro cluster representou a funcionalidade com eficiência relativa inferior a 19%, correspondente a FS20. O melhor desempenho foi percebido pelo quinto cluster, com eficiência relativa média acima de 99%. Este *cluster* é considerado como o *benchmark* e representa as melhores práticas para que os demais busquem a eficiência relativa de 100%. A prioridade do tratamento das falhas de funcionalidades de *software* foi estabelecida na seguinte ordem de ranqueamento de *clusters* conforme representado na Tabela 17.

Tabela 17.Agrupamento por método DEA orientado a entrada (VRS)

Clusters	FS	FS Eficientes	Eficiência Média	Rank
1	20	0	18,55%	1°
2	1,2	0	20,22%	2°
3	8,16	0	32,77%	3°
4	9,18	0	40,96%	4°
5	30,28,26, 24,22,21, 19,17,14, 13,11,10, 7,6,5,3,4	16	99,05%	9°
6	15,23	0	70,46%	8°
7	12,25	0	44,37%	5°
8	27	0	48,40%	6°
9	29	0	52,02%	7°

Fonte: O autor.

1.1.16 Modelo DEA orientado a saída (VRS)

O método DEA orientado a saída (produto) com retornos variáveis de escala (VRS) estabeleceu como metas de aumento quatro variáveis: horas de teste unitário, horas de teste de integração, horas de teste de sistema e horas de teste de aceitação. Esta modelagem forneceu as melhores funcionalidades de *software* que foram classificadas como *benchmarks* em um conjunto de 30 FS. As melhores apresentaram a eficiência relativa de 100%. Isto é relevante a fim de identificar para as funcionalidades ineficientes quais variáveis de produto deverão aumentar para atingir a eficiência. Como exemplo para a FS1 a eficiência relativa foi de 72,92%, 28 horas de teste unitário, 36 horas de teste de integração, 83 horas de teste de sistema e 58 horas de teste de aceitação. As metas de aumento estabelecidas foram de 38,40 horas para teste unitário, 49,37 horas para teste de integração, 113,83 horas de teste de sistema e 100,37 horas de teste de aceitação. Desta forma, se a FS1 alcançar as metas de aumento de testes, ela passará de uma funcionalidade ineficiente para eficiente (de 72,92% para 100%). Para cada funcionalidade há metas específicas de aumento dos testes na fase de validação. A Tabela 18 apresenta os valores das metas de aumento para este cenário de aplicação do método DEA orientado a saída (produto) com retornos variáveis de escala (VRS), considerando o ranqueamento das funcionalidades de *software* (FS) analisadas neste estudo de caso.

Tabela 18. Metas para aumento das saídas (VRS)

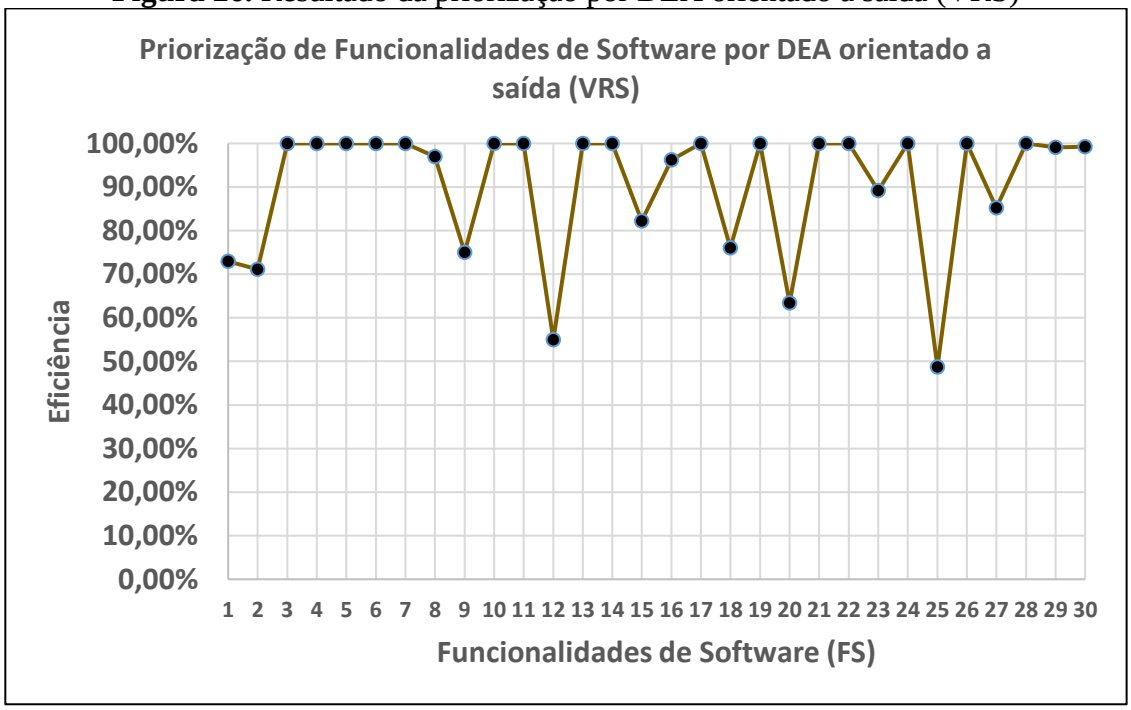
Metas (*Benchmarking*)						
FS (DMU)	**Teste Unitário (Horas)**	**Teste de Integração (Horas)**	**Teste de Sistema (Horas)**	**Teste de Aceitação (Horas)**	**Eficiência (θ)**	**Rank**
1	38,40	49,37	113,83	100,37	72,92%	5°
2	34,00	62,00	90,00	128,00	71,09%	4°
3	42,00	43,00	115,00	48,00	100,00%	15°
4	38,00	50,00	114,00	106,00	100,00%	15°
5	18,00	31,00	31,00	48,00	100,00%	15°
6	39,00	11,00	66,00	115,00	100,00%	15°
7	43,00	26,00	123,00	40,00	100,00%	15°
8	21,66	42,50	121,72	90,78	96,94%	12°
9	34,00	62,00	90,00	128,00	75,00%	6°
10	23,00	59,00	89,00	37,00	100,00%	15°
11	8,00	46,00	119,00	101,00	100,00%	15°
12	35,15	54,80	103,73	114,65	54,95%	2°
13	30,00	62,00	89,00	59,00	100,00%	15°
14	34,00	58,00	49,00	34,00	100,00%	15°
15	30,07	62,04	86,59	59,30	82,21%	8°
16	39,50	48,99	100,83	93,55	96,20%	11°
17	34,00	62,00	90,00	128,00	100,00%	15°
18	32,90	43,42	118,43	90,08	76,00%	7°
19	10,00	38,00	121,00	80,00	100,00%	15°
20	36,29	53,64	80,27	118,33	63,38%	3°
21	41,00	58,00	62,00	93,00	100,00%	15°
22	11,00	33,00	56,00	121,00	100,00%	15°
23	34,44	60,67	92,67	125,56	89,20%	10°
24	16,00	35,00	129,00	74,00	100,00%	15°
25	21,15	41,06	121,14	86,79	48,71%	1°
26	32,00	63,00	24,00	67,00	100,00%	15°
27	41,09	47,68	79,60	81,01	85,18%	9°
28	29,00	38,00	29,00	15,00	100,00%	15°
29	32,59	62,58	51,48	88,20	99,07%	13°
30	34,00	62,00	90,00	128,00	99,22%	14°

Fonte: O autor.

A análise por meio do gráfico da Figura 26 representa que a primeira no ranqueamento foi a FS25, seguida da FS12 e FS20. Nota-se que 16 funcionalidades de *software* foram consideradas como eficientes neste método, uma quantidade superior em comparação ao modelo com retornos variáveis de escala. Os resultados apresentados identificaram quais testes deverão ser aumentados e quantas horas serão essenciais a fim

de direcionar as equipes de desenvolvimento na solução do tratamento das falhas de funcionalidades ineficientes.

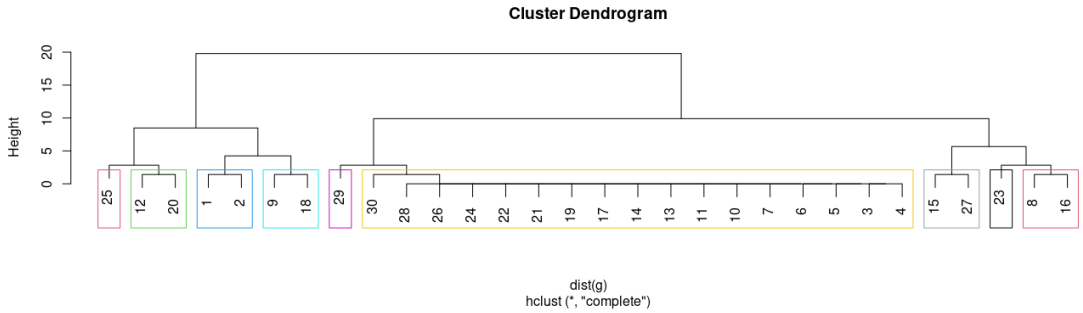
Figura 26. Resultado da priorização por DEA orientado a saída (VRS)



Fonte: O autor.

Com o apoio do agrupamento hierárquico aglomerativo, permitiu-se visualizar o resultado do dendrograma representado na Figura 27. Assim como nos métodos antecessores, o nível de similaridade foi plotado no eixo vertical representando o ranqueamento das funcionalidades de *software* (FS). No eixo horizontal foi destacado o conjunto composto pelas 30 FS. Este dendrograma representou 9 grupos (*clusters*) compostos de funcionalidades que apresentaram proximidade da eficiência relativa no ranqueamento da aplicação do método DEA com orientação saída (produto) e retornos variáveis de escala (VRS).

Figura 27. Dendrograma para o método por DEA orientado a saída (VRS)



Fonte: O autor.

Enumerando os *clusters* da esquerda para a direita de 1 até 9, os resultados analisados destacaram que o primeiro cluster representou a funcionalidade com eficiência relativa inferior a 49%, correspondente a FS25. O melhor desempenho foi percebido pelo sexto cluster, com eficiência relativa média acima de 99,9%. Este *cluster* é considerado como o *benchmark* e representa as melhores práticas para que os demais busquem a excelência. A prioridade do tratamento das falhas de funcionalidades de *software* foi estabelecida na seguinte ordem de ranqueamento de *clusters* conforme representado na Tabela 19.

