RESEARCH-ARTICLE

Impacts of the Usage of Generative Artificial Intelligence on Software Development Process

PATRICIA DE SANTOS, Federal University of the State of Rio de Janeiro, Rio de Janeiro, RJ, Brazil

ALLAN CHAMON FIGUEIREDO, Federal University of the State of Rio de Janeiro, Rio de Janeiro, RJ, Brazil

PEDRO MOURA, Federal University of the State of Rio de Janeiro, Rio de Janeiro, RJ, Brazil

BRUNA DIIRR, Federal University of the State of Rio de Janeiro, Rio de Janeiro, RJ, Brazil

ADRIANA CESARIO F ALVIM, Federal University of the State of Rio de Janeiro, Rio de Janeiro, RJ, Brazil

RODRIGO PEREIRA DOS SANTOS, Federal University of the State of Rio de Janeiro, Rio de Janeiro, RJ, Brazil

Published: 20 May 2024

SBSI '24: XX Brazilian Symposium on Information Systems
May 20 - 23, 2024
Juiz de Fora, Brazil

Impacts of the Usage of Generative Artificial Intelligence on Software Development Process

Patrícia de Oliveira Santos
Universidade Federal do Estado do Rio de Janeiro (UNIRIO)
Rio de Janeiro, RJ, Brazil
patricia.santos@edu.unirio.br

Allan Chamon
Universidade Federal do Estado do Rio de Janeiro (UNIRIO)
Rio de Janeiro, RJ, Brazil
allanchamon@edu.unirio.br

Pedro Nuno de Souza Moura
Universidade Federal do Estado do Rio de Janeiro (UNIRIO)
Rio de Janeiro, RJ, Brazil
pedro.moura@uniriotec.br

Bruna Diirr
Universidade Federal do Estado do Rio de Janeiro (UNIRIO)
Rio de Janeiro, RJ, Brazil
bruna.diirr@uniriotec.br

Adriana Cesario de Faria Alvim
Universidade Federal do Estado do Rio de Janeiro (UNIRIO)
Rio de Janeiro, RJ, Brazil
adriana@uniriotec.br

Rodrigo Pereira dos Santos
Universidade Federal do Estado do Rio de Janeiro (UNIRIO)
Rio de Janeiro, RJ, Brazil
rps@uniriotec.br

ABSTRACT

Context: Over the years, tools have been created to improve the execution of development process activities. The emergence of generative Artificial Intelligence (AI) and, more recently, the launch and dissemination of Copilot, ChatGPT-3 and other generative tools, have broadened the discussion about the possibility of using conversational generative AI tools in diverse development tasks. **Problem:** There is still a lack of secondary studies to map the literature about how software development process activities can be affected by the usage of generative AI tools. **Solution:** This study aims to identify in which activities of the software development process Natural Language (NL) generative AI tools have been used and how they can impact requirements specification, design/architecture, development and testing activities. **IS Theory:** The study was developed under the aegis of the Task Technology Fit theory. **Method:** This work presents the results of a Systematic Mapping Review (SMR) carried out to collect research results that investigate the application of generative AI tools in the software development process. **Results:** Results indicate that the main activities affected are development and testing and that, although there are still some issues to be addressed, there are benefits in using AI generative tools compared to using more traditional methods like human-human pair programming and code testing made by software engineering professionals. **Contribution:** It was possible to collect studies to identify in which activities of the software development process generative AI tools can be applied and what are the impacts of using this technology.

SBSI '24, May 20–23, 2024, Juiz de Fora, Brazil

ACM ISBN 979-8-4007-0996-8/24/05
https://doi.org/10.1145/3658271.3658337

CCS CONCEPTS

• **Software and its engineering** → **Software creation and management**; • **Computing methodologies** → *Natural language processing*.

KEYWORDS

Generative AI, Software Process, Software Engineering, ChatGPT, Copilot

ACM Reference Format:
Patrícia de Oliveira Santos, Allan Chamon, Pedro Nuno de Souza Moura, Bruna Diirr, Adriana Cesario de Faria Alvim, and Rodrigo Pereira dos Santos. 2024. Impacts of the Usage of Generative Artificial Intelligence on Software Development Process. In *XX Brazilian Symposium on Information Systems (SBSI '24), May 20–23, 2024, Juiz de Fora, Brazil.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3658271.3658337

1 INTRODUÇÃO

Melhorar a produtividade do desenvolvedor e, consequentemente, a qualidade dos sistemas, tem sido uma preocupação fundamental na Engenharia de Software há décadas [30]. Ao longo dos anos, ferramentas de software foram desenvolvidas para apoiar as partes de todo o ciclo de vida do processo de desenvolvimento [19]. Estas ferramentas visam otimizar a execução das diversas tarefas relacionadas ao processo de desenvolvimento de software.

Os avanços recentes na Inteligência Artificial (IA) promovidos pelas técnicas de Aprendizagem Profunda abriram novas oportunidades para automatizar várias tarefas no desenvolvimento de software [45] e a integração da IA, especificamente de ferramentas generativas de IA, como *Large Language Models* (LLMs), mostrou-se promissora em processos de desenvolvimento de software, levando a avanços potenciais em vários aspectos desse campo [38].

A IA generativa pode ser usada para simplificar processos de desenvolvimento de software, por meio da automação de tarefas como elaboração de *user stories* [8], teste, depuração e implantação [14]. Atualmente, já existem ferramentas de IA para auxiliar desenvolvedores como GitHub Copilot e Copilot X que usam o modelo GPT-4 [14] e alguns estudos [14, 20] indicam que a produtividade no

desenvolvimento de software pode ser impactada positivamente pela integração de ferramentas de IA, sendo previsto um aumento de produtividade que pode implicar redução de custos funcionais variando de 30% a 45% [38].

Entretanto, apesar dos benefícios, também devem ser considerados os aspectos envolvendo riscos no uso da tecnologia da IA generativa como as chamadas "alucinações" [14], que ocorrem quando a ferramenta cria dados/informações que não existem [43] , geração de códigos com viés [16] e brechas de segurança [52]. Em decorrência destes fatores, surge a necessidade de estratégias como supervisão [14], revisão [48] e validação das respostas geradas por este tipo de ferramenta [13].

Neste contexto, este trabalho apresenta os resultados de um Mapeamento Sistemático da Literatura (MSL) conduzido com a finalidade de responder à seguinte questão: **Como a IA generativa impacta o processo de desenvolvimento de software?**. Tal MSL visa investigar (a) as implicações o uso de IA generativa nas atividades do processo de desenvolvimento de software, (b) como os papéis dos profissionais envolvidos no processo de desenvolvimento podem ser afetados, além de (c) discutir a viabilidade do uso da IA generativa emergente no processo de software.Por meio dos estudos selecionados, e com base na teoria de sistemas de informação (SI) *Task Technology Fit*, pretende-se abordar a adaptabilidade das ferramentas de AI generativa no execução das atividades do processo de desenvolvimento de software.

O artigo está organizado da seguinte forma: a Seção 2 apresenta os conceitos fundamentais relacionados a este trabalho; a Seção 3 detalha o método de pesquisa utilizado; por sua vez, a Seção 4 descreve os resultados obtidos; a Seção 5 aborda a discussão acerca dos resultados; já a Seção 6 pontua as ameaças à validade identificadas durante a condução do trabalho; e, finalmente, a Seção 7 apresenta as conclusões e os trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 Atividades do Processo de Desenvolvimento de Software

Um processo de software pode ser definido como um conjunto coerente de políticas, estruturas organizacionais, tecnologias, procedimentos e artefatos necessários para conceber, desenvolver, implantar e manter um produto de software [18]. A adoção de um processo de software bem executado pode agregar benefícios como consistência e repetibilidade nas execução da tarefas, viabilização da colaboração e coordenação entre membros da equipe de desenvolvimento, gerenciamento de riscos durante as etapas do processo e melhoria continua pela observação da aplicação do processo em projetos de software executados anteriormente.

