

# Investigating requirements change requests

Eduardo dos Santos Gonçalves  
edusantos@edu.unirio.br  
UNIRIO  
Rio de Janeiro, Brazil

Pablo Oliveira Antonino  
pablo.antonino@iese.fraunhofer.de  
Fraunhofer IESE  
Germany

Paulo Malcher  
malcher@edu.unirio.br  
UNIRIO & UFRA  
Rio de Janeiro, Brazil

Rodrigo Pereira dos Santos  
rps@uniriotec.br  
UNIRIO  
Rio de Janeiro, Brazil

## Abstract

**Context:** Software development is a highly dynamic, collaborative, and constantly changing process. Hence, the efficient management of requirements change requests (RCR) is essential to avoid negative impacts such as delays and quality compromise. **Problem:** The management of RCR in software development demands an understanding of associated motivator factors, solutions, and challenges. **Solution:** In this study, we aim to identify factors, solutions, and challenges in RCR management. **IS Theory:** This study is motivated by socio-technical theory, which addresses the technical and social aspects of requirements management. It highlights the interplay between stakeholder collaboration and technical processes in effectively managing RCR. **Method:** Employing the rapid review of the literature (RR) method, the research analyzes 37 selected studies from a traditional scientific database in the field of computing for this purpose—out of 1,061 retrieved results. **Summary Results:** The research reveals 19 factors (e.g., change in users/clients needs), 35 solutions (e.g., COSMIC Functional Size Measurement), and 12 challenges (e.g., time and resource consumption), providing a comprehensive overview of the RCR management landscape in software development. **Contributions to IS:** This study contributes to understanding factors, solutions, and challenges for managing RCR in software development. For academia, it contributes to disseminating knowledge about RCR, which contributes to advancing the state of the art of requirements change management. For the industry, it assists professionals in identifying, adopting, and implementing effective strategies to face the challenges in the area.

## CCS Concepts

• **Social and professional topics** → **Software management**; • **Software and its engineering** → **Software development process management**.

## Keywords

Requirement change request, Change management, Requirements management, Rapid Review

## 1 Introdução

As mudanças de requisitos são inevitáveis no desenvolvimento de software [1, 44]. Elas podem surgir em qualquer estágio do processo de desenvolvimento de software devido a diferentes fatores internos e externos, como por exemplo, necessidades do cliente, mudanças tecnológicas, de mercado, de orçamento e competição global [29, 33].

Nesse sentido, as mudanças dinâmicas de requisitos estão entre as principais dificuldades que as empresas de software enfrentam [49]. Elas impactam naturalmente o custo, a qualidade e o cronograma do projeto, criando trabalho adicional em *design* e codificação [1, 67].

De acordo com Rahman et al. [57], mudanças nos requisitos do sistema têm impactos negativos e positivos em projetos de desenvolvimento de sistemas [57]. Segundo os autores, as mudanças terão um impacto negativo se os processos não forem gerenciados adequadamente. Para Ali and Lai [6], o efeito das mudanças de requisitos é positivo para um projeto se as mudanças forem abordadas adequadamente no momento certo, caso contrário, elas podem resultar em falha do projeto. Neste sentido, projetos de software estão cada vez mais complexos e os profissionais encontram vários desafios para lidar com mudanças de requisitos, como a distribuição geográfica e o relacionamento entre múltiplos atores [21]. Portanto, não é apenas importante, mas crucial para o sucesso do produto final entender e gerenciar as mudanças de requisitos durante o desenvolvimento de software [29, 33, 45, 49].

O processo de gerência de mudanças de requisitos começa quando uma parte interessada envia uma solicitação de mudança de requisitos (SMR), na qual é detalhada a mudança necessária para o sistema [50]. SMR normalmente são expressas em formato de linguagem natural e são difíceis de analisar, pois podem afetar diferentes tipos de requisitos [37]. Gerenciar essas SMR apresenta vários desafios, principalmente devido à natureza frequente e às vezes imprevisível de tais solicitações [55]. Um desses desafios é o grande volume de SMR que podem surgir, especialmente quando os produtos atingem a maturidade ou entram em estágios de manutenção. De acordo com Kumar et al. [40], quanto mais tempo um produto estiver em desenvolvimento ou manutenção, maior será a probabilidade de encontrar inúmeras SMR. Além disso, as mudanças de requisitos podem colocar imensa pressão sobre recursos, cronogramas e gerenciamento de projetos [40]. Portanto, sem uma abordagem sistemática, as SMR podem interromper os cronogramas de desenvolvimento, atrasar as entregas e aumentar os custos [55].

Diante desse cenário, este estudo tem como objetivo identificar o que se sabe sobre o gerência de SMR no contexto de desenvolvimento de software. Para atingir esse objetivo, foi realizada uma revisão da literatura existente para mapear e analisar o estado da arte da gerência de SMR. A abordagem utilizada foi a *rapid review* (RR), que é um tipo de estudo secundário destinado a identificar evidências que auxiliem na resolução de problemas práticos [13].

Como resultados, foram identificados 19 fatores que motivam as SMR e 35 soluções utilizadas para gerenciar SMR, que foram

classificadas em modelos, métodos, *frameworks*, ferramentas, teorias e práticas. Além disso, foram identificados 12 desafios relacionados à gerência de SMR no contexto de desenvolvimento de sistemas. Nesse sentido, os resultados deste estudo podem ampliar a compreensão acerca dos fatores motivadores, soluções e desafios relacionados à gerência de SMR. Assim, este estudo busca auxiliar profissionais e pesquisadores na identificação, adoção e implementação de estratégias eficazes para enfrentar os desafios da área.

O restante deste artigo está organizado da seguinte forma: a Seção 2 apresenta a fundamentação teórica; a Seção 3 apresenta os trabalhos relacionados; a Seção 4 descreve a método de pesquisa; a Seção 5 apresenta os resultados; a Seção 6 discute as contribuições do estudo e apresenta as ameaças à validade; e, por fim, a Seção 7 conclui o artigo com considerações finais e trabalhos futuros.

## 2 Fundamentação Teórica

### 2.1 Engenharia de requisitos

Engenharia de requisitos (ER) é uma disciplina interdisciplinar que atua como mediadora entre as partes interessadas, visando estabelecer e manter os requisitos que devem ser cumpridos pelo sistema, software ou serviço de interesse [27]. Ela é um processo composto por atividades definidas e executadas sequencialmente, visando identificar os requisitos de um projeto de desenvolvimento de software [22]. Para Pohl [54], a ER é fundamental para desenvolver software que atenda aos clientes dentro de restrições orçamentárias e de tempo do projeto de desenvolvimento de software. Assim, seu objetivo é documentar os requisitos do cliente da forma mais completa possível com boa qualidade e identificar e resolver problemas nos requisitos o mais cedo possível [54].

A ER é composta por dois processos principais: o desenvolvimento de requisitos e a gerência de requisitos. O desenvolvimento de requisitos é realizado de forma iterativa, com o objetivo de criar o documento de requisitos [71]. O documento de requisitos serve como um referencial para todas as alterações nos requisitos que podem ocorrer devido a diversas razões [25]. A gerência de requisitos abrange as atividades de gerenciamento na ER [71].

### 2.2 Gerência de Requisitos

A gerência de requisitos é um processo que acompanha o planejamento e o desenvolvimento de um sistema, capturando e mapeando a origem e o contexto das mudanças [69]. Ela fornece visibilidade para o processo de desenvolvimento do software, o que é uma garantia para o sucesso do software [73]. Xu et al. [73] identificou três atividades importantes da gerência de requisitos: manutenção da linha de base de requisitos, rastreabilidade de requisitos e gerência de mudanças de requisitos. A identificação de mudanças de requisitos, que é uma atividade da gerência de mudanças de requisitos, é o foco deste trabalho e será detalhada a seguir.

A mudança é uma característica intrínseca de sistemas de informação em comparação com outros sistemas e um aspecto desafiador do processo de ER [3, 29]. Jayatilleke and Lai [29] afirmaram que é previsto que os requisitos mudarão durante o ciclo de vida de um projeto, e tal volatilidade constitui um dos dez principais riscos para o desenvolvimento bem-sucedido de projetos de software. Uma mudança de requisitos é uma adição/modificação/exclusão/correção

de *bug* ou combinação destes em termos de requisitos funcionais e não funcionais apresentados em qualquer formato [43].

A gerência de mudanças de requisitos é um processo orientado à colaboração que requer comunicação e coordenação entre várias partes interessadas para ser bem-sucedido [5]. Jayatilleke and Lai [29] identificaram três áreas da gerência de mudanças de requisitos: identificação de mudanças, análise de mudanças e estimativa de custos de mudanças. A identificação de mudanças de requisitos é uma área que indica por que, por quem e quando uma mudança de requisitos é necessária [29, 34]. As outras áreas da gerência de mudanças de requisitos não são autônomas e serão baseadas na identificação correta de mudanças de requisitos [29]. De acordo com Jayatilleke and Lai [29], há duas atividades principais na identificação de mudanças de requisitos: elicitación de mudanças de requisitos e representação de mudanças de requisitos por meio de SMR.