Tabela 19. Agrupamento por método DEA orientado a saída (VRS)

Clusters	FS	FS Eficientes	Eficiência Média	Rank
1	25	0	48,71%	1°
2	12,20	0	59,16%	2°
3	1,2	0	72 %	3°
4	9,18	0	75,5%	4°
5	29	0	99,07%	8°
6	30,28,26, 24,22,21, 19,17,14, 13,11,10, 7,6,5, 3,4	16	99,95%	9°
7	15,27	0	83,69%	5°
8	23	0	89,20%	6°
9	8,16	0	96,57%	7°

Fonte: O autor.

COMPARATIVO DOS MÉTODOS FMEA, FMEA ESTENDIDO E DEA

Com a implementação dos três métodos, o ranqueamento das 30 funcionalidades de *software* foi estabelecido conforme a Tabela 20.

Tabela 20. Comparativo do ranqueamento na aplicação dos métodos do estudo

FS (DMU)	Ranqueamento					
	FMEA Tradicional	**FMEA Estendido**	**DEA CRS (Insumo)**	**DEA CRS (Produto)**	**DEA VRS (Insumo)**	**DEA VRS (Produto)**
1	4°	1°	2°	2°	2°	5°
2	14°	5°	3°	3°	3°	4°
3	2°	7°	18°	18°	15°	15°
4	18°	24°	22°	22°	15°	15°
5	25°	30°	22°	22°	15°	15°
6	14°	20°	22°	22°	15°	15°
7	22°	23°	20°	20°	15°	15°
8	26°	18°	4°	4°	4°	12°
9	18°	14°	10°	10°	6°	6°
10	7°	17°	22°	22°	15°	15°
11	30°	28°	11°	11°	15°	15°
12	9°	10°	7°	7°	8°	2°
13	6°	19°	22°	22°	15°	15°
14	8°	11°	21°	21°	15°	15°
15	12°	22°	17°	17°	13°	8°
16	17°	8°	8°	8°	5°	11°
17	16°	21°	22°	22°	15°	15°
18	29°	25°	9°	9°	7°	7°
19	23°	29°	22°	22°	15°	15°
20	21°	6°	1°	1°	1°	3°
21	10°	4°	12°	12°	15°	15°
22	1°	9°	22°	22°	15°	15°
23	18°	16°	14°	14°	12°	10°
24	13°	13°	15°	15°	15°	15°
25	10°	12°	5°	5°	9°	1°
26	3°	3°	19°	19°	15°	15°
27	23°	15°	13°	13°	10°	9°
28	26°	26°	22°	22°	15°	15°
29	5°	2°	6°	6°	11°	13°
30	28°	27°	16°	16°	14°	14°

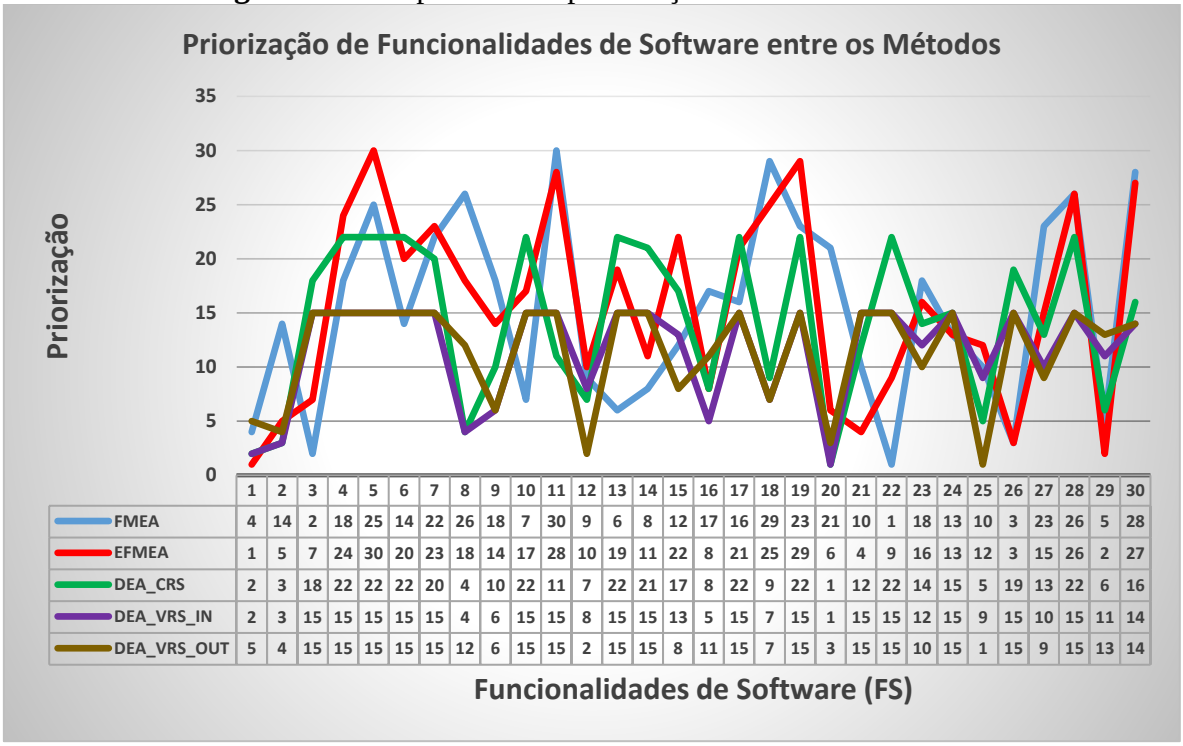
Fonte: O autor.

No FMEA tradicional, observa-se que uma mesma priorização (18°) foi pontuada para 3 diferentes funcionalidades (FS4, FS9 e FS23), gerando mesma prioridade da ação como comprovado neste trabalho. As variáveis de tempo de reparo e quantidade de falhas foram adicionadas na análise como proposta para extensão do método FMEA tradicional. O resultado para o ranqueamento por este método apresenta

uma ordenação única e específica, diferentemente do FMEA tradicional. Desta forma, o poder de discernimento é superior no FMEA estendido. Por outro lado a modelagem DEA proporciona o conceito da eficiência relativa, que distingui as funcionalidades eficientes das ineficientes através do coeficiente (θ) que atribui o valor de 1 (100%) para as DMUs eficientes. Para um conjunto de 30 DMUs que neste caso foram as funcionalidades de *software*, nota-se que as prioritárias foram as que apresentaram a menor eficiência relativa. No método DEA-CRS, um total de 9 funcionalidades resultaram na eficiência relativa (θ) de 100% (FS4, FS5, FS6, FS10, FS13, FS17, FS19, FS22 e FS28) e foram classificadas como eficientes e de menor prioridade para o tratamento das falhas (ranqueadas em 22°). No método DEA-VRS, a modelagem considera uma restrição adicional (Eq. 2.17) e um total de 16 funcionalidades resultaram na eficiência relativa (θ) de 100% (FS3, FS4, FS5, FS6, FS7, FS10, FS11, FS13, FS14, FS17, FS19, FS21, FS22 , FS24, FS26 e FS28) e foram classificadas como eficientes e de menor prioridade para o tratamento das falhas (ranqueadas em 15°). Este método DEA é vantajoso comparado aos demais métodos pois além de indicar e distinguir funcionalidades eficientes das ineficientes, fornece os esforços a serem feitos para que a unidade em estudo atinja a eficiência. Além disso, para a modelagem DEA-CRS orientado a entrada e saída resultaram no mesmo ranqueamento. Para a modelagem DEA-VRS a quantidade de unidades eficientes foi superior comparado ao modelo DEA-CRS. Nota-se também que tanto para a orientação a entrada quanto saída, a modelagem DEA-VRS classificou a mesma quantidade de unidades eficientes e na mesma posição no ranqueamento (15°).

A análise por meio do gráfico da Figura 28 representa o comparativo da priorização das funcionalidades de *software* para os diferentes métodos deste estudo de caso.

Figura 28. Comparativo da priorização entre os métodos



Fonte: O autor.

Foi construído por meio de dendrogramas associados a cada método implementado o agrupamento das funcionalidades de *software*. A Tabela 19 contém *clusters* formados para cada um dos métodos propostos.

Tabela 21. Agrupamento das funcionalidades de *software* em clusters para os métodos

	Métodos				
Clusters	**FMEA**	**EFMEA**	**DEA (CRS)**	**DEA (VRS) Insumo**	**DEA (VRS) Produto**
1	30,11, 18	5,19, 11,30	20,1,2	20	25
2	5,8, 28	18,28	12,16	1,2	12,20
3	20,7, 19,27	4,7, 15,17	29,8,25	8,16	1,2
4	15,24, 2,6	1,29, 21,26	30,3, 15	9,18	9,18
5	17,16, 23,4,9	2,20,22 3,16	14,28,22, 19,17,13,	30,28,26, 24,22,21,	29

			10,6,4,5	19,17,14, 13,11,10, 7,6,5,3,4	
6	3,22	12,14	7,26	15,23	30,28,26, 24,22,21, 19,17,14, 13,11,10, 7,6,5,3,4
7	29,1,26	25,9,24	18,9,11	12,25	15,27
8	10,13	23,27	21,27	27	23
9	21,25, 12,14	6,13, 8,10	23,24	29	8,16

Fonte: O autor.

Os resultados apresentados demonstraram que, com exceção do método DEA-CRS, cada método implementado gera uma agrupamento de funcionalidades de *software* e priorização do tratamento de falhas único e específico. Não há correlação entre os agrupamentos, ou seja, as respostas a qualquer tratamento a partir destes agrupamentos serão diferentes.

PROPOSIÇÃO DE AÇÕES DE MELHORIA

Os métodos FMEA e EFMEA aplicados neste estudo de caso priorizam os riscos para o tratamento de falhas das funcionalidades de *software* através da maior pontuação resultante do produto entre os valores estabelecidos pelas equipes de desenvolvimento. Da mesma forma, a proposição de ações recomendadas para a melhoria da qualidade de *software* são consideradas pelas equipes multiculturais e multidisciplinares, que interagem em todo o ciclo de desenvolvimento do produto. Nestes métodos não é indicado o que e o quanto deverá ser estabelecido para alcançar a excelência. Em contrapartida, a abordagem do método DEA apresenta a modelagem orientada a entrada com retornos constantes de escala que estabelece as metas para atingir a excelência. A Tabela 22 apresenta a proposição de melhorias no método DEA(CRS) com orientação a entrada, estabelecendo as metas de redução dos insumos, considerados como ocorrência de falhas e horas de reparo para o conjunto de funcionalidades de *software* em estudo.

