Desenvolvimento de Editores Colaborativos em Tempo Real: Revisão Rápida

Laurentino Augusto Dantas
laurentino.dantas@ifms.edu.br
Instituto Federal de Mato Grosso do Sul
Naviraí, MS

Joab Cavalcante da Silva
joab@uems.br
Universidade Estadual de Mato Grosso do Sul
Naviraí, MS

Maria da Graça C. Pimentel
mgp@icmc.usp.br
Universidade de São Paulo
São Carlos, SP

ABSTRACT

Currently, users expect to collaborate synchronously with others, including over the Web. Developing a real-time collaborative editor (RCE) that enables geographically dispersed users to simultaneously edit the same document relies on specific algorithms and techniques. To understand how real-time collaborative editors are developed and tested, and which algorithms or techniques are crucial for their development, we conducted a systematic review addressing the following questions: For which types of objects does the literature present real-time collaborative editing systems (RTCE)? What is the most commonly used architecture by researchers to implement RTCE systems, how are they represented, and which works present models or roadmaps for the development of RCEs? Between Operational Transformation (OT) and Conflict-free Replicated Data Types (CRDTs) algorithms, which one appears most frequently in the selected works? What are the main types of tests conducted to evaluate collaborative editors, which articles conducted user tests, and what is the average number of users involved in these tests? Which works discuss undo/redo techniques? Which works discuss the use of comments, chat, or history as tools to support collaborative work? After screening 365 records published between 1993 and 2024, the review analyzed 23 studies published between 2002 and 2022. The results revealed various approaches and techniques employed in the implementation of RTCEs, offering a comprehensive view of research in this area. This, in turn, allowed for the identification of challenges that future research should address.

KEYWORDS

Edição colaborativa em tempo real, editor colaborativo em tempo real, groupware, revisão sistemática

1 INTRODUÇÃO

A colaboração síncrona remota em ambientes de autoria tornou-se uma opção cada vez mais acessível para os usuários. Em 2024, por exemplo, o *Google Workspace* conta com mais de três bilhões de usuários, enquanto o *Google Docs* sozinho possui mais de um bilhão de usuários ativos mensais.[1] Embora nem todos esses usuários utilizem a colaboração síncrona, esses números destacam o vasto alcance das ferramentas de colaboração e seu crescente potencial para suportar interações em tempo real.

A literatura registra que usuários estão cada vez mais familiarizados com a interação síncrona em plataformas como Google Docs [35, 51] e Etherpad [9]. Videoconferências evoluíram [6] para apoiar pessoas com deficiência [1] e oferecer *blended whiteboards* [21]. Pesquisadores investigam aspectos da autoria colaborativa textual [9] e de software [11] via web, da autoria multimídia via dispositivos móveis [68], opções de colaboração em redes sociais [13], e a demanda por serviços web que suportem comunicação e coordenação [70].

A capacidade de múltiplos usuários trabalharem em sincronia e de forma eficiente vai além da simples edição de documentos, configurando-se como um paradigma fundamental para inovação, educação e comunicação global. Organizações têm observado um aumento significativo na colaboração, eficiência e produtividade de suas equipes ao permitir que documentos, como relatórios técnicos e propostas de trabalho, sejam editados simultaneamente por diversos usuários [56].

Na computação, sistemas que permitem a colaboração simultânea entre usuários geograficamente dispersos são conhecidos como *groupware* em tempo real [15]. Estes sistemas baseiam-se em algoritmos complexos que garantem a consistência dos documentos e a sincronização das edições, evitando restrições de usabilidade e de conflitos [19].

A edição colaborativa em tempo real (RTCE) se tornou uma característica essencial para muitos sistemas, oferecendo benefícios como a eliminação de conflitos e perdas de dados [10]. A evolução contínua da internet impulsiona o desenvolvimento dessas tecnologias, refletindo uma tendência em direção a plataformas verdadeiramente interativas e inclusivas nas quais a distância geográfica não é obstáculo para a criação e o compartilhamento de conhecimento.

A literatura sobre escrita colaborativa *online* e programação colaborativa em tempo real fornece direções sobre desafios e métodos na área de editores colaborativos em tempo real (RCEs). Montanelli and Ruskov [38] realizaram uma revisão sistemática da escrita colaborativa de histórias online, identificando formas de envolvimento dos usuários e propondo direções de pesquisa futuras, como recrutamento de colaboradores e modalidades de contribuição. Tan et al. [65] estudaram a programação colaborativa em tempo revelando cenários de uso, requisitos não atendidos por ferramentas atuais e desafios como atrasos e conflitos de permissões. Tsan et al. [67] compararam configurações de programação em par para estudantes do ensino fundamental, destacando preferências por pares com computadores separados. Esses estudos ilustram a diversidade de

[1] https://explodingtopics.com/blog/google-workspace-stats

In: I Workshop de Revisões Sistemáticas de Literatura em Sistemas Multimídia e Web (WRSL+ 2024). Anais Estendidos do XXX Simpósio Brasileiro de Sistemas Multimídia e Web (WRSL+'2024). Juiz de Fora/MG, Brasil. Porto Alegre: Brazilian Computer Society, 2024.

ISSN 2596-1683

abordagens e desafios na colaboração em tempo real de modo geral, e na construção de sistemas apoio à colaboração de modo particular.

Diante da relevância e da atualidade desse tema, o presente trabalho se propõe a realizar uma revisão sistemática da literatura, com o objetivo de elucidar aspectos cruciais no desenvolvimento de editores colaborativos em tempo real (RCE). Esse esforço está alinhado a vários tópicos de interesse do WebMedia, inclusive "autoria e anotação" e "interação multiusuário."

Assim, apresentamos uma Revisão Rápida (RR) para identificar, de maneira objetiva, sistemática e reprodutível, os trabalhos relevantes sobre o tema. As Revisões Rápidas utilizam métodos abreviados em comparação com as revisões sistemáticas padrão para acelerar o processo de tomada de decisão [57], mas mantêm um caráter sistemático [50]. Seguindo as melhores práticas para RR [16], inicialmente foi definido um protocolo PRISMA[2] para especificar os passos e os resultados esperados da pesquisa.

Um dos motivos de classificarmos nosso trabalho como uma Revisão Rápida é o fato de utilizarmos a indexação realizada pela *ACM Digital Library* para acessar os registros de trabalhos da área de computação disponibilizados em outras bases de dados. O outro motivo é que aplicamos a *string* de busca para a seleção dos artigos nos títulos e nos resumos dos registros, e não no texto do artigo como um todo.

O restante deste texto está organizado como segue. A Seção 2 apresenta uma compilação de conceitos importantes para o desenvolvimento de editores colaborativos em tempo real. A Seção 3 detalha o protocolo adotado na revisão bem como o processo de seleção dos estudos. A Seção 4 apresenta os resultados obtidos relativamente às questões de pesquisa e um resumo dos estudos analisados relativamente às questões. A Seção 5 identifica riscos à validade dos resultados da revisão. A Seção 6 discute desafios de pesquisa associados aos resultados obtidos. A Seção 7 sumariza o trabalho reportado neste estudo.

2 EDITORES COLABORATIVOS EM TEMPO REAL (RCE)

Editores colaborativos em tempo real, ou *Real-Time Collaborative Editors* (RCEs), são ferramentas de software que permitem a múltiplos usuários trabalharem simultaneamente em um mesmo documento, independentemente de suas localizações geográficas. Esses editores oferecem uma interface onde um grupo de usuários pode visualizar e editar o mesmo documento em tempo real, com todas as alterações sendo propagadas e exibidas instantaneamente para todos os participantes [5].

O processo de edição colaborativa é complexo e envolve múltiplas considerações [22, 36]. A principal delas é a manutenção da sincronização entre diversas cópias do documento editadas simultaneamente por usuários distintos. É essencial que os sistemas de RCE integrem camadas de controle de acesso para garantir responsividade e consistência, sem introduzir cargas excessivas. Além disso, deve-se assegurar que todos os usuários vejam a versão atualizada do documento compartilhado [7].

Os RCEs são amplamente utilizados em ambientes como desenvolvimento de software, educação e colaboração em documentos. Eles suportam a edição de uma variedade de tipos de conteúdo, incluindo texto, imagens [5], objetos JSON [27], documentos PDF [28] e objetos 3D [49], entre outros.

Para funcionar eficazmente, um RCE deve atender a vários requisitos comuns, de acordo com Cherif [7]:

- *Alta responsividade local*: O sistema deve ter uma performance comparável àquelas de editores de usuário único [14, 61, 63];
- *Alta concorrência*: O sistema deve permitir que os usuários modifiquem simultaneamente qualquer parte do documento [14, 61];
- *Consistência*: Todos os usuários devem eventualmente visualizar uma versão convergente do documento [14, 61];
- *Coordenação descentralizada*: Atualizações concorrentes devem ser sincronizadas de forma descentralizada para evitar pontos únicos de falha [14, 61];
- *Escalabilidade do número de usuários*: O sistema deve suportar um número dinâmico de usuários, permitindo que se juntem ou saiam do grupo a qualquer momento [23].

Os algoritmos de RCE que suportam esses requisitos são geralmente classificados em dois grupos [2]:

- *Algoritmos centralizados*: Esses algoritmos exigem um servidor central para coordenar as atualizações do documento, gerenciando a concorrência e a ordem das operações. Exemplos incluem SOCT4 [69], GOT [61], e Jupiter [40].
- *Algoritmos descentralizados*: Esses algoritmos permitem que as atualizações sejam executadas em qualquer ordem, sem a necessidade de um servidor central. Exemplos incluem adOPTed [48] e SOCT2 [59].

Além de permitir a edição simultânea e visualização em tempo real das alterações, muitos RCEs oferecem recursos adicionais, e.g.:

- Histórico de alterações: Registra a sequência de operações e os usuários envolvidos [17];
- Comunicação em tempo real: Inclui recursos como chat para facilitar a comunicação durante a edição [17];
- Desfazer e Refazer (*undo/redo*): Permite reverter ações realizadas [20];
- Comentários: Possibilita a inserção de notas em partes específicas do documento [28].

Atualmente, existem diversos RCEs disponíveis na web, incluindo:

- *Google Docs*: Plataforma para edição colaborativa de documentos, planilhas e apresentações com recursos de histórico de versões e comentários;
- *Microsoft Office Online*: Versões online do Word, Excel e PowerPoint com funcionalidades colaborativas;
- *Notion*: Ferramenta que combina notas, tarefas, wikis e bases de dados com colaboração em tempo real;
- *Visual Studio Code com Live Share*: Permite edição colaborativa de código com colaboração simultânea em projetos;
- *Overleaf* : Plataforma para edição colaborativa de documentos LaTeX, amplamente utilizada no meio acadêmico.

Conforme Gadea [19], o design e a implementação de algoritmos de controle de consistência otimista apresentam desafios significativos. Ao longo de mais de três décadas de pesquisa em coedição,

[2]https://www.prisma-statement.org/prisma-2020-statement

surgiram duas principais abordagens para o controle de consistência otimista: Transformações Operacionais (OT) e Tipos de Dados Replicados Comutativos (CmRDTs).