O processo de desenvolvimento de software é composto por atividades dentre as quais, conforme definido pela ABNT ISO/IEC 12207, estão Análise de Requisitos de Software, Projeto Arquitetural do Software, Codificação e Teste de Software [1], atividades que antecedem a entrega do produto final de software.

Já o ciclo de vida de software define os diferentes estágios da vida útil de um produto de software e normalmente englobam análise e especificação de requisitos, design, desenvolvimento, verificação e validação, implantação, operação, manutenção e desativação [18]. A adoção de um modelo de ciclo de vida adequado e bem definido ajuda a produzir produtos de software de boa qualidade em organizações que atuam com desenvolvimento de software [42]. Alguns exemplos de ciclo de vida são *Rapid Application Development*, *Extreme Programming* (XP) e Scrum [37].

2.2 Agentes Conversacionais e *Chatbots*

Agentes Conversacionais (ACs) existem desde a década de 60, quando Joseph Weizenbaum desenvolveu a agente ELIZA para produzir interações humanas típicas entre um usuário e um computador [50].

Os ACs exploram tecnologias de linguagem natural para envolver os usuários em busca de informações baseadas em texto e em diálogos orientados a tarefas para uma ampla gama de aplicações como serviços de atendimento ao cliente, navegação em websites e suporte técnico [24]. O desenvolvimento de ACs pressupõe que as pessoas interajam com sistemas que sejam capazes de capturar, interpretar e responder em linguagem natural, comparável a iniciar uma conversa com outro ser humano [30].

Chatbots são uma classe de agentes de software conversacionais inteligentes ativados por entrada em linguagem natural, que pode ser na forma de texto, voz ou ambos. Eles fornecem saída conversacional em resposta e, a depender da aplicação, também podem executar tarefas [34].

2.3 IA Generativa, ChatGPT e Copilot

A IA generativa é um campo de inteligência artificial que se concentra na geração de informações novas e originais por aprendizado de máquina em grandes bases de dados de experiências [6]. Esse campo se ocupa de uma classe de algoritmos de aprendizado de máquina que podem aprender a partir de conteúdos como texto, imagens e áudio para gerar novos conteúdos. Em contraste com os modelos discriminativos (condicionais) de Aprendizado de Máquina (*Machine Learning* - ML), que aprendem limiares de decisão, modelos de IA generativa produzem artefatos como saída, que podem ter uma ampla variedade e complexidade [44].

Ferramentas de *chatbot* correspondem a uma das áreas de aplicação da IA generativa [6], sendo o ChatGPT[1] um dos mais mais recentes desenvolvimentos no grupo de sistemas de *chatbot* [46]. O ChatGPT é um *chatbot* dotado de IA criado pela empresa OpenAI [46] que possui como cerne o modelo GPT (*Generative Pretrained Transformer*), treinado em uma grande quantidade de dados de texto a fim de gerar textos em linguagem natural (humana). Esse modelo pode ser usado para uma variedade de tarefas de processamento de linguagem natural, como tradução de idiomas, resumo de texto, resposta a perguntas e também para aplicativos de *chatbot* [12].

Os modelos GPT geram textos em diferentes idiomas e podem criar palavras, frases e parágrafos com sonoridade humana sobre quase qualquer tópico e estilo de escrita [22]. Além destas funcionalidades, de acordo com Sakib et al.[39], o ChatGPT apresenta um bom desempenho na geração de código-fonte.

GitHub Copilot[2] é uma ferramenta de desenvolvimento de software que oferece geração de linhas código, trechos de código ou

[1] https://chat.openai.com/
[2] https://copilot.github.com/

até programas inteiros baseados em código e comentários existentes [20]. O Copilot, que é baseado no modelo LLM Codex também da OpenAI, tem acesso a um grande número de projetos de código aberto, o que permite que ele utilize códigos de forma mais extensiva em mais linguagens de programação do que outras ferramentas de geração de código [52].

Mais especificamente, o Codex é uma versão modificada do GPT-3 projetado para criar código de computador e treinado em código-fonte aberto e linguagem natural, por isso compreende tanto linguagens de programação quanto linguagens humanas [26].

Atualmente estas ferramentas estão sendo cada vez mais adotadas por profissionais no processo de desenvolvimento por serem gratuitas, terem bom desempenho, especialmente para geração de código-fonte, e devido ao seu potencial para facilitar a execução de diversas tarefas, conforme relatado nos diversos estudos que são abordados nas próxima seções.

3 MÉTODO DE PESQUISA

Para a realização deste estudo, foi conduzido um MSL com a finalidade de buscar na literatura os principais estudos relacionados ao uso de IA generativa nas atividades do processo de desenvolvimento de software.

Foi definido e seguido um protocolo organizado em etapas com a finalidade de estabelecer de forma clara o trabalho necessário para a realização do mapeamento. Como material de referência para definição do protocolo de mapeamento foram utilizados conceitos definidos por Petersen et al. [32] e orientações definidas por Kitchenham e Chartes [23]. As etapas do protocolo foram definidas da seguinte forma:

(1) Definir objetivos da pesquisa
(2) Utilizar a estratégia PICO (detalhada na Seção 3.1) para construção de uma *string* de busca e das perguntas do projeto
(3) Definir critérios de inclusão e exclusão
(4) Selecionar estudo de controle nas bases de pesquisa para validar a *string* de busca
(5) Executar a *string* e extrair estudos relevantes nas bases de pesquisa
(6) Eliminar possíveis duplicações de resultados de buscas utilizando Parsifal (CE1)
(7) Leitura parcial dos estudos (título e palavras-chave) para aplicação de critérios de exclusão
(8) Leitura parcial dos estudos (*abstract*) e aplicação de critérios de exclusão
(9) Leitura parcial dos estudos (introdução e conclusão) para aplicação de critérios de inclusão/exclusão – Leitura realizada por dois autores
(10) Leitura completa dos estudos selecionados e aplicação do critério de inclusão
(11) Relato dos resultados encontrados

3.1 PICO

O uso do modelo PICO (*population*, *intervention*, *comparison*, *outcomes*), que permite definir os critérios para elaboração da pergunta de pesquisa, foi a estratégia escolhida para seleção dos termos de busca relevantes, seguindo os princípios definidos por Petticrew e Roberts [33] e Kitchenham e Chartes [23]. Com base na principal Questão de Pesquisa (QP), definida como:
Como a **IA generativa (I) impacta (O)** o **processo de desenvolvimento de software (P)**?
foram definidos os critérios de população, intervenção e resultados da seguinte forma:

- (P) População: desenvolvimento de software
- (I) Intervenção: IA generativa
- (O) Resultados: impacto (não utilizado na busca)

O elemento "Comparação" (*Comparison*) não foi definido, pois este estudo visa realizar um MSL, cujo objetivo é fornecer uma visão mais abrangente acerca de um tópico de pesquisa [28], diferenciando-se da Revisão Sistemática de Literatura (RSL) na qual, além da composição do estado da arte, também é necessário definir critérios de comparação, analisar e interpretar todas as evidências relacionadas à questão de pesquisa e identificar melhores práticas baseadas em evidências empíricas [28].

