Fairness in Multi-Agent Systems for Software Engineering: An SDLC-Oriented Rapid Review

Corey Yang-Smith
University of Calgary
Calgary, Canada
corey.yangsmith@ucalgary.ca

Ronnie de Souza Santos
University of Calgary
Calgary, Canada
ronnie.desouzasantos@ucalgary.ca

Ahmad Abdellatif
University of Calgary
Calgary, Canada
ahmad.abdellatif@ucalgary.ca

Abstract

Transformer-based large language models (LLMs) and multi-agent systems (MAS) are increasingly embedded across the software development lifecycle (SDLC), yet their fairness implications for developer-facing tools remain underexplored despite their growing role in shaping what code is written, reviewed, and released. We present a rapid review of recent work on fairness in MAS, emphasizing LLM-enabled settings and relevance to software engineering. Starting from an initial set of 350 papers, we screened and filtered the corpus for relevance, retaining 18 studies for final analysis. Across these 18 studies, fairness is framed as a combination of trustworthy AI principles, bias reduction across groups, and interactional dynamics in collectives, while evaluation spans accuracy metrics on bias benchmarks, demographic disparity measures, and emergent MAS-specific notions such as conformity and bias amplification. Reported harms include representational, quality-of-service, security and privacy, and governance failures, which we relate to SDLC stages where evidence is most and least developed. We identify three persistent gaps: (1) fragmented, rarely MAS-specific evaluation practices that limit comparability, (2) limited generalization due to simplified environments and narrow attribute coverage, and (3) scarce, weakly evaluated mitigation and governance mechanisms aligned to real software workflows. These findings suggest MAS fairness research is not yet ready to support deployable, fairness-assured software systems, motivating MAS-aware benchmarks, consistent protocols, and lifecycle-spanning governance.

CCS Concepts

• **Software and its engineering** → Software creation and management; • **Computing methodologies** → **Multi-agent systems**; *Artificial intelligence*.

Keywords

fairness, multi-agent systems, large language models, software engineering, software development life cycle

1 Introduction

Transformer-based large language models (LLM) are increasingly used across the Software Development Life Cycle (SDLC), supporting tasks such as code generation [46], testing [63], deployment [37], and maintenance [60]. The capabilities of these models have been further enhanced through research in reasoning and tool-use frameworks such as ReAct [61] and chain-of-thought prompting [56]. Recent architectural trends increasingly favor Multi-Agent Systems (MAS), in which multiple LLM-based agents collaborate on complex tasks [22], often integrating with external tools and platforms through emerging standards such as MCP [5] and A2A [54]. This trend expands their influence on the SDLC, both in developer use and product integration, raising urgent questions about fairness, accountability, and responsible AI in software deployments.

Despite rapid advances in LLM research, systematic study of fairness in software engineering remains limited [57]. Transparency challenges persist due to limited disclosure of training data, which can propagate bias into downstream tasks [14, 45]. AI assistants and low-code agentic tools are increasingly embedded in developer workflows and accessible to non-technical users [12, 18, 32, 39, 42, 48, 64]. As software engineering shifts towards AI-native development, often described as "Software 3.0" [20, 21], agent-based architectures support collaboration, delegation, and autonomous execution alongside human developers. In this context, fairness extends beyond model outputs to include decision visibility, responsibility allocation, and human oversight, with implications for both developer workflows and downstream software products [49].

This rapid review synthesizes current research on fairness in multi-agent software systems. After screening an initial set of 350 papers, retaining 18 studies for final analysis, we examine how fairness is defined and measured across existing LLM-based MAS studies, identify the types of harms and biases that arise within the SDLC, and highlight where methodological or empirical gaps persist. This review contributes to responsible AI practice by offering guidance on how transparency, accountability, and equity can be strengthened as multi-agent architectures become increasingly explored and integrated into software engineering workflows and products. Our review focuses on three research questions:

- **RQ1:** How is fairness defined and measured within existing studies of multi-agent systems?
- **RQ2:** What types of biases, harms, or inequitable outcomes have been identified across different stages of the SDLC in these systems?
- **RQ3:** Where do current research gaps lie in promoting fairness, accountability, and transparency within multi-agent software ecosystems?

The remainder of this paper is organized as follows: Section 2 reviews background and related work; Section 3 describes the rapid review protocol; Section 4 presents the results and synthesis across the three research questions; and Sections 5 and 6 discuss limitations and conclusions.