Tabela 22. Proposição de melhorias por meio do método DEA orientado a entrada (CRS)

(FS)	Metas de redução (CRS)		Proposição de melhorias
	Ocorrências de Falhas	**Horas Estimadas de Reparo**	**Reduzir os insumos**
1	5,60	23,72	Reduzir em 81% a ocorrência de falhas e horas de correção
2	5,12	20,68	Reduzir em 80% a ocorrência de falhas e horas de correção
3	7,14	36,51	Reduzir em 21% a ocorrência de falhas e horas de correção
7	2,60	92,64	Reduzir em 13% a ocorrência de falhas e horas de correção
8	5,28	20,05	Reduzir em 74% a ocorrência de falhas e horas de correção
9	8,63	13,33	Reduzir em 61% a ocorrência de falhas e horas de correção
11	3,63	49,41	Reduzir em 55% a ocorrência de falhas e horas de correção
12	7,56	9,96	Reduzir em 66% a ocorrência de falhas e horas de correção
14	24,17	18,37	Reduzir em 3% a ocorrência de falhas e horas de correção
15	1,48	62,85	Reduzir em 26% a ocorrência de falhas e horas de correção
16	7,02	32,14	Reduzir em 63% a ocorrência de falhas e horas de correção
18	2,71	35,61	Reduzir em 61% a ocorrência de falhas e horas de correção
20	4,74	19,80	Reduzir em 83% a ocorrência de falhas e horas de correção
21	13,56	27,12	Reduzir em 53% a ocorrência de falhas e horas de correção
23	6,79	23,76	Reduzir em 43% a ocorrência de falhas e horas de correção
24	6,94	13,98	Reduzir em 65% a ocorrência de falhas e horas de correção
25	5,76	7,28	Reduzir em 70% a ocorrência de falhas e horas de correção
26	23,05	19,76	Reduzir em 18% a ocorrência de falhas e horas de correção
27	10,59	23,58	Reduzir em 52% a ocorrência de falhas e horas de correção
29	6,63	19,57	Reduzir em 68% a ocorrência de falhas e horas de correção
30	9,87	20,40	Reduzir em 34% a ocorrência de falhas e horas de correção

Fonte: O autor.

A Tabela 23 apresenta a proposição de melhorias no método DEA(VRS) com orientação a entrada, estabelecendo as metas de redução dos insumos, considerados

como ocorrência de falhas e horas de reparo para o conjunto de funcionalidades de *software* em estudo.

Tabela 23. Proposição de melhorias por meio do método DEA orientado a entrada (VRS)

(FS)	Metas de redução (VRS)		Proposição de melhorias
	Ocorrências de Falhas	Horas Estimadas de Reparo	Reduzir os insumos
1	5,91	25,02	Reduzir em 80% a ocorrência de falhas e horas de correção
2	5,19	20,95	Reduzir em 79% a ocorrência de falhas e horas de correção
8	4,08	21,70	Reduzir em 80% a ocorrência de falhas e em 71% as horas de correção
9	8,90	13,75	Reduzir em 60% a ocorrência de falhas e horas de correção
12	9,29	12,25	Reduzir em 58% a ocorrência de falhas e horas de correção
15	1,52	64,74	Reduzir em 24% a ocorrência de falhas e horas de correção
16	7,03	32,18	Reduzir em 63% a ocorrência de falhas e horas de correção
18	2,90	38,15	Reduzir em 59% a ocorrência de falhas e horas de correção
20	5,19	21,70	Reduzir em 81% a ocorrência de falhas e horas de correção
23	7,77	27,20	Reduzir em 35% a ocorrência de falhas e horas de correção
25	8,83	11,16	Reduzir em 54% a ocorrência de falhas e horas de correção
27	10,65	23,72	Reduzir em 52% a ocorrência de falhas e horas de correção
29	10,92	32,25	Reduzir em 48% a ocorrência de falhas e horas de correção
30	12,59	21,73	Reduzir em 16% a ocorrência de falhas e em 30% as horas de correção

Fonte: O autor.

A Tabela 24 apresenta a proposição de melhorias no método DEA(CRS) com orientação a saída, estabelecendo as metas de aumento dos produtos, considerados como as horas de teste unitário, teste de integração, teste de sistema e de aceitação para o conjunto de funcionalidades de *software* em estudo.

Tabela 24. Proposição de melhorias por meio do método DEA orientado a saída (CRS)

(FS)	Metas de aumento (CRS)				Proposição de melhorias
	Teste Unitário (Horas)	Teste de Integração (Horas)	Teste de Sistema (Horas)	Teste de Aceitação (Horas)	Maximizar os produtos
1	149,92	192,75	444,40	456,40	Aumentar em 435% as horas de teste unitário, integração, sistema e em 687% as horas de teste de aceitação.
2	107,43	211,40	523,97	444,36	Aumentar em 388% as horas de teste unitário e aceitação, 392% de integração e 2811% de sistema.
3	52,91	60,34	147,00	149,03	Aumentar em 26% as horas de teste unitário, 40% de integração, 28% de sistema e 210% de aceitação.
7	49,67	88,61	147,97	113,06	Aumentar em 16% as horas de teste unitário, 241% de integração, 20% de sistema e 183% de aceitação.
8	79,59	153,87	447,24	351,81	Aumentar em 279% as horas de teste unitário e sistema, 710% de integração e 300% de aceitação.
9	45,90	87,09	293,67	244,82	Aumentar em 155% as horas de teste unitário e aceitação, 223% de integração e 1532% de sistema.
11	79,81	123,64	262,53	222,82	Aumentar em 898% as horas de teste unitário, 169% de integração e 121% de sistema e aceitação.
12	52,39	45,48	165,91	185,83	Aumentar em 191% as horas de teste unitário e sistema,

					313% de integração e 195% de aceitação.
14	35,17	60,00	82,80	82,61	Aumentar em 3% as horas de teste unitário e integração, 69% de sistema e 143% de aceitação.
15	31,17	68,97	116,42	77,12	Aumentar em 73% as horas de teste unitário, 35% de integração, 128% de sistema e 88% de aceitação.
16	102,86	113,22	280,60	290,34	Aumentar em 171% as horas de teste unitário, 1032% de integração, 189% de sistema e 223% de aceitação.
18	64,58	106,85	232,50	190,39	Aumentar em 158% as horas de teste unitário e sistema, 224% de integração e em 184% de aceitação.
20	135,88	211,27	426,16	443,08	Aumentar em 491% as horas de teste unitário e aceitação, 521% de integração e em 1370% de sistema.
21	87,68	124,04	224,81	295,33	Aumentar em 114% as horas de teste unitário e integração, 263% de sistema e em 218% de aceitação.
23	45,95	94,35	229,90	197,95	Aumentar em 77% as horas de teste unitário e aceitação, 372% de integração e em 858% de sistema.
24	27,47	66,89	221,46	165,92	Aumentar em 72% as horas de teste unitário e sistema, 91% de integração e em 124% de

					aceitação.
25	32,99	65,98	194,63	167,15	Aumentar em 230% as horas de teste unitário, integração e sistema e em 352% de aceitação.
26	38,87	76,52	135,84	123,90	Aumentar em 21% as horas de teste unitário e integração, 466% de sistema e 85% de aceitação.
27	72,72	74,80	177,54	221,10	Aumentar em 108% as horas de teste unitário e integração, 186% de sistema e 220% de aceitação.
29	51,73	196,44	624,83	414,26	Aumentar em 269% as horas de teste unitário, 217% de integração, 1125% de sistema e 729% de aceitação.
30	39,50	92,63	205,98	192,96	Aumentar em 52% as horas de teste unitário e aceitação, 1058% de integração e 636% de sistema.

Fonte: O autor.

A Tabela 25 apresenta a proposição de melhorias no método DEA(VRS) com orientação a saída, estabelecendo as metas de aumento dos produtos, considerados como as horas de teste unitário, teste de integração, teste de sistema e de aceitação para o conjunto de funcionalidades de *software* em estudo.

Tabela 25. Proposição de melhorias por meio do método DEA orientado a saída (VRS)

(FS)	**Metas de aumento (VRS)**				**Proposição de melhorias**
	Teste Unitário (Horas)	**Teste de Integração (Horas)**	**Teste de Sistema (Horas)**	**Teste de Aceitação (Horas)**	**Maximizar os produtos**
1	38,40	49,37	113,83	100,37	Aumentar em 37% as horas de teste unitário, integração, sistema e em 73% as horas de

					teste de aceitação.
2	34,00	62,00	90,00	128,00	Aumentar em 55% as horas de teste unitário, 44% de integração, 400% de sistema e 41% de aceitação.
8	21,66	42,50	121,72	90,78	Aumentar em 3% as horas de teste unitário, sistema, aceitação e em 124% as horas de teste de integração.
9	34,00	62,00	90,00	128,00	Aumentar em 89% as horas de teste unitário, 130% de integração, 400% de sistema e 33% de aceitação.
12	35,15	54,80	103,73	114,65	Aumentar em 95% as horas de teste unitário, 398% de integração e em 82% de sistema e aceitação.
15	30,07	62,04	86,59	59,30	Aumentar em 67% as horas de teste unitário, 22% de integração, 70% de sistema e 45% de aceitação.
16	39,50	48,99	100,83	93,55	Aumentar em 4% as horas de teste unitário, sistema, aceitação e em 390% as horas de teste de integração.
18	32,90	43,42	118,43	90,08	Aumentar em 32% as horas de teste unitário, integração e sistema e em 34% as horas de teste de aceitação.
20	36,29	53,64	80,27	118,33	Aumentar em 58% as horas de teste unitário, integração e aceitação e em 177% as horas de teste de sistema.
23	34,44	60,67	92,67	125,56	Aumentar em 32% as horas de teste unitário, 203% de integração, 286% de sistema e 12% de aceitação.
25	21,15	41,06	121,14	86,79	Aumentar em 111% as horas de teste unitário, 105% de integração e

					sistema e em 135% de aceitação.
27	41,09	47,68	79,60	81,01	Aumentar em 17% as horas de teste unitário e de aceitação,32% de integração e em 28% de sistema.
29	32,59	62,58	51,48	88,20	Aumentar em 133% as horas de teste unitário, 1% de integração e sistema, e em 76% de aceitação.
30	34,00	62,00	90,00	128,00	Aumentar em 31% as horas de teste unitário, 675% de integração, 221% de sistema e 1% de aceitação.