Além da QP, foram definidas e enumeradas subquestões (SQ) de pesquisa, conforme exposto na Tabela 1, para apoiar a identificação da relevância dos estudos existentes e selecionados para análise. Ressalta-se que, na subquestão SQ3, o termo produtividade pode ser definido como a redução de tempo de execução de tarefas e produção de entregas. Embora não exista uma definição comum de produtividade, parece haver consenso de que a produtividade descreve a relação entre produção e insumo (saídas e entradas) [47].

Tabela 1: Questões de Pesquisa

ID	Descrição
QP	Como a IA generativa impacta o processo de desenvolvimento de software?
SQ1	Como a IA generativa é usada nas atividades do processo de desenvolvimento de software?
SQ2	Como o uso de IA generativa impacta os papéis dos profissionais de TI envolvidos no processo de desenvolvimento de software?
SQ3	Como a IA generativa pode impactar a produtividade das atividades de desenvolvimento de software?

Após a definição das questões de pesquisa e dos termos relevantes, foram identificados os principais sinônimos para os termos PI com a finalidade de compor uma *string* de busca mais abrangente, apresentada na Tabela 3 e cuja construção está detalhada na Seção 3.2.

3.2 String de Busca

Os termos utilizados na *string* de busca foram selecionados pela análise PICO e, para obtenção de maior número de estudos de interesse, foram adicionados termos relacionados aos termos identificados na QP, conforme apresentado na Tabela 2.

Foi utilizado o *wildcard* "*" em algumas palavras para que os resultados das buscas contemplassem derivações do termo como conjugações verbais distintas, substantivos relacionados e termos no plural.

Foram definidos termos para o item "Resultados"(O), entretanto estes termos não foram utilizados na formulação da *string* por não serem considerados fundamentais para a obtenção dos resultados. O uso destes termos na *string* de busca também poderia restringir a quantidade de estudos obtidos que respondem adequadamente à

Tabela 2: Termos para construção da string de busca (PI).

(P) População	application develop* software architecture software building software construction software develop* software engineering software lifecycle software process
(I) Intervenção	generative AI generative artificial intelligence Chat GPT ChatGPT chatbot copilot

Subquestão de pesquisa 1 (SQ1). A conjunção dos termos definidos em P e I resultou na *string* apresentada na Tabela 3:

Tabela 3: String de busca obtida.

("application develop*" OR "software architecture" OR "software building" OR "software construction" OR "software develop*" OR "software engineering" OR "software lifecycle" OR "software process") AND ("Chat GPT" OR "ChatGPT" OR "chatbot" OR "copilot" OR "generative AI" OR "generative artificial intelligence")

3.3 Critérios de Seleção

Critérios de seleção de estudos têm o objetivo de identificar quais estudos primários fornecem evidência direta sobre a questão de pesquisa [23] e podem ser classificados como Critério de Exclusão (CE) ou Critério de Inclusão (CI). Os critérios são definidos para avaliar se os estudos apresentam respostas que atendam às QPs e para eliminar estudos não relevantes da seleção.

Os estudos foram incluídos na avaliação quando atendiam ao CI1 e eliminados da seleção quando não estavam de acordo com pelo menos um dos CEs definidos. Os critérios foram estabelecidos conforme apresentado na Tabela 4.

Tabela 4: Critérios de inclusão e exclusão.

ID	Descrição	Tipo
CI1	Estudo relaciona pelo menos uma atividade do processo de desenvolvimento de software ao uso de IA generativa	Inclusão
CE1	Estudo duplicado	Exclusão
CE2	Estudo não é revisado por pares	
CE3	Estudo não é completo	
CE4	Estudo não foi publicado em português ou inglês	
CE5	Estudo não pode ser obtido na íntegra através das bases de busca	
CE6	Estudo não responde a alguma das questões de pesquisa	

3.4 Extração dos Dados

Foram utilizadas três bases para busca dos estudos:

- ACM Digital Library[3]
- IEEE Xplore[4]
- Scopus[5]

Estas bases foram selecionadas devido à diversidade de estudos disponíveis e à sua relevância para pesquisas em Ciência da computação. As buscas foram realizadas em setembro de 2023 e foram utilizados filtros específicos em cada base para se obter estudos mais propensos a atender aos critério estabelecidos na Tabela 4. A consulta à base Scopus foi limitada especificamente para a área de Ciência da computação e, na base ACM Digital Library, foram buscados apenas estudos que atendem ao filtro *Research Article*. Não foi necessário especificar filtros de busca para a base IEEE Xplore, dados os critérios de seleção estabelecidos.

Na etapa 4, foram selecionados dois estudos de controle que, ao final do processo de seleção, foram identificados como E4 e E26 e estão destacados na Tabela 6.

As buscas nas bases selecionadas retornaram um total de 649 estudos, dos quais foram removidos 95 duplicados, identificados com auxílio da ferramenta Parsifal, e um estudo identificado como duplicado posteriormente. Após a leitura parcial dos artigos e atendendo ao critério CE1, 554 estudos prosseguiram na análise. A Figura 1 ilustra os passos executados e os resultados do processo de seleção dos estudos. Entre parênteses, ao lado da descrição, está o número correspondente à etapa do protocolo.

Ao final da execução de todo o processo de mapeamento, foram selecionados 27 artigos que foram identificados e estão listados na Tabela 6.

4 RESULTADOS

4.1 Visão Geral dos Estudos Selecionados

A maioria dos estudos selecionados relata uso e impacto das ferramentas de IA generativa para dar suporte às tarefas da atividade de Desenvolvimento de Sistemas e Testes. A Tabela 5 apresenta a relação dos assuntos abordados nos estudos selecionados e as atividades do processo de desenvolvimento de software. Nesta tabela, a atividade "Codificação e Teste de Software" foi separada em "Codificação" e "Teste" para tornar clara a forma de uso de IA generativa abordada em cada estudo.

Tabela 5: Relação entre atividades do processo de desenvolvimento e estudos selecionados.

Atividade		Estudos
Análise de Requisitos de Software		E3, E4, E7, E16
Projeto Arquitetural do Software		E2, E10
Codificação e Teste de Software	Codificação	E1, E2, E3, E4, E5, E6, E7, E8, E9, E11 E12, E13, E14, E15, E17, E18, E19, E20, E21, E22, E23, E24, E25, E26, E27
	Teste	E2, E4, E6, E13

Conforme é possível observar na Tabela 5, muitos estudos discorrem sobre mais de uma atividade do processo de desenvolvimento

[3] https://dl.acm.org/
[4] https://ieeexplore.ieee.org/Xplore/home.jsp
[5] https://www.elsevier.com/pt-br/solutions/scopus

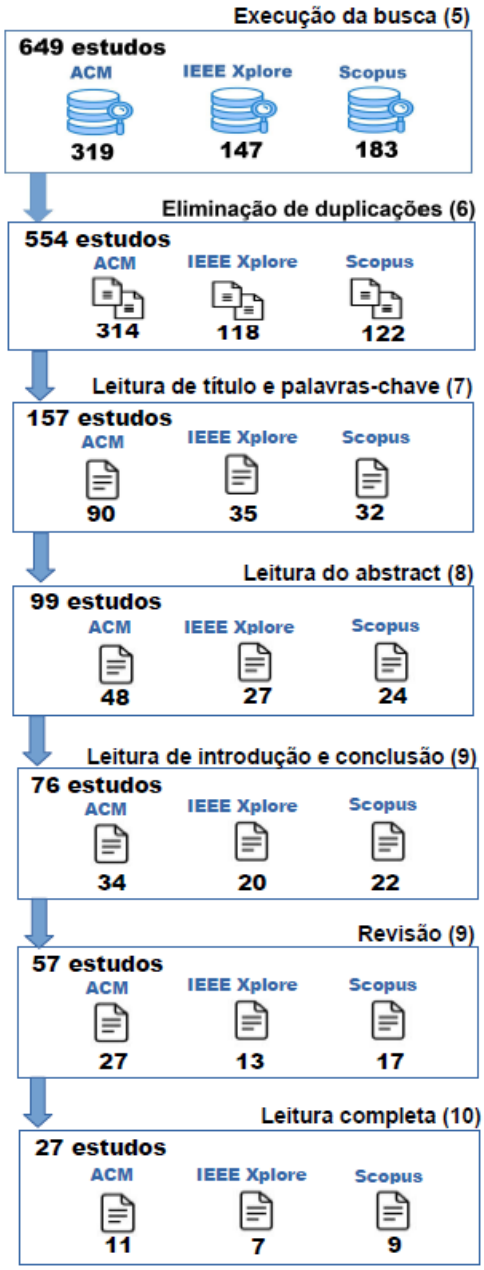

Figura 1: Resultados do processo de seleção dos estudos.