2 Background and Related Work

LLMs and MAS in Software Engineering. Since the introduction of the transformer [55], LLMs have been applied across many Software Engineering (SE) tasks, including code generation, review, refactoring, and log analysis [24]. Early work largely studied single-agent assistants spanning one or more SDLC stages, evaluating how

much a single model can support development and the resulting changes in reliability, maintainability, and control [28].

Recent research increasingly adopts multi-agent architectures that emulate software teams by assigning roles or coordinating agent pipelines [33]. Systems such as ChatDev [46] and MetaGPT [23] implement waterfall-style processes while AgileCoder [40] emphasizes iterative collaboration. Other work targets specific development activities such as IaC generation [30] and unit test synthesis and evaluation [58]. These MAS-based approaches align with AI-native "Software 3.0" development [20, 21], but there remains a gap in understanding fairness, bias, and equitable treatment in MAS-supported SE [57].

Fairness in LLM-Assisted Software Engineering. Fairness has emerged as an important but underexplored concern in applying AI to SE. Initial studies have shown that LLM generated code and technical suggestions can exhibit social biases toward certain demographic groups, for example by producing stereotyped identifiers, comments, or documentation [16, 31]. Benchmark datasets such as MALIBU [38] and BBQ [44] reveal that persona-based LLM interactions can surface implicit gender, racial, and religious biases in model behavior. Other work focuses on real-world, human-centered coding scenarios, such as designing coding tasks and judgmental prompts that intentionally reveal latent biases in how LLMs complete or explain code [34].

Within text-to-code settings, research by Liu [35] and Huang [25] propose structured prompts, method signatures, and translation tasks to measure and mitigate bias in code synthesis and analysis. While these studies mainly address single model systems, they show that SE applications are subject to the same fairness challenges observed in natural language generation. They also highlight tradeoffs between bias reduction and downstream performance, described as an "alignment tax" by Xu [59], which are especially relevant when LLMs are embedded in developer tools. Beyond SE, a growing body of literature investigates fairness in MAS more generally, including:

- MAS benchmarks evaluating implicit bias and differential treatment across personas or identity attributes [7, 38]
- Debate and deliberation frameworks that study group conformity [10], political bias [6], and minority suppression in LLM-based social simulations [10], and
- Conceptual and empirical work on responsibility allocation [49], group blameworthiness [62], and collective misalignment in multi agent decision making [15].

While many of these studies are not grounded in SE, they propose definitions, metrics, and evaluation patterns that can be adapted to MAS embedded in software workflows and deployed within MAS-based software systems. For example, they introduce measures for bias amplification [41], group conformity [10], and collective misalignment [15] that become relevant when multiple agents coordinate to propose designs, prioritize requirements, or vote on candidate patches.

Responsible AI, Governance, and Trustworthiness Frameworks. Parallel to technical advances in LLMs and MAS, there is growing emphasis directed toward responsible AI frameworks and regulatory guidance. The EU AI Act [2], alongside related initiatives [50], formalize expectations around fairness, transparency, accountability, human oversight, and justice in AI systems. Recent studies [13, 47, 49] operationalize these principles within software engineering contexts, outlining concrete requirements for AI-enabled workflows such as recruitment systems, DevOps automation, and human-in-the-loop collaboration. These works typically frame fairness as one dimension of trustworthy AI, alongside robustness, safety, privacy, and explainability. They discuss practices such as risk assessment, model documentation, continuous monitoring, and human-centric validation. However, they often focus on single systems or high-level governance rather than multi-agent architectures, and they rarely provide detailed fairness metrics or SDLC applicable evaluation protocols.

Taken together, existing research on MAS in SE, fairness in LLM-based tools, and responsible AI governance reveals a fragmented landscape. Although multi-agent architectures are increasingly integrated into software workflows, fairness-focused analyses remain fragmented and are often disconnected from SDLC practices. These disparities highlight the need for work that consolidates definitions, evaluation approaches, and fairness implications specific to MAS in SE.

3 Methodology

This study followed a **rapid review** methodology as described by Cartaxo [8], prioritizing a timely synthesis by narrowing the search scope and using time-bounded screening of contemporary research, rather than the exhaustive coverage typical of traditional systematic literature reviews, as performed by other papers within the SE/LLM field [17, 27, 29]. This approach was chosen to capture recent work on fairness in multi-agent software systems, given the rapid pace of development in this area and the emerging need to address fairness concerns. The following subsections describe the procedures used throughout the study.