Fonte: O autor.

Diferentemente do FMEA e EFMEA, o DEA destaca quais funcionalidades de *software* são eficientes, bem como estabelece as metas para que as funcionalidades ineficientes alcancem a excelência. Isto torna-se relevante para auxiliar as organizações na tomada de decisão em um ambiente de desenvolvimento de *software* que interage com equipes multiculturais e multidisciplinares, no sentido de alocar os recursos na correção de funcionalidades com menor desempenho, a fim de buscar a melhoria contínua da qualidade de *software*.

CONSIDERAÇÕES FINAIS

CONCLUSÕES

Este trabalho apresenta uma abordagem dos principais desafios que o segmento automotivo enfrenta no desenvolvimento das funcionalidades de *software* embarcado. Destaca que a preocupação constante com a qualidade nas fases do projeto, manufatura e operação são essenciais para garantir a sustentabilidade econômica das organizações. Nas montadoras automotivas e sistemistas, é reconhecido que as falhas em funcionalidades de *software* geram prejuízos financeiros, retrabalhos, campanhas de *recall* e perda de reputação das marcas. Para mitigar estes riscos e o tratamento das falhas, os métodos propostos visam contribuir com a qualidade e confiabilidade no desenvolvimento de *software*. Os recursos humanos qualificados bem como os custos no desenvolvimento são finitos. Nesse sentido, a priorização do tratamento de riscos das falhas e alocação dos recursos para a garantia da qualidade demonstra que este trabalho é de extrema importância.

Este trabalho buscou analisar métodos de priorização e categorização de riscos. A abordagem do FMEA é amplamente divulgada em vários campos da indústria, entretanto passa por críticas na comunidade científica para a classificação dos riscos e tomada de decisão. Desta forma, para solucionar esta lacuna foram apresentados neste trabalho o método FMEA estendido e o método DEA.

Por fim, é importante enfatizar que os métodos aplicados neste trabalho estão condicionados aos atributos considerados por uma equipe de desenvolvimento de *software* embarcado para o segmento automotivo. Qualquer valor de variável ou funcionalidade alterada modificará os resultados apresentados.

CONTRIBUIÇÕES DO TRABALHO

Este trabalho tem como principal contribuição a proposta da metodologia para apoio à tomada de decisão no desenvolvimento e correção de falhas de *software* embarcado automotivo.

Do ponto de vista do produto, o aumento da confiabilidade e qualidade são os elementos que destacam-se neste estudo. O tratamento de falhas é de extrema importância visando a qualidade e reduzindo os custos. Para as organizações é

importante ressaltar o impacto financeiro e de reputação da marca causado por falhas em funcionalidades de *software*.

Este trabalho contribui para a qualidade de *software* com proposta de métodos científicos para avaliação e tratamento de falhas. Com a aplicação dos métodos (FMEA, FMEA estendido e DEA), cria-se uma solução de análise, avaliação e priorização de riscos eficiente e robusta, que coopera com a tomada de decisão e integridade das políticas de qualidade organizacionais, melhorando a confiabilidade e maximizando a eficiência das operações.

SUGESTÕES PARA TRABALHOS FUTUROS

Como trabalhos futuros propõe-se a integração de um *framework* de aprendizado de máquina para auxiliar no tratamento das falhas e avaliação dos riscos. No que tange específico o método DEA, o trabalho poderá ser estendido através do uso do algoritmo DEA-SBM, também conhecido como DEA aditivo baseado em folgas (do inglês *Slacks-Based Measure* – SBM). Além disso, as metodologias apresentadas neste trabalho poderão ser implementadas nas mais diversas aplicações e segmentos, visando a contribuição no planejamento estratégico das organizações, otimização de recursos, *benchmarking*, avaliação de riscos e benefícios financeiros nos projetos.

Por fim, como perspectivas futuras pode-se complementar e estender o estudo com a implementação de outros métodos e estratégias de gerenciamento de riscos, considerando vulnerabilidades, perigos (ameaças) e exposição. Neste trabalho foram utilizados os métodos no ambiente de desenvolvimento de *software* embarcado automotivo. Diversos contextos e projetos de desenvolvimento podem realizar o tratamento das falhas com a proposta deste estudo, tais como: desenvolvimento de *software Web*, internet das coisas (do inglês *Internet of Things*) ou computação em nuvem (do inglês *Cloud Computing*). Também poderá ser alvo de estudos futuros o uso destes métodos em projetos que não sejam de domínio de desenvolvimento de *software*, como exemplos: setores administrativos, industriais e financeiros.

REFERÊNCIAS

ABDULLAH, D.; SUSILO, S.; AHMAR, A. S.; R RUSLI, R. E HIDAYAT, R. (2022). The application of K-means clustering for province clustering in Indonesia of the risk of the COVID-19 pandemic based on COVID-19 data. Quality & Quantity, v. 56, n. 3, pp.1283-1291. https://doi.org/10.1007/s11135-021-01176-w

AIAG/VDA. (2019). Failure Mode and Effect Analysis. Michigan, Vol. 1.

AIAG_VDA_FMEA, AIAG. (2023). FMEA - Failure Mode & Effects Analysis. Disponível em: https://www.aiag.org/quality/automotive-core-tools/fmea. Acesso em 2 de setembro de 2023.

ÁLVAREZ, I., BARBERO, J. e ZOFIO, P.J. (2020). A Data Envelopment Analysis Toolbox for MATLAB. 2020, Journal of Statistical Software, v. 95, n. 3.

ANACKOVSKI, F.; KUZMANOV, I. e PASIC, R. (2021). Action Priority in new FMEA as factor for Resources Management in Risk Reduction. International Journal of Science & Engineering Research, v. 12, n. 4

ANWAR, A. (2014). A review of RUP (Rational Unified Process). Atlanta : IJSE - International Journal of Software Engineering.

ASPICE (2017). Automotive SPICE® Process Reference and Assessment Model. VDA QMC Quality Management in the Automotive Industry. Disponível em: https://www.automotivespice.com/fileadmin/softwaredownload/AutomotiveSPICE_PAM_31.pdf. Acesso em 10 de novembro de 2023.

ASQ (2023). American Society for Quality. Disponível em: https://asq.org/ Acesso em 10 de outubro de 2023.

ATLASSIAN (2023). O Manifesto Ágil ainda é válido? *Atlassian.* Disponível em] https://www.atlassian.com/br/agile/manifesto. Acesso em 3 de outubro de 2023.

AVEN, T. (2015). Risk analysis. John Wiley & Sons, doi:10.1002/9781119057819.

BAGHERI, E.; ASADI, M.; GASEVIC, D.; e SOLTANI, S. (2010). Stratified Analytic Hierarchy Process: Prioritization and Selection of Software Features. In: Bosch, J., Lee, J. (eds) Software Product Lines: Going Beyond. SPLC 2010. Lecture Notes in Computer Science, vol 6287. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-15579-6_21

BANKER, R.D., CHARNES, A, e COOPER, W.W. (1984). Some models for estimating technical and scale inefficiencies in data envelopment analysis. Management Science, v. 30, n. 9, pp. 1078-1092.

BANXIA SOFTWARE (2023). Frontier Analyst versão 4.0. Disponível em: https://banxia.com/frontier/. Acesso em 10 de novembro de 2023.

BARSALOU, M. (2022). Investigation into a Potential Reduction of FMEA Efforts Using Action Priority. Management and Production Engineering Review, v. 13, n. 4, pp. 59 - 71.

BAUER, E.; ZHANG, X. e KIMBER, D. A. (2009). Practical system reliability. Wiley-IEEE Press; 1st edition.

BAZOVSKY, I. (1961). Reliability Theory and Practice. Printice Hall Inc., New Jersey. Disponível em: https://babel.hathitrust.org/cgi/pt?id=mdp.39015000449549&seq=11

BBC (2017). *Tesla recalls 53,000 cars over brake issue. Disponível em:* http://www.bbc.com/news/business-39663382?ocid=socialflow_twitter. Acesso em 2 de novembro de 2023.

BECKER, J.; SCHOCH, D.; LEEPER, T. J. (2023). A Swiss-Army Knife for Data I/O. [Online] 2023. Disponivel em:https://cran.rproject.org/web/packages/rio/index.html. Acesso em 9 de dezembro de 2023.

BERTSCHE, B. (2008). Reliability in Automotive and Mechanical Engineering: Determination of component and system reliability. Springer Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-34282-3

BLOKUS-ROSZKOWSKA, A.; KOLOWROCKI, K. (2014). Reliability analysis of complex shipyard transportation system with dependent components. Journal of Polish Safety and Reliability Association Summer Safety and Reliability Seminars, v. 5, n. 1.

BOGETOFT, P.; OTTO,L. (2022). Benchmark and Frontier Analysis Using DEA and SFA. Disponível em: https://cran.r-project.org/web/packages/Benchmarking/index.html. Acesso em 10 de novembro de 2023.

BOEHMKE, B. (2023). Hierarchical Cluster Analysis. *UC Business Analytics R Programming Guide.* Disponível em: https://uc-r.github.io/hc_clustering. Acesso em 5 de setembro de 2023.

HUNTER, P. (2015). Takata airbag recall biggest in history at 33.8 million vehicles. [Online] CBC News,. Disponível em: https://www.cbc.ca/news/business/takata-airbag-recall-biggestin-history-at-33-8-million-vehicles-1.3079490. Acesso em 3 de novembro de 2023.

CELEBI, M. E. (2015). Partitional clustering algorithms. Springer International Publishing Switzerland 2015. DOI: https://doi.org/10.1007/978-3-319-09259-1

CHARNES, A., COOPER, W. W. e RHODES, E. (1978). Measuring the efficiency of decision making units. European Journal of Operational Research, v. 2, n. 6, November 1978, pp. 429-444

CHIN, K-S.; WANG, Y-M.; POON, G. K. K. e YANG, J.-B. (2009). Failure mode and effects analysis by data envelopment analysis. Decision Support Systems, v. 48, pp. 246--256.

CMMI. (2023). *What is CMMI?* s.l. : ISACA, 2023.