de software, principalmente nos casos em que o estudo trata do uso do ChatGPT (E1, E2, E4, E7, E10, E12, E13, E14, E16 e E21) e outras ferramentas de *chatbot* (E3, E24 e E27). Os estudos E8, E11, E15, E17, E22 e E26 tratam do uso do GitHub Copilot para realização das diversas tarefas que envolvem geração de código-fonte além do próprio processo de codificação, tais como testes automatizados e verificação de segurança. O estudo E9 faz uma comparação entre ferramentas distintas de IA generativa para geração de código-fonte. No estudo E3, é proposto um *framework* para uso de ferramentas de IA em todas as fases do ciclo de vida de um software. Os estudos E5, E19, E23 e E25 abordam o uso de outras ferramentas de IA generativa além do ChatGPT e Copilot como Themisto, TransCoder e ferramentas de *chatbot* alternativas.

A maioria dos estudos (18 estudos) foi publicada em 2023. Não foram encontrados estudos relevantes publicados antes de 2019.

4.2 Questões de Pesquisa

Nesta subseção, serão respondidas as SQs de acordo com os resultados encontrados nos estudos selecionados, objetivando responder também à questão principal de pesquisa (QP).

4.2.1 SQ1: Como a IA generativa é usada nas atividades do processo de desenvolvimento de software? Os estudos selecionados apresentam diversas formas de aplicação da IA generativa nas atividades de desenvolvimento de software. No contexto da atividade de "Análise de Requisitos de Software", o ChatGPT-4 foi apresentado como uma alternativa para elicitação de requisitos utilizando a abordagem de engenharia de requisitos orientada a objetivos. O estudo E7 apresenta a possibilidade de uso do ChatGPT-3 para geração e classificação de *user stories*. *User story* é um artefato simplificado de engenharia de requisitos, comumente utilizado em metodologias de desenvolvimento ágil, que permite uma descrição padronizada dos recursos do sistema requeridos pelos usuários [35]. Também foi abordado pelo estudo E12 o uso de um *chatbot* durante a realização de reuniões de retrospectiva no processo de desenvolvimento ágil para controle das tarefas realizadas na iteração e facilitação da rastreabilidade. Já o estudo E3 sugere o uso de algoritmos de ML e AI para automatizar tarefas como classificação de requisitos, análise e mitigação de riscos, geração de esqueletos de código entre outras, abrangendo diversas fases do ciclo de vida do desenvolvimento de software.

No contexto da atividade "Projeto Arquitetural do Software", o estudo E10 apresenta um estudo de caso no qual o ChatGPT é utilizado para geração de um modelo arquitetural de software. O estudo E2 faz referência ao estudo E10 como exemplo de estudo sobre especificação e criação de uma arquitetura de software supervisionada por humano com auxílio da IA.

Em relação à atividade "Codificação e Teste de Software", diversos estudos abordam o uso de IA generativa para geração de código-fonte para viabilizar a codificação automatizada. Os artigos E1, E2, E4, E14 e E21 abordam o uso do ChatGPT para geração de código-fonte, e os estudos E6, E8, E11, E15, E17, E22, E26 e E27 tratam do uso de Codex e/ou Copilot para este mesmo fim. O estudo E11 apresenta um caso prático de uso do Copilot para geração de código seguro. O uso da IA generativa para tradução de código-fonte entre linguagens de programação distintas, que é necessária durante o processo de modernização de código legado, também foi abordado pelos estudos E4, E19 e E22. O estudo E20 apresenta a possibilidade de geração de interfaces gráficas de usuário (*Graphical User Interfaces* ou GUIs) pelo uso da ferramenta de busca de interfaces Instigator, que é baseada em um modelo LLM e usa a arquitetura GPT.

O estudo E4 cita a viabilidade do uso de ferramentas de IA generativa para a geração de casos de teste e identificação e correção de *bugs* de forma mais rápida. Outro estudo (E6) faz uma avaliação sobre a capacidade de geração e correção de *bugs* simples de código-fonte pelo Codex. O ChatGPT também foi utilizado para inspeção e