### 2.3 Solicitação de Mudanças de Requisitos

Mudanças de requisitos são geralmente representadas por meio de SMR, que são especificações de requisitos que definem as mudanças solicitadas de forma completa e sistemática [57]. De acordo com Rahman et al. [57], lidar com SMR é um processo intensivo de decisão, pois envolve custos e recursos. SMR abordam não apenas requisitos novos ou alterados, mas também correções de *bugs* e defeitos. Elas são revisadas para determinar o impacto que a mudança terá nos artefatos relacionados (e.g., código-fonte, modelos e diagramas), bem como no orçamento e cronograma. Essa verificação é necessária porque nem todas as SMR exigem ação. Por exemplo, um *bug* relatado pode já ter sido corrigido ou novos recursos já estão planejados para serem incorporados ao sistema. Nessas situações, a SMR é fechada com o motivo do seu fechamento [50].

SMR aparecem durante todas as partes do desenvolvimento e manutenção do sistema [42]. Elas se originam de mudanças na demanda das partes interessadas, mercado, ferramentas e tecnologia [18, 48]. SMR precisam ser tratadas de forma eficiente para reduzir o custo do software e melhorar a qualidade [72]. Elas são geralmente armazenadas em repositórios para acesso futuro e manutenção de registros. Como mais de 85% dos requisitos de software aparecem na forma de linguagem natural [59], cada SMR é descrita com uma longa elaboração textual chamada “*descrição*” e um assunto denominado “*título*”. O assunto dessas SMR abrange uma ampla variedade de aspectos de desenvolvimento, por exemplo, relatórios de *bugs*, melhorias, solicitações de recursos e problemas de compatibilidade.

## 3 Trabalhos Relacionados

Jayatilleke and Lai [29] conduziram uma revisão sistemática da literatura (RSL) sobre gerência de mudanças de requisitos que forneceu informações sobre as técnicas e práticas de última geração para gerência de mudanças de requisitos e as lacunas de pesquisa no trabalho existente. Em um contexto mais específico, Khan and Akbar [35] realizaram uma RSL sobre desafios da gerência de mudanças de requisitos no contexto de desenvolvimento global de software. Uma taxonomia que fornece uma estrutura para enfrentar os desafios da gerência de mudanças de requisitos foi proposta por Khan and Akbar [35]. Kausar et al. [33] também conduziram uma RSL sobre os desafios da gerência de mudanças de requisitos no desenvolvimento global de software. Foram identificados 12 desafios que, de acordo

com os autores, auxiliaram o entendimento das práticas da gerência de mudanças de requisitos no desenvolvimento global de software. Este trabalho, diferente dos anteriores, visa identificar fatores motivadores, soluções e desafios relacionados especificamente às SMR em software, sem limitação de contexto específico.

## 4 Método de Pesquisa

Para este estudo, foi conduzida uma RR durante os meses de outubro e novembro de 2024. RR são estudos secundários orientados para a prática, cujo principal objetivo é fornecer evidências para apoiar a tomada de decisões visando a solução, ou pelo menos a atenuação, dos problemas enfrentados pelos profissionais na prática [13].

Para identificar o problema prático deste estudo, um profissional da indústria de software com mais de 15 anos de experiência foi entrevistado. O profissional, que trabalha em um instituto de pesquisa orientado para aplicações na Europa, ocupa atualmente o cargo de chefe de departamento, gerenciando uma equipe que se encontra globalmente distribuída. Durante a entrevista, o profissional relatou suas experiências, limitações e dificuldades em relação a gerência de SMR. Inicialmente, foi identificada a dificuldade de entender como se origina uma SMR. O profissional mencionou que: *“Eu tenho dificuldade de ver onde as pontas estão amarradas na origem de uma solicitação”*. Também foi identificada a necessidade de investigar quais soluções voltadas à gerência de SMR estão disponíveis e de como essas soluções tem sido avaliadas. Nesse contexto, o profissional mencionou: *“Gostaria de ver as solicitações com diferentes graus de detalhes. Caso eu queira ver isso tenho que ir de um a um projeto para verificar. Eu como gestor tenho que confiar no que é falado pelos meus sêniores”*. Por fim, foi identificada a necessidade de entender quais os desafios na gerência das SMR.

Portanto, a demanda por uma RR surgiu a partir do seguinte problema prático: **a dificuldade em lidar com as SMR que emergem de diferentes canais de comunicação utilizados pelo time e as diferentes partes interessadas do projeto**. Assim, a dificuldade em lidar com as SMR resultam na necessidade de explorar na literatura os fatores motivadores, soluções propostas e seus desafios. Nesse contexto, a RR se mostrou mais adequada para abordar os problemas práticos enfrentados pelos profissionais da área, pois permite uma conexão mais direta com suas necessidades.

Uma RR se caracteriza como um método de pesquisa realizado em prazos mais curtos quando comparado às RSL, com algumas etapas da RSL sendo omitidas ou simplificadas [12, 13]. Para conduzir esta RR, foi seguido o modelo de protocolo proposto por Cartaxo et al. [13]. Além disso, foram consideradas as diretrizes para a condução de RSL indicadas por Kitchenham and Charters [39].

### 4.1 Questões de Pesquisa

Para atingir o objetivo da pesquisa, foram formuladas quatro questões de pesquisa (QP) em estreita colaboração com o profissional da indústria envolvido na RR. As QP estão detalhadas na Tabela 1.

### 4.2 Estratégia de Busca

Para identificar os estudos primários que abordam às QP descritas na Tabela 1, foi utilizada a base de dados da *Scopus*<sup>1</sup>, um indexador

<sup>1</sup><https://www.scopus.com/>

Tabela 1: Questões de pesquisa

ID	Questão de Pesquisa	Meta
QP1	Quais são os fatores que motivam as SMR?	Essa QP visa identificar os fatores que motivam a formalização de uma SMR.
QP2	Quais são as soluções utilizadas para gerenciar SMR?	Essa QP busca examinar e documentar os métodos, técnicas, ferramentas e práticas predominantes no gerenciamento de SMR.
QP3	Como essas soluções têm sido avaliadas?	Essa QP visa identificar como essas soluções são avaliadas e quais os critérios utilizados para determinar sua eficácia no gerenciamento de SMR.
QP4	Quais são os desafios na gerência de SMR?	Essa QP busca destacar os principais desafios no gerenciamento de SMR e as lacunas existentes na literatura.

de várias bibliotecas digitais [12]. Para conduzir as buscas automatizadas, foi definida uma *string* de busca com base nos principais termos utilizados nas QP. A elaboração e teste de diferentes versões da *string* foram realizadas na base de dados selecionada. A *string* de busca definida também foi baseada em termos utilizados em trabalhos relacionados a SMR e mudanças de requisitos [5, 29]. Por fim, a *string* foi revisada pelo profissional da indústria e por um pesquisador experiente em engenharia de software experimental. A seguinte *string* de busca foi utilizada neste estudo: *(“request” OR “demand” OR “proposal” OR “identification” OR “solicitation” OR “requisition” OR “submission”) AND (“requirements change” OR “requirements modification” OR “requirements volatility”) AND (“software”*.

### 4.3 Critérios de Inclusão e Exclusão

Para a seleção dos estudos, foram definidos e aplicados critérios de inclusão (CI) e exclusão (CE), conforme listado abaixo na Tabela 2. Os estudos foram incluídos se atendiam a todos os CI e foram excluídos caso se enquadrassem em pelo menos um dos CE.

Tabela 2: Critérios de seleção

	ID	Descrição
Critério de Inclusão	CI1	Estudo responde pelo menos uma QP.
	CE1	Estudo secundário (e.g., mapeamento sistemático da literatura e revisão sistemática da literatura).
	CE2	Estudo duplicado.
Critérios de Exclusão	CE3	Estudo que não esteja disponível para download gratuito ou por meio de acesso institucional.
	CE4	Estudo que não esteja escrito em inglês.
	CE5	Estudo que seja prefácio, livro, capítulo de livro, resumo, pôster, painel, palestra, keynote, tutorial, editorial ou demonstração.
	CE6	Estudo não atende nenhum dos CI.

### 4.4 Processo de Seleção dos Estudos

Este estudo contou com o processo de seleção em seis etapas: (1) Execução da busca automatizada dos estudos primários (aplicação da *string* de busca); (2) 1º Filtro: Remoção de estudos duplicados (aplicação do CE2); (3) Aplicação do 2º Filtro: leitura do título, resumo e palavras-chave; (4) Aplicação do 3º Filtro: leitura da introdução e conclusão; (5) Aplicação do 4º Filtro: leitura completa do estudo; e (6) Extração de dados e síntese.