Transformações Operacionais (OT) utilizam transformações que preservam a intenção das operações em ambientes colaborativos, mesmo diante de concorrência. O primeiro algoritmo de OT foi introduzido por Ellis and Gibbs [14] e era baseado em uma abordagem ponto-a-ponto. Por outro lado, Oster et al. [43] propuseram os CmRDTs, que utilizam estruturas de dados projetadas para garantir a comutatividade das operações, eliminando a necessidade de transformações [53]. Os CmRDTs baseiam-se em operações posicionais e são projetados para garantir a consistência sem conflitos [52]. Sun et al. [62, 64] analisaram e compararam as duas estratégias, e detalham as vantagens de OT em relação CRDT.

3 REVISÃO RÁPIDA

3.1 Protocolo da revisão

Foi definido como protocolo de trabalho para esta pesquisa, o método denominado Revisão Rápida (RR) [16, 50] utilizando a base de dados indexada pela ACM e dividindo a seleção e a análise entre dois avaliadores apoiados por uma pessoa árbitra de conflitos. Para análise dos estudos, foram selecionados trabalhos que descrevessem o desenvolvimento de sistemas com suporte a RTCE, incluindo trabalhos que descrevessem a implementação de um RCE. Também foram analisados trabalhos que, mesmo que não apresentassem a implementação de um RCE, descrevessem algoritmos ou técnicas importantes para o seu desenvolvimento.

3.1.1 Fatores PICO.

Conforme descrito por Moher et al. [37], no contexto de uma revisão sistemática, critérios de elegibilidade são os critérios pré-definidos que formam a base para a definição de uma boa questão de pesquisa. Esses critérios incluem fatores como o tipo de população alvo estudada (P), a intervenção realizada ou tratamento aplicado (I), a comparação com outras intervenções (C), e os desfechos de interesse ou resultados esperados (O). De acordo com Shin et al. [55], o termo *intervenção*, no caso de revisão sistemática na área de computação, corresponde à inovação investigada pela pesquisa relatada no artigo analisado.

P (População/Problema): Tipos de objetos editáveis, arquiteturas de sistemas RTCE, tipos de diagramas usados, modelos e roteiros para desenvolvimento, algoritmos de sincronização OT e CRDT, tipos de testes realizados, técnicas de *undo/redo*, e utilização de comentários, chat ou histórico.

I (Intervenção): Sistemas RTCE, implementação de sistemas RTCE, representação de editores colaborativos, desenvolvimento de sistemas RTCE, algoritmos de OT e CRDT, e técnicas de undo/redo.

C (Comparação): Comparação entre diferentes tipos de objetos, arquiteturas, modelos ou roteiros, algoritmos de OT e CRDT, tipos de testes, e técnicas de *undo/redo*.

O (Resultado): Tipos de objetos editáveis, arquitetura mais utilizada e modelos para desenvolvimento, principais tipos de diagramas utilizados, frequência de algoritmos de OT e CRDT, tipos e número de testes realizados, discussões sobre técnicas de *undo/redo*, e uso de ferramentas de apoio ao trabalho colaborativo.

3.1.2 Objetivos e Questões de Pesquisa.

Utilizamos os fatores PICO para definir as questões de pesquisa.

QP1. Para edição de quais tipos de objetos a literatura apresenta sistemas RTCE?
QP2. Qual a arquitetura mais utilizada pelos pesquisadores para implementar sistemas RTCE , como eles são representados e quais trabalhos apresentam modelos ou roteiros para o desenvolvimento de RCEs?
QP3. Entre os algoritmos OT e CRDT, qual o que aparece com mais frequência nos trabalhos selecionados?
QP4. Quais os principais tipos de testes realizados para avaliar os editores colaborativos, quais os artigos realizaram testes com usuários e qual o número médio de usuários que realizaram os testes?
QP5. Quais trabalhos discutem técnicas de *undo/redo*?
QP6. Quais trabalhos discutem a utilização dos comentários, *chat* ou histórico como ferramentas de apoio ao trabalho colaborativo?

A seguir, detalhamos os aspectos específicos que buscamos investigar em nossa revisão.

QP1. *Para edição de quais tipos de objetos a literatura apresenta sistemas RTCE?* Buscamos identificar os tipos de objetos que são alvo de edições em tempo real nos sistemas colaborativos, como texto, código-fonte, gráficos, multimídia, entre outros.

QP2. *Qual a arquitetura mais utilizada pelos pesquisadores para implementar sistemas RTCE, como eles são representados e quais trabalhos apresentam modelos ou roteiros para o desenvolvimento de RCEs?* Analisamos quais arquiteturas são preferidas para a implementação de sistemas RTCE, como cliente-servidor, peer-to-peer, entre outras. Investigamos também como essas arquiteturas são representadas nos artigos e quais deles fornecem modelos ou roteiros detalhados para o desenvolvimento de editores colaborativos.

QP3. *Entre os algoritmos OT e CRDT, qual o que aparece com mais frequência nos trabalhos selecionados?* Verificamos a frequência de uso dos algoritmos de Transformação Operacional (OT) e Tipos de Dados Replicados sem Conflitos (CRDT) nos artigos, identificando qual deles é mais recorrente e em quais contextos são aplicados.

QP4. *Quais os principais tipos de testes realizados para avaliar os editores colaborativos, quais os artigos realizaram testes com usuários e qual o número médio de usuários que realizaram os testes?* Examinamos os tipos de testes empregados para avaliar a eficácia dos editores colaborativos, distinguindo entre testes de desempenho, usabilidade, robustez, entre outros. Procuramos identificar artigos que realizaram testes com usuários e a média de participantes envolvidos nesses estudos.

QP5. *Quais trabalhos discutem técnicas de undo/redo?* Investigamos quais artigos abordam a implementação de técnicas de desfazer/refazer (*undo/redo*) nos editores colaborativos, destacando as abordagens utilizadas e os desafios enfrentados.

QP6. *Quais trabalhos discutem a utilização dos comentários, chat ou histórico como ferramentas de apoio ao trabalho colaborativo?* Analisamos artigos que discutem a integração de funcionalidades

como comentários, *chat* e histórico nos editores colaborativos, avaliando como essas ferramentas contribuem para a eficácia do trabalho colaborativo.

Essas questões de pesquisa nos orientaram a focar nos aspectos mais relevantes e recorrentes do desenvolvimento de editores colaborativos em tempo real, garantindo uma análise abrangente e detalhada das práticas, algoritmos e inovações documentadas na literatura. Através dessas perguntas, buscamos compreender as metodologias e tecnologias envolvidas no desenvolvimento de sistemas RTCE.

Além disso, este estudo também se interessa em entender como se dá o processo de desfazer e refazer em sistemas colaborativos em tempo real, assim como em examinar como a literatura aborda os recursos de histórico, comentários e chat nesses sistemas, que muitas vezes são utilizados como suporte ao trabalho colaborativo.

3.1.3 Critérios de elegibilidade.

Apresentamos os critérios de inclusão dos estudos com base nos fatores PICO utilizados na definição da questão de pesquisa.

Population (P): Devem ser incluídos artigos que reportam modelos, algoritmos, técnicas ou formas de desenvolver editores colaborativos em tempo real, bem como aqueles que discutem formas originais ou customizadas para implementar recursos como histórico, desfazer/refazer, *chat*, comentários, etc.

Intervention (I): Devem ser incluídos artigos que apresentam novos editores colaborativos em tempo real, novas formas de implementação desses editores, ou novos recursos que possam ser aplicados aos mesmos, além de formas inovadoras de implementar esses recursos.

Comparison (C): Devem ser incluídos artigos que explicitam o desenvolvimento de editores colaborativos em tempo real, algoritmos e técnicas utilizados, implementação de novos recursos e novas formas de implementá-los. Preferencialmente, artigos que apresentam as avaliações realizadas e os dispositivos empregados.

Outcome (O): Devem ser incluídos artigos que explicitam técnicas, algoritmos, arquiteturas ou modelos empregados no desenvolvimento de editores colaborativos em tempo real, dando preferência àqueles que apresentam testes e resultados obtidos.

3.1.4 String de busca.

Considerando que as palavras necessárias ao estudo são comuns e utilizadas com frequência no texto acadêmico, restringimos a consulta ao resumo e ao título dos trabalhos. Portanto, a *string* de busca foi criada a identificar, no *abstract* e/ou no título, estudos que trouxessem as palavras edição ou editores, colaborativos em tempo real ou síncrono.

Além disso, não foi utilizado filtro para seleção de categorias (e.g. *research article* ou *short paper*).

Ainda, não foi incluída restrição quanto ao período de publicação dos trabalhos.

Assim, *string* de busca utilizada foi:

full query syntax: "query": Abstract:(edit*) AND Abstract:(synchron* OR "real-time") AND Abstract:(collaborat*) "filter": {}

A URL correspondente é como segue:
https://dl.acm.org/action/doSearch?fillQuickSearch=false&target=advanced&expand=all&field1=Abstract&text1=edit*&field2=Abstract&text2=synchron*+OR+%22real-time%22&field3=Abstract&text3=collaborat*

3.2 Processo de seleção dos estudos

A seleção dos estudos, sumarizada no fluxograma da Figura 1, ocorreu como detalhado a seguir.

A consulta foi realizada na base de dados *ACM Digital Library* com seleção de artigos estendida ao serviço *ACM Guide to Computing Literature* que engloba trabalhos da área de computação disponibilizados por outras editoras como IEEE, Springer-Verlag e Elsevier.

Destacamos que a base *The ACM Guide to Computing Literature* contém, na data de preparação deste manuscrito (27 de agosto de 2024) um total de 3.748.089 registros, dos quais 754.670 registros (20%) são publicados pela ACM (*The ACM Full-Text Collection*). Esse valores corroboram a afirmação da própria ACM de que esse é o banco de dados bibliográfico mais abrangente existente hoje focado exclusivamente no campo da computação.[3]

3.2.1 Identificação. A consulta foi realizada no dia 20/03/2024. Como resultado, foram obtidos 370 registros, dos quais 5 (cinco) foram identificados como duplicados e removidos.[4]

Uma verificação dos 365 registros obtidos depois da eliminação das duplicatas permitiu identificar as editoras responsáveis pela publicação dos trabalhos: ACM (162), IEEE (64), Springer-Verlag (41), Elsevier (12), Kluwer (7), Addison-Wesley (5), e os 74 trabalhos restantes foram publicados por outras editoras como Microsoft Press e Australian Computer Society.

A Figura 2 mostra a distribuição do ano de publicação dos 365 artigos. Nota-se que o período de maior ocorrência foi entre 2010 e 2017 e que, nos últimos cinco anos, o número de registros está entre quatro e dez por ano.

Utilizamos a ferramenta Rayyan[5] para apoiar a triagem de artigos e a extração de dados.

3.2.2 Seleção. Considerando os critérios de elegibilidade enumerados na Seção 3.1.3, a triagem correspondente à leitura dos títulos e dos resumos foi realizada paralelamente por dois pesquisadores, que avaliaram todos os registros.

A análise foi conduzida em modo cego, de forma que cada pesquisador realizou sua análise sem tomar conhecimento da decisão do outro. Ao final dessa etapa, o modo cego foi desativado. As divergências entre as classificações dos revisores foram arbitradas por uma terceira pessoa revisora. Do total de 365 trabalhos, foram selecionados 65 artigos completos.

Embora todos os tipos de materiais tenham sido analisados, a fase de análise resultou na seleção exclusivamente de artigos completos.

3.2.3 Elegibilidade. Na etapa de elegibilidade, dois pesquisadores fizeram a leitura completa de cada artigo e os classificaram de acordo com os critérios de elegibilidade. A seguir, as divergências foram sanadas pela terceira pessoa revisora. Além disso, nesta etapa foi possível identificar a língua utilizada nos artigos, e foram considerados artigos em inglês ou português.