Tabela 6: Estudos selecionados

ID	Titulo	Autores	Ano	Fonte	Ref
E1	ChatGPT in Software Development: Methods and Cross-Domain Applications	Bale, A. S.; Vada, Y. R.; Oshiojum, B. E.; Lakkineni, U. K.; Rao, C.; V., K. e Rani, I.	2023	Scopus	[7]
E2	Decoding ChatGPT: A taxonomy of existing research, current challenges, and possible future directions	Sohail, S. S. ; Farhat, F.; Himeur, Y.; Nadeem, M.; Madsen, D. Ø.; Singh, Y.; Atalla, S. e Mansoor, W.	2023	Scopus	[43]
E3	Framework for the Automation of SDLC Phases using Artificial Intelligence and Machine Learning Techniques	Shankar, Sahana P. e Chaudhari, S. S.	2023	Scopus	[40]
E4	**Generative AI for Software Practitioners**	Ebert, C. e Louridas, P.	2023	IEEExplore Scopus	[14]
E5	How Data Scientists Improve Generated Code Documentation in Jupyter Notebooks	Muller, M.; Wang, A. Y.; Ross, S. I.; Weisz, J. D.; Agarwal, M.; Talamadupula, K.; Houde, S.; Martinez, F.; Richards, J.; Drozdal, J.; Liu, X.; Piorkowski, D. e Wang, D.	2023	Scopus	[27]
E6	Large Language Models and Simple, Stupid Bugs	Jesse, K.; Ahmed, To.; Devanbu, P. T. e Morgan, E.	2021	Scopus	[21]
E7	On the Use of ChatGPT to Support Agile Software Development	Bera, P.; Wautelet, Y. e Poels, G.	2023	Scopus	[8]
E8	Practices and Challenges of Using GitHub Copilot: An Empirical Study	Zhang, B.; Liang, P.; Zhou, X.; Ahmad, A. e Waseem, M.	2023	Scopus	[54]
E9	The Potential of Artificial Intelligence as a Method of Software Developer's Productivity Improvement	Moroz, E. A.; Grizkevich, V. O. e Novozhilov, I. M.	2022	Scopus	[26]
E10	Towards Human-Bot Collaborative Software Architecting with ChatGPT	Ahmad, A.; Waseem, M.; Liang, P.; Fahmideh, M.; Aktar, M. S. e Mikkonen, T.	2022	Scopus	[3]
E11	AI-Assisted Security: A Step towards Reimagining Software Development for a Safer Future	Shi, Y.and Sakib, N.; Shahriar, H.; Lo, D.; Chi, H. e Qian, K.	2023	IEEExplore	[41]
E12	An Additional Set of (Automated) Eyes: Chatbots for Agile Retrospectives	Matthies, C.; Dobrigkeit, F. e Hesse, G.	2019	IEEExplore	[25]
E13	ChatGPT's use case for software engineers	Ashraf, A. e Imam, A.	2023	IEEExplore	[5]
E14	Investigating Code Generation Performance of ChatGPT with Crowdsourcing Social Data	Feng, Y.; Vanam, S.; Cherukupally, M.; Zheng, W.; Qiu, M. e Chen, H.	2023	IEEExplore	[16]
E15	Is GitHub Copilot a Substitute for Human Pair-programming? An Empirical Study	Imai, S.	2022	IEEExplore	[20]
E16	On the Use of GPT-4 for Creating Goal Models: An Exploratory Study	Chen, B.; Chen, K.; Hassani, S.; Yang, Y.; Amyot, D.; Lessard, L.; Mussbacher, G.; Sabetzadeh, M. e Varró, D.	2023	IEEExplore	[11]
E17	Assessing the Quality of GitHub Copilot's Code Generation	Yetistiren, B.; Ozsoy, I. e Tuzun, E.	2022	ACM	[53]
E18	Automated Repair of Programs from Large Language Models	Fan, Z.; Gao, X.; Mirchev, M.; Roychoudhury, A. e Tan, S. H.	2023	ACM	[15]
E19	Better Together? An Evaluation of AI-Supported Code Translation	Weisz, J. D.; Muller, M.; Ross, S. I.; Martinez, F.; Houde, S.; Agarwal, M.; Talamadupula, K. e Richards, J. T.	2022	ACM	[49]
E20	Evaluating a Large Language Model on Searching for GUI Layouts	Brie, P.; Burny, N.; Sluÿters, A. e Vanderdonckt, J.	2023	ACM	[10]
E21	Generative AI for Reengineering Variants into Software Product Lines: An Experience Report	Acher, M. e Martinez, J.	2023	ACM	[2]
E22	How Readable is Model-Generated Code? Examining Readability and Visual Inspection of GitHub Copilot	Al Madi, N.	2023	ACM	[4]
E23	In-IDE Code Generation from Natural Language: Promise and Challenges	Xu, F. F.; Vasilescu, B. e Neubig, G.	2023	ACM	[51]
E24	Investigating User Perceptions of Conversational Agents for Software-Related Exploratory Web Search	Frazier, M.; Kumar, S.; Damevski, K. e Pollock, L.	2022	ACM	[17]
E25	Perfection Not Required? Human-AI Partnerships in Code Translation	Weisz, Justin D.; Muller, Michael; Houde, Stephanie; Richards, John; Ross, Steven I.; Martinez, F.; Agarwal, M. e Talamadupula, K.	2021	ACM	[48]
E26	**Taking Flight with Copilot: Early Insights and Opportunities of AI-Powered Pair-Programming Tools**	Bird, C.; Ford, D.; Zimmermann, T.; Forsgren, N.; Kalliamvakou, E.; Lowdermilk, T. e Gazit, I.	2023	ACM	[9]
E27	The Programmer's Assistant: Conversational Interaction with a Large Language Model for Software Development	Ross, Steven I.; Martinez, F.; Houde, S.; Muller, M. e Weisz, J. D.	2023	ACM	[36]

resolução de *bugs* de programação em substituição a ferramentas tradicionais de depuração (*debugging*) de código no estudo E13. Já o estudo E24 avalia o potencial de uso de ACs para auxiliar a busca de informações e aquisição de conhecimento pelos desenvolvedores de software.

4.2.2 SQ2: Como o uso de IA generativa impacta os papéis dos profissionais de TI envolvidos no processo de desenvolvimento de software? O estudo E10, que trata do uso do ChatGPT para geração de um modelo de arquitetura, traz a possibilidade de arquitetos de software delegarem suas tarefas de arquitetura ao *bot* e supervisioná-lo por meio de diálogo em linguagem(s) natural(is) para obter automação, aliviando os arquitetos de realizar tarefas tediosas na engenharia de software orientada à arquitetura [3].

Outros estudos reforçam que o papel dos desenvolvedores de software que adotam a técnica de *pair programming* (programação em pares) durante a codificação pode ser impactado pela substituição de um dos desenvolvedores por ferramentas capazes de gerar e corrigir código-fonte como o Copilot (E15 e E26) e pela utilização destas ferramentas para realizar tarefas monótonas e repetitivas (E9).

Os estudos E4, E15 e E21 mencionam a necessidade de revisão dos artefatos gerados pelas ferramentas de IA generativa que pode afetar o papel do desenvolvedor visto que, além das tarefas relacionadas à codificação, este profissional deverá realizar tarefas relacionadas à supervisão e revisão dos códigos gerados. Esta necessidade é justificada pelo risco de geração de códigos com erros, artefatos incompletos, códigos com viés e as possíveis "alucinações" que fazem com que a ferramenta gere novos dados/informações que não existem [43].

Por fim, conforme o estudo E4, desenvolvedores de software precisarão de novas competências, tais como melhoria de software gerado automaticamente, alimentação dos mecanismos de aprendizagem e exploração dos comportamentos (da ferramente de IA generativa) que não são explicáveis.

4.2.3 SQ3: Como a IA generativa pode impactar a produtividade das atividade do desenvolvimento de software? Os estudos E4, E8, E9 e E15 trazem mais ênfase aos impactos positivos da IA generativa na

produtividade dos profissonas de Engenharia de Software. O estudo E4 sugere que a IA generativa pode ser usada para simplificar o processo de desenvolvimento de software automatizando tarefas, como teste, depuração e implantação, o que pode ajudar a reduzir tempo e custos de desenvolvimento e melhorar a produtividade geral. Este estudo ainda sugere que pode haver otimização na criatividade por meio de sugestões relacionadas a *User Experience (UX)* e design.

No estudo E8, usuários relatam que conseguem codificar de forma mais rápida utilizando o Copilot, o que gera "economia de tempo". No estudo E26, usuários também relatam que, ao utilizar a ferramenta avaliada (Copilot), empregam menos tempo em codificação, mas são necessárias mais revisões de código. Neste estudo também há relatos de que os usuários se sentem mais produtivos ao aceitar sugestões de código da ferramenta, mesmo quando é necessário editar essas sugestões. De acordo com o estudo E19, os modelos generativos podem amplificar o desempenho ou as capacidades de seus usuários.

Os resultados do estudo E15 sugerem aumento de produtividade ao utilizar o Copilot durante a programação em pares com um programador utilizando a ferramenta como copiloto, em comparação à programação em pares realizada por dois programadores humanos.

Contudo, apesar das evidências e relatos de melhoria na produtividade, em alguns estudos, como E22 e E23, não foi possível comprovar ganho de produtividade pelo uso das ferramentas de geração de código-fonte de IA generativa avaliadas. O estudo E22 levanta alguns fatores que podem influenciar a produtividade entre eles a aceitação de códigos gerados pela ferramenta sem a realização de inspeção, que pode introduzir erros na aplicação e demandar mais tempo para a correção destes erros.

5 DISCUSSÃO

Nos estudos em que foram realizadas pesquisas sobre a opinião dos usuários, ocorrerem relatos variados sobre o uso de IA generativa. No estudo E7, foi realizada uma pesquisa de análise de sentimentos de usuários com base em dados de redes sociais sobre o uso do ChatGPT no processo de desenvolvimento Ágil. Segundo o estudo, estes usuários demonstram insegurança sobre a confiabilidade do uso do chatGPT na metodologia ágil.