### 4.5 Extração de Dados e Síntese

A extração dos dados foi conduzida por dois pesquisadores, seguida por uma verificação dos resultados por dois pesquisadores mais experientes em engenharia de software. A extração dos dados foi realizada de maneira sistemática com o auxílio da ferramenta Parsifal<sup>2</sup> e com apoio da ferramenta Microsoft Excel para registrar as informações necessárias. Para responder às QP, foi utilizado um formulário contendo os seguintes campos: (1) identificador do estudo (ID); (2) título; (3) autores; (4) ano; (5) local de publicação; (6) país do primeiro autor; (7) fator/motivo que leva a uma SMR; (8) tipo de solução voltada ao gerenciamento de SMR; (9) tipo de avaliação realizado na solução; e (10) desafio no gerenciamento de SMR.

Para responder às QP, utilizou-se a síntese de dados. Considerando as diretrizes fornecidas por Cartaxo et al. [13] e King et al. [38], uma RR geralmente emprega um método de síntese descritiva (qualitativo) em vez de uma meta-análise quantitativa. Assim, neste estudo se utilizou o método qualitativo para analisar os dados extraídos. Inicialmente, foi adotada a codificação aberta, inspirada no procedimento inicial da abordagem *Grounded Theory* [15]. Nesse sentido, os dados extraídos dos estudos selecionados foram codificados. Para esta análise, os estudos selecionados foram lidos detalhadamente, segmentando o texto em **unidades coerentes**, como frases ou parágrafos. Posteriormente, foram atribuídos **códigos** representativos aos pontos-chave relevantes para responder às QP. Seguindo a codificação aberta, foi aplicada a codificação axial, conforme a abordagem de Charmaz [15], para organizar os códigos em **categorias**. A Tabela 3 mostra exemplos do processo de codificação aplicado aos estudos selecionados, incluindo os códigos resultantes e suas respectivas categorias. Os detalhes sobre a extração e a síntese dos dados podem ser encontrados na Seção 5.

### 5 Resultados

A RR realizada teve como enfoque identificar o máximo possível de estudos pertinentes ao tópico de pesquisa. Portanto, não foi aplicado nenhum filtro quanto ao ano de publicação, significando que todos os estudos disponíveis até outubro de 2024 foram incluídos. A Figura 1 mostra o número de estudos restantes em cada etapa do processo de seleção da RR. Ao final do processo de seleção, foram selecionados 37 estudos para extração e síntese dos dados. Os dados brutos e todas as etapas necessárias para reproduzir a pesquisa estão detalhados no material suplementar disponível abertamente<sup>3</sup>.

A Tabela 4 apresenta os 37 estudos em ordem decrescente por ano de publicação, com identificadores numéricos (E1-E37) usados para referenciar os estudos nas próximas seções. A Figura 2 mostra a distribuição dos estudos selecionados ao longo dos anos. Nota-se que, pelo menos uma publicação anual foi realizada de 2017 até 2023, atingindo o pico em 2019 com 6 estudos, conforme apresentado na 2. Também pode-se observar que o tema que já vem sendo pesquisado desde o século passado, com 2 estudos identificados entre os anos de 1997 e 1999. A Figura 3 mostra a distribuição dos países dos estudos selecionados, considerando apenas o país do primeiro autor. Paquistão e Tunísia destacaram-se com o maior número de publicações (5 estudos cada).

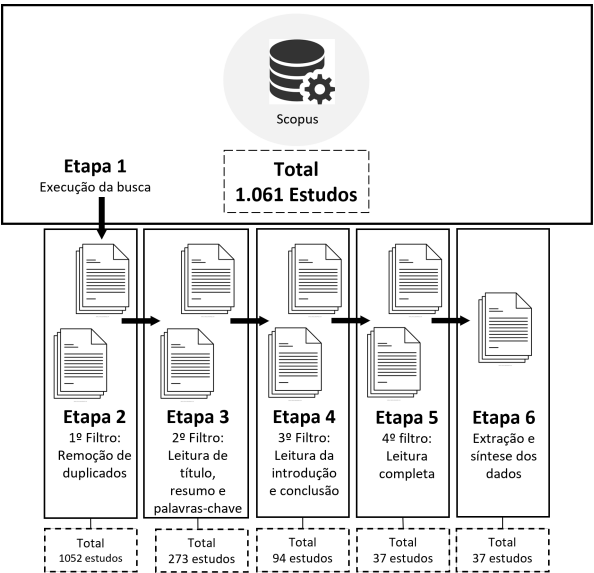


Figura 1: Resultados do processo de seleção dos estudos

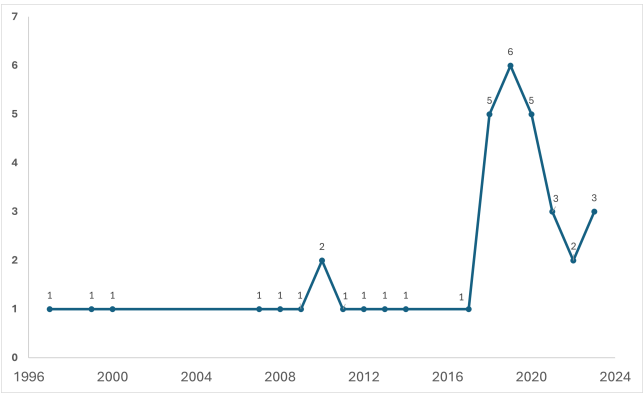


Figura 2: Número de estudos por ano

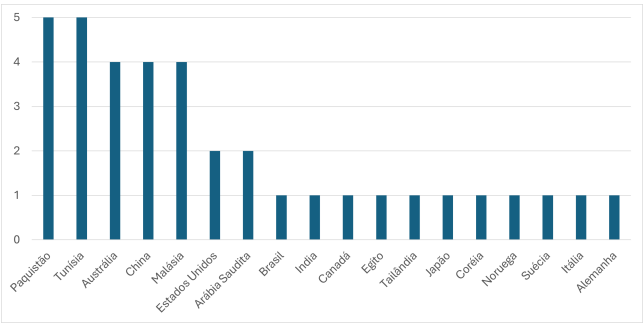


Figura 3: Número de estudos por país

<sup>2</sup><https://parsif.al/>

<sup>3</sup><https://doi.org/10.5281/zenodo.14914670>

**Tabela 3: Ilustração do processo de codificação**

<b>Unidade:</b> “This paper proposed a new approach for change request classification. <i>Proposed approach is based on ontology to classify the different change requests as either Functional Change or Technical Change.</i> ” E7		
<b>Código</b>	<b>Código focado</b>	<b>Categoria</b>
Proposed approach is based on ontology to classify the different change requests as either Functional Change or Technical Change.	Ontology to classify change requests as Functional Change or Technical Change	Solução