3.2.4 Inclusão. Ao final foram identificados 23 artigos que atendiam aos critérios de elegibilidade. A Figura 3 apresenta a distribuição dos estudos ao longo do período entre 2002 e 2021, sendo que 2012 apresenta o maior número de publicações (4).

Uma síntese da análise desses 23 artigos, relativamente às questões de pesquisa, é apresentada na Seção 4.1. O resumo de cada um dos artigos é apresentado na Seção 4.2.

[3] https://libraries.acm.org/digital-library/acm-guide-to-computing-literature
[4] Observamos que a consulta realizada na data de conclusão deste manuscrito (27/08/2024), utilizando a URL informada, retorna 381 registros.
[5] https://rayyan.ai/

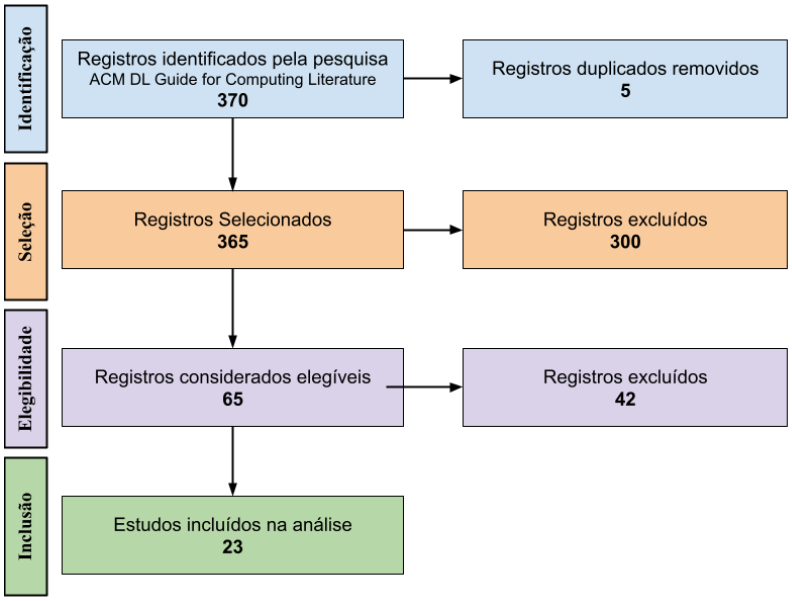

Figura 1: Síntese da estratégia de identificação dos estudos.

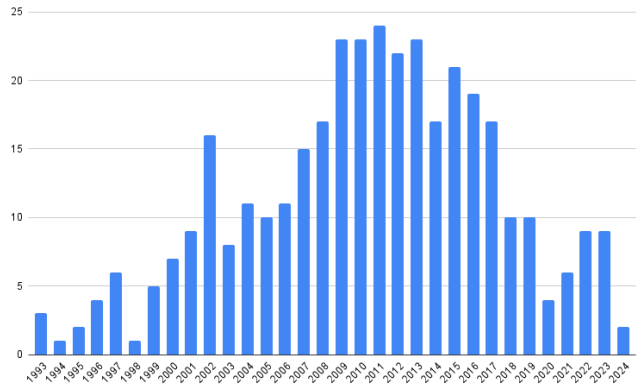

Figura 2: Distribuição dos 365 registros retornados por ano.

Figura 3: Distribuição dos 23 artigos selecionados por ano.

4 RESULTADOS

4.1 Síntese das respostas às questões da RR

A etapa de análise de cada artigo permitiu extrair as respostas às questões de pesquisa, sumarizadas nesta seção.

A Tabela 1 sumariza o tipo de objeto editado, de arquitetura utilizada, de algoritmo de sincronização adotado, bem como se o editor foi desenvolvido ou não como uma aplicação Web. A Figura 4 apresenta graficamente os resultados descritos na Tabela 1.

A Tabela 2 indica quais trabalhos contemplaram testes de modo geral, e os que realizaram testes com usuários. A Figura 5 apresenta graficamente os trabalhos que executaram testes.

A Tabela 3 especifica quais trabalhos oferecem quais recursos de edição. A Figura 6 apresenta os totais referentes aos trabalhos que exploraram os recursos de *undo/redo*, *chat*, histórico e comentários.

Para edição de quais tipos de objetos a literatura apresenta sistemas RTCE?

A análise dos estudos identificou editores para os seguintes tipos de objeto: texto [3, 31, 32, 39, 54, 71, 73], imagens [5] [20] [49], páginas web [24, 44, 45], diagramas UML [12, 66], modelos de recursos [30], modelo de software [41], objetos JSON [27], dados geográficos [17], documentos PDF [28], e documentos XML [72].

Qual a arquitetura mais utilizada pelos pesquisadores para implementar sistemas RTCE , como eles são representados e quais trabalhos apresentam modelos ou roteiros para o desenvolvimento de RCEs?

A arquitetura predominante nos trabalhos analisados foi a cliente-servidor, presente em 18 dos 23 estudos, o que representa 75% do total.

Tabela 1: Objeto editado, arquitetura empregada, algoritmos de sincronização (se informados) e plataforma web

Referência	Tipo de objeto editado	Arquitetura	OT / CRDT	Web
Bath et al. [5]	Raster e Vetor de imagem	Cliente-Servidor	OT	Sim
Kuiter et al. [30]	Modelos de Recursos (gráfico)	Cliente-Servidor	OT	Sim
Nicolaescu et al. [41]	Modelo de software	Centralizada e ponto a ponto	OT	Sim
Alsulami and Cherif [2]	Não especificado	Redes Oportunísticas	OT	Não
Gao et al. [20]	Imagens Bitmap	Ponto a Ponto	-	Não
Jungnickel and Herb [27]	Objetos JSON	Cliente-Servidor	OT	Sim
Nédelec et al. [39]	Documentos de texto	Descentralizado	CRDT	Não
Fechner et al. [17]	Dados geográficos	Cliente-Servidor	-	Sim
Salvati et al. [49]	Imagens Poligonais	Ponto a Ponto	-	Sim
Katayama et al. [28]	Documentos PDF	Cliente-Servidor	-	Sim
Inoue et al. [24]	Páginas WEB	Cliente-Servidor	-	Sim
Ozono et al. [45]	Páginas WEB	Cliente-Servidor	-	Sim
Ozono et al. [44]	Páginas WEB	Cliente-Servidor	-	Sim
Lautamäki et al. [31]	Código fonte Java (texto)	Cliente-Servidor	-	Sim
Thum et al. [66]	Diagramas UML	Cliente-Servidor	-	Sim
Wei et al. [71]	Textos que representam moléculas	Cliente-Servidor	-	Não
Bani-Salameh et al. [3]	Código fonte de software (texto)	Cliente-Servidor	-	Não
De Lucia et al. [12]	Diagramas UML	Cliente-Servidor	-	Não
Lin et al. [33]	Documentos Visio Microsoft	Cliente-Servidor	OT	Não
Leone et al. [32]	Documentos de texto	Cliente-Servidor	-	Não
Wong [72]	Documentos XML	Cliente-Servidor	-	Não
Xia et al. [73]	Documentos de texto	Cliente-Servidor	OT	Não
Shen and Sun [54]	Texto	Cliente-Servidor	-	Não

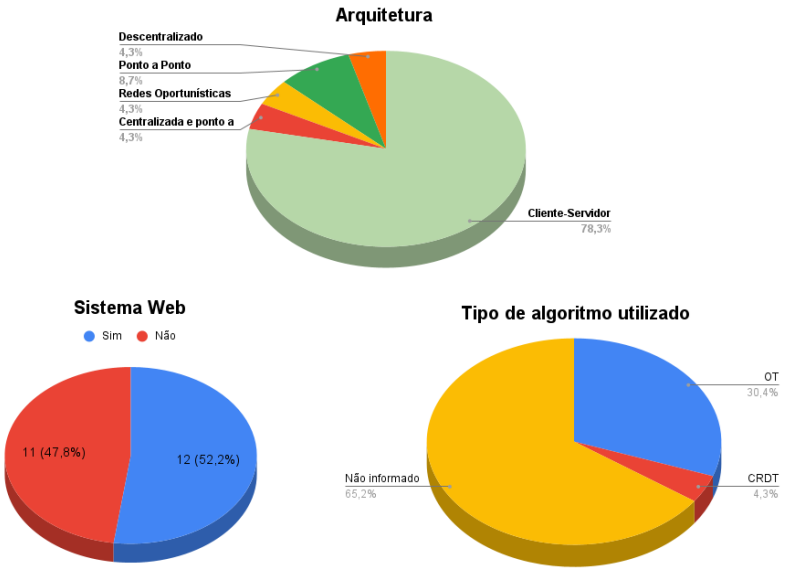

Figura 4: Arquitetura adotada, algoritmo de sincronização, e plataforma Web

A prevalência da arquitetura cliente-servidor pode ser explicada pelo fato de ser o modelo mais adotado na internet atualmente. Essa popularidade é reforçada pela estatística de que, em 2023, 86% dos desenvolvedores utilizaram APIs REST [47] que segue a arquitetura cliente-servidor [18].

Entre os algoritmos OT e CRDT, qual o que aparece com mais frequência nos trabalhos selecionados?

Os algoritmos de Transformação de Operações (OT) foram mais frequentemente utilizados do que os algoritmos de Estado Replicado (CRDT): 7 (sete) estudos utilizaram OT enquanto 2 (dois) utilizaram CRDT. Os demais estudos não especificaram o algoritmo de resolução de conflitos adotado. A preferência por OT pode ser atribuída ao fato de ser o método mais tradicional e melhor adaptado ao modelo cliente-servidor [19].

Tabela 2: Avaliação com ou sem envolvimento de usuários

Referência	Testes	Testes com usuários	Quantidade de usuários
Bath et al. [5]	Sim	Sim	27 + 16
Kuiter et al. [30]	Sim	Sim	17
Nicolaescu et al. [41]	Sim	Sim	20 + 16
Alsulami and Cherif [2]	Não	Não	-
Gao et al. [20]	Não	Não	-
Jungnickel and Herb [27]	Não	Não	-
Nédelec et al. [39]	Sim	Não	-
Fechner et al. [17]	Sim	Sim	39
Salvati et al. [49]	Sim	Sim	12
Katayama et al. [28]	Sim	Não	-
Inoue et al. [24]	Não	Não	-
Ozono et al. [45]	Sim	Não	5 (simulados)
Ozono et al. [44]	Sim	Não	Simulado de 3 a 30 com intervalos de 3
Lautamäki et al. [31]	Não	Não	-
Thum et al. [66]	Não	Não	-
Wei et al. [71]	Não	Não	-
Bani-Salameh et al. [3]	Não	Não	-
De Lucia et al. [12]	Não	Não	-
Lin et al. [33]	Não	Não	-
Leone et al. [32]	Não	Não	-
Wong [72]	Sim	Não	-
Xia et al. [73]	Não	Não	-
Shen and Sun [54]	Não	Não	-

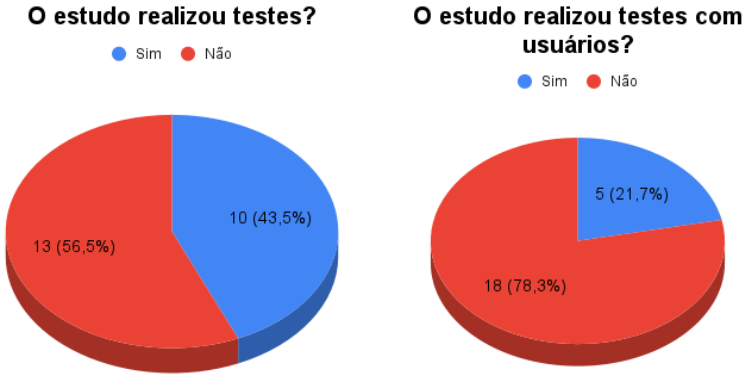

Figura 5: Testes gerais e testes com usuários.