3.1 Search Strategy

Sources and Time Span: We searched the ACM Digital Library [1], IEEE Xplore [4], and Google Scholar [3] for works published from 2017 to 2025. The search was conducted on November 15, 2025.

Search Strategy: We conducted a primary search across all databases using the string in Figure 1, which was formed by combining fairness, multi-agent, and software engineering terms, retaining the top 100 results per database by relevance (300 total). We then ran a focused secondary search in Google Scholar using Figure 2, retaining 50 additional records.

Snowballing: One round of backward and forward snowballing was applied to seed papers identified through both searches to retrieve further relevant work.

3.2 Inclusion, Exclusion and Screening Process

After conducting the initial identification, papers were screened in multiple stages outlined below using the inclusion and exclusion criteria in Table 1. We included both peer-reviewed conference and journal papers as well as preprints to capture recent developments in this area.

- **Deduplication:** Remove duplicates and superseded versions.

```
("fairness" OR "bias" OR "equity" OR "justice" OR "discrimination" OR "harm" OR "accountability") AND ("multi-agent" OR "agentic system" OR
"autonomous agent" OR "LLM-based agent" OR "multi-agent system" OR "MAS" OR "AI agent") AND ("software engineering" OR "software development
lifecycle" OR "SDLC" OR "requirements engineering" OR "software design" OR "software architecture" OR "implementation" OR "testing" OR "software
maintenance" OR "software evolution" OR "deployment" OR "operations" OR "MLOps" OR "DevOps")
```

Figure 1: Search strings to use across all databases in primary search.

- **Title and Abstract Screening:** Screen titles and abstracts to exclude irrelevant works; include peer-reviewed conference/journal papers and preprints.
- **Partial Text Screening:** Review Abstract, Introduction, Results, and Conclusion to assess relevance and record preliminary notes.

Study Selection Results. Figure 3 summarizes the number of records retained and excluded at each stage, resulting in 18 included studies.

3.3 Analysis Approach

To address our research questions, we conducted a structural analysis of all studies that passed the screening process. The analysis was based on targeted data extraction and combined descriptive summarization with thematic synthesis. Our goal was to characterize how fairness is defined and assessed in multi-agent systems, identify documented harms across the SDLC, and map remaining research gaps. For studies not explicitly situated in SE, we applied a predefined SDLC mapping scheme that assigns reported harms and evaluation mechanisms to SDLC stages based on their role in software workflows, rather than the original application domain.

3.3.1 Information Extraction. After study selection, we performed structured data extraction on each included paper to support the research questions. The first author skimmed the full text end-to-end, with particular attention to the Introduction, Methodology, Evaluation, Threats to Validity, and Future Work sections, revisiting specific parts as needed to clarify definitions, experimental setup, or findings related to fairness or multi-agent interactions. During the process, detailed notes were recorded in a spreadsheet[1] using a predefined extraction schema aligned with the research questions and organized into five categories:

- **Fairness Definitions (RQ1):** How each study defined or conceptualized fairness in multi-agent contexts.
- **Fairness Evaluation Metrics (RQ1):** The metrics, criteria, or evaluation methods used to assess fairness or bias.
- **Harms and Inequities (RQ2):** The types of biases, harms, or inequitable outcomes reported or analyzed.
- **SDLC Relevance (RQ2):** The stages of the SDLC to which the harms, fairness concerns, or interventions applied.
- **Gaps and Future Work (RQ3):** Reported limitations, open problems, or directions for future research.

[1]The coding spreadsheet used in this study is available at: Google Sheets

```
"Multi-agent LLM" AND "Bias"
```

Figure 2: Search string used across Google Scholar during secondary review

Considering the inclusion of non-SE studies, we treated a study as transferable to the SE domain when its agent interaction mechanism was domain-agnostic (e.g., debate, role assignment, consensus), its fairness failure mode was interaction-driven (e.g., amplification, conformity, minority suppression), and its evaluation could be instantiated on SE artifacts or workflows (e.g., requirements prioritization, design review, code review, test generation, or incident response). Figure 4 illustrates the analytical traceability from extracted study evidence to SDLC-aligned harms and themes using a representative example. Following extraction, we conducted iterative thematic synthesis by grouping recurring definitions, metrics, and reported harms across studies and refining themes until stable categories emerged that aligned with the research questions. These themes structure the synthesis presented in the Results and Discussion section. Excluding preprints did not materially change the review's main findings, as the same high-level themes and research gaps remained visible in the peer-reviewed subset. Due to the rapid review scope, full independent double screening and coding were not feasible. A second author reviewed the extracted data and coding decisions for all included studies, and disagreements were resolved by consensus. This improved consistency, but does not replace formal inter-rater reliability assessment.