**Tabela 4: Lista de estudos selecionados**

<b>ID</b>	<b>Título</b>	<b>Autores</b>	<b>País</b>	<b>Ano</b>
E1	Expediting Time to Market: Evaluating the Effects of Change Control Board Performance in Emerging Markets	Irfan et al. [26]	Paquistão	2023
E2	Improving the evaluation of change requests using past cases	Mello and Fontoura [51]	Brasil	2023
E3	Enhancing Requirements Change Request Categorization and Prioritization in Agile Software Development Using Analytic Hierarchy Process (AHP)	Asad and Muqem [10]	Índia	2023
E4	Elicitation, Computational Representation, and Analysis of Mission and System Requirements	Masoudi et al. [46]	Estados Unidos	2022
E5	An Intelligent Methodology to Enhance Requirements Engineering in Multidisciplinary Projects	Salmani et al. [63]	Canadá	2022
E6	A Conceptual Model to Address the Communication and Coordination Challenges during Requirements Change Management in Global Software Development	Qureshi et al. [56]	Paquistão	2021
E7	Requirements change requests classification: An ontology-based approach	Sakhrawi et al. [62]	Tunísia	2021
E8	Extracting Software Change Requests from Mobile App Reviews	Nadeem et al. [53]	Paquistão	2021
E9	Discovering Relationships among Software Artifacts	Champagne and Carver [14]	Estados Unidos	2020
E10	Measuring ripple effect of natural language requirements change for ULS dynamic requirements	Aly et al. [9]	Egito	2020
E11	A method of assessing rework for implementing software requirements changes	Jayatilleke and Lai [30]	Austrália	2020
E12	An in-Depth requirements change evaluation process using functional and structural size measures in the context of agile software development	Hakim et al. [23]	Tunísia	2020
E13	Survey on requirement-driven microservice system evolution	Wang et al. [70]	China	2020
E14	Specification of a hybrid effort estimation system using UML	Kadir et al. [32]	Malásia	2019
E15	AZ-Model of software requirements change management in global software development	Akbar et al. [4]	China	2019
E16	Extending function point analysis effort estimation method for software development phase	Shah and Kama [66]	Malásia	2018
E17	A Domain Ontology for Software Requirements Change Management in Global Software Development Environment	Alsanad et al. [7]	Arábia Saudita	2019
E18	An ontology framework of software requirements change management process based on causality	Yan et al. [74]	China	2018
E19	Managing software requirements changes through change specification and classification	Jayatilleke et al. [31]	Austrália	2018
E20	Towards a software requirements change classification using support vector machine	Khelifa et al. [37]	Tunísia	2018
E21	Orchestrating functional change decisions in scrum process using COSMIC FSM method	Sellami et al. [64]	Tunísia	2019
E22	Towards an assessment tool for controlling functional changes in scrum process	Sellami et al. [65]	Tunísia	2018
E23	Risk factors for software requirements change implementation	Rahman et al. [58]	Malásia	2019
E24	Multilevel Ontology Framework for Improving Requirements Change Management in Global Software Development	Alsanad et al. [8]	Arábia Saudita	2019
E25	Requirement change taxonomy and categorization in agile software development	Saher et al. [60]	Paquistão	2017
E26	Diagram change types taxonomy based on analysis and design models in UML	Inpirom and Prompoon [24]	Tailândia	2013
E27	Tracking requirements evolution by using issue tickets: A case study of a document management and approval system	Saito et al. [61]	Japão	2014
E28	A process model for requirements change management in collocated software development	Khan et al. [36]	Malásia	2012
E29	Managing software requirements changes based on negotiation-style revision	Mu et al. [52]	China	2011
E30	Requirements changes rework effects: A case study	Chua [16]	Austrália	2010
E31	Requirements change management on feature-oriented requirements tracing	Ahn and Chong [2]	Coreia	2007
E32	Using business process modelling to reduce the effects of requirements changes in software projects	Mathisen et al. [47]	Noruega	2009
E33	A methodology to manage the changing requirements of a software project	Bhatti et al. [11]	Paquistão	2010
E34	Criteria for estimating effort for requirements changes	Chua et al. [17]	Austrália	2008
E35	Processing requirements by software configuration management	Crnkovic et al. [19]	Suécia	1999
E36	Requirements-based estimation of change costs	Lavazza and Valetto [41]	Itália	2000
E37	Causal analysis of the requirements change process for a large system	El Emam et al. [20]	Alemanha	1997

## 5.1 QP1: Quais são os fatores que motivam as SMR?

Essa questão visou identificar os fatores que motivam a formalização de uma SMR. Em via de melhor representatividade e apresentação, os fatores que tinham similaridade foram agrupados em um único código. O fator com maior evidência entre os estudos selecionados foi **mudanças nas necessidades dos usuários/clientes**, apresentado em 7 estudos (E1, E3, E4, E15, E23, E29, E37). A Tabela 5 apresenta os fatores identificados.

**Tabela 5: Lista de fatores identificados**

ID	Fator	Estudos
F1	Mudanças nas necessidades dos usuários/clientes	E1, E3, E4, E15, E23, E29, E37
F2	Feedback ou reviews dos usuários	E1, E8, E13
F3	Mercado ou ambiente de negócios	E3, E15, E23
F4	Entendimento inadequado dos requisitos iniciais	E4, E32
F5	Mudanças tecnológicas	E15, E23
F6	Solicitações organizacionais ou da corporação	E3, E15
F7	Aumento na especialização do trabalhador do software	E3
F8	Mudanças no ambiente do produto	E4
F9	Redução no escopo do projeto	E4
F10	Melhoria na testabilidade	E4
F11	Descoberta de novos requisitos	E4
F12	User stories	E12
F13	Fontes públicas	E13
F14	Tempo de execução	E13
F15	Ocorrência de requisitos ruins	E18
F16	Aderência a política organizacional e operacional	E23
F17	Solicitações fora do escopo ou controle do projeto	E32
F18	Requisitos iniciais incompletos, inconsistentes ou tecnicamente não-implementáveis	E33
F19	Linha de base técnica	E37

## 5.2 QP2: Quais são as soluções utilizadas para gerenciar SMR?

Essa questão buscou examinar e documentar as soluções utilizadas na gerência de SMR. A taxonomia definida por Shaw [68] foi utilizada para categorizar as soluções identificadas nos estudos selecionados: (i) teoria: um sistema de ideias destinado a explicar algo; (ii) modelo: estrutura ou taxonomia para uma área de problema; (iii) método/técnica: uma forma particular de procedimento para realizar algo; (iv) prática: uma maneira nova ou melhor de fazer alguma tarefa; (v) ferramenta: dispositivo que incorpora uma técnica; e (vi) *framework*: uma estrutura de suporte para fazer algo.

A categoria de solução com maior representação entre os estudos selecionados foi **método**, apresentada em 10 estudos (E5, E8, E9, E11, E19, E20, E21, E22, E36, E37). Em relação as outras categorias de soluções identificadas, foram também encontrados **modelos** em 9 estudos (E10, E15, E16, E25, E26, E28, E29, E31, E35), **framework** em 5 estudos (E3, E13, E24, E30, E34), **ferramenta** também em 5 estudos (E2, E14, E21, E22, E27), **teoria** em 2 estudos (E7, E17) e, finalmente, **práticas** em 2 estudos (E12, E31). A Tabela 6 apresenta as soluções identificadas.

**5.2.1 Modelos.** E10 utilizou processamento de linguagem natural (PLN) para desenvolver um modelo que avalia o impacto de SMR de sistemas de ultra larga escala (ULSS). De forma semelhante, E15 propôs um modelo que visa lidar com solicitações de mudança de forma eficiente através de gerenciamento de projetos especializado e *strong time boxing*. E16 apresentou o modelo *Function Point Analysis for Software Development Phase* (FPA-SDP), integrando a análise

de impacto de mudanças com a análise de pontos de função, com o objetivo de estimar o esforço em mudanças de software para solicitações de mudança durante o desenvolvimento. Em outra perspectiva, E25 desenvolveu uma taxonomia para explicar elementos relacionados a mudanças de requisitos e propor uma categorização de mudanças para identificar claramente as solicitações de mudança. Complementando esse enfoque, E26 apresentou uma taxonomia para a classificação de tipos de mudança baseada em uma representação UML da mudança requisitada, composta por diagramas de caso de uso, diagramas de sequência e diagramas de classe.

Outros modelos também foram elaborados com diferentes enfoques. E28 introduziu um modelo de processo voltado ao gerenciamento de mudanças de requisitos, que trata as solicitações de mudança desde o início. E29 apresentou um modelo para gerenciar mudanças de requisitos baseado em revisão de crenças no estilo de negociação, considerando a solicitação de mudança e a especificação atual como partes de uma negociação. E31 criou um modelo em árvore de mudanças de requisitos para gerar solicitações de mudança generalizadas por nível de funcionalidade. E35 propôs um modelo para gerenciar requisitos usando funções de gerenciamento de configuração e gerenciamento de mudanças, estruturando a especificação de requisitos como itens individuais sob controle de versão e relacionando-os com as solicitações de mudança.

**5.2.2 Métodos.** E5 propôs uma solução automatizada para estimar o esforço necessário para implementar novas funcionalidades ou mudanças de requisitos. E8 apresentou uma abordagem que trazem 2 componentes que podem apoiar a gerência de SMR: *guidelines* de sentenças de solicitações e não-solicitações de mudanças de software e um *dataset* de solicitações e não-solicitações de mudanças de software. E9 introduziu um método de análise textual de requisitos utilizando técnicas de *information retrieval* (IR) como TF-IDF e LSI para detectar funcionalidades com potencial de afetar requisitos existentes. E11 apresentou um método voltado à avaliação do re-trabalho necessário para implementar mudanças de requisitos. O método inclui a identificação de solicitações de mudança, especificação da mudança e métodos de classificação propostos pelo estudo. E19 propôs dois métodos, o primeiro voltado à especificação de mudanças e o segundo voltado à classificação de mudanças, ambos baseados em um layout originado de uma SMR, cujo objetivo é melhorar a comunicação e elicitação de mudanças de requisitos.

E20 apresentou a utilização de processamento de linguagem natural para classificar solicitações de mudança automaticamente em 10 subgrupos (1 funcional + 9 subgrupos não-funcionais). E21 utilizou o método COSMIC Functional Size Measurement (FSM) para implementação de uma abordagem de avaliação de mudanças funcionais aplicada ao processo Scrum em uma ferramenta. E36 propôs um método para estimar o impacto e o custo das mudanças nos requisitos do usuário, explorando modelos do produto e do processo de implementação da alteração, caracterizados quantitativamente por medições adequadas. E37 apresentou um método para analisar o baixo custo que pode ser aplicado para identificar problemas de implementação no processo de mudanças de requisitos e suas causas, transformando solicitações de mudança que endereçam problemas da *Technical Baseline* e propostas de mudança.