Grande parte dos estudos selecionados apresenta o uso de IA generativa para facilitação da codificação e para realização de tarefas relacionadas à construção de código seguro [41] e validação de código-fonte gerado por automatização [5], o que indica a aceitação dos usuários e a eficácia das ferramentas para uso em geração de código.

O estudo E21 considera a natureza estocástica dos LLMs como uma barreira que vai exigir revisões mais cautelosas e mais frequentes do código gerado.Os resultados gerados pelo ChatGPT e Copilot não são determinísticos e podem variar em cada execução [3, 31, 53]. O não-determinismo afeta a confiabilidade e a reprodutibilidade da Engenharia de Software Empírica [29]. Esta característica também cria um desafio para abordagens tradicionais de detecção de *malware* baseada em assinaturas de código [26].

O viés de código e informações [3, 16, 43] geradas pelas ferramentas ChatGPT e Copilot, a possibilidade de divulgação de dados privados [7] e o risco de violação de direitos autorais [3, 26] também são preocupações relatadas nos estudos.

Os estudos que tratam de produtividade trazem o assunto de forma genérica e subjetiva, baseando-se na opinião de usuários e especialistas sem apresentar resultados quantitativos. Não foi encontrado nenhum estudo cujo foco fosse retratar resultados a respeito de medições de impactos sobre a produtividade utilizando variáveis específicas para este propósito. Estudos com esta finalidade seriam interessantes para avaliar estes impactos de forma quantitativa.

6 LIMITAÇÕES E AMEAÇAS À VALIDADE

Quanto às limitações, este artigo mapeou apenas estudos que relacionam o uso de IA generativa a três das treze atividades que compõem o processo de desenvolvimento conforme a norma ABNT ISO/IEC 12207 [1]. Além disso, foram utilizadas apenas três bases acadêmicas para busca dos estudos, limitando a seleção de estudos ao conteúdos destas bases. Desta forma, é possível que estudos relevantes disponíveis em outras bases acadêmicas não tenham sido selecionados.

As ameaças à validade identificadas para este estudo, considerando as definições de Zhou et al. [55] e Kitchenham e Charters [23], são as seguintes:

Validade interna: para reduzir a possibilidade de viés durante a seleção dos estudos por parte dos pesquisadores, na 9ª etapa do protocolo, foi feita a leitura parcial dos estudos selecionados até esta etapa por dois pesquisadores e eventuais conflitos sobre a exclusão de estudo foram resolvidos conjuntamente.

Validade externa: trata da capacidade de generalização dos resultados deste estudo. Este estudo não tem como objetivo a generalização dos resultados. Foram consultados apenas estudos disponíveis em bases acadêmicas que podem não refletir todos os usos possíveis das ferramentas e modelos de IA generativa no processo de desenvolvimento. Por esse motivo, não é possível generalizar os resultados para todos os contextos de processo de desenvolvimento de software.

Validade de confiabilidade: diz respeito ao potencial de reprodutibilidade do estudo. Foi definido um protocolo em etapas e, neste estudo, estão detalhados todos os passos para a obtenção dos resultados. Além disso, os autores utilizaram a ferramenta Parsifal para acompanhamento da execução do protocolo e uma planilha específica para controle de leitura e classificação dos artigos selecionados a partir da etapa 9.

7 CONCLUSÃO E TRABALHOS FUTUROS

Apesar de a popularização da IA generativa ser um fenômeno recente, a adoção desta tecnologia nas atividades do processo de desenvolvimento já é uma realidade. Conforme os resultados deste MSL, Codificação e Teste de Software são as atividades em que o uso de ferramentas de IA generativa tem mais adesão e está sendo mais propagado. É provável que isto ocorra devido à existência do Copilot e do modelo Codex que estão se popularizando e são direcionadas para a geração de código-fonte.

As demais atividades carecem de ferramentas de IA generativa que sejam específicas para a realização de suas tarefas e ofereçam o mesmo nível de confiabilidade e especificidade do Copilot. Neste sentido, a maioria dos estudos relacionados a "Análise de Requisitos de Software" e "Projeto Arquitetural do Software" tratam do uso

de ChatGPT, que não é uma ferramenta específica para realização de tarefas de desenvolvimento como o Copilot, ou de ferramentas alternativas baseadas em LLM que ainda não estão popularizadas entre os engenheiros de software. Apesar disso, o uso de IA generativa para realização de tarefas destas atividades também se mostra promissor.

Alguns dos estudos selecionados também relatam que o uso das ferramentas de IA generativa implicará mudanças nos papeis dos profissionais de TI e o desenvolvimento de novas competências para supervisão e revisão de artefatos gerados, alimentação de mecanismos de aprendizagem dos modelos de LM e exploração dos comportamentos das ferramentas que ainda não são explicáveis. Apesar dos resultados positivos relatados em alguns estudos, não há consenso entre os estudos selecionados sobre melhoria da produtividade como consequência do uso das ferramentas de IA generativa.

Entre os principais desafios identificados neste estudo estão o tratamento das questões relacionadas à segurança dos artefatos gerados e seleção e uso de métricas adequadas para avaliação de produtividade na execução de tarefas com o uso das ferramentas de IA generativa.

Como trabalho futuro, a fim de complementar as informações obtidas por meio deste mapeamento, poderá ser realizado um mapeamento mais abrangente contemplando outras bases acadêmicas e outras atividades do processo de desenvolvimento, contemplando também as atividades das etapas de Operação e a Manutenção do processo de desenvolvimento. Por se tratar de um tema recente, os estudos encontrados são majoritariamente novos e, por conta disto, pretende-se incluir o *snowballing* futuramente para que seja possível extrair mais estudos atualizados que referenciem os estudos selecionados neste trabalho.

AGRADECIMENTOS

Os autores agradecem ao CNPq (Proc. 316510/2023-8), FAPERJ (Proc. 211.583/2019), CAPES e UNIRIO pelo suporte.

REFERENCES

[1] ABNT ISO/IEC 12207:2017 2017. *Tecnologia de informação - Processos de ciclo de vida de software.* Standard. Associação Brasileira de Normas Técnicas, Rio de janeiro, BR.

[2] M. Acher and J. Martinez. 2023. Generative AI for Reengineering Variants into Software Product Lines: An Experience Report. In *Proceedings of the 27th ACM International Systems and Software Product Line Conference - Volume B* (Tokyo, Japan) *(SPLC '23)*. Association for Computing Machinery, New York, NY, USA, 57–66. https://doi.org/10.1145/3579028.3609016

[3] A. Ahmad, M. Waseem, P. Liang, M. Fahmideh, M. S. Aktar, and T. Mikkonen. 2023. Towards Human-Bot Collaborative Software Architecting with ChatGPT. *ACM International Conference Proceeding Series* (2023), 279 – 285. https://doi.org/10.1145/3593434.3593468

[4] N. Al Madi. 2023. How Readable is Model-Generated Code? Examining Readability and Visual Inspection of GitHub Copilot. In *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering* (Rochester, MI, USA) *(ASE '22)*. Association for Computing Machinery, New York, NY, USA, Article 205, 5 pages. https://doi.org/10.1145/3551349.3560438

[5] A. Ashraf and A. Imam. 2023. ChatGPT's use case for software engineers. In *8th International Conference on Computing in Engineering and Technology (ICCET 2023)*, Vol. 2023. 487–492. https://doi.org/10.1049/icp.2023.1537

[6] Ö. Aydın and E. Karaarslan. 2023. Is ChatGPT leading generative AI? What is beyond expectations? *Academic Platform Journal of Engineering and Smart Systems* (2023). https://doi.org/10.2139/ssrn.4341500