COOPER, H. C. (2015). Capture all critical failure modes into FMEA in half the time with a simple decomposition table. RAMS - 2015 Annual Reliability and Maintainability Symposium. DOI: 10.1109/RAMS.2015.7105140

COSTA, M.A; LOPES-AHN, A. L.; KILGER, A. C. e MICAS, A. F. (2023). Limitations of weight restrictions in data envelopment analysis for benchmarking Brazilian electricity distribution system operators. Utilities Policy, v. 82, June 2023, 101540

COLL-SERRANO, V.; BOLOS, V.; SUAREZ, R. B. (2023). deaR: Conventional and Fuzzy Data Envelopment Analysis. Disponível em https://cran.r-project.org/web/packages/deaR/index.html. Acesso em 2 de novembro de 2023.

CRISTEA, G.; CONSTANTINESCU, D. M. (2017). A comparative critical study between FMEA and FTA risk analysis methods. IOP Conf. Ser.: Mater. Sci. Eng., v. 252, 012046 DOI 10.1088/1757-899X/252/1/012046

CROSBY, Philip B. (1985) “Qualidade é investimento”, José Olympio, Rio de Janeiro

DEMING, W.E. (1990). Qualidade: a revolução na administração.Rio de Janeiro: Marques Saraiva.

DUCOTE, P.; LOBITZ, B.; DAVID VIROST, D.; MARTIN, M. A. (2022). Failsafe Motor Control Design to Prevent Runaway Motors. IEEE Transactions on Industry Applications, v. 58, n. 4, pp. 4271 – 4278.

DYER, M. K.; LITTLE, D. G.; HOARD, E. G.; TAYLOR, A. C.; CAMPBELL, R.(1972). *Applicability of NASA contract quality management and failure mode effect analysis procedures to the USGS outer continental shelf oil and gas lease management program. NASA Technical Memorandum TX-2567, Washington, DC, 1972*

EMROUZNEJAD, A. e AMIN, G. R. (2009). DEA models for ratio data: Convexity consideration. Applied Mathematical Modelling, v. 33,, n. 1, pp. 486-498.

EMROUZNEJAD, A. e YANG, G.L. (2018). A survey and analysis of the first 40 years of scholarly literature in DEA: 1978–2016. Socio-Economic Planning Sciences, v. 61, pp. 4--8

ENGADGET (2019). Nissan disables its Leaf remote control app (update). Disponível em: https://www.engadget.com/2016/02/24/nissan-leafsconnected-climate-control-has-a-security-flaw. Acesso em 10 de setembro de 2023.

FAPESP (2023). Obstacles to growth of the electric car market in Brazil. Disponível em: https://revistapesquisa.fapesp.br/en/obstacles-to-growthof-the-electric-car-market-in-brazil/. Acesso em 10 de dezembro de 2023.

FEIGENBAUM, A. V. (1990). "Total Quality Control", Third Edition, Pittsfield, Massachussets.

FITZSIMMONS, J. A. e FITZSIMMONS, M. J. (2014). Administração de Serviços: Operações, Estratégia e Tecnologia da Informação. AMGH; 7ª edição.

FLACH, L.; MATTOS, L.K.; WILL, A.R. e ROSCHEL, L.F. (2017). Efficiency of expenditure on education and learning by Brazilian states: A study with Data Envelopment Analysis. Contabilidad y Negocios, v. 12, n. 23, pp. 111-128.

FREITAS, M.M.; FARIAS, R.A.S; FLACH, L. (2017). Efficiency determinants in Brazilian football clubs. Brazilian Business Review, v. 14(Special Ed), pp.1– 23 https://doi.org/10.15728/edicaoesp.2017.1

GUEORGUIEV, T., KOKALAROV, M. e SAKAKUSHEV, B. (2020). Recent trends in FMEA methodology. 7th International Conference on Energy Efficiency and Agricultural Engineering (EE&AE), Ruse, Bulgaria. DOI: 10.1109/EEAE49144.2020.9279101

HAHSLER, M., PIEKENBROCK, M. e DORAN, D. (2019). dbscan: Fast Density-Based Clustering with R. Journal of Statistical Software, v. 91, n. 1, pp. 1-30. https://doi.org/10.18637/jss.v091.i01

HAN, X. e ZHANG, J. (2013). A combined analysis method of FMEA and FTA for improving the safety analysis quality of safety-critical software. In: 2013 IEEE International Conference on Granular Computing (GrC), TBD, China, 2013 pp. 353-356. doi: 10.1109/GrC.2013.6740435

HAUPTMANNS, U.; WERNER, W. (2012). Engineering risks: Evaluation and valuation. Springer, Softcover reprint of the original 1st ed. 1991 edition

HUMPHREY, W. S. (1989) Managing the software process. Addison-Wesley Professional.

IATF. (2016). Norma IATF 16949. International Automotive Task Force.

IEC. (1985). Analysis techniques for system reliability - procedure for failure mode and effects analysis (FMEA). International Electrotechnical Commission.

IEC 60812. (2018). Failure modes and effects analysis (FMEA and FMECA). Disponível em: https://webstore.iec.ch/preview/info_iec60812%7Bed3.0%7Db.pdf. Acesso em 6 de novembro de 2023.

IEEE. (2023). The professional home for the engineering and technology community worldwide.

ISO 21434 (2021). Road Vehicles - Cybersecurity Engineering. Disponível em: https://www.iso.org/standard/70918.html. Acesso em 3 de setembro de 2023.

ISO 25010. (2011). *Systems and Software Engineering: Systems and Software Quality Requirements and Evaluation (SQuaRE).* 2011.

ISO 26262. (2018). *Road Vehicles - Functional Safety.*

ISO 31000. (2018). *Risk management guidelines.*

ISO 9001. (2015). Quality Management Systems. Disponível em: https://www.iso.org/standard/62085.html. Acesso em 5 de setembro de 2023.

ISO 9126. (1991). *Software Engineering - Product Quality.* 1991.

JIRA. (2023). Bug tracking done right with Jira Software. Atlassian, 2023 Disponível em: https://www.atlassian.com/software/jira/features/bug-tracking. Acesso em 5 de setembro de 2023.

JOMTHANACHAI, S.; WONG, W. P.; LIM, C. P. (2021). An Application of Data Envelopment Analysis and Machine Learning Approach to Risk Management, in *IEEE Access*, vol. 9, pp. 85978-85994. DOI: 10.1109/ACCESS.2021.3087623.

JULIUSSEN, E. (2022). Global EV Market Outlook: China, U.S., and Europe. EE Times publicado em 18 de julho de 2022. Disponível em: https://www.eetimes.com/global-ev-market-outlook-china-u-s-andeurope/?utm_source=eetimes_linkedin&utm_medium=social&utm_campaign=tuesdayarticl.

JURAN, J. M. (1951). Quality-Control Handbook

KAPUR, K. C.; PECHT, M. (2014). Reliability engineering. John Wiley & Sons. ISBN: 978-1-118-14067-3

KMENTA, S.; ISHII, K. (2000). Scenario-based FMEA: a life cycle cost perspective. *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference.* 2000, Vol. 35159, pp. 163-173.

MALLORY, W.C.; WALLER, R. (1973). Application of Selected Industrial Engineering Techniques to Wastewater Treatment Plants. United States Environmental Protection Agency, Washington.

MILLER, J.; POSADA, F. (2019). Brazil PROCONVE P-8 emission standards. ICCT - International Council on Clean Transportation - Policy Update. Disponível em: https://theicct.org/sites/default/files/publications/P8_emissions_Brazil_update_20190227.pdf. Acesso em 2 de dezembro de 2023.

MIL-P1629. (1949). USA military standard, procedure for performing a failure mode, effects, and analysis. United States Military Procedure, 1949.

MIL-STD-1629A NOTICE 3 (1998). Military Standard: procedures for performing a failure mode, effects, and criticality analysis. Disponível em: http://everyspec.com/MIL-STD/MIL-STD-1600-1699/MIL-STD-1629_NOTICE-3_23100. Acesso em 3 de novembro de 2023.

MINITAB. (2023). Suporte ao Minitab 20. *Etapa 1: Examinar os níveis de similaridade e de distância.* [Online] Minitab, Disponível em: https://support.minitab.com/pt-br/minitab/20/help-and-how-to/statistical-modeling/multivariate/how-to/cluster-observations/interpret-the-results/key-results/. Acesso em 2 de setembro de 2023.

MONTOYA-QUINTERO, D.M., LARREA-SERNA, O.L.; JIMÉNEZ-BUILES, J.A. (2022). Evaluation of the Efficiency of Regional Airports Using Data Envelopment Analysis. 2022, Vol. 9, p. 90.

MORDOR. (2023). Mercado de Big Data na Indústria Automotiva. Análise de Tamanho e Ações - Tendências e Previsões de Crescimento (2023-2028). Mordor Intelligence, 2023. Disponível em: https://www.mordorintelligence.com/pt/industry-reports/big-data-market-in-the-automotive-industry. Acesso em 2 de novembro de 2023.

MORGAN, M. G.; HENRION, M.; SMALL, M. (1990). Uncertainty: a guide to dealing with uncertainty in quantitative risk and policy analysis. Cambridge University Press; 6th edition.

MOTORSAFETY. (2022). *Stellantis recalls Jeep, RAM vehicles over fuel pump failure, stall risk*. Disponível em: https://www.motorsafety.org/stellantis-recalls-jeep-ram-vehicles-over-fuel-pump-failure-stall-risk. Acesso em 6 de setembro de 2023.

MPS.BR. 2003. *Melhoria do Processo de Software Brasileiro.* s.l. : Softex, 2003.

MÜLLNER, D. (2013). Fast Hierarchical Clustering Routines for R and Python. Disponível em https://cran.r-project.org/web/packages/fastcluster/fastcluster.pdf. Acesso em 2 de dezembro de 2023.

MURTAGH, F.; CONTRERAS, P. (2012). Algorithms for hierarchical clustering: an overview.Wires Data Mining and Knowledge Discovery, v.2, n. 1, pp. 86--97.

NHTSA. (2023). Safety Issues & Recalls. NHTSA, 2023. Disponível em https://www.nhtsa.gov/recalls. Acesso em 4 de novembro de 2023.

NIU, G.; JI, Y.; ZHANG, Z.; WANG, W.; CHEN, J. e YU, P. (2021). Clustering analysis of typical scenarios of island power supply system by using cohesive hierarchical clustering based K-Means clustering method. Energy Reports, v. 7, Supplement 6, Pages 250-256. https://doi.org/10.1016/j.egyr.2021.08.049

PUTHANPURA,A.K. (2022). Multiplier DEA. Repositório CRAN 2022. Disponível em: https://cran.r-project.org/web/packages/MultiplierDEA/index.html. Acesso em 3 de novembro de 2023.