**Tabela 6: Lista de soluções identificadas**

ID	Solução	Tipo de solução	Estudos
S1	Tool that implements an approach to facilitate the risk analysis of a change request	Ferramenta	E2
S2	Framework for classifying agile software development change requests into small change requests (SCRs) and large change requests (LCRs) using Analytic Hierarchy Process (AHP)	Framework	E3
S3	Automated process for estimating the required effort for developing new features or implementing requirement changes	Método	E5
S4	Ontology to classify change requests as Functional Change or Technical Change	Teoria	E7
S5	SCR-Dataset	Método	E8
S6	SCR Sentence and Non-SCR Sentence guidelines	Método	E8
S7	Information Retrieval and Data Mining-Based Requirements Change Impact Analysis Method	Método	E9
S8	Model based on the Natural Language Processing (NLP) similarity models to measure the impact of group of Changes on ULS Requirements represented in Natural Language	Modelo	E10
S9	Method of assessing rework for implementing software requirements changes	Método	E11
S10	In-depth evaluation process of requirement changes	Processo	E12
S11	Conceptual Framework for Requirement-Driven Microservice System Evolution	Framework	E13
S12	Tool that implements the Change Effort Prediction Model (CEPM)	Ferramenta	E14
S13	AZ-Model of RCM in GSD	Modelo	E15
S14	Function Point Analysis for Software Development Phase (FPA-SDP) Model	Modelo	E16
S15	Requirement change ontology (RCO)	Teoria	E17
S16	Method for change specification	Método	E19
S17	Method for change classification	Método	E19
S18	Use of NLP and Support Vector Machine (SVM) to classify requirements change requests	Método	E20
S19	COSMIC Functional Size Measurement (FSM)	Método	E21, E22
S20	Tool for measuring the Functional Size of User Stories using COSMIC FSM	Ferramenta	E21
S21	Tool for change controlling in scrum process	Ferramenta	E22
S23	Multilevel ontology framework to support RCM process for GSD	Framework	E24
S24	Requirement Change Taxonomy	Modelo	E25
S25	Taxonomy of UML diagram for change types	Modelo	E26
S26	Requirements Evolution Chart (REC)	Ferramenta	E27
S27	A process model for requirements change management in collocated software development	Modelo	E28
S28	Belief negotiation models to execute requirements changes	Modelo	E29
S29	Conceptual change management framework	Framework	E30
S30	Requirements change tree model	Modelo	E31
S31	Feature-oriented requirements change management process	Processo	E31
S32	Framework for estimating person-effort for requirements changes	Framework	E34
S33	Model for managing requirements using Configuration Management functions	Modelo	E35
S34	Method for estimation of impact and cost of changes in user requirements	Método	E36
S35	Low cost analysis method	Método	E37

**5.2.3 Framework.** E3 apresentou um *framework* para classificar SMR no desenvolvimento de software ágil em “pequenas solicitações de mudanças” e “grandes solicitações de mudanças” utilizando o processo analítico hierárquico para ranquear esses requisitos. Essa classificação é baseada no nível de análise de impacto, tempo e esforço necessários para implementar as mudanças. E13 introduziu um *framework* conceitual seguindo a filosofia de engenharia de software contínua para tratar solicitações de mudança em sistemas baseados em microsserviços. E24 propôs um *framework* de ontologias visando melhorar a semântica de SMR para reduzir problemas de comunicação e mal-entendidos no contexto de desenvolvimento global de software. E30 desenvolveu um *framework* conceitual para gerenciamento de mudanças visando controlar ou evitar o custo do esforço aplicado ao retrabalho causado por mudanças de requisitos. E34 propôs um *framework* voltado a estimar o esforço de pessoal em mudanças de requisitos, baseado em uma análise específica.

**5.2.4 Ferramentas.** E2 desenvolveu uma ferramenta para implementar uma abordagem que combina Case-Based Reasoning e Association Rule Mining para facilitar a análise de riscos de SMR, fornecendo informações sobre casos passados semelhantes, soluções adotadas e suporte à decisão. E14 propôs uma ferramenta que instancia uma abordagem híbrida combinando métodos de estimação de esforço em software para fornecer uma decisão mais precisa ao gestor de projeto quando houver uma SMR. E21 desenvolveu

uma ferramenta que implementou uma abordagem de avaliação de mudanças funcionais (*Functional Change*) aplicada ao processo Scrum, baseada no método COSMIC FSM. E22 introduziu uma ferramenta que implementa o método COSMIC FSM para quantificar e avaliar uma solicitação de mudança, visando controlar as mudanças no processo Scrum. E27 desenvolveu uma ferramenta que implementou uma abordagem de visualização da evolução de requisitos.

**5.2.5 Teorias.** E7 apresentou uma ontologia para classificar solicitações de mudança como mudanças funcionais ou técnicas, seguindo a norma ISO-25010. E17 propôs uma ontologia para a gerência de mudanças de requisitos visando mitigar problemas de comunicação e mal-entendimentos no contexto de desenvolvimento global de software.

**5.2.6 Práticas.** E12 desenvolveu um processo para a avaliação de SMR no método Scrum, baseado em *user stories* que possam afetar requisitos funcionais ou não-funcionais. E31 propôs um processo para a gerência de mudanças de requisitos sob traços de requisitos orientados a funcionalidades.

### 5.3 QP3: Como essas soluções têm sido avaliadas?

Essa QP visou identificar como as soluções identificadas pela QP anterior são avaliadas e quais os critérios utilizados para determinar sua eficácia no gerenciamento de SMR. Nem todas as soluções ou

componentes para compor uma solução identificados na QP anterior tiveram uma avaliação realizada. Dentre os que tiveram uma avaliação realizada, as avaliações baseadas em **estudos de caso** tiveram maior representatividade entre os estudos selecionados com 9 estudos (E2, E3, E5, E9, E10, E16, E27, E30, E36). Em relação aos outros métodos de avaliação, foram também encontrados **survey** em 2 estudos (E15, E24), **experimento com algoritmos de aprendizado de máquina** em 1 estudo (E8), **validação e verificação de propósito** em 1 estudo (E17), e **análise causal** em 1 estudos (E37).

#### 5.4 QP4: Quais são os desafios na gerência de SMR?

Essa QP busca destacar os principais desafios no gerenciamento de SMR e as lacunas existentes na literatura. Foram identificadas 2 desafios com maior representatividade entre os estudos selecionados (3 estudos cada): **consumo de tempo e recursos** (E6, E15, E23) e **tomada de decisão sobre aceitar ou rejeitar mudanças** (E16, E19, E23). A Tabela 7 apresenta os desafios identificados.

**Tabela 7: Desafios no gerenciamento de SMR**

ID	Desafio	Estudos
D1	Consumo de tempo e recursos	E6, E15, E23
D2	Tomada de decisão sobre aceitar ou rejeitar mudanças	E16, E19, E23
D3	Comunicação e coordenação	E6, E19
D4	Impacto no cronograma e custo de projeto	E14, E16
D5	Satisfação das partes interessadas	E16
D6	Inconsistência de artefatos de software durante o desenvolvimento	E16
D7	Informações insuficientes para avaliar mudanças	E19
D8	Interpretação errônea dos objetivos da mudança	E19
D9	Representação incorreta das mudanças no projeto do sistema	E19
D10	Discrepâncias na análise das mudanças	E19
D11	Impacto em múltiplos estágios de desenvolvimento	E29
D12	Relacionar requisitos não-funcionais a mudanças de requisitos	E35

## 6 Discussão

Este estudo identificou fatores, soluções e desafios voltados ao gerenciamento de SMR, mapeando teorias, modelos, métodos/técnicas, práticas, ferramentas e *frameworks* propostos na literatura. Durante a análise dos 1.061 estudos recuperados, foi constatado que a literatura científica ainda dá pouca atenção específica a esse tema. Embora as SMR sejam inevitáveis para a dinâmica das relações entre as partes interessadas envolvidas no desenvolvimento de software, muitos estudos apenas citam a existência das SMR, mas sem aprofundar na discussão sobre como gerenciá-las efetivamente. Apenas 37 estudos foram selecionados para análise detalhada, o que revela a necessidade de investigar esse tópico com mais profundidade.

Os dados evidenciam que a pesquisa sobre SMR ainda está em um estágio emergente. Sakhravi et al. [62] destacaram este cenário e afirmaram que é necessário fornecer uma compreensão apropriada de uma SRM de forma sistemática. Além disso, a maioria das soluções identificadas são métodos, *frameworks* ou ferramentas que nem sempre são avaliadas em contextos industriais. Isso destaca a importância de estudos experimentais que testem essas abordagens em cenários reais, conforme citado por Jayatilleke and Lai [29].