Quais os principais tipos de testes realizados para avaliar os editores colaborativos, quais os artigos realizaram testes como usuários e qual o número médio de usuários que realizaram os testes?
Entre os 23 trabalhos selecionados, 10 realizaram algum tipo de testes, o que representa aproximadamente 42% dos trabalhos selecionados. Apenas 5 (cinco) entre os 23 trabalhos realizaram testes com usuários. Desses cinco trabalhos que realizaram testes com usuários, 2 (dois) reportaram testes preliminares. O maior número de usuários foi 39, no trabalho de Fechner et al. [17], e o número mínimo foi 12, no trabalho de Salvati et al. [49]. Além disso, durante a análise dos 23 trabalhos, não foi possível identificar um processo padronizado que definisse um modelo de teste para o desenvolvimento de um RCE.

Quais trabalhos discutem técnicas de *undo/redo*?
Dos 23 trabalhos incluídos, 3 (três) discutem explicitamente técnicas e recursos de *undo/redo*. No trabalho de Gao et al. [20] foram apresentados e discutidos os algoritmos BTMVIC e AnyUndo, ambos com o objetivo de resolver problemas de consistência nas operações de *undo/redo*. No trabalho de Shen and Sun [54] foi apresentado o algoritmo SUCA.

Quais trabalhos discutem a utilização dos comentários, *chat* ou histórico como ferramentas de apoio ao trabalho colaborativo?
De forma diversificada, 6 (seis) dos 23 trabalhos discutiram sobre recursos de colaboração utilizando *chat*, anotações ou histórico. No trabalho de Fechner et al. [17] foram verificadas as razões da utilização, ou não, do *chat* e do recurso de histórico como ferramenta de auxílio. No trabalho de Salvati et al. [49], foi permitido o compartilhamento do histórico de edição para auxílio aos outros usuários. Katayama et al. [28] registram o compartilhamento

Tabela 3: Recursos de edição oferecidos

Referência	*undo/redo*	*chat*, comentário e histórico
Bath et al. [5]	Não	Não
Kuiter et al. [30]	Não	Não
Nicolaescu et al. [41]	Não	Não
Alsulami and Cherif [2]	Não	Não
Gao et al. [20]	Algoritmos BTMVIC e AnyUndo	Não
Jungnickel and Herb [27]	Não	Não
Nédelec et al. [39]	Não	Não
Fechner et al. [17]	Não	*chat* e histórico[3]
Salvati et al. [49]	Não	Compartilhamento de históricos de edição
Katayama et al. [28]	Não	Compartilhamento das anotações e *chat* por texto
Inoue et al. [24]	Não	Não
Ozono et al. [45]	Não	*chat* por texto
Ozono et al. [44]	Não	Não
Lautamäki et al. [31]	Não	Não
Thum et al. [66]	Não	Não
Wei et al. [71]	Não	Não
Bani-Salameh et al. [3]	Não	*chat* com Voip
De Lucia et al. [12]	Não	Não
Lin et al. [33]	Não	Não
Leone et al. [32]	Sim	Não
Wong [72]	Não	Não
Xia et al. [73]	Não	Não
Shen and Sun [54]	Algoritmo SUCA	Destaques no documento

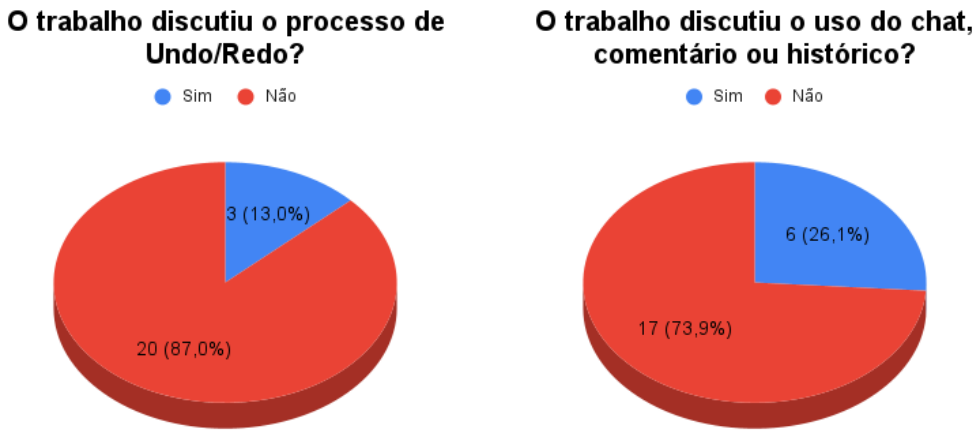

Figura 6: Undo/Redo, histórico, chat e comentários.

das anotações (comentários) e a uso de *chat* por texto. Ozono et al. [45] também reportam o uso de *chat* por texto. Bani-Salameh et al. [3] registram a disponibilização de um *chat* com recursos de VoIP (Voice over Internet Protocol). No editor reportado por Shen and Sun [54], era possível fazer destaques nos textos que eram compartilhados com outros usuários.

4.2 Resumo dos artigos selecionados

O trabalho de **Bath et al. [5]** centra-se na concepção e na implementação de um aplicativo web para edição colaborativa em tempo real de imagens *raster* e vetoriais. Para compreender melhor os requisitos dos usuários, foi realizada uma avaliação preliminar, identificando comunicação e sincronização como elementos essenciais. O sistema desenvolvido utiliza um modelo de documento centralizado, mantido por um servidor que sincroniza as alterações com múltiplos clientes. A implementação prototípica é baseada em uma arquitetura cliente-servidor escalável, utilizando WebGL para renderização interativa no navegador e *WebSocket* para manter a sincronização em tempo real. A sincronização é gerenciada de forma centralizada, com um sistema de bloqueio que permite apenas a um usuário editar uma camada

por vez. As atualizações são transmitidas por *broadcast* e serializadas para garantir a consistência. O trabalho foi avaliado qualitativamente através de um estudo como usuários.

O variED é um editor para modelagem de características (*feature modeling*) colaborativa e em tempo real apresentado por **Kuiter et al. [30]**. O editor foi desenvolvido para permitir que múltiplos usuários trabalhem simultaneamente na criação e modificação de modelos de características, que são essenciais para o desenvolvimento de software orientado a linha de produtos. A ferramenta aborda desafios críticos, como a manutenção da consistência do modelo e a sincronização das alterações feitas por diferentes usuários em tempo real. A arquitetura do variED é baseada em uma infraestrutura distribuída que utiliza algoritmos de OT para gerenciar as operações concorrentes, garantindo que todas as edições sejam integradas de forma consistente sem conflitos. Os autores conduziram uma série de testes experimentais e estudos de caso para avaliar o desempenho do variED, demonstrando que a ferramenta é capaz de suportar um grande número de usuários simultâneos sem comprometer a integridade e a consistência do modelo. A análise dos resultados mostra que variED melhora a eficiência e a produtividade do processo de modelagem de características, e também proporciona uma experiência de usuário fluida e interativa.

Nicolaescu et al. [41] abordam a necessidade de colaboração eficiente entre *stakeholders* geograficamente distribuídos em ambientes de modelagem conceitual. Para atender a essa necessidade, os autores introduzem a SyncMeta, uma ferramenta de modelagem colaborativa que opera em navegadores web, permitindo a edição compartilhada em tempo quase real. SyncMeta é projetada para suportar a modelagem baseada em visões, na qual diferentes perspectivas de *stakeholders* podem ser integradas e negociadas de forma eficaz. A ferramenta permite que *stakeholders* contribuam simultaneamente, minimizando conflitos e inconsistências nas operações de modelagem. A ferramenta permite a rápida geração de protótipos a partir de especificações, facilitando a análise de impacto e a tomada de decisões. Os estudos conduzidos pelos autores investigaram tanto os requisitos fundamentais da ferramenta quanto opções de design. Entre os requisitos fundamentais avaliados, os autores destacam a aceitação da abordagem de modelagem baseada em visões. As opções de design exploradas no trabalho incluem a resolução de pontos de vista de maneira centralizada versus ponto-a-ponto, considerando o compromisso entre desempenho e complexidade de implementação. Os resultados dos estudos indicam que a SyncMeta oferece uma solução robusta para os desafios de colaboração em ambientes de modelagem distribuídos, melhorando a eficiência e a eficácia das equipes.

Alsulami and Cherif [2] exploram a viabilidade e os desafios da edição colaborativa em redes oportunísticas (ON), na qual a conectividade entre dispositivos é intermitente e imprevisível, focalizando na adaptação de algoritmos baseados em OT para RTCE em ON. São discutidos os principais desafios como alta mobilidade, dinâmica dos nós e atrasos de rede, e revisadas as versões dos algoritmos OT segundo critérios relevantes para ON. O artigo oferece uma visão abrangente para o desenvolvimento e avaliação de editores colaborativos em ambientes ON. O artigo também aborda as estratégias para sincronização de dados em tempo real, adaptando-os à natureza assíncrona e intermitente das comunicações em redes oportunísticas, e métodos são propostos para reconciliar versões divergentes de documentos editados por dispositivos desconectados. Resultados experimentais são apresentados, demonstrando o desempenho das abordagens propostas em cenários simulados e reais, com análises quantitativas de consistência de dados, tempo de sincronização e eficiência operacional.

Nédelec et al. [39] apresentam o desenvolvimento e a implementação de um sistema de edição colaborativa em tempo real, chamado CRATE, que é um editor colaborativo descentralizado em tempo real que funciona diretamente em navegadores da web utilizando o WebRTC. CRATE foi um editor em tempo real pioneiro na utilização apenas de navegadores para suportar a edição colaborativa, manejando de forma transparente grupos pequenos e grandes de usuários. As propriedades do CRATE dependem de dois avanços científicos principais: 1) uma estrutura de sequência replicada com limite superior sublinear na complexidade do espaço, o que evita a necessidade de coletores de lixo distribuídos dispendiosos; 2) um protocolo adaptativo de amostragem por pares, que evita o superdimensionamento de tabelas de roteamento, permitindo que redes pequenas não paguem o preço das redes grandes.

Jungnickel and Herb [27] ampliam o escopo do algoritmo de controle de consistência OT para a edição simultânea de objetos JSON compartilhados. A pesquisa detalha as estruturas de dados e as mecânicas subjacentes necessárias para suportar a edição colaborativa de objetos JSON, fornecendo uma base teórica robusta para desenvolvedores de aplicações web. Além disso, o artigo discute o design de aplicativos web que utilizam essa extensão da OT, destacando as vantagens em termos de versatilidade e robustez na colaboração em tempo real. Esta extensão da OT para objetos JSON abre novas possibilidades, permitindo a edição colaborativa de uma gama mais ampla de dados estruturados, desde configurações de software até dados de aplicações complexas. A implementação desta técnica oferece benefícios significativos para desenvolvedores e usuários, criando um ambiente colaborativo mais flexível e poderoso. A capacidade de editar simultaneamente objetos JSON em tempo real pode transformar a maneira como dados estruturados são manipulados colaborativamente, promovendo uma maior eficiência e inovação no desenvolvimento de software e em outras áreas que dependem de dados complexos.