3.3.2 Analysis Plan. We applied a mixed-method analysis that combined structural coding with qualitative synthesis.

RQ1: Fairness Conceptualization and Measurement. We coded how fairness was defined or framed in each study and catalogued all reported evaluation metrics. These codes were then grouped to identify recurring conceptualizations, theoretical perspectives, and measurement practices. Labels were assigned based on the study's primary fairness objective stated in the framing and

Table 1: Screening criteria for study selection.

Include
Empirical evaluation or explicit conceptual/method contribution on multi-agent systems.
Fairness focus (bias, equity, discrimination, accountability) in multi-agent settings relevant to SE workflows or inter-agent interaction.
Proposes/evaluates fairness methods, metrics, frameworks, or benchmarks with transferable technical value to AI/SE.
English; 2017–2025.
Exclude
Non-SE applications where fairness analysis is not transferable to AI/SE practice.
Single-agent LLM studies without multi-agent interaction.
Non-technical ethics discussion without MAS details, evaluation, or SE implications.
Duplicates/superseded versions; full text unavailable.

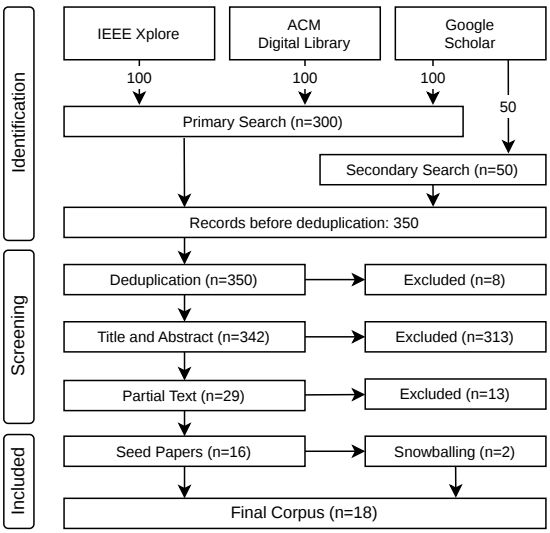

Figure 3: Study selection flow for the rapid review.

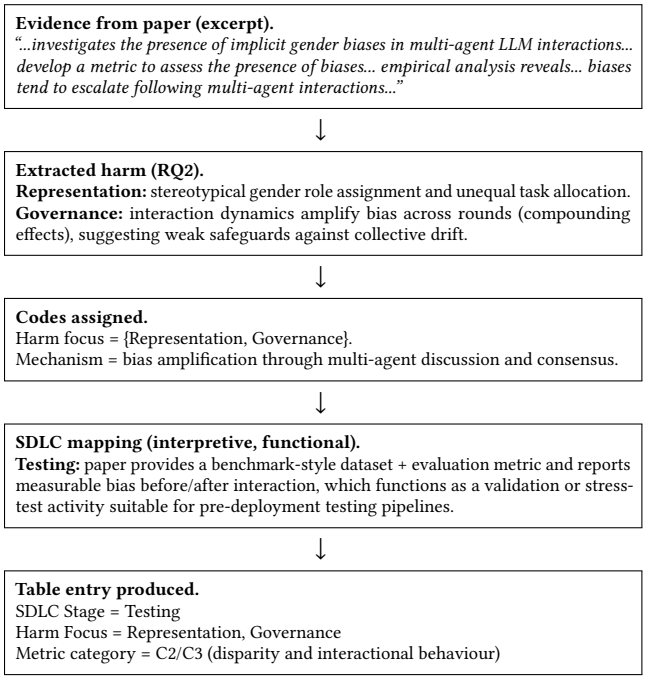

Figure 4: Coding example: how extracted evidence about implicit gender bias amplification is coded as representation and governance harm and mapped to the Testing stage via a functional SDLC interpretation [7].

evaluation design. Metric categories (C1 to C4) were assigned based on the measurement used in the main evaluation; studies using multiple measurement types were assigned multiple categories.

RQ2: Biases, Harms, and Inequitable Outcomes. We analyzed reported harms and biases using thematic synthesis, grouping similar inequities and failure modes across studies. In parallel, we mapped each identified harm to relevant stages of the SDLC to understand where fairness concerns arise within SE workflows. For studies not explicitly situated in SE contexts, this mapping was performed interpretively based on the functional role of the reported mechanism, evaluations, or interactions (e.g., testing, validation, or deployment), rather than the original application domain.