Considerando a teoria socio-técnica, que enfatiza a interdependência entre fatores sociais e técnicos nos processos organizacionais, os desafios identificados neste estudo, como a tomada de decisão sobre mudanças e a comunicação entre partes interessadas, reforçam

a importância de abordagens integradas para o gerenciamento de SMR. A gestão eficaz dessas mudanças não pode se limitar apenas a soluções técnicas, mas deve considerar também a colaboração entre diferentes partes interessadas e a adaptação contínua dos processos. Qureshi et al. [56] destacam que desafios como comunicação insuficiente e falta de coordenação podem ser mitigados com ferramentas que promovam melhor rastreabilidade e integração.

### 6.1 Ameaças à validade

Esta seção apresenta as possíveis ameaças à validade desta RR. Durante o decorrer desta pesquisa, procurou-se minimizar a influência dessas ameaças e reduzir seus possíveis riscos. **Validade descritiva:** para reduzir essa ameaça, foi desenvolvido um formulário de coleta de dados para apoiar o registro das informações necessárias para responder às QP. **Validade teórica:** os estudos foram analisados e selecionados sob os conceitos de ER e gerência de requisitos [22, 27, 28, 69]. A *string* de busca foi definida de forma inclusiva para capturar estudos que apresentassem respostas a alguma das QP. Vale destacar que a *string* de busca definida foi baseada em termos já utilizados em trabalhos relacionados à SMR [5, 29].

Em relação à **Generalização:** não é uma grande ameaça, pois foram utilizados os protocolos propostos por Cartaxo et al. [13] e Kitchenham and Charters [39], o que facilita a replicação da pesquisa. Além disso, a generalização não é o foco de um estudo de RR [12], uma vez que o objetivo é analisar o estado da arte de um tema específico. Os conjuntos de dados também estão disponíveis no material suplementar. **Validade interpretativa:** para minimizar o viés do pesquisador, quando havia dúvidas na execução do processo de seleção, estas eram discutidas entre outros dois pesquisadores, e as diferenças eram analisadas em conjunto com estes pesquisadores até se chegar a um consenso.

## 7 Conclusão

Neste estudo, foi conduzida uma RR com o objetivo de identificar o que se sabe sobre SMR no contexto de desenvolvimento de software. Para isso, foram selecionados 37 estudos que contribuíram para analisar o estado da arte relacionado a fatores, soluções e desafios na gerência de SMR. Como principal contribuição, este estudo oferece uma análise detalhada destes pontos encontradas na literatura, destacando os fatores que levam a uma SMR (QP1); métodos/técnicas, modelos, práticas, ferramentas, teorias e *frameworks* que são utilizados para gerenciar SMR (QP2) e os tipos de avaliação aplicados nestas soluções (QP3) e os desafios na gerência de SMR (QP4).

Este estudo também tem implicações para a academia e a indústria. Para os acadêmicos, este trabalho oferece uma visão geral do que tem sido estudado até agora sobre SMR, podendo contribuir para o avanço do estado da arte. Para a indústria, as descobertas deste estudo podem auxiliar os profissionais na compreensão de SMR a partir dos fatores e desafios identificados e as atividades que envolvam-as a partir das soluções mapeadas. A principal mensagem deste estudo é que a pesquisa sobre SMR está em crescimento, mas ainda há uma falta de estudos que se concentrem mais no gerenciamento de SMR.

Como trabalhos futuros, a partir da identificação das soluções e desafios, uma investigação adicional pode ser realizada para examinar como essas soluções e desafios foram utilizadas em seus



contexto de software (e.g. ecossistemas de software, desenvolvimento ágil). Também é recomendável desenvolver ferramentas automatizadas que ajudem na gestão de SMR, promovendo melhor comunicação e colaboração entre as partes interessadas, além de rastreabilidade e mapeamento dos requisitos modificados. Além disso, é necessário conduzir estudos de campo para entender como os desenvolvedores e engenheiros de requisitos percebem e lidam com as SMR na prática, explorando a eficácia das soluções propostas e identificando novas oportunidades de melhoria.

## DISPONIBILIDADE DE ARTEFATOS

Os dados brutos e todas as etapas necessárias para reproduzir o trabalho estão detalhados no material suplementar, que está disponível abertamente em <https://doi.org/10.5281/zenodo.14914670>.

## AGRADECIMENTOS

Este estudo foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001, CNPq (Proc. 316510/2023-8), FAPERJ (Proc. E-26/204.404/2024) e UNIRIO.