O trabalho de **Gao et al. [20]** foca nos desafios de garantir que múltiplos usuários possam editar simultaneamente uma imagem *bitmap* sem causar inconsistências ou conflitos nas alterações. Para resolver esses desafios, os autores propõem um modelo que integra algoritmos de OT para gerenciar e sincronizar as operações de edição realizadas por diferentes usuários. Este modelo permite que as operações de *undo* e *redo* sejam tratadas de forma consistente, mantendo a integridade do estado do *bitmap* ao longo do tempo, o que é demonstrado através de análises e um protótipo de sistema chamado CoGraphical Editor. Estudos de caso e simulações reforçam a viabilidade do sistema em cenários de uso real, destacando melhorias significativas na precisão e na eficiência das operações de edição colaborativa.

Fechner et al. [17] apresentam uma abordagem para a edição colaborativa em tempo real de dados geográficos, exemplificada pelo Ethermap. O sistema Ethermap foi desenvolvido para permitir que múltiplos usuários possam editar simultaneamente mapas geográficos, garantindo a sincronização imediata e precisa das alterações realizadas. A arquitetura do Ethermap utiliza tecnologias avançadas de comunicação e OT para gerenciar operações concorrentes, assegurando que todas as edições sejam integradas de maneira consistente. O sistema é projetado para suportar tanto a criação de novos mapas quanto a edição de mapas existentes, oferecendo uma interface intuitiva que facilita a colaboração entre os usuários. Estudos de caso são apresentados para demonstrar a eficácia do sistema em melhorar a colaboração e a produtividade dos usuários durante a manipulação e a atualização de informações geoespaciais.

Salvati et al. [49] propõem o editor MeshHisto como uma abordagem inovadora para a modelagem colaborativa de objetos 3D, centrada no compartilhamento e redirecionamento de históricos de edição. MeshHisto permite que múltiplos usuários colaborem em projetos de modelagem 3D, compartilhando seus históricos de edição de forma granular e reutilizável. A metodologia proposta utiliza uma estrutura de dados eficiente para capturar e armazenar as operações de edição, permitindo que estas possam ser reaplicadas ou adaptadas a diferentes modelos 3D. Isso facilita a transferência de técnicas e estilos de modelagem entre colaboradores, promovendo uma integração harmoniosa e eficiente do trabalho em equipe. A arquitetura do sistema MeshHisto incorpora mecanismos para resolver conflitos de edição e manter a consistência dos modelos em cenários de edição concorrente. Estudos de caso demonstram a eficácia do sistema em diversos contextos de modelagem, evidenciando melhorias significativas na produtividade e

na qualidade dos modelos produzidos. Além disso, a análise empírica dos resultados revela que o compartilhamento de históricos de edição não apenas acelera o processo de modelagem, mas também enriquece a experiência colaborativa ao possibilitar a aprendizagem e a inspiração mútua entre os usuários.

Katayama et al. [28] reportam uma aplicação web colaborativa para editar documentos PDF usando navegadores. O sistema permite que os usuários editem o mesmo documento em tempo real e compartilhem anotações em documentos simultaneamente. Os autores propõem um mecanismo de sincronização eficiente para aplicações web colaborativas utilizando as capacidades avançadas do HTML5. Este mecanismo visa resolver desafios de latência e consistência que surgem quando múltiplos usuários interagem simultaneamente com uma aplicação web. A abordagem se baseia na utilização de *WebSockets* para comunicação bidirecional em tempo real entre clientes e servidores, permitindo atualizações instantâneas e redução de atrasos. O trabalho detalha a arquitetura do sistema, que incorpora algoritmos de OT para gerenciar as operações concorrentes e assegurar que todas as mudanças feitas pelos usuários sejam integradas de maneira consistente. A infraestrutura inclui um servidor central que coordena as operações de sincronização, garantindo que todas as instâncias da aplicação web permaneçam atualizadas.

Lautamäki et al. [31] detalham o desenvolvimento e a implementação de CoRED, um editor colaborativo em tempo real baseado em navegador para aplicações web Java. O CoRED foi projetado para permitir que múltiplos desenvolvedores editem simultaneamente o código fonte de aplicações Java diretamente em um navegador web, promovendo uma colaboração eficiente e integrada. O CoRED é um editor Java completo que inclui verificação de erros e recursos de geração automática de código, além de ser complementado por funcionalidades comumente associadas às mídias sociais. A arquitetura do CoRED incorpora tecnologias avançadas como *WebSockets* para comunicação bidirecional em tempo real, garantindo a sincronização imediata das edições feitas por diferentes usuários. Além disso, o sistema utiliza algoritmos de OT para gerenciar operações concorrentes, assegurando que todas as modificações sejam aplicadas de maneira consistente e sem conflitos. A ferramenta foi testada em diversos cenários de desenvolvimento colaborativo, demonstrando melhorias significativas na produtividade e eficiência das equipes de desenvolvimento.

Inoue et al. [24] apresentam o WFE, um RCE que permite a um grupo de pessoas editarem uma página web em um navegador compartilhando o conteúdo de edição em tempo real. Além disso, os autores também propõem a aplicação do WFE para um ambiente de computação em nuvem, chamada WFE-S, com o objetivo de melhorar a escalabilidade, a elasticidade, a capacidade de resposta, bem como de reduzir a necessidade de manutenção. A arquitetura do sistema é baseada em uma combinação de algoritmos de OT e técnicas de controle de versão, que juntos asseguram que as edições concorrentes sejam integradas sem conflitos. Os autores detalham a infraestrutura técnica, incluindo o uso de servidores intermediários para gerenciar as comunicações entre os clientes e assegurar a integridade das operações de edição. Os estudos experimentais realizados demonstram a eficácia do sistema, mostrando que ele é capaz de lidar com um grande número de usuários simultâneos sem degradação significativa de desempenho. Além disso, o sistema inclui funcionalidades como a visualização das edições em tempo real e a capacidade de desfazer e refazer mudanças, oferecendo uma experiência de usuário rica e interativa. A análise dos resultados indica que o mecanismo proposto melhora a eficiência da colaboração em tempo real e aumenta a produtividade e a qualidade do trabalho colaborativo.

Ozono et al. [44] também apresentam o ambiente WFE-S para edição colaborativa de páginas web em tempo real. Além de detalhar a infraestrutura e os algoritmos, os autores realizaram uma série de testes experimentais e estudos de caso para avaliar o desempenho do WFE-S, demonstrando que o sistema é altamente eficiente em termos de latência e escalabilidade. Os resultados mostram que o WFE-S mantém a consistência dos dados e proporciona uma experiência de usuário suave, mesmo sob alta carga de usuários simultâneos.

Thum et al. [66] descrevem o desenvolvimento e a implementação do SLIM, um ambiente projetado para suportar a modelagem colaborativa síncrona. O sistema SLIM visa facilitar a colaboração em tempo real entre múltiplos usuários na criação e edição de modelos, com foco especial em modelos de sistemas de software. A arquitetura do SLIM incorpora tecnologias que permitem a sincronização imediata de alterações feitas por diferentes participantes, garantindo consistência e integridade do modelo em tempo real. O ambiente utiliza técnicas avançadas de OT para gerenciar operações concorrentes de maneira eficiente, minimizando conflitos e assegurando que todas as alterações sejam devidamente integradas. O artigo também discute estudos de caso e experimentos realizados para avaliar o desempenho e a eficácia do SLIM em cenários reais de uso. Os resultados demonstram que o ambiente é capaz de suportar uma colaboração eficiente e produtiva, promovendo uma interação fluida entre os participantes durante o processo de modelagem.

Wei et al. [71] abordam o desenvolvimento e a implementação de um ambiente colaborativo de edição científica focado na química. O sistema foi projetado para facilitar a colaboração entre cientistas e pesquisadores na criação e edição colaborativa de documentos científicos relacionados à química. A arquitetura do ambiente colaborativo incorpora funcionalidades específicas para apoiar a edição em tempo real de documentos científicos complexos, incluindo formulações químicas, estruturas moleculares, diagramas e textos técnicos especializados. O ambiente oferece recursos como controle de versões, que permitem rastrear e gerenciar alterações feitas por diferentes colaboradores ao longo do tempo. Além disso, são implementadas ferramentas para suportar a análise e a visualização de dados químicos, facilitando a comunicação e a colaboração entre os membros da equipe de pesquisa. O artigo também destaca estudos de caso e experimentos práticos realizados para validar a eficácia e a usabilidade do ambiente colaborativo em cenários reais de pesquisa química. Os resultados demonstram que o sistema é capaz de melhorar a eficiência e a produtividade dos pesquisadores, ao mesmo tempo que promove uma colaboração mais integrada e eficaz no desenvolvimento de documentos científicos na área da química.

Bani-Salameh et al. [3] registram o design e a implementação de uma de um IDE colaborativo chamado ICI (Idaho Collaborative IDE) que permite que desenvolvedores em diferentes locais colaborem em uma variedade de atividades de desenvolvimento de software em tempo real. O ICI combina um editor de programa colaborativo síncrono e um depurador colaborativo em tempo real em um ambiente virtual multiusuário 3D. O ICI permite que desenvolvedores compartilhem, em tempo real, o processo de edição, compilação, execução e depuração de seus projetos de software. A arquitetura do ICI é composta por quatro componentes principais: 1) um editor colaborativo 2) um ambiente colaborativo shell 3) um conjunto de ferramentas de comunicação, como *chat* de texto e voz, e 4) uma interface para colaboração controle. A contribuição deste trabalho é uma IDE colaborativa que integra edição colaborativa responsiva e em tempo real e depuração. Os resultados de testes de usabilidade e estudos de caso indicam que o ICI melhora a produtividade e a eficiência da equipe, reduzindo o tempo necessário para detectar e corrigir erros.

De Lucia et al. [12] apresentam e ferramenta STEVE, destinada à modelagem colaborativa síncrona com gerenciamento de versionamento, que suporta a modelagem distribuída em UML de sistemas de software. STEVE permite que desenvolvedores distribuídos editem simultaneamente o mesmo diagrama UML, decompostos em sub-artefatos gerenciados hierarquicamente, oferecendo gerenciamento de alterações e configurações tanto para diagramas quanto para objetos gráficos. Isso possibilita a reutilização e o compartilhamento consistentes de componentes de diagramas em diferentes projetos. Integrada ao sistema ADAMS, que oferece funcionalidades refinadas de gerenciamento de artefatos e compartilhamento de diagramas, a

ferramenta destaca a importância do controle preciso de versões. Ela incorpora mecanismos de versionamento que rastreiam alterações nos elementos individuais do modelo, promovendo a sincronização eficiente e a resolução de conflitos. A arquitetura do sistema, que combina ferramentas de modelagem UML com um robusto sistema de controle de versões, permite operações colaborativas em tempo real. Comparações com métodos tradicionais de versionamento demonstram melhorias significativas em precisão e eficiência na gestão de alterações concorrentes. Estudos de caso e testes empíricos validam a eficácia da ferramenta, mostrando que ela facilita a colaboração contínua e a integração de mudanças de maneira fluida.