[7] A. S. Bale, Y. R. Vada, B. E. Oshiojum, U. K. Lakkineni, C. Rao, K. Venkatesh, and I. Rani. 2023. ChatGPT in Software Development: Methods and Cross-Domain Applications. *International Journal of Intelligent Systems and Applications in Engineering* 11, 9s (2023), 636 – 643. https://www.scopus.com/inward/record.uri?eid=2-s2.0-85171329846&partnerID=40&md5=4283848fd2a7f64e2f54575369e84b6a

[8] P. Bera, Y. Wautelet, and G. Poels. 2023. On the Use of ChatGPT to Support Agile Software Development. *CEUR Workshop Proceedings* 3414 (2023), 1 – 9. https://www.scopus.com/inward/record.uri?eid=2-s2.0-85163636200&partnerID=40&md5=95f0a9c9935d75f8ec69d52a7c21670d

[9] C. Bird, D. Ford, T. Zimmermann, N. Forsgren, E. Kalliamvakou, T. Lowdermilk, and I. Gazit. 2023. Taking Flight with Copilot: Early Insights and Opportunities of AI-Powered Pair-Programming Tools. *Queue* 20, 6 (jan 2023), 35–57. https://doi.org/10.1145/3582083

[10] P. Brie, N. Burny, A. Sluÿters, and J. Vanderdonckt. 2023. Evaluating a Large Language Model on Searching for GUI Layouts. *Proc. ACM Hum.-Comput. Interact.* 7, EICS, Article 178 (jun 2023), 37 pages. https://doi.org/10.1145/3593230

[11] B. Chen, K. Chen, S. Hassani, Y. Yang, D. Amyot, L. Lessard, G. Mussbacher, M. Sabetzadeh, and D. Varró. 2023. On the Use of GPT-4 for Creating Goal Models: An Exploratory Study. In *2023 IEEE 31st International Requirements Engineering Conference Workshops (REW)*. 262–271. https://doi.org/10.1109/REW57809.2023.00052

[12] G. Cooper. 2023. Examining science education in chatgpt: An exploratory study of generative artificial intelligence. *Journal of Science Education and Technology* 32, 3 (2023), 444–452. https://doi.org/10.1007/s10956-023-10039-y

[13] P. Dhoni. 2023. Exploring the Synergy between Generative AI, Data and Analytics in the Modern Age. (2023). https://doi.org/10.36227/techrxiv.24045792.v1

[14] C. Ebert and P. Louridas. 2023. Generative AI for Software Practitioners. *IEEE Software* 40, 4 (July 2023), 30–38. https://doi.org/10.1109/MS.2023.3265877

[15] Z. Fan, X. Gao, M. Mirchev, A. Roychoudhury, and S. H. Tan. 2023. Automated Repair of Programs from Large Language Models. In *Proceedings of the 45th International Conference on Software Engineering* (Melbourne, Victoria, Australia) *(ICSE '23)*. IEEE Press, 1469–1481. https://doi.org/10.1109/ICSE48619.2023.00128

[16] Y. Feng, S. Vanam, M. Cherukupally, W. Zheng, M. Qiu, and H. Chen. 2023. Investigating Code Generation Performance of ChatGPT with Crowdsourcing Social Data. In *2023 IEEE 47th Annual Computers, Software, and Applications Conference (COMPSAC)*. 876–885. https://doi.org/10.1109/COMPSAC57700.2023.00117

[17] M. Frazier, S. Kumar, K. Damevski, and L. Pollock. 2022. Investigating User Perceptions of Conversational Agents for Software-Related Exploratory Web Search. In *Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: New Ideas and Emerging Results* (Pittsburgh, Pennsylvania) *(ICSE-NIER '22)*. Association for Computing Machinery, New York, NY, USA, 51–55. https://doi.org/10.1145/3510455.3512778

[18] A. Fuggetta. 2000. Software Process: A Roadmap. In *Proceedings of the Conference on The Future of Software Engineering* (Limerick, Ireland) *(ICSE '00)*. Association for Computing Machinery, New York, NY, USA, 25–34. https://doi.org/10.1145/336512.336521

[19] J. Grundy and J. Hosking. 2001. *Software tools.* The Software Engineering Encyclopedia. Wiley. https://doi.org/10.1002/0471028959.sof332

[20] S. Imai. 2022. Is GitHub Copilot a Substitute for Human Pair-programming? An Empirical Study. In *2022 IEEE/ACM 44th International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*. 319–321. https://doi.org/10.1145/3510454.3522684

[21] K. Jesse, T. Ahmed, P. T. Devanbu, and E. Morgan. 2023. Large Language Models and Simple, Stupid Bugs. *Proceedings - 2023 IEEE/ACM 20th International Conference on Mining Software Repositories, MSR 2023* (2023), 563 – 575. https://doi.org/10.1109/MSR59073.2023.00082

[22] M. Jovanovic and M. Campbell. 2022. Generative artificial intelligence: Trends and prospects. *Computer* 55, 10 (2022), 107–112. https://doi.org/10.1109/MC.2022.3192720

[23] B. A. Kitchenham and S. Charters. 2007. *Guidelines for performing Systematic Literature Reviews in Software Engineering.* Technical Report EBSE 2007-001. Keele University and Durham University Joint Report. https://www.elsevier.com/__data/promis_misc/525444systematicreviewsguide.pdf

[24] J. Lester, K. Branting, and B. Mott. 2004. Conversational agents. *The practical handbook of internet computing* (2004), 220–240.

[25] C. Matthies, F. Dobrigkeit, and G. Hesse. 2019. An Additional Set of (Automated) Eyes: Chatbots for Agile Retrospectives. In *2019 IEEE/ACM 1st International Workshop on Bots in Software Engineering (BotSE)*. 34–37. https://doi.org/10.1109/BotSE.2019.00017

[26] E. A. Moroz, V. O. Grizkevich, and I. M. Novozhilov. 2022. The Potential of Artificial Intelligence as a Method of Software Developer's Productivity Improvement. *Proceedings of the 2022 Conference of Russian Young Researchers in Electrical and Electronic Engineering, ElConRus 2022* (2022), 386 – 390. https://doi.org/10.1109/ElConRus54750.2022.9755659

[27] M. Muller, A. Y. Wang, S. I. Ross, J. D. Weisz, M. Agarwal, K. Talamadupula, S. Houde, F. Martinez, J. Richards, J. Drozdal, X. Liu, D. Piorkowski, and D. Wang. 2021. How Data Scientists Improve Generated Code Documentation in Jupyter Notebooks. *CEUR Workshop Proceedings* 2903 (2021). https://www.scopus.com/inward/record.uri?eid=2-s2.0-85110412324&partnerID=40&md5=a5c242d318b8bd3780fdbc3dac45fcce

[28] B. M. Napoleão, K. R. Felizardo, É. F. de Souza, and N. L. Vijaykumar. 2017. Practical similarities and differences between Systematic Literature Reviews and Systematic Mappings: a tertiary study. In *International Conference on Software Engineering and Knowledge Engineering*. https://api.semanticscholar.org/CorpusID:39847813

[29] S. Ouyang, J. M Zhang, M. Harman, and M. Wang. 2023. LLM is Like a Box of Chocolates: the Non-determinism of ChatGPT in Code Generation. *arXiv preprint arXiv:2308.02828* (2023).