## Referências

- [1] Syed Adnan Afaq and Mohammad Faisal. 2021. An Efficient Approach For Software Requirement Change Identification. *Webology* 18, 3 (2021), 1919–1926.
- [2] Sangim Ahn and Kiwon Chong. 2007. Requirements Change Management on Feature-Oriented Requirements Tracing. In *Computational Science and Its Applications – ICCSA 2007*, Osvaldo Gervasi and Marina L. Gavrilova (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 296–307.
- [3] Muhammad Azeem Akbar, Sajjad Mahmood, Ahmed Alsanad, Muhammad Shafiq, Abdu Guma, and Abeer Abdul-Aziz Alsanad. 2020. Organization Type and Size Based Identification of Requirements Change Management Challenges in Global Software Development. *IEEE Access* 8 (2020), 94089–94111. <https://doi.org/10.1109/ACCESS.2020.2995238>
- [4] Muhammad Azeem Akbar, Nasrullah, Muhammad Shafiq, Jawad Ahmad, Muhammad Mateen, and Muhammad Tanveer Riaz. 2018. AZ-Model of software requirements change management in global software development. In *2018 International Conference on Computing, Electronic and Electrical Engineering (ICE Cube)*. 1–6. <https://doi.org/10.1109/ICECUBE.2018.8610964>
- [5] Muhammad Azeem Akbar, Jun Sang, Nasrullah, Arif Ali Khan, Sajjad Mahmood, Syed Furqan Qadri, Haibo Hu, and Hong Xiang. 2019. Success factors influencing requirements change management process in global software development. *Journal of Computer Languages* 51 (2019), 112–130. <https://doi.org/10.1016/j.col.2018.12.005>
- [6] Naveed Ali and Richard Lai. 2016. A method of requirements change management for global software development. *Information and Software Technology* 70 (2016), 49–67. <https://doi.org/10.1016/j.infsof.2015.09.005>
- [7] Abeer Abdulaziz Alsanad, Azeddine Chikh, and Abdulrahman Mirza. 2019. A Domain Ontology for Software Requirements Change Management in Global Software Development Environment. *IEEE Access* 7 (2019), 49352–49361. <https://doi.org/10.1109/ACCESS.2019.2909839>
- [8] Abeer Abdulaziz Alsanad, Azeddine Chikh, and Abdulrahman Mirza. 2019. Multilevel Ontology Framework for Improving Requirements Change Management in Global Software Development. *IEEE Access* 7 (2019), 71804–71812. <https://doi.org/10.1109/ACCESS.2019.2916782>
- [9] Ahmed Aly, Mohamed Senousy, and Alaa el-din Riad. 2020. MEASURING RIPPLE EFFECT OF NATURAL LANGUAGE REQUIREMENTS CHANGE FOR ULS DYNAMIC REQUIREMENTS. *Journal of Theoretical and Applied Information Technology* 98 (07 2020), 2214–2226.
- [10] K. Asad and M. Muqeem. 2023. Enhancing Requirements Change Request Categorization and Prioritization in Agile Software Development Using Analytic Hierarchy Process (AHP). *International Journal on Recent and Innovation Trends in Computing and Communication* 11, 5 (May 2023), 148–159. <https://doi.org/10.17762/ijritcc.v11i5.6589>
- [11] Muhammad Wasim Bhatti, Farah Hayat, Nadeem Ehsan, Azam Ishaque, Sohail Ahmed, and Ebtisam Mirza. 2010. A methodology to manage the changing requirements of a software project. In *2010 International Conference on Computer Information Systems and Industrial Management Applications (CISIM)*. 319–322. <https://doi.org/10.1109/CISIM.2010.5643642>
- [12] Bruno Cartaxo, Gustavo Pinto, and Sergio Soares. 2018. The role of rapid reviews in supporting decision-making in software engineering practice. In *International Conference on Evaluation and Assessment in Software Engineering* 2018. 24–34. <https://doi.org/10.1145/3210459.3210462>
- [13] Bruno Cartaxo, Gustavo Pinto, and Sergio Soares. 2020. *Rapid reviews in software engineering*. Springer, 357–384. [https://doi.org/10.1007/978-3-030-32489-6\\_13](https://doi.org/10.1007/978-3-030-32489-6_13)
- [14] Job M. Champagne and Doris L. Carver. 2020. Discovering Relationships Among Software Artifacts. In *2020 IEEE Aerospace Conference*. 1–11. <https://doi.org/10.1109/AERO47225.2020.9172288>
- [15] Kathy Charmaz. 2006. *Constructing Grounded Theory: A Practical Guide Through Qualitative Analysis*. Vol. 1. Sage Publications, Thousand Oaks.
- [16] B.B. Chua. 2010. Requirements Changes Rework Effects: A Case Study. *Proceedings of the IASTED International Conference on Software Engineering and Applications, SEA 2010* (01 2010). <https://doi.org/10.2316/P.2010.725-073>
- [17] Bee Bee Chua, Danilo Valeros Bernardo, and June Verner. 2008. Criteria for Estimating Effort for Requirements Changes. In *Software Process Improvement*, Rory V. O'Connor, Nathan Baddoo, Kari Smolander, and Richard Messnarz (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 36–46.
- [18] Ricardo Colomo-Palacios, Cristina Casado-Lumbreras, Pedro Soto-Acosta, Francisco García-Peñalvo, and Edmundo Tovar. 2014. Project managers in global software development teams: a study of the effects on productivity and performance. *Software Quality Journal* 22, 1 (2014), 3–19. <https://doi.org/10.1007/s11219-012-9191-x>
- [19] I. Crnkovic, P. Funk, and M. Larsson. 1999. Processing requirements by software configuration management. In *Proceedings 25th EUROMICRO Conference. Informatics: Theory and Practice for the New Millennium*, Vol. 2. 260–265 vol.2. <https://doi.org/10.1109/EURMIC.1999.794789>
- [20] K. El Emam, D. Holtje, and N.H. Madhavji. 1997. Causal analysis of the requirements change process for a large system. In *International Conference on Software Maintenance*. 214–221. <https://doi.org/10.1109/ICSM.1997.624248>
- [21] Ridda Fatima, Furkh Zeshan, Adnan Ahmad, Muhammad Hamid, Ashfaq Ahmad, and Sohail Afzal Tahir. 2021. Software Requirements Change Prediction Model. In *2021 International Conference on Decision Aid Sciences and Application*. IEEE, Sakheer, Bahrain, 607–612. <https://doi.org/10.1109/DASA53625.2021.9682217>
- [22] D Méndez Fernández, Stefan Wagner, Marcos Kalinowski, Michael Felderer, Priscilla Mafra, Antonio Vetrò, Tayana Conte, M-T Christiansson, Des Greer, Casper Lassenius, et al. 2017. Naming the pain in requirements engineering: Contemporary problems, causes, and effects in practice. *Empirical software engineering* 22 (2017), 2298–2338. <https://doi.org/10.1007/s10664-016-9451-7>
- [23] Hela Hakim, Asma Sellami, and Hanène Ben-Abdallah. 2020. An in-Depth Requirements Change Evaluation Process using Functional and Structural Size Measures in the Context of Agile Software Development. In *15th International Conference on Software Technologies*. 361–375. <https://doi.org/10.5220/0009876003610375>
- [24] Akapon Inpirom and Nakornthip Prompoon. 2013. Diagram change types taxonomy based on analysis and design models in UML. In *2013 IEEE 4th International Conference on Software Engineering and Service Science*. 283–287. <https://doi.org/10.1109/ICSESS.2013.6615306>
- [25] M Aqeel Iqbal and Asadullah Shah. 2021. Stakeholder’s evaluation process for GSD based requirements elicitation frameworks. *Pakistan Journal of Engineering and Technology* 4, 1 (2021), 175–183. <https://doi.org/10.51846/vol4iss1pp175-183>
- [26] Sanaullah Irfan, Jamshid Ali, Imdadullah Hidayat-ur Rehman, Muddasar Ghani Khwaja, Joanna Rosak-Szyrocka, and Attila Kovacs. 2023. Expediting Time to Market: Evaluating the Effects of Change Control Board Performance in Emerging Markets. *Sustainability* 15, 22 (2023). <https://doi.org/10.3390/su152216085>
- [27] ISO/IEC/IEEE 29148. 2018. ISO/IEC/IEEE International Standard - Systems and software engineering – Life cycle processes – Requirements engineering. (2018), 1–104. <https://doi.org/10.1109/IEEESTD.2018.8559686>
- [28] Rahat Izhar, Dr. Shahid Nazir Bhatti, Saba Izhar, and Dr. Amr Mohsen Jada. 2018. Impact and Challenges of Requirements Management in Enterprise Resource Planning (ERP) via ERP Thesaurus. *International Journal of Advanced Computer Science and Applications* 9, 8 (2018). <https://doi.org/10.14569/IJACSA.2018.090832>
- [29] Shalinka Jayatilake and Richard Lai. 2018. A Systematic Review of Requirements Change Management. *Information and Software Technology* 93 (2018), 163–185. <https://doi.org/10.1016/j.infsof.2017.09.004>
- [30] Shalinka Jayatilake and Richard Lai. 2021. A method of assessing rework for implementing software requirements changes. *Computer Science and Information Systems* 18, 1 (2021), 129–154.
- [31] Shalinka Jayatilake, Richard Lai, and Karl Reed. 2018. Managing software requirements changes through change specification and classification. *Computer Science and Information Systems* 15, 2 (2018), 321–346.
- [32] Nur Kadir, Haslina Sarkan, Azri Azmi, Othman Yusop, and Mohd Kama. 2019. Specification of a Hybrid Effort Estimation System using UML. In *2019 6th International Conference on Research and Innovation in Information Systems (ICRIIS)*. 1–7. <https://doi.org/10.1109/ICRIIS48246.2019.9073665>
- [33] Maryam Kausar, Anas Wali Muhammad, Rabia Jabbar, and Muhammad Ishfaq. 2022. Key challenges of requirement change management in the context of global software development: systematic literature review. *Pakistan Journal of Engineering and Applied Sciences* 30 (2022), 41–51.