Leone et al. [32] reportam TeNDaX, um editor colaborativo em tempo real, baseado em um sistema de banco de dados, conforme descrito no trabalho de . O editor armazena documentos, incluindo conteúdo, estrutura, tabelas e imagens, em um banco de dados semiestruturado, o que facilita a edição colaborativa e o layout dos documentos. Além disso, o TeNDaX oferece funcionalidades avançadas como operações de desfazer e refazer, definição e execução de processos de negócios, e recursos de segurança e conscientização do contexto. Durante a criação e uso dos documentos, metadados são coletados automaticamente, permitindo a criação de pastas dinâmicas, rastreamento da proveniência dos dados, e a realização de mineração e pesquisa visual e de texto. A plataforma é comparada a uma "LAN-Party"de processamento de texto, suportando múltiplos editores e sistemas operacionais diferentes. Esse ambiente colaborativo permite a execução de operações locais e globais de desfazer e refazer, facilitando a edição em tempo real. Além disso, o TeNDaX demonstra a utilização eficiente de dados e metadados para criar pastas dinâmicas, visualizar a proveniência dos dados, realizar mineração visual de texto e oferecer funcionalidades de pesquisa avançadas. O sistema também destaca a extensão do banco de dados para gerenciar texto, mostrando como a integração de dados e metadados pode aprimorar significativamente a experiência de edição colaborativa em tempo real.

Wong [72] estudam como múltiplos usuários podem colaborar de forma eficaz na criação e edição de documentos hipertextuais usando dispositivos móveis, considerando as limitações típicas desses ambientes, como conectividade intermitente e recursos limitados. Os autores apresentam um protótipo de sistema de edição colaborativa com um mecanismo para sincronizar atualizações de editores simultâneos, integrado ao sistema de gerenciamento de banco de dados XML nativo SODA. A pesquisa discute diversas estratégias e tecnologias para facilitar a colaboração em tempo real, incluindo o uso de protocolos de comunicação eficientes, algoritmos de reconciliação de conflitos e interfaces de usuário adaptadas para dispositivos móveis. São exploradas técnicas para garantir a consistência e a sincronização das edições realizadas por diferentes usuários, mesmo em condições adversas de rede e mobilidade. O artigo também apresenta estudos de caso e experimentos práticos para avaliar a viabilidade e a eficácia das soluções propostas. Resultados experimentais são discutidos para demonstrar a performance e a usabilidade do sistema em cenários reais de uso, destacando melhorias na colaboração e na produtividade dos usuários móveis durante a edição de documentos hipertextuais.

Xia et al. [73] investigam a transformação de aplicações de usuário único em plataformas colaborativas multiusuário através da abordagem CoWord. Os autores apresentam um trabalho com o objetivo central de adaptar o Microsoft Word, uma aplicação originalmente concebida para uso individual, para suportar a edição colaborativa em tempo real. A metodologia empregada envolve a integração de um mecanismo de sincronização que permite a múltiplos usuários editar simultaneamente o mesmo documento, mantendo a consistência e a integridade das alterações. Utilizando técnicas de OT e algoritmos de resolução de conflitos, o CoWord assegura que as edições concorrentes sejam harmonizadas eficientemente. O artigo descreve a arquitetura do sistema CoWord, destacando componentes essenciais como o servidor de sincronização, os clientes distribuídos e o protocolo de comunicação utilizado para garantir uma colaboração fluida. Resultados experimentais são apresentados para demonstrar a eficácia do CoWord, evidenciando melhorias significativas na produtividade e na experiência colaborativa dos usuários. Estudos de caso exemplificam a aplicabilidade da abordagem em diversos contextos profissionais e educacionais, realçando a versatilidade e a robustez da solução proposta.

Shen and Sun [54], considerando oferta do recurso de destacar texto de modo síncrono, identificaram três necessidades principais: 1) diferenciar destaques feitos por diferentes usuários; 2) resolver problemas de inconsistência causados por operações simultâneas; 3) oferecer uma função de desfazer flexível, permitindo reverter qualquer operação de destaque a qualquer momento. Para atender a essas necessidades, foi desenvolvido o sistema REDUCE (Real-time Distributed Unconstrained Collaborative Editing). O artigo apresenta soluções específicas, incluindo os algoritmos de transformação IT_EH e IT_HH, além do algoritmo de controle OSCA, que resolvem problemas de divergência causados por operações de destaque sobrepostas. Adicionalmente, o algoritmo GOTO foi estendido para controlar a transformação de operações de destaque em relação às operações de edição. A função de desfazer foi aprimorada com os algoritmos de transformação IT_HU e o algoritmo de controle SUCA, permitindo desfazer seletivamente qualquer operação de destaque a qualquer momento. Resultados experimentais, obtidos através de estudos de caso e de testes de usabilidade, demonstram que o uso do *highlighting* melhora significativamente a compreensão mútua, reduz o tempo de resolução de tarefas colaborativas e aumenta a satisfação dos usuários. A análise quantitativa e qualitativa dos dados coletados sustenta a eficácia da ferramenta em diversos cenários de aplicação, incluindo edição de documentos, design gráfico e programação em pares.

5 RISCO À VALIDADE

Em relação aos resultados relatados nos artigos, o principal viés está associado à fase inicial e exploratória de muitos dos estudos, dado que apenas 10 dos estudos apresentam algum tipo de avaliação e, dentre eles, apenas 5 (cinco) relatam estudos com a participação de usuários.

No viés associado à realização da revisão, os principais riscos e as ações tomadas para minimizá-los foram:

(1) A pesquisa foi restrita a trabalhos classificados como sendo da área de Computação pela ACM. Não foram tomadas ações neste caso pois considerou-se que os principais veículos estão indexados;
(2) A *string* de busca restringiu a pesquisa aos campos de título e resumo. Não foram tomadas atitudes porque considerou-se que os trabalhos mais relevantes foram identificados;
(3) A distinta experiência dos 3 avaliadores. Para diminuir o risco de viés associado, a seleção foi realizada individualmente por dois avaliadores de forma que um não soubesse as escolhas do outro, e divergências foram definidas por uma pessoa árbitra.

6 DESAFIOS DE PESQUISA

Os resultados obtidos com esta revisão fornecem uma base para o desenvolvimento futuro de sistemas com suporte a RTCE. Os estudos identificados sugerem múltiplas vias para alcançar eficiência e eficácia na edição colaborativa em tempo real, cujos temas são reportados na literatura recente.

Testes e Avaliação com Usuários: Realizar testes abrangentes é um desafio fundamental para a pesquisa em RTCE. A maioria dos estudos ainda é limitada em termos de número de usuários e de cenários avaliados. Garantir a condução de estudos que envolvam um número apropriado de participantes, participantes com diferentes papéis e em ambientes variados, como no estudo de de Lange et al. [11], pode proporcionar uma compreensão mais completa dos desafios enfrentados pelos usuários e das características desejadas nos sistemas colaborativos.

Algoritmos de resolução de conflitos: Estudos ainda são necessários neste tema dado que, apesar das análises de Sun et al. [62, 64] explicitarem as vantagens de OT em relação CRDT, pesquisadores continuam a advogar pelo uso de CRDT. Por exemplo, Kleppmann [29] apresenta um algoritmo para operações de movimentação de elementos em CRDTs de listas, e Litt et al. [34] tratam da edição de texto rico. Outros exemplos são os frameworks *YJS* [26], para compartilhamento de dados, e Hocuspocus [25], para edição colaborativa.

Diversidade de Objetos Editáveis e ambientes heterogêneos: Um desafio significativo na pesquisa de sistemas RTCE é expandir a capacidade desses sistemas para suportar uma gama mais ampla de objetos editáveis. Atualmente, muitos sistemas são otimizados para tipos de objetos relativamente simples, como textos e planilhas. No entanto, a demanda está crescendo por sistemas que possam lidar com modelos gráficos complexos e diagramas interativos, entre outros [11, 29, 34]. Desenvolver abordagens que garantam a consistência visual e a integridade dos dados para esses tipos de objetos apresenta uma oportunidade para avanços significativos na área. Outro desafio é a construção de sistemas de co-edição heterogêneos, permitindo que múltiplos usuários utilizem editores diferentes para editar documentos compartilhados na mesma sessão [8]. Essa abordagem destaca a necessidade de maior flexibilidade e adaptabilidade nos sistemas de edição colaborativa, evidenciando a importância de suportar uma diversidade maior de objetos e editores para melhorar a experiência colaborativa.

Edição Colaborativa de Texto Rico: A edição colaborativa de texto rico, que inclui formatação e outras características visuais, apresenta desafios específicos em comparação com a edição de texto simples. Por exemplo, Litt et al. [34] descrevem um algoritmo CRDT para edição colaborativa de texto rico. A estratégia armazena *spans* de formatação ao lado da sequência de caracteres do texto e deriva o texto formatado final de forma determinística para que operações concorrentes sejam comutativas.

Edição Colaborativa de Vídeo: O design de ferramentas para edição colaborativa de vídeo também levanta questões importantes e oportunidades de pesquisa. Por exemplo, Okopnyi et al. [42] exploram as complexidades do design para edição colaborativa de vídeo, destacam a necessidade de integrar recursos colaborativos específicos para software de edição não-linear, e discutem os desafios associados à introdução desses recursos. A pesquisa sugere que, além dos recursos colaborativos convencionais, pode ser necessário desenvolver representações abstratas alternativas para mídias baseadas em tempo para facilitar a colaboração eficaz na edição de vídeo.

Integração de Modelagem e Edição Colaborativa em Processos de Desenvolvimento: Além dos avanços na edição colaborativa de textos e vídeos, a integração de técnicas colaborativas em processos de desenvolvimento web também apresenta desafios. Por exemplo, de Lange et al. [11] investigam a integração de modelagem colaborativa e edição de código em um processo de engenharia web orientado a modelos (MDWE). O estudo propõe uma abordagem que combina edição ao vivo e *wireframing* com modelagem colaborativa em tempo real, abordando a sincronização entre código-fonte, *wireframes* e modelos.

Implementação de *Undo*/*Redo*: A implementação de operações de *undo* e *redo* continua a ser um desafio de pesquisa no contexto de edição colaborativa em tempo real, em particular no contexto de objetos complexos e interativos. Sun [60] propõe uma solução para operações de *undo* em editores colaborativos, permitindo desfazer qualquer operação a qualquer momento, independentemente do contexto de *undo*. A solução usa OT para garantir que uma operação possa ser desfeita, suportando múltiplos modos de *undo* (único, cronológico e seletivo) na mesma sessão. Já Stewen and Kleppmann [58] utilizam CRDT para implementar um novo algoritmo que implementa *undo*/*redo* de acordo com a semântica modelada pelos autores a partir de um conjunto de ambientes de edição investigados.

Representação e Diagramação: Outro desafio importante é o desenvolvimento de técnicas e de ferramentas de representação que capturem de forma eficaz as interações dinâmicas e o comportamento em tempo real dos sistemas RTCE. Inovar na criação de diagramas e modelos que possam representar adequadamente essas interações é crucial para melhorar a compreensão e o gerenciamento desses sistemas. Por exemplo, Yu and Lu [74] propõem uma ferramenta de discussão colaborativa adaptada para ambientes MOOC, utilizando de um esquema de *Address Space Transformation* para simular a interação entre usuários históricos e atuais. Essa abordagem pode inspirar novas maneiras de representar interações dinâmicas em sistemas RTCE, especialmente em ambientes educacionais.