We mapped each harm or mechanism to SDLC stages using decision rules: (i) governance, policy, or compliance constraints were coded as Requirements, (ii) mechanisms introduced as architectural or interaction controls were coded as Design; (iii) harms surfaced via benchmarks, stress tests, or disparity analyses were coded as Testing; (iv) and runtime amplification, drift, or operational failures were coded as Maintenance.

RQ3: Research Gaps and Future Directions. We analyzed limitations and future work statements across studies and clustered them into higher-level themes. These themes were then cross-referenced with findings from RQ1 and RQ2 to identify underexplored SDLC stages, missing evaluation approaches, and inconsistencies in how fairness is assessed and operationalized.

This analysis produced both a conceptual characterization of fairness in MAS and a structural map of where literature is concentrated and where gaps remain.

4 Results and Discussion

In this section, we synthesize findings from the included studies with respect to our three research questions. We highlight common patterns, points of divergence, research gaps, and implications for multi-agent software systems across the SDLC. Table 2 summarizes the included studies; the grouping rationale and synthesis dimensions used in the table are introduced in RQ1.

4.1 RQ1: Fairness Definitions and Measurement

Fairness Definitions. Across the multi-agent LLM literature, fairness tends to appear in three overlapping strands that reflect how authors define fairness goals and what they treat as evidence of progress. The first strand focuses on **reducing social bias** and group fairness in persona-based multi-agent simulations, where fairness means avoiding implicit or explicit stereotypes across protected groups such as gender [7], race, religion, disability, and politics [6]. This line of work often relies on dedicated evaluation resources, such as the bias benchmark BBQ [44], including some proposed benchmarks such as MALIBU [38]. Bias mitigation approaches such as MOMA [59] aim to reduce bias. Across the 18 included studies, gender was the most commonly examined attribute ($n = 7$), followed by race ($n = 6$), age ($n = 4$), politics ($n = 3$), disability ($n = 3$), and intersectional bias ($n = 1$).

A second strand ties fairness explicitly to normative and regulatory notions of justice, transparency, and non-discrimination, often referencing principles from the EU AI Act [2] and ISO/IEC 42001:2023 [26] and considering fairness as a property of responsible human-agent collaboration and **ethical governance** rather than a purely statistical target [13, 47, 49].

A third strand frames fairness through interactional and procedural dimensions within **multi-agent dynamics**. This view focuses on how responsibility and influence are distributed among

Table 2: Overview of sources grouped by fairness approach. Metric categories: C1 Benchmark Performance, C2 Group Disparity, C3 Interactional or MAS Behavioural, C4 Conceptual or Normative.

Fairness Definition	Metrics	SDLC Stage	Harm Focus	Peer Reviewed
Reduce Bias (n = 9)				
Gosmar and Dahl [19]	C4	Maintenance	Security, Quality of Service, Governance, Trust	–
Lünstedt and Schlippe [36]	C1	Design, Testing	Representation	✓
Bandaru et al. [6]	C2, C3	Testing	Representation, Quality of Service, Governance	–
Nguyen et al. [41]	C1, C3	N/A	Representation, Security	–
Coppolillo et al. [11]	C2, C3	Testing	Representation, Security, Quality of Service, Governance	–
Xu et al. [59]	C1	N/A	Representation, Quality of Service	✓
Solomon et al. [53]	C1	Design, Testing	Security, Quality of Service	–
Borah and Mihalcea [7]	C2	Testing	Representation, Governance	✓
Mirza et al. [38]	C1, C2	Testing	Representation, Governance	✓
Ethical and Trustworthy AI (n = 5)				
Sharanarthi [52]	C4	Development	Equity, Sustainability	✓
Cerqueira et al. [13]	C4	Requirements, Development	Representation, Quality of Service, Governance, Trust	–
Ronanki [49]	C4	Requirements, Design	Governance, Trust	✓
Oriol et al. [43]	C1	Requirements	Quality of Service	✓
Raza et al. [47]	C4	Full SDLC	Security, Quality of Service, Governance	✓
Inter-Agent Dynamics (n = 4)				
Choi et al. [9]	C3	Testing	Representation, Governance, Trust	–
Flint et al. [15]	C3	Testing	Quality of Service	–
Choi et al. [10]	C3	Design, Testing	Quality of Service	✓
Yazdanpanah et al. [62]	C4	Requirements	Governance	✓

agents [62], and how inter-agent dependency can contribute to unequal outcomes or propagate bias [15]. Flint et al. conceptualize fairness as the mitigation of political bias amplification during conversational debate, where agent viewpoints can converge and reinforce prejudice [15]. Choi et al. expands on this by treating fairness as the prevention of group conformity effects that accumulate over multiple interaction rounds [10], with another study proposing agent anonymization as a prevention mechanism [9].