- [34] Arif Ali Khan and Muhammad Azeem Akbar. 2020. Systematic literature review and empirical investigation of motivators for requirements change management process in global software development. *Journal of Software: Evolution and Process* 32, 4 (2020), e2242. <https://doi.org/10.1002/smr.2242>
- [35] Arif Ali Khan and Muhammad Azeem Akbar. 2020. Systematic literature review and empirical investigation of motivators for requirements change management process in global software development. *Journal of Software: Evolution and Process* 32, 4 (2020), e2242. <https://doi.org/10.1002/smr.2242>
- [36] Arif Ali Khan, Shuib Basri, P. D. D. Dominic, and Fazal e Amin. 2012. A process model for Requirements Change Management in collocated software development. In *2012 IEEE Symposium on E-Learning, E-Management and E-Services*. 1–6. <https://doi.org/10.1109/IS3e.2012.6414949>
- [37] Amani Khelifa, Mariem Haoues, and Asma Sellami. 2018. Towards a Software Requirements Change Classification using Support Vector Machine. In *Conference on Language Processing and Knowledge Management*. 1–10.
- [38] Valerie J. King, Adrienne Stevens, Barbara Nussbaumer-Streit, Chris Kamel, and Chantelle Garritty. 2022. Paper 2: Performing rapid reviews. *Systematic Reviews* 11, 1 (30 Jul 2022), 151. <https://doi.org/10.1186/s13643-022-02011-5>
- [39] Barbara Kitchenham and Stuart Charters. 2007. *Guidelines for performing structural literature reviews in software engineering*. Technical Report. Evidence-Based Software Engineering (EBSE) Project.
- [40] Mahesh Kumar, Omkarprasad S Vaidya, and Rajiv Kumar Srivastava. 2021. Impact of task priority on software supply chain: a simulation approach. *South Asian Journal of Business Studies* 10, 3 (2021), 326–341.
- [41] Luigi Lavazza and Giuseppe Valetto. 2000. Requirements-Based Estimation of Change Costs. *Empirical Software Engineering* 5, 3 (01 Nov 2000), 229–243. <https://doi.org/10.1023/A:1026590615963>
- [42] Jingzhou Li, Guenther Ruhe, Ahmed Al-Emran, and Michael M. Richter. 2007. A flexible method for software effort estimation by analogy. *Empirical Software Engineering* 12, 1 (2007), 65–106. <https://doi.org/10.1007/s10664-006-7552-4>
- [43] Kashumi Madampe, Rashina Hoda, and John Grundy. 2022. A Faceted Taxonomy of Requirements Changes in Agile Contexts. *IEEE Transactions on Software Engineering* 48, 10 (2022), 3737–3752. <https://doi.org/10.1109/TSE.2021.3104732>
- [44] Kashumi Madampe, Rashina Hoda, and John Grundy. 2024. Supporting Emotional Intelligence, Productivity and Team Goals while Handling Software Requirements Changes. *ACM Transactions on Software Engineering and Methodology* 33, 6, Article 153 (2024), 38 pages. <https://doi.org/10.1145/3664600>
- [45] Kashumi Madampe2023, Rashina Hoda, and John Grundy. 2023. The Emotional Roller Coaster of Responding to Requirements Changes in Software Engineering. *IEEE Transactions on Software Engineering* 49, 3 (2023), 1171–1187. <https://doi.org/10.1109/TSE.2022.3172925>
- [46] N. Masoudi, R. Rai, J. Ortiz, M. Sutton, et al. 2023. Elicitation, Computational Representation, and Analysis of Mission and System Requirements. *SAE International Journal of Advances and Current Practices in Mobility* 5, 1 (2023), 315–325. <https://doi.org/10.4271/2022-01-0363>
- [47] Eystein Mathisen, Kjell Ellingsen, and Terje Fallmyr. 2009. Using business process modelling to reduce the effects of requirements changes in software projects. In *2009 2nd International Conference on Adaptive Science Technology (ICAST)*. 14–19. <https://doi.org/10.1109/ICASTECH.2009.5409754>
- [48] Sharon McGee and Des Greer. 2011. Software requirements change taxonomy: Evaluation by case study. In *2011 IEEE 19th International Requirements Engineering Conference*. 25–34. <https://doi.org/10.1109/RE.2011.6051641>
- [49] Faisal Mehmood and Sukana Zulfqar. 2021. Effect of Human Related Factors on Requirements Change Management in Offshore Software Development Outsourcing: A theoretical framework. *Software Computing and Machine Intelligence Journal* 1, 1 (2021), 36–52.
- [50] Otávio Mello and Lisandra Fontoura. 2023. Improving the evaluation of change requests using past cases. *International Journal of Information Systems and Project Management* 11, 1 (2023), 74–89.
- [51] Otávio Mello and Lisandra Fontoura. 2023. Improving the evaluation of change requests using past cases. *International Journal of Information Systems and Project Management* 11, 1 (2023), 74–89. <https://doi.org/10.12821/ijispm110104>
- [52] Ke-Dian Mu, Weiru Liu, Zhi Jin, Jun Hong, and David Bell. 2011. Managing Software Requirements Changes Based on Negotiation-Style Revision. *Journal of Computer Science and Technology* 26, 5 (01 Sep 2011), 890–907. <https://doi.org/10.1007/s11390-011-0187-y>
- [53] Muhammad Nadeem, Khurram Shahzad, and Nadeem Majeed. 2021. Extracting Software Change Requests from Mobile App Reviews. In *2021 36th IEEE/ACM International Conference on Automated Software Engineering Workshops (ASEW)*. 198–203. <https://doi.org/10.1109/ASEW52652.2021.00047>
- [54] Klaus Pohl. 2016. *Requirements engineering fundamentals: a study guide for the certified professional for requirements engineering exam-foundation level-IREB compliant*. Rocky Nook, Inc.
- [55] Yudha Prambudya, Muhammad Almaududi Pulungan, et al. 2024. Software Change Request in Software Development Project: Factors and Methods (Scoping Review Methods). *Asian Journal of Engineering, Social and Health* 3, 10 (2024), 2203–2216.
- [56] Saim Qureshi, Saif Ur Rehman Khan, Inayat-Ur-Rehman, Yasir Javed, Shahzad Saleem, and Abid Iqbal. 2021. A Conceptual Model to Address the Communication and Coordination Challenges During Requirements Change Management in Global Software Development. *IEEE Access* 9 (2021), 102290–102303. <https://doi.org/10.1109/ACCESS.2021.3091603>
- [57] Marfizah A Rahman, Rozilawati Razali, and Fatin Filzahti Ismail. 2019. Risk factors for software requirements change implementation. *International Journal of Advanced Computer Science and Applications* 10, 3 (2019).
- [58] Marfizah A. Rahman, Rozilawati Razali, and Fatin Filzahti Ismail. 2019. Risk Factors for Software Requirements Change Implementation. *International Journal of Advanced Computer Science and Applications* 10, 3 (2019). <https://doi.org/10.14569/IJACSA.2019.0100316>
- [59] Abderahman Rashwan, Olga Ormandjieva, and René Witte. 2013. Ontology-Based Classification of Non-functional Requirements in Software Specifications: A New Corpus and SVM-Based Classifier. In *2013 IEEE 37th Annual Computer Software and Applications Conference*. 381–386. <https://doi.org/10.1109/COMPSAC.2013.64>
- [60] Najia Saher, Fauziah Baharom, and Osman Ghazali. 2017. Requirement change taxonomy and categorization in agile software development. In *2017 6th International Conference on Electrical Engineering and Informatics (ICEEI)*. 1–6. <https://doi.org/10.1109/ICEEI.2017.8312441>
- [61] Shinobu Saito, Yukako Iimura, Kenji Takahashi, Aaron K. Massey, and Annie I. Antón. 2014. Tracking requirements evolution by using issue tickets: a case study of a document management and approval system. In *Companion Proceedings of the 36th International Conference on Software Engineering (Hyderabad, India) (ICSE Companion 2014)*. Association for Computing Machinery, New York, NY, USA, 245–254. <https://doi.org/10.1145/2591062.2591194>
- [62] Zaineb Sakhravi, Asma Sellami, and Nadia Bouassida. 2021. Requirements Change Requests Classification: An Ontology-Based Approach. In *Intelligent Systems Design and Applications*, Ajith Abraham, Patrick Siarry, Kun Ma, and Arturas Kaklauskas (Eds.). Springer International Publishing, Cham, 487–496.
- [63] Ali Salmani, Alireza Imani, Majid Bahrehvar, Linda Duffett-Leger, and Mohammad Moshirpour. 2022. An Intelligent Methodology to Enhance Requirements Engineering in Multidisciplinary Projects. In *2022 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*. 452–457. <https://doi.org/10.1109/CCECE49351.2022.9918286>
- [64] Asma Sellami, Mariem Haoues, Nour Borchani, and Nadia Bouassida. 2018. Orchestrating Functional Change Decisions in Scrum Process using COSMIC FSM Method. In *13th International Conference on Software Technologies*. 482–493. <https://doi.org/10.5220/0006853804820493>
- [65] Asma Sellami, Mariem Haoues, Nour Borchani, and Nadia Bouassida. 2018. Towards an Assessment Tool for Controlling Functional Changes in Scrum Process. In *IWSM-Mensura*. <https://api.semanticscholar.org/CorpusID:53313479>
- [66] Jalal Shah and Nazri Kama. 2018. Extending Function Point Analysis Effort Estimation Method for Software Development Phase. In *Proceedings of the 2018 7th International Conference on Software and Computer Applications (Kuantan, Malaysia) (ICSCA '18)*. Association for Computing Machinery, New York, NY, USA, 77–81. <https://doi.org/10.1145/3185089.3185137>
- [67] Jalal Shah, Nazri Kama, Nur Azaliah A Bakar, and Zuhairuddin Bhutto. 2019. Software Requirement Change Effort Estimation Model Prototype Tool for Software Development Phase. *International Journal of Software Engineering & Applications* 10, 3 (2019), 9–19.
- [68] M. Shaw. 2003. Writing good software engineering research papers. In *25th International Conference on Software Engineering, 2003. Proceedings*. 726–736. <https://doi.org/10.1109/ICSE.2003.1201262>
- [69] Wenyan Song. 2017. Requirement management for product-service systems: Status review and future trends. *Computers in Industry* 85 (2017), 11–22. <https://doi.org/10.1016/j.compind.2016.11.005>
- [70] Zhongjie Wang, Xiang He, Lei Liu, Zhiying Tu, and Hanchuan Xu. 2020. Survey on Requirement-Driven Microservice System Evolution. In *2020 IEEE International Conference on Services Computing (SCC)*. 186–193. <https://doi.org/10.1109/SCC49832.2020.00032>
- [71] Karl E Wiegiers and Joy Beatty. 2013. *Software requirements*. Pearson Education.
- [72] Byron J. Williams, Jeffrey C. Carver, and Rayford B. Vaughn. 2006. Change Risk Assessment: Understanding Risks Involved in Changing Software Requirements. In *Proceedings of the International Conference on Software Engineering Research and Practice & Conference on Programming Languages and Compilers, SERP 2006, Las Vegas, Nevada, USA, June 26-29, 2006, Volume 2*, Hamid R. Arabnia and Hassan Reza (Eds.). CSREA Press, 966–971.
- [73] Chi Xu, Yuanbang Li, Bangchao Wang, and Shi Dong. 2023. A systematic mapping study on machine learning methodologies for requirements management. *IET Software* 17, 4 (2023), 405–423. <https://doi.org/10.1049/sfw2.12082>
- [74] Yuqing Yan, Pupei Liao, and Zhenhua Zhang. 2018. An Ontology Framework of Software Requirements Change Management Process Based on Causality. In *Proceedings of the 2nd International Conference on Information System and Data Mining (Lakeland, FL, USA) (ICISDM '18)*. Association for Computing Machinery, New York, NY, USA, 107–111. <https://doi.org/10.1145/3206098.3206117>