Integração de Comentários, Chat e Histórico: Melhorar a integração e a gestão de ferramentas de apoio colaborativo, como comentários, chat e histórico de edições, representa um desafio significativo, entre outros, relativamente à interação. No contexto da edição de imagens, por exemplo, Bath et al. [4] optaram por posicionar a janela de troca de mensagens sobre a imagem sendo editada. Desenvolver métodos que otimizem o uso dessas ferramentas e avaliem seu impacto na produtividade e na eficácia do trabalho em equipe são relevantes para permitir integração sem comprometer a interação dos usuários nem o desempenho do sistema.

Colaboração Humano-Computador: A colaboração entre humanos e computadores na edição pode representar um avanço significativo na eficácia e eficiência dos sistemas RTCE. O trabalho de Pan et al. [46] introduz uma ferramenta de edição colaborativa que divide a tarefa entre o humano e o computador, permitindo que cada um se concentre em suas especialidades. Explorar e desenvolver ferramentas que integrem a colaboração humano-computador pode reduzir o esforço necessário para operações complexas e melhorar a eficiência na edição colaborativa.

Inclusão de Usuários com Deficiências: Um desafio constante na pesquisa em RTCE é garantir que os sistemas de edição colaborativa sejam acessíveis e eficazes para pessoas com deficiências. O trabalho de Akter et al. [1] revela as barreiras enfrentadas por facilitadores de reuniões com deficiência visual ao usar ferramentas de videoconferência, destacando a necessidade de interfaces mais inclusivas e adaptáveis. Investigar como as tecnologias de edição colaborativa podem ser projetadas para atender às necessidades específicas de diferentes grupos de usuários é crucial para promover a equidade e a acessibilidade no ambiente colaborativo.

7 CONSIDERAÇÕES FINAIS

No presente trabalho apresentamos uma Revisão Rápida que buscou identificar pesquisas que descrevessem o desenvolvimento de sistemas com suporte a RTCE, bem como algoritmos ou técnicas importantes para o seu desenvolvimento.

Os dados extraídos desses artigos foram organizados de maneira a destacar aspectos importantes como o tipo de objeto editado, a arquitetura adotada, os algoritmos de sincronização utilizados, os tipos de avaliações realizadas, e os recursos de edição oferecidos. A análise revelou diversas abordagens e técnicas empregadas na implementação de RTCE, proporcionando uma visão abrangente das pesquisas nessa área, e permitiu identificar desafios para pesquisas futuras.

AGRADECIMENTOS

Agradecemos as pessoas revisoras pelas valiosas sugestões que contribuíram para o aprimoramento do conteúdo e da apresentação deste trabalho.

REFERÊNCIAS

[1] Taslima Akter, Yoonha Cha, Isabela Figueira, Stacy M. Branham, and Anne Marie Piper. 2023. "If I'm supposed to be the facilitator, I should be the host": Understanding the Accessibility of Videoconferencing for Blind and Low Vision Meeting Facilitators. In *Proceedings of the 25th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '23)*. ACM, Article 45, 14 pages. https://doi.org/10.1145/3597638.3608420

[2] Noha Alsulami and Asma Cherif. 2017. Collaborative editing over opportunistic networks: State of the art and challenges. *International Journal of Advanced Computer Science and Applications* 8, 11 (2017).

[3] Hani Bani-Salameh, Clinton Jeffery, Ziad Al-Sharif, and Iyad Abu Doush. 2008. Integrating collaborative program development and debugging within a virtual environment. In *Groupware: Design, Implementation, and Use: 14th International Workshop, CRIWG 2008, 2008, Revised Selected Papers 14*. Springer, 107–120.

[4] Ulrike Bath, Sumit Shekhar, Jürgen Döllner, and Matthias Trapp. 2021. Colier: Collaborative editing of raster images. In *2021 International Conference on Cyberworlds (CW)*. IEEE, 33–40.

[5] Ulrike Bath, Sumit Shekhar, Julian Egbert, Julian Schmidt, Amir Semmo, Jürgen Döllner, and Matthias Trapp. 2022. CERVI: collaborative editing of raster and vector images. *The Visual Computer* 38, 12 (2022), 4057–4070.

[6] Zane L Berge. 1995. Facilitating computer conferencing: Recommendations from the field. *Educational technology* 35, 1 (1995), 22–30.

[7] Asma Cherif. 2012. *Access control models for collaborative applications*. Ph. D. Dissertation. Université de Lorraine.

[8] Bryden Cho, Chengzheng Sun, and Agustina Ng. 2019. Issues and Experiences in Building Heterogeneous Co-Editing Systems. *Proc. ACM Hum.-Comput. Interact.* 3, GROUP, Article 245 (dec 2019), 28 pages. https://doi.org/10.1145/3361126

[9] Gabriele D'Angelo, Angelo Di Iorio, and Stefano Zacchiroli. 2018. Spacetime Characterization of Real-Time Collaborative Editing. *Proc. ACM Hum.-Comput. Interact.* 2, CSCW, Article 41 (nov 2018), 19 pages. https://doi.org/10.1145/3274310

[10] Gabriele d'Angelo, Angelo Di Iorio, and Stefano Zacchiroli. 2018. Spacetime characterization of real-time collaborative editing. *Proceedings of the ACM on Human-Computer Interaction* 2, CSCW (2018), 1–19.

[11] Pieter de Lange, Paul Nicolaescu, Andreas T. Neumann, et al. 2020. Integrating Web-Based Collaborative Live Editing and Wireframing into a Model-Driven Web Engineering Process. *Data Sci. Eng.* 5 (2020), 240–260. https://doi.org/10.1007/s41019-020-00131-3

[12] Andrea De Lucia, Fausto Fasano, Giuseppe Scanniello, and Genny Tortora. 2007. Enhancing collaborative synchronous UML modelling with fine-grained versioning of software artefacts. *Journal of Visual Languages & Computing* 18, 5 (2007), 492–503.

[13] Thiago A. de S. Silva, Glívia A. R. Barbosa, and Ismael S. Santana. 2018. Evaluation of User Experience and Sociability on Platforms of Ephemeral Narratives: an Instagram Stories Case Study. In *Proceedings of the 24th Brazilian Symposium on Multimedia and the Web (WebMedia '18)*. ACM, 331–337. https://doi.org/10.1145/3243082.3243084

[14] Clarence A Ellis and Simon J Gibbs. 1989. Concurrency control in groupware systems. In *Proceedings of the 1989 ACM SIGMOD international conference on Management of data*. 399–407.

[15] Clarence A Ellis, Simon J Gibbs, and Gail Rein. 1991. Groupware: some issues and experiences. *Commun. ACM* 34, 1 (1991), 39–58.

[16] R M Featherstone, D M Dryden, M Foisy, J Guise, M D Mitchell, R A Paynter, K A Robinson, C A Umscheid, and L Hartling. 2015. Advancing knowledge of rapid reviews: an analysis of results, conclusions and recommendations from published review articles examining rapid reviews. *Systematic reviews* 4, 1 (2015), 1–8.

[17] Thore Fechner, Dennis Wilhelm, and Christian Kray. 2015. Ethermap: real-time collaborative map editing. In *Proceedings of the 33rd annual ACM conference on human factors in computing systems*. 3583–3592.

[18] Roy Thomas Fielding. 2000. REST: architectural styles and the design of network-based software architectures. *Doctoral dissertation, University of California* (2000).

[19] Cristian Gadea. 2021. *Architectures and Algorithms for Real-Time Web-Based Collaboration*. Ph. D. Dissertation. Université d'Ottawa/University of Ottawa.

[20] Liping Gao, Fangyu Yu, Qingkui Chen, and Naixue Xiong. 2016. Consistency maintenance of do and undo/redo operations in real-time collaborative bitmap editing systems. *Cluster Computing* 19 (2016), 255–267.

[21] Jens Emil Sloth Grønbæk, Juan Sánchez Esquivel, Germán Leiva, Eduardo Velloso, Hans Gellersen, and Ken Pfeuffer. 2024. Blended Whiteboard: Physicality and Reconfigurability in Remote Mixed Reality Collaboration. In *Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI '24)*. ACM, Article 798, 16 pages. https://doi.org/10.1145/3613904.3642293

[22] Claudia-Lavinia Ignat and Moira C Norrie. 2003. Customizable collaborative editor relying on treeOPT algorithm. In *ECSCW 2003: Proceedings of the Eighth European Conference on Computer Supported Cooperative Work 14–18 September 2003, Helsinki, Finland*. Springer, 315–334.

[23] Abdessamad Imine. 2009. Coordination model for real-time collaborative editors. In *Coordination Models and Languages: 11th International Conference, COORDINATION 2009, Lisboa, Portugal, June 9-12, 2009. Proceedings 11*. Springer, 225–246.

[24] Ryota Inoue, Yudai Kato, Takushi Goda, Tadachika Ozono, Shun Shiramatsu, and Toramatsu Shintani. 2012. A real-time collaborative mechanism for editing a web page and its applications. In *2012 Fifth International Symposium on Parallel Architectures, Algorithms and Programming*. IEEE, 186–193.

[25] Kevin Jahns. 2018. HocusPocus - Collaborative editing. https://tiptap.dev/docs/hocuspocus/guides/collaborative-editing

[26] Kevin Jahns. 2018. YJS - A CRDT framework with a powerful abstraction of shared data. https://github.com/yjs/yjs

[27] Tim Jungnickel and Tobias Herb. 2016. Simultaneous editing of JSON objects via operational transformation. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*. 812–815.

[28] Shin-Ya Katayama, Takushi Goda, Shun Shiramatsu, Tadachika Ozono, and Toramatsu Shintani. 2013. A fast synchronization mechanism for collaborative web applications based on HTML5. In *2013 14th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*. IEEE, 663–668.

[29] Martin Kleppmann. 2020. Moving elements in list CRDTs. In *Proceedings of the 7th Workshop on Principles and Practice of Consistency for Distributed Data (PaPoC '20)*. ACM, Article 4, 6 pages. https://doi.org/10.1145/3380787.3393677

[30] Elias Kuiter, Sebastian Krieter, Jacob Krüger, Gunter Saake, and Thomas Leich. 2021. variED: an editor for collaborative, real-time feature modeling. *Empirical Software Engineering* 26, 2 (2021), 24.

[31] Janne Lautamäki, Antti Nieminen, Johannes Koskinen, Timo Aho, Tommi Mikkonen, and Marc Englund. 2012. CoRED: browser-based Collaborative Real-time Editor for Java web applications. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*. 1307–1316.

[32] Stefania Leone, Thomas B Hodel-Widmer, Michael Boehlen, and Klaus R Dittrich. 2006. Tendax, a collaborative database-based real-time editor system. In *Advances in Database Technology-EDBT 2006: 10th International Conference on Extending Database Technology, Munich, Germany, March 26-31, 2006 10*. Springer, 1135–1138.

[33] Kai Lin, David Chen, Chengzheng Sun, and Geoff Dromey. 2007. Leveraging Single-User Microsoft Visio for Multi-user Real-Time Collaboration. In *Cooperative Design, Visualization, and Engineering*, Yuhua Luo (Ed.). Springer Berlin Heidelberg, 353–360.

[34] Geoffrey Litt, Sarah Lim, Martin Kleppmann, and Peter van Hardenberg. 2022. Peritext: A CRDT for Collaborative Rich Text Editing. *Proc. ACM Hum.-Comput. Interact.* 6, CSCW2, Article 531 (nov 2022), 36 pages. https://doi.org/10.1145/3555644

[35] Ming Liu, Leping Liu, and Li Liu. 2018. Group awareness increases student engagement in online collaborative writing. *The Internet and Higher Education* 38 (2018), 1–8.