Fairness Measurement. Measurement choices align with the four metric categories summarized in Table 2.

C1 (Benchmark Performance) captures conventional predictive quality, where studies report accuracy, F1, or related scores on bias benchmarks such as BBQ [44]. In this setting, improved benchmark performance is treated as a proxy for reduced bias or more equitable behavior [36, 43].

C2 (Group Disparity) measures fairness through between-group comparisons, examining whether scores, labels, or response distributions differ across protected attributes. These analyses may incorporate statistical testing to assess whether the observed gaps are significant [10] and, in some cases, rely on LLM-as-a-judge protocols [65] to score outputs before computing disparities [11].

C3 (Interactional or MAS Behavioural) reflects metrics tailored to multi-agent settings, where fairness concerns emerge from coordination dynamics rather than single-agent outputs. Representative approaches include measuring conformity or herding in debate and deliberation [10], estimating collective misalignment in coordination problems [15], and tracking propagated social cost or amplification effects over repeated interaction rounds [41].

Finally, **C4 (Conceptual or Normative)** is common in SE-oriented agent frameworks, where fairness is discussed as part of broader Trustworthy AI objectives such as explainability [51, 53], security [19], accountability, and sustainability [52]. While these works provide important design guidance, many of these studies do not define explicit fairness metrics, leaving a methodological gap and a lack of alignment between conceptual proposals and evaluation practices.

Taken together, the three fairness strands identified not only define fairness differently but also measure it in largely non-overlapping ways: bias-reduction studies rely on benchmark accuracy and group disparity (C1/C2) but rarely assess interactional effects; inter-agent dynamics work introduces MAS-specific metrics such as conformity and amplification (C3) but seldom connects these to downstream demographic disparities; and governance-oriented studies discuss fairness normatively (C4) without specifying measurable thresholds. Because each strand evaluates a different facet of fairness using self-contained methods, results cannot be compared across strands, and no single study captures the full range of fairness concerns that arise when multiple agents coordinate within a software workflow.

4.2 RQ2: Biases and Harms Across the SDLC

Across the reviewed work, relatively few studies directly examine LLM-based multi agent systems as developer-facing tools or as integrated components in software products. Instead, most evaluate MAS in general purpose settings such as social simulations, political discourse, or safety monitoring. Therefore, to answer RQ2 we first synthesize the harms identified in general MAS use, then map where these harms appear (or are likely to appear) across the SDLC.

At a system level, the literature consistently reports several types of harm: representation harms (stereotypes, underrepresented viewpoints, and marginalization of protected groups), quality of service harms (unequal accuracy, hallucinations, misdiagnosis or unsafe recommendations), security and privacy harms (prompt injection, data exfiltration, and cascading attacks), and governance and trust challenges (opaque decision pathways, unclear responsibility, weak oversight). Several works also highlight bias amplification in multi

agent interactions, where group conformity and persona reinforcement intensify existing bias [10], as well as over-correction risks where mitigation strategies such as anonymization reduce accuracy or introduce new inequities [9]. Among the reviewed MAS interaction patterns, two are particularly consequential: (i) iterative consensus through debate produces group conformity that progressively suppresses minority viewpoints, with severity scaling with group size and round count [10, 15], while (ii) role-based task decomposition propagates stereotypical associations embedded in persona descriptions through the division of labor, causing downstream agents to inherit and amplify upstream biases [41].

When mapped onto the SDLC, these harms appear most predominantly at requirements, design, testing, and maintenance. Requirements engineering can encode or obscure ethical and legal constraints when MAS are used in domains such as recruitment or media integrity, raising representational harms if scenarios or prompts are biased [13]. Architectural work treats fairness, security, and observability as cross-cutting design concerns, proposing role based access control, modular security and monitoring layers, and anonymization pipelines, each of which can shape downstream inequities [9]. Testing oriented benchmarks for conversational bias provide stress tests that could be integrated into pre-deployment pipelines, though they currently remain largely separate from mainstream SE testing practice [59]. Finally, security and observability frameworks position MAS as part of the operational fabric, where prompt injections, data leakage, hallucinations, and cascading failures surface in production logs or incident response [19]. Taken together, these studies outline a wide range of harms across the SDLC, but the evidence remains uneven. Much of the work examines requirements, bias benchmarking, and security scenarios, while everyday developer activities such as design iteration, code review, and debugging of MAS-enabled tools receive limited attention.