VLASIC,B.; APUZZO,M. (2014). Toyota Is Fined $1.2 Billion for Concealing Safety Defects. Disponível em https://www.nytimes.com/2014/03/20/business/toyota-reaches-1-2-billion-settlement-in-criminal-inquiry.html. Acesso em 10 de novembro de 2023.

OAKLAND, J. S. (1994). Gerenciamento da qualidade total. Editora Nobel.

PALIOTTA, J. (2015). The quality of your code is the quality of your brand-And it's time to pay attention to software testing. IEEE AUTOTESTCON, pp. 186–193. https://doi.org/10.1109/AUTEST.2015.7356487

PANCIK, J.; VÉMOLA, A.; KLEDUS, R.; SEMELA, M. AND BRADÁČ, A. (2018). Auto recalls and software quality in the automotive sector. EAI Endorsed Transactions on Scalable Information Systems, v. 5, n. 17, e3. https://eudl.eu/pdf/10.4108/eai.29-5-2018.154808

PASUPATHI, S.; SHANMUGANATHAN, V.; MADASAMY, K.; YESUDHAS, H. R. & KIM, M. (2021). Trend analysis using agglomerative hierarchical clustering approach for Big Data. The Journal of Supercomputing, v. 77, pp. 6505–6524

PEÑA, C.R. (2008). A model of evaluation of the efficiency of the public sector through the method data envelopment analysis (DEA). Revista de Administração Contemporânea, v. 12, pp. 83--106.

PIECH,C. (2013). K-Means. Stanford CS221. Disponível em https://stanford.edu/~cpiech/cs221/handouts/kmeans.html. Acesso em 2 de novembro de 2023.

PMBOK. (2021). Guia do conhecimento em gerenciamento de projetos e o padrão de gerenciamento de projetos. PMI - PROJECT MANAGEMENT INSTITUTE, 7ª Edição. Disponível em: https://www.pharmabooks.com.br/livros/gestao-de-projetos/guia-pmbok-7-guia-do-conhecimento-em-gerenciamento-de-projetos-e-o-padrao-de-gerenciamento-de-projetos-7a-edicao-2021-pmi-project-management-institute. Acesso em 7 de novembro de 2023.

POSIT. (2022). Posit Cloud Guide. Disponível em https://posit.cloud/learn/guide. Acesso em 6 de setembro de 2023.

QIAN, H. H.T.; LIU, Z. J.; XU, Y. B. (2017). Systematic maintenance and applications of Failure Modes and Effects Analysis (FMEA) in semiconductor manufacturing. CSTIC - China Semiconductor Technology International Conference. DOI: 10.1109/CSTIC.2017.7919853

QUALYTEAM. (2021). Norma IATF 16949: O que é, interações e os Core Tools. Disponível em: https://qualyteam.com/pb/blog/iatf-16949-o-que-e-interacao-da-norma-e-os-core-tools/. Acesso em 7 de setembro de 2023.

SAE (1967). Design analysis procedure for failure mode, effects and criticality analysis (FMECA). Disponível em: https://www.sae.org/standards/content/arp926/. Acesso em 4 de setembro de 2023.

SAE (2021). Potential Failure Mode and Effects Analysis (FMEA) Including Design FMEA, Supplemental FMEA-MSR, and Process FMEA. Disponível em https://www.sae.org/standards/content/j1739_202101. Acesso em 3 de dezembro de 2023.

SCIKIT-LEARN. (2023). Scikit-learn: Machine Learning in Python, JMLR, v. 12, pp. 2825-2830.

SINAGA, K.P.; YANG, M.S. (2020). Unsupervised K-means clustering algorithm. IEEE Access, Vol. 8, pp. 80716-80727. DOI: 10.1109/ACCESS.2020.2988796

SINNAMON, R. M.; ANDREWS, J. D. (1997). Improved efficiency in qualitative fault tree analysis. Wiley Online Library, v. 13, n. 5, pp. 293-298. 5.

SOLVER (2023). Frontline Systems, 2023. Disponível em https://www.solver.com/excel-solver-online-help. Acesso em 6 de dezembro de 2023.

SOMMERVILLE, I. (2015). Software Engineering. Pearson; 10th Revised ed. ISBN-13 : 978-0133943030

SOTERIADES, A.D. (2022). Additive Data Envelopment Analysis Models. Disponível em: https://cran.r-project.org/web/packages/additiveDEA/index.html. Acesso em 10 de dezembro de 2023.

STAMATIS, Diomidis H. (2003). *Failure mode and effect analysis.* s.l. : Quality Press, 2003.

TAVARES, B. G.; DA SILVA, C. E. S. e DE SOUZA, A. D. (2019). Risk management analysis in Scrum software projects. Wiley Online Library, v. 26, n. 5, pp. 1884–1905.

TEXEIRA, M.C.; FLACH, L. e MATTOS, L.K. (2020). Analysis of expenses quality of credit cooperatives of free admission of associates. Disponível no SSRN Electronic Journal: http://dx.doi.org/10.2139/ssrn.3567853

TINGA, T. (2013). Principles of Loads and Failure Mechanisms. Springer-Verlag London.

TOLOO, M.; BARAT, M.; MASOUMZADEH, A. (2015). Selective measures in data envelopment analysis. Annals of Operations Research, v. 226, pp. 623--642.

VBA. (2023). Introdução ao VBA no Office. Disponível em https://learn.microsoft.com/pt-br/office/vba/library-reference/concepts/getting-started-with-vba-in-office. Acesso em 3 de dezembro de 2023.

WANG, Z.; FENG, C. (2015). Sources of production inefficiency and productivity growth in China: A global data envelopment analysis. Energy Economics, v. 49, May 2015, pp. 380-389

WICKHAM, H.; BRYAN, J.; KALICINSKI, M.; VALERY, K.; LEITIENNE, C.; COLBERT, B.; HOERL, D. E MILLER, E. (2023). Read Excel Files. Repositório CRAN. Disponível em: https://cran.r-project.org/web/packages/readxl/readxl.pdf. Acesso em 2 de setembro de 2023.

WIRED. (2015). Hackers Remotely Kill a Jeep on the Highway—With Me in It. Disponível em: https://www.wired.com/2015/07/hackers-remotely-kill-jeephighway/. Acesso em 1 de novembro de 2023.

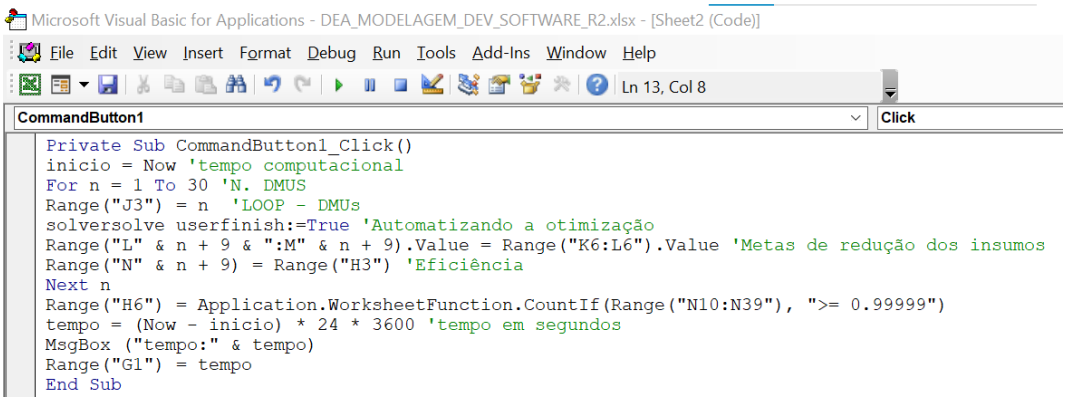
WILEY. (2012). Giants of Quality—W. Edwards Deming. Giants of Quality—W. Edwards Deming. 2012.

YI, H. e MA, H. (2019). Optimization of Municipal Solid Waste Logistics System based on Data Envelopment Analysis. Ekoloji Dergisi, n. 107, pp. 2643-2654

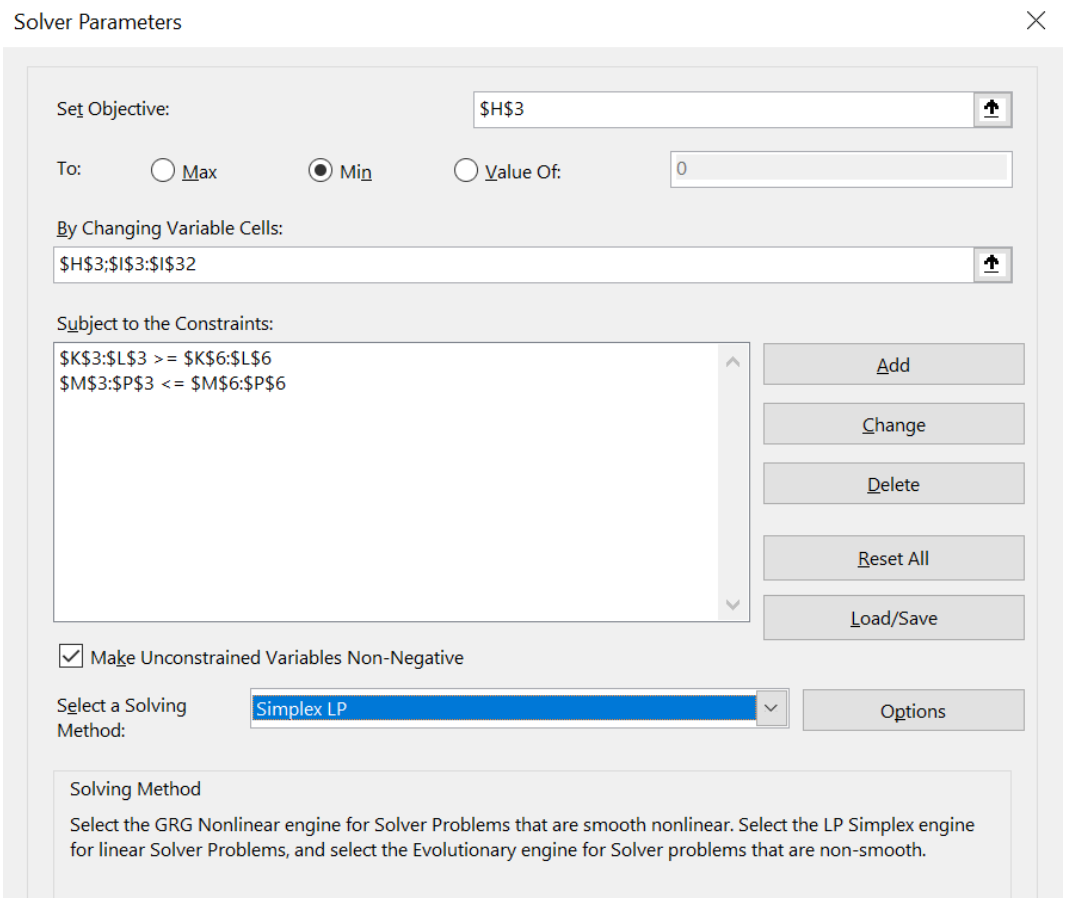
YOUSEFI, S.; ALIZADEH, A.; HAYATI, J. e BAGHERY, M. (2017). HSE risk prioritization using robust DEA-FMEA approach with undesirable outputs: a study of automotive parts industry in Iran. Safety Science, v. 102, February 2018, Pages 144-158. https://doi.org/10.1016/j.ssci.2017.10.015