[36] Paul Benjamin Lowry, Aaron Mosiah Curtis, and Michelle Rene Lowry. 2004. A taxonomy of collaborative writing to improve empirical research, writing practice, and tool development. *International Journal of Business Communication* 41, 1 (2004), 66–99.

[37] David Moher, Larissa Shamseer, Mike Clarke, Davina Ghersi, Alessandro Liberati, Mark Petticrew, Paul Shekelle, Lesley A Stewart, and Prisma-P Group. 2015. Preferred reporting items for systematic review and meta-analysis protocols (PRISMA-P) 2015 statement. *Systematic reviews* 4 (2015), 1–9.

[38] Stefano Montanelli and Martin Ruskov. 2023. A Systematic Literature Review of Online Collaborative Story Writing. In *Human-Computer Interaction – INTERACT 2023: 19th IFIP TC13 International Conference, York, UK, August 28 – September 1, 2023, Proceedings, Part III* (York, United Kingdom). Springer-Verlag, 73–93. https://doi.org/10.1007/978-3-031-42286-7_5

[39] Brice Nédelec, Pascal Molli, and Achour Mostefaoui. 2016. Crate: Writing stories together with our browsers. In *Proceedings of the 25th International Conference Companion on World Wide Web*. 231–234.

[40] David A Nichols, Pavel Curtis, Michael Dixon, and John Lamping. 1995. High-latency, low-bandwidth windowing in the Jupiter collaboration system. In *Proceedings of the 8th annual ACM symposium on User interface and software technology*. 111–120.

[41] Petru Nicolaescu, Mario Rosenstengel, Michael Derntl, Ralf Klamma, and Matthias Jarke. 2018. Near real-time collaborative modeling for view-based web information systems engineering. *Information Systems* 74 (2018), 23–39.

[42] Pavel Okopnyi, Oskar Juhlin, and Frode Guribye. 2022. Designing for Collaborative Video Editing. In *Norddic Human-Computer Interaction Conference (NordiCHI '22)*. ACM, Article 3, 11 pages. https://doi.org/10.1145/3546155.3546664

[43] Gérald Oster, Pascal Urso, Pascal Molli, and Abdessamad Imine. 2006. Data consistency for P2P collaborative editing. In *Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*. 259–268.

[44] Tadachika Ozono, Robin ME Swezey, Shun Shiramatsu, Toramatsu Shintani, Takushi Goda, Yudai Kato, and Ryota Inoue. 2012. Differential Synchronizaiton Mechanism for a Real-Time Collaborative Web Page Editing System WFE-S. In *2012 IIAI International Conference on Advanced Applied Informatics*. IEEE, 242–247.

[45] Tadachika Ozono, Robin ME Swezey, Shun Shiramatsu, Toramatsu Shintani, Ryota Inoue, Yudai Kato, and Takushi Goda. 2012. A real-time collaborative web

page editing system WFE-S based on cloud computing environment. In *2012 IIAI International Conference on Advanced Applied Informatics*. IEEE, 224–229.

[46] Lihang Pan, Chun Yu, Zhe He, and Yuanchun Shi. 2023. A Human-Computer Collaborative Editing Tool for Conceptual Diagrams. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23)*. ACM, Article 360, 29 pages. https://doi.org/10.1145/3544548.3580676

[47] Postman. 2023. 2023 State of the API Report. Accessed: 2024-06-10.

[48] Matthias Ressel, Doris Nitsche-Ruhland, and Rul Gunzenhäuser. 1996. An integrating, transformation-oriented approach to concurrency control and undo in group editors. In *Proceedings of the 1996 ACM conference on Computer supported cooperative work*. 288–297.

[49] Gabriele Salvati, Christian Santoni, Valentina Tibaldo, and Fabio Pellacini. 2015. Meshhisto: Collaborative modeling by sharing and retargeting editing histories. *ACM Transactions on Graphics (TOG)* 34, 6 (2015), 1–10.

[50] Holger J Schünemann and Lorenzo Moja. 2015. Reviews: rapid! rapid! rapid!… and systematic. *Systematic reviews* 4, 1 (2015), 1–3.

[51] Sharifahtun Naim Shahidan, Zuraina Ali, and Norsuhaily Abu Bakar. 2022. Motivational Impacts of the Google Docs Integration to Support Collaborative Writing: A Review Approach. *Intl. Journal of Advances in Social Sciences and Humanities* 1, 3 (2022), 166–171.

[52] Marc Shapiro and Nuno Preguiça. 2007. Designing a commutative replicated data type. *arXiv preprint arXiv:0710.1784* (2007).

[53] Marc Shapiro, Nuno Preguiça, Carlos Baquero, and Marek Zawirski. 2011. Conflict-free replicated data types. In *Stabilization, Safety, and Security of Distributed Systems: 13th International Symposium, SSS 2011, Grenoble, France, October 10-12, 2011. Proceedings 13*. Springer, 386–400.

[54] Haifeng Shen and Chengzheng Sun. 2002. Highlighting: a gesturing communication tool for real-time collaborative systems. In *Fifth International Conference on Algorithms and Architectures for Parallel Processing, 2002. Proceedings.* IEEE, 180–187.

[55] Jaemyung Shin, Bumsoo Kang, Taiwoo Park, Jina Huh, Jinhan Kim, and Junehwa Song. 2016. Beupright: Posture correction using relational norm intervention. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 6040–6052.

[56] Scott Solomon. 2016. *Google and Microsoft's Battle for the Enterprise: How Users Are Interacting With SaaS Applications - BetterCloud Monitor. BetterCloud Monitor.* https://www.bettercloud.com/monitor/google-apps-and-office-365-differences/ Accessado em 10/06/2024.

[57] Adrienne Stevens, Mona Hersi, Chantelle Garritty, Lisa Hartling, Beverley J Shea, Lesley A Stewart, Vivian Andrea Welch, and Andrea C Tricco. 2024. Rapid review method series: interim guidance for the reporting of rapid reviews. *BMJ Evidence-Based Medicine* (2024). https://doi.org/10.1136/bmjebm-2024-112899 arXiv:https://ebm.bmj.com/content/early/2024/07/21/bmjebm-2024-112899.full.pdf

[58] Leo Stewen and Martin Kleppmann. 2024. Undo and Redo Support for Replicated Registers. In *Proceedings of the 11th Workshop on Principles and Practice of Consistency for Distributed Data* (Athens, Greece) *(PaPoC '24)*. Association for Computing Machinery, New York, NY, USA, 1–7. https://doi.org/10.1145/3642976.3653029

[59] Maher Suleiman, Michele Cart, and Jean Ferrié. 1998. Concurrent operations in a distributed and mobile collaborative environment. In *Proceedings 14th International Conference on Data Engineering*. IEEE, 36–45.

[60] Chengzheng Sun. 2000. Undo any operation at any time in group editors. In *Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work (CSCW '00)*. ACM, 191–200. https://doi.org/10.1145/358916.358990

[61] Chengzheng Sun, Xiaohua Jia, Yanchun Zhang, Yun Yang, and David Chen. 1998. Achieving convergence, causality preservation, and intention preservation in real-time cooperative editing systems. *ACM Transactions on Computer-Human Interaction (TOCHI)* 5, 1 (1998), 63–108.

[62] Chengzheng Sun, David Sun, Agustina Ng, Weiwei Cai, and Bryden Cho. 2020. Real Differences between OT and CRDT under a General Transformation Framework for Consistency Maintenance in Co-Editors. *Proc. ACM Hum.-Comput. Interact.* 4, GROUP, Article 06 (jan 2020), 26 pages. https://doi.org/10.1145/3375186

[63] Chengzheng Sun, Steven Xia, David Sun, David Chen, Haifeng Shen, and Wentong Cai. 2006. Transparent adaptation of single-user applications for multi-user real-time collaboration. *ACM Transactions on Computer-Human Interaction (TOCHI)* 13, 4 (2006), 531–582.

[64] David Sun, Chengzheng Sun, Agustina Ng, and Weiwei Cai. 2020. Real Differences between OT and CRDT in Correctness and Complexity for Consistency Maintenance in Co-Editors. *Proc. ACM Hum.-Comput. Interact.* 4, CSCW1, Article 21 (may 2020), 30 pages. https://doi.org/10.1145/3392825

[65] Xin Tan, Xinyue Lv, Jing Jiang, and Li Zhang. 2024. Understanding Real-Time Collaborative Programming: A Study of Visual Studio Live Share. *ACM Trans. Softw. Eng. Methodol.* 33, 4, Article 110 (apr 2024), 28 pages. https://doi.org/10.1145/3643672

[66] Christian Thum, Michael Schwind, and Martin Schader. 2009. SLIM—A lightweight environment for synchronous collaborative modeling. In *Model Driven Engineering Languages and Systems: 12th International Conference, MODELS 2009, Denver, CO, USA, October 4-9, 2009. Proceedings 12*. Springer, 137–151.

[67] Jennifer Tsan, Jessica Vandenberg, Zarifa Zakaria, Joseph B. Wiggins, Alexander R. Webber, Amanda Bradbury, Collin Lynch, Eric Wiebe, and Kristy Elizabeth Boyer. 2020. A Comparison of Two Pair Programming Configurations for Upper Elementary Students. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (Portland, OR, USA) *(SIGCSE '20)*. ACM, 346–352. https://doi.org/10.1145/3328778.3366941

[68] A. Omar M. Uscamayta, Bruna C.R. Cunha, Diogo S. Martins, and Maria G. Pimentel. 2017. Collaborative Ad-hoc Multimedia Authoring via Mobile Devices. In *Proceedings of the 23rd Brazillian Symposium on Multimedia and the Web (WebMedia '17)*. ACM, 61–64. https://doi.org/10.1145/3126858.3131572

[69] Nicolas Vidot, Michelle Cart, Jean Ferrié, and Maher Suleiman. 2000. Copies convergence in a distributed real-time collaborative environment. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work*. 171–180.

[70] Victor H. Vieira, Daniel G. Sante, André P. Freire, and Renata P. M. Fortes. 2005. A web service for CSCW applications. In *Proceedings of the 11th Brazilian Symposium on Multimedia and the Web (WebMedia '05)*. ACM, 1–3. https://doi.org/10.1145/1114223.1114241

[71] Ruipeng Wei, Ruisheng Zhang, Chen Zhao, Dongmei Yue, and Lian Li. 2009. Design and Implementation of Scientific Collaborative Editing Environment on Chemistry. In *2009 Eighth International Conference on Grid and Cooperative Computing*. IEEE, 188–192.

[72] Raymond K Wong. 2004. Collaborative hypertext editing in mobile environment. In *10th International Multimedia Modelling Conference, 2004. Proceedings.* IEEE, 300–307.

[73] Steven Xia, David Sun, Chengzheng Sun, David Chen, and Haifeng Shen. 2004. Leveraging single-user applications for multi-user collaboration: the coword approach. In *Proceedings of the 2004 ACM conference on Computer supported cooperative work*. 162–171.

[74] Xinyue Yu and Tun Lu. 2022. An AST-Based Collaborative Discussion Tool for the MOOC Environment. In *CCF Conference on Computer Supported Cooperative Work and Social Computing*. Springer, 284–294.