4.3 RQ3: Research Gaps and Implications for Fair MAS

Although interest in multi-agent systems is accelerating, the evidence base remains hard to trust in practice because key claims are rarely tested under comparable agent-relevant conditions. Across the reviewed studies, shortcomings cluster around (i) evaluation that does not isolate agentic effects, (ii) coverage that is too configuration bound to generalize, and (iii) mitigations that are proposed more often than they are implemented.

Evaluation Infrastructure. A consistent limitation is that "fairness" (and related accountability or transparency goals) is usually evaluated in ad-hoc settings that conflate agent interaction with task choice and evaluation design. Studies frequently vary core factors that directly shape group behaviour, such as the coordination protocol (debate, voting, delegation), agent count and role structure, memory and tool access, and prompting strategies. Many evaluations also rely on single tasks or general-purpose bias datasets that were not designed to probe interactional harms like conformity, polarization, dominance by a single agent, or bias amplification over multi-turn conversations. This makes cross-paper comparison difficult and weakens any architectural-level takeaway, as the differences in outcomes may reflect evaluation design rather than properties of multi-agent governance.

Generalization. A second gap is that results are often demonstrated in one narrow configuration (one foundation model, one interaction protocol, or one programming language), leaving it unclear which observed effects are intrinsic to multi-agent settings versus artifacts of the chosen setup. Coverage is also narrow in terms of harm dimensions: bias frequently centers on a small set of attributes (commonly gender or political preference), with limited attention to intersectional, linguistic, or domain-specific harms that are plausibly triggered by agent specialization and division of labor. Only one study in our corpus explicitly reports uneven performance across bias categories [36], and the literature rarely investigates *why* these disparities emerge, for example whether they are driven by aggregation rules, role assignment, or task decomposition strategies.

Mitigation and Integration. Finally, the corpus is richer in problem framing and risk identification than in deployable mitigations. Many papers surface interactional failure modes (bias amplification, group conformity, misaligned collective decisions), but few implement mitigations with ablations that separate "agent effects" from "prompting effects", compare alternatives under shared protocols, or evaluate practical constraints such as computing cost, scaling with agent count, and human oversight requirements. Without mitigations that are evaluated under MAS-relevant stressors and embedded into realistic SE pipelines, current progress remains closer to diagnosis than to operational governance for fair, accountable, and transparent multi-agent software ecosystems.

5 Study Limitations

This rapid review is subject to several limitations that should be considered when interpreting the findings.

Rapid Review Constraints: As a rapid review, this study prioritizes timeliness over exhaustiveness. Searches were limited to selected databases, and pragmatic screening decisions were made under time and resource constraints, meaning some relevant studies may have been missed. We also included preprints to capture emerging work; however, their inclusion did not materially alter the main themes or conclusions. Since LLM-based MAS are evolving rapidly, this review should also be understood as a snapshot of the field at the time of study, and some conclusions may become outdated as new architectures, evaluation methods, and fairness interventions emerge. Therefore, the corpus is best interpreted as a well-motivated sample rather than a complete census. In addition, because the review focuses on LLM-based MAS in SE, the findings may not generalize to non-LLM MAS or to domains with different agent capabilities, interaction patterns, or fairness concerns.

Primary Author-Led Review Process: Screening, data extraction, and initial coding were conducted primarily by the first author. Although a second author reviewed the extracted data and coding decisions for all included studies, it does not substitute for formal inter-rater-reliability assessment. Accordingly, some subjectivity may remain in the selection and classification of studies. Future works should incorporate more rigorous multi-coder procedures.

Temporal and Scope Limitations: This review provides a snapshot of the field at a specific point in time. Because LLM-based MAS are evolving rapidly, new architectures, evaluation methods, and fairness interventions may emerge after the review window,

and some conclusions may become outdated. Additionally, this study focuses on LLM-based MAS in SE; findings may not generalize to non-LLM MAS or to domains with substantially different agent capabilities, interaction patterns, or fairness concerns.