[30] I. Ozkaya. 2023. The Next Frontier in Software Development: AI-Augmented Software Development Processes. *IEEE Software* 40, 4 (July 2023), 4–9. https://doi.org/10.1109/MS.2023.3278056

[31] O. S. Ozturk, E. Ekmekcioglu, O. Cetin, B. Arief, and J. Hernandez-Castro. 2023. New Tricks to Old Codes: Can AI Chatbots Replace Static Code Analysis Tools? *(EICC '23)*. Association for Computing Machinery, New York, NY, USA, 13–18. https://doi.org/10.1145/3590777.3590780

[32] K. Petersen, S. Vakkalanka, and L. Kuzniarz. 2015. Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and software technology* 64 (2015), 1–18.

[33] M. Petticrew and H. Roberts. 2008. *Systematic reviews in the social sciences: A practical guide*. John Wiley & Sons.

[34] N. M. Radziwill and M. C. Benton. 2017. Evaluating quality of chatbots and intelligent conversational agents. *arXiv preprint arXiv:1704.04579* (2017).

[35] M.J. Rees. 2002. A feasible user story tool for agile software development?. In *Ninth Asia-Pacific Software Engineering Conference, 2002.* 22–30. https://doi.org/10.1109/APSEC.2002.1182972

[36] S. I. Ross, F. Martinez, S. Houde, M. Muller, and J. D. Weisz. 2023. The Programmer's Assistant: Conversational Interaction with a Large Language Model for Software Development. In *Proceedings of the 28th International Conference on Intelligent User Interfaces* (Sydney, NSW, Australia) *(IUI '23)*. Association for Computing Machinery, New York, NY, USA, 491–514. https://doi.org/10.1145/3581641.3584037

[37] N. B. Ruparelia. 2010. Software Development Lifecycle Models. *SIGSOFT Softw. Eng. Notes* 35, 3 (may 2010), 8–13. https://doi.org/10.1145/1764810.1764814

[38] D. Russo. 2023. Navigating the complexity of generative ai adoption in software engineering. *arXiv preprint arXiv:2307.06081* (2023). https://doi.org/10.48550/arXiv.2307.06081

[39] F. A. Sakib, S. H. Khan, and A. Karim. 2023. Extending the Frontier of ChatGPT: Code Generation and Debugging. *arXiv preprint arXiv:2307.08260* (2023). https://doi.org/10.48550/arXiv.2307.08260

[40] S. P. Shankar and S. S. Chaudhari. 2023. Framework for the Automation of SDLC Phases using Artificial Intelligence and Machine Learning Techniques. *International Journal on Recent and Innovation Trends in Computing and Communication* 11, 6 s (2023), 379 – 390. https://doi.org/10.17762/ijritcc.v11i6s.6944

[41] Y. Shi, N. Sakib, H. Shahriar, D Lo, H. Chi, and K. Qian. 2023. AI-Assisted Security: A Step towards Reimagining Software Development for a Safer Future. In *2023 IEEE 47th Annual Computers, Software, and Applications Conference (COMPSAC)*. 991–992. https://doi.org/10.1109/COMPSAC57700.2023.00142

[42] B. Singh and S. Gautam. 2016. The Impact of Software Development Process on Software Quality: A Review. In *2016 8th International Conference on Computational Intelligence and Communication Networks (CICN)*. 666–672. https://doi.org/10.1109/CICN.2016.137

[43] S. S. Sohail, F. Farhat, Y. Himeur, M. Nadeem, D. Ø. Madsen, Y. Singh, S. Atalla, and W. Mansoor. 2023. Decoding ChatGPT: A taxonomy of existing research, current challenges, and possible future directions. *Journal of King Saud University - Computer and Information Sciences* 35, 8 (2023). https://doi.org/10.1016/j.jksuci.2023.101675

[44] J. Sun, Q. V. Liao, M. Muller, M. Agarwal, S. Houde, K. Talamadupula, and J. D. Weisz. 2022. Investigating Explainability of Generative AI for Code through Scenario-Based Design. In *27th International Conference on Intelligent User Interfaces* (Helsinki, Finland) *(IUI '22)*. Association for Computing Machinery, New York, NY, USA, 212–228. https://doi.org/10.1145/3490099.3511119

[45] N. M. S. Surameery and M. Y. Shakor. 2023. Use chat gpt to solve programming bugs. *International Journal of Information Technology & Computer Engineering (IJITC) ISSN: 2455-5290* 3, 01 (2023), 17–22.

[46] V. Taecharungroj. 2023. "What Can ChatGPT Do?" Analyzing Early Reactions to the Innovative AI Chatbot on Twitter. *Big Data and Cognitive Computing* 7, 1 (2023), 35. https://doi.org/10.3390/bdcc7010035

[47] S. Wagner and F. Deissenboeck. 2019. Defining productivity in software engineering. *Rethinking productivity in software engineering* (2019), 29–38.

[48] J. D. Weisz, M. Muller, S. Houde, J. Richards, S. I. Ross, F. Martinez, M. Agarwal, and K. Talamadupula. 2021. Perfection Not Required? Human-AI Partnerships in Code Translation. In *26th International Conference on Intelligent User Interfaces* (College Station, TX, USA) *(IUI '21)*. Association for Computing Machinery, New York, NY, USA, 402–412. https://doi.org/10.1145/3397481.3450656

[49] J. D. Weisz, M. Muller, S. I. Ross, F. Martinez, S. Houde, M. Agarwal, K. Talamadupula, and J. T. Richards. 2022. Better Together? An Evaluation of AI-Supported Code Translation. In *27th International Conference on Intelligent User Interfaces* (Helsinki, Finland) *(IUI '22)*. Association for Computing Machinery, New York, NY, USA, 369–391. https://doi.org/10.1145/3490099.3511157

[50] J. Weizenbaum. 1966. ELIZA—a Computer Program for the Study of Natural Language Communication between Man and Machine. *Commun. ACM* 9, 1 (jan 1966), 36–45. https://doi.org/10.1145/365153.365168

[51] F. F. Xu, B. Vasilescu, and G. Neubig. 2022. In-IDE Code Generation from Natural Language: Promise and Challenges. *ACM Trans. Softw. Eng. Methodol.* 31, 2, Article 29 (mar 2022), 47 pages. https://doi.org/10.1145/3487569

[52] B. Yetistiren, I. Ozsoy, and E. Tuzun. 2022. Assessing the Quality of GitHub Copilot's Code Generation. In *Proceedings of the 18th International Conference on Predictive Models and Data Analytics in Software Engineering* (Singapore, Singapore) *(PROMISE 2022)*. Association for Computing Machinery, New York, NY, USA, 62–71. https://doi.org/10.1145/3558489.3559072

[53] B. Yetistiren, I. Ozsoy, and E. Tuzun. 2022. Assessing the Quality of GitHub Copilot's Code Generation. In *Proceedings of the 18th International Conference on Predictive Models and Data Analytics in Software Engineering* (Singapore, Singapore) *(PROMISE 2022)*. Association for Computing Machinery, New York, NY, USA, 62–71. https://doi.org/10.1145/3558489.3559072

[54] B. Zhang, P. Liang, X. Zhou, A. Ahmad, and M. Waseem. 2023. Practices and Challenges of Using GitHub Copilot: An Empirical Study. *Proceedings of the International Conference on Software Engineering and Knowledge Engineering, SEKE* 2023-July (2023), 124 – 129. https://doi.org/10.18293/SEKE2023-077

[55] X. Zhou, Y. Jin, H. Zhang, S. Li, and X. Huang. 2016. A Map of Threats to Validity of Systematic Literature Reviews in Software Engineering. In *2016 23rd Asia-Pacific Software Engineering Conference (APSEC)*. 153–160. https://doi.org/10.1109/APSEC.2016.031