APÊNDICE A

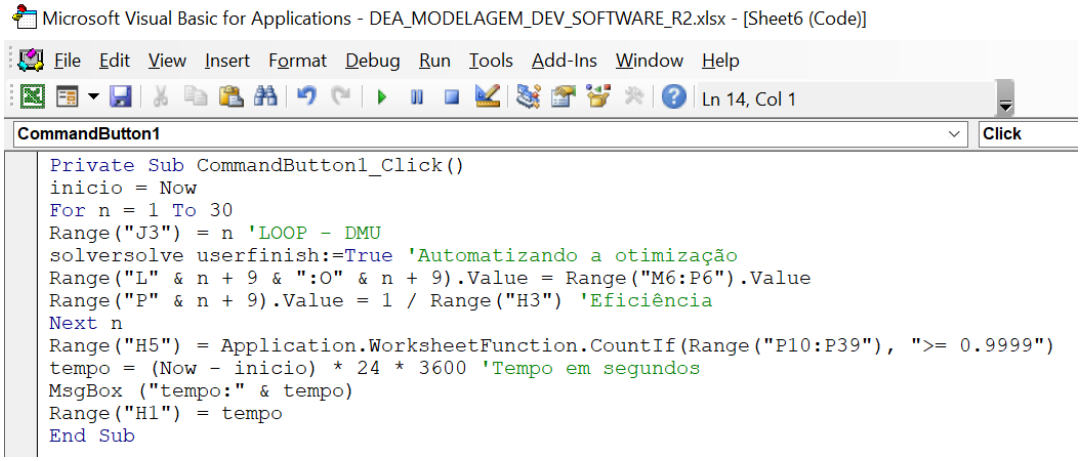
Código em VBA para geração das eficiências das DMUs e metas de redução dos insumos



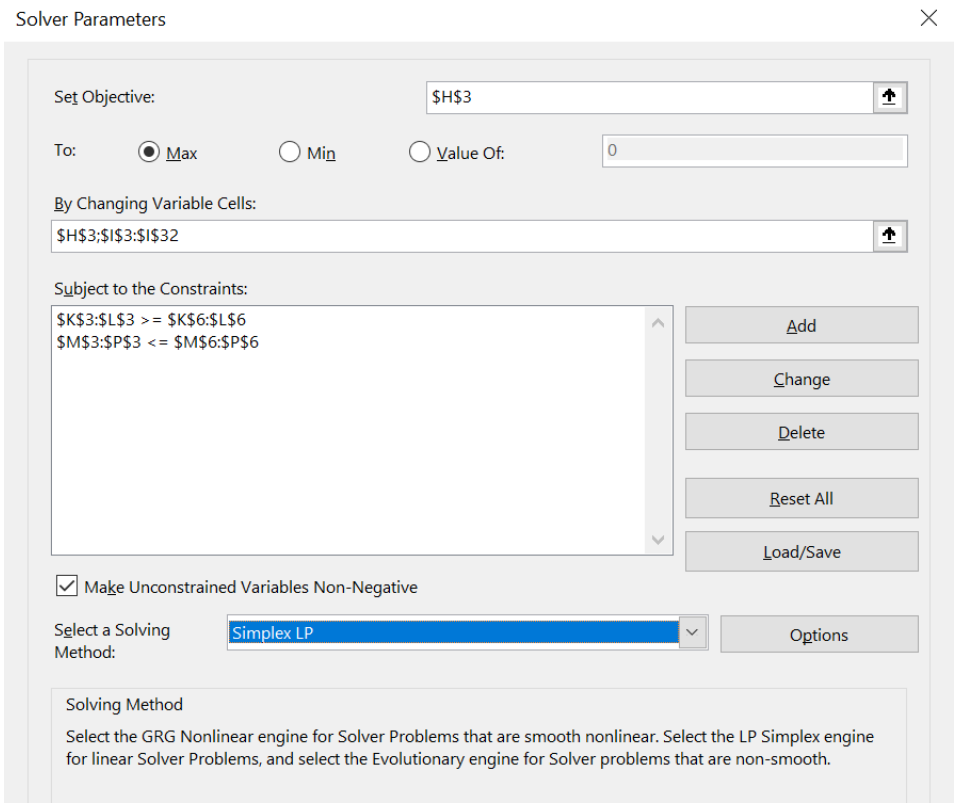
Configuração dos parâmetros de restrições no *Solver* e seleção do método *Simplex LP (Modelo CRS com orientação a entrada)*



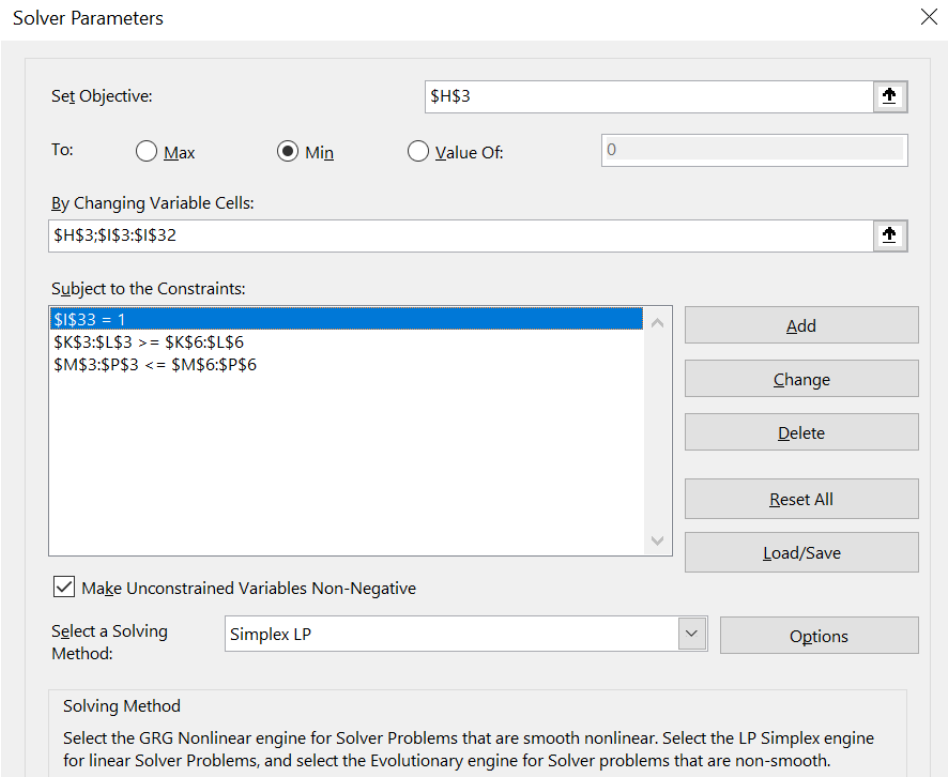
Código em VBA para geração das eficiências das DMUs e metas de aumento dos produtos



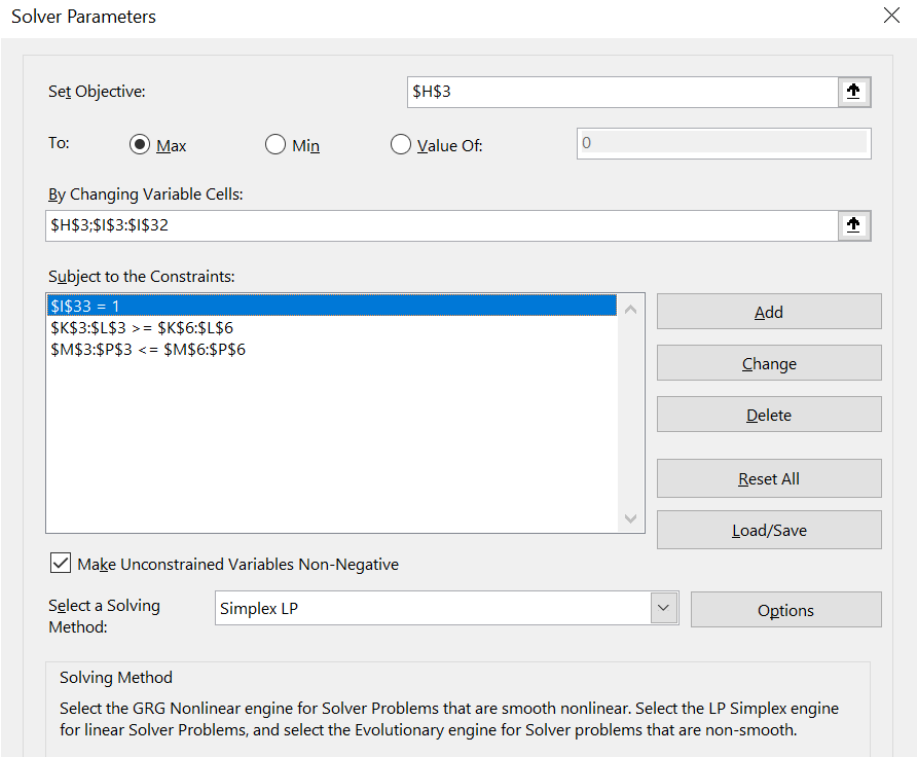
Configuração dos parâmetros de restrições no *Solver* e seleção do método *Simplex LP (Modelo CRS com orientação a saída)*



Configuração dos parâmetros de restrições no *Solver* e seleção do método *Simplex LP (Modelo VRS com orientação a entrada)*