Transferability from Non-SE Contexts: A key limitation of this review is that many included studies are not grounded in SE contexts, instead focusing on general-purpose MAS settings such as social simulations or political discourse. Although we applied a functional role mapping, this approach primarily supports interaction-level generalization (e.g., bias amplification, conformity, coordination failures). In contrast, artifact-level outcomes that are central to SE, such as code quality, security vulnerabilities, and maintainability, may not be directly captured by these studies. As a result, our findings should be interpreted as identifying transferable fairness risks in multi-agent interaction patterns rather than providing empirical evidence of their impact on software artifacts or developer workflows. Future work should validate these interaction-driven risks in real-world SE settings.

6 Conclusion

As foundation models and LLM-based multi-agent systems become embedded across the SDLC, fairness must be evaluated as both an outcome property and an interactional system property. This rapid review synthesizes 18 studies and shows that MAS fairness work clusters into three strands (bias reduction, trustworthy AI governance, and inter-agent dynamics), but measurement remains fragmented across benchmark performance, group disparity, MAS-behavior metrics, and conceptual discussions that often lack operationalization. Reported harms span representation, quality of service, security and privacy, and governance and trust, with evidence concentrated in requirements, design, testing, and maintenance settings and comparatively limited attention to developer workflows such as code review and debugging. Overall, the literature is not yet positioned to support deployable fairness-assured MAS in SE, motivating MAS-aware benchmarks, consistent evaluation protocols, broader attribute and setting coverage, and mitigation and governance mechanisms validated in realistic pipelines.

References

[1] [n. d.]. ACM Digital Library. https://dl.acm.org/. Accessed: 2025-10-11.

[2] [n. d.]. EU Artificial Intelligence Act: Up-to-date developments and analyses. https://artificialintelligenceact.eu/. Accessed: 2025-11-29.

[3] [n. d.]. Google Scholar. https://scholar.google.ca/. Accessed: 2025-10-11.

[4] [n. d.]. IEEE Xplore Digital Library. https://ieeexplore.ieee.org. Accessed: 2025-10-11.

[5] Anthropic. 2024. Introducing the Model Context Protocol. https://www.anthropic.com/news/model-context-protocol. Accessed: April 10, 2026.

[6] Aishwarya Bandaru, Fabian Bindley, Trevor Bluth, Nandini Chavda, Baixu Chen, and Ethan Law. 2025. Revealing Political Bias in LLMs through Structured Multi-Agent Debate. arXiv:2506.11825 [cs.AI] https://arxiv.org/abs/2506.11825

[7] Angana Borah and Rada Mihalcea. 2024. Towards Implicit Bias Detection and Mitigation in Multi-Agent LLM Interactions. arXiv:2410.02584 [cs.CL] https://arxiv.org/abs/2410.02584

[8] Bruno Cartaxo, Gustavo Pinto, and Sergio Soares. 2020. Rapid Reviews in Software Engineering. arXiv:2003.10006 [cs.SE] https://arxiv.org/abs/2003.10006

[9] Hyeong Kyu Choi, Xiaojin Zhu, and Sharon Li. 2025. Measuring and Mitigating Identity Bias in Multi-Agent Debate via Anonymization. arXiv:2510.07517 [cs.AI] https://arxiv.org/abs/2510.07517

[10] Min Choi, Keonwoo Kim, Sungwon Chae, and Sangyeop Baek. 2025. An Empirical Study of Group Conformity in Multi-Agent Systems. In *Findings of the Association for Computational Linguistics: ACL 2025*, Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (Eds.). Association for Computational Linguistics, Vienna, Austria, 5123–5139. doi:10.18653/v1/2025.findings-acl.265

[11] Erica Coppolillo, Giuseppe Manco, and Luca Maria Aiello. 2025. Unmasking Conversational Bias in AI Multiagent Systems. arXiv:2501.14844 [cs.CL] https://arxiv.org/abs/2501.14844

[12] Cursor. 2025. Cursor — The AI Code Editor. https://cursor.com/. Accessed: April 10, 2026.

[13] José Antonio Siqueira de Cerqueira, Mamia Agbese, Rebekah Rousi, Nannan Xi, Juho Hamari, and Pekka Abrahamsson. 2025. Can We Trust AI Agents? A Case Study of an LLM-Based Multi-Agent System for Ethical AI. arXiv:2411.08881 [cs.CY] https://arxiv.org/abs/2411.08881

[14] Shangbin Feng, Chan Young Park, Yuhan Liu, and Yulia Tsvetkov. 2023. From Pretraining Data to Language Models to Downstream Tasks: Tracking the Trails of Political Biases Leading to Unfair NLP Models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (Eds.). Association for Computational Linguistics, Toronto, Canada, 11737–