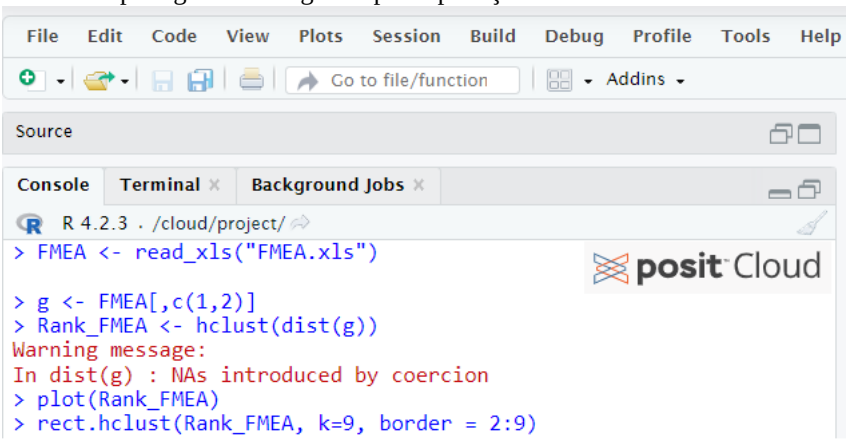


Configuração dos parâmetros de restrições no *Solver* e seleção do método *Simplex LP (Modelo VRS com orientação a saída)*

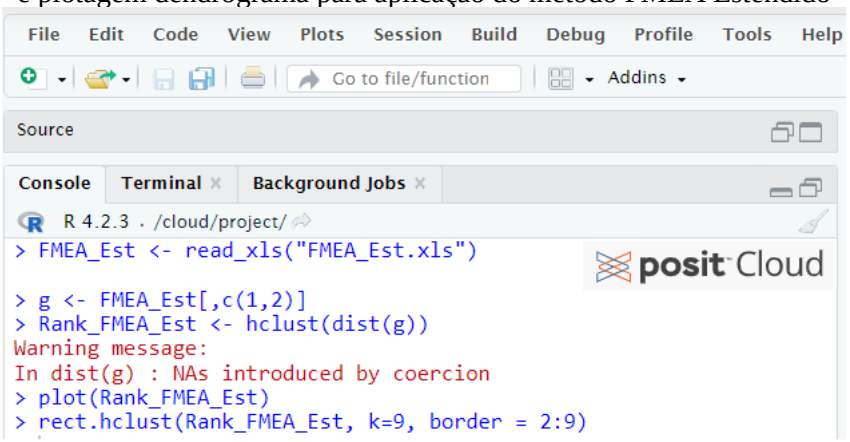


APÊNDICE B

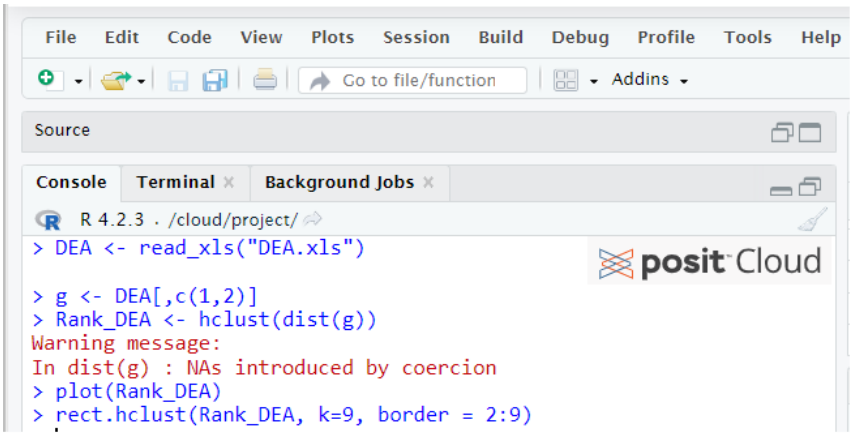
Código em linguagem R com implementação de agrupamento hierárquico aglomerativo e plotagem dendrograma para aplicação do método FMEA



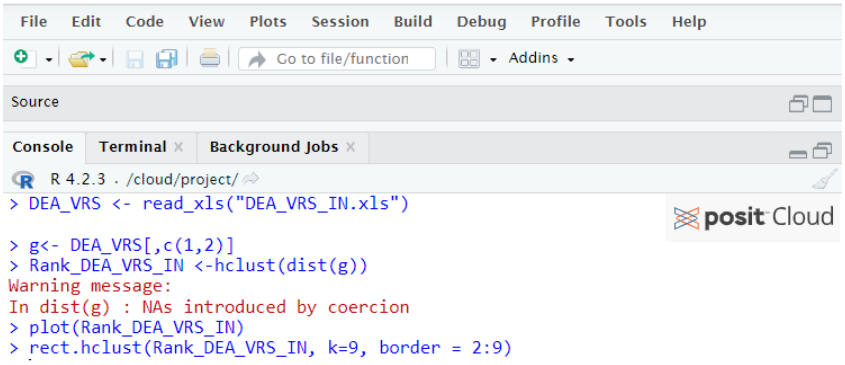
Código em linguagem R com implementação de agrupamento hierárquico aglomerativo e plotagem dendrograma para aplicação do método FMEA Estendido



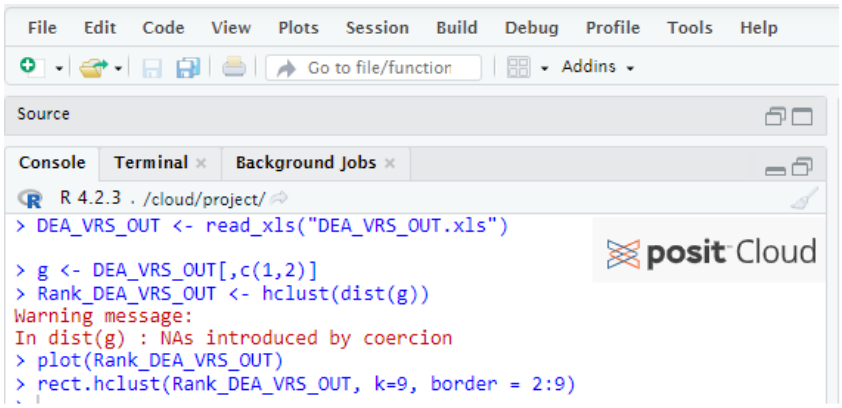
Código em linguagem R com implementação de agrupamento hierárquico aglomerativo e plotagem dendrograma para aplicação do método DEA (CRS)



Código em linguagem R com implementação de agrupamento hierárquico aglomerativo e plotagem dendrograma para aplicação do método DEA (VRS) orientado a entrada

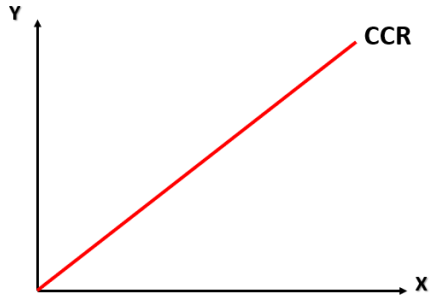


Código em linguagem R com implementação de agrupamento hierárquico aglomerativo e plotagem dendrograma para aplicação do método DEA (VRS) orientado a saída

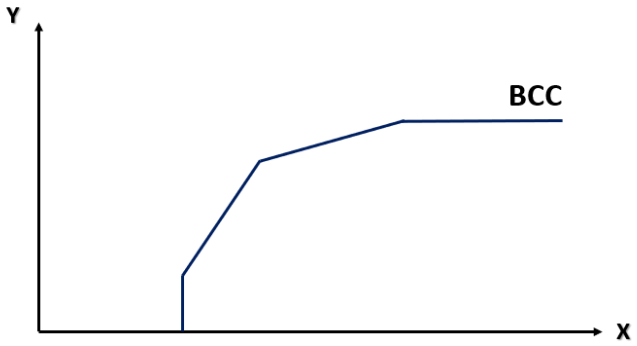


APÊNDICE C

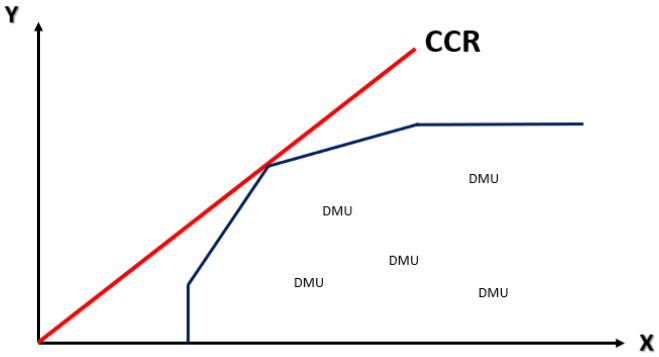
Modelo DEA CCR – Retornos Constantes de Escala (CRS)



Modelo DEA BCC – Retornos Variáveis de Escala (VRS)



Comparativo entre modelo DEA CCR e BCC



APÊNDICE D

Código em linguagem R com implementação da eficiência por DEA orientado a entrada (CRS)

```
R 4.2.3 · /cloud/project/
> library(Benchmarking)
> library(psych)
> ccr <- dea(x,y, RTS="crs", ORIENTATION =
"in")
> eff(ccr)
 [1] 0.1867689 0.2047875 0.7937491 1.00000
00 1.0000000 1.0000000 0.8657529 0.2638422
 [9] 0.3921183 1.0000000 0.4532868 0.34354
99 1.0000000 0.9666907 0.7394532 0.3694310
[17] 1.0000000 0.3870997 1.0000000 0.16926
94 0.4676005 1.0000000 0.5657945 0.5824916
[25] 0.3031431 0.8233185 0.4812993 1.00000
00 0.3156206 0.6581739
> data.frame(ccr$eff)
     ccr.eff
1  0.1867689
2  0.2047875
3  0.7937491
4  1.0000000
5  1.0000000
6  1.0000000
7  0.8657529
8  0.2638422
9  0.3921183
10 1.0000000
11 0.4532868
12 0.3435499
13 1.0000000
14 0.9666907
15 0.7394532
16 0.3694310
17 1.0000000
18 0.3870997
19 1.0000000
20 0.1692694
21 0.4676005
22 1.0000000
23 0.5657945
24 0.5824916
25 0.3031431
26 0.8233185
27 0.4812993
28 1.0000000
29 0.3156206
30 0.6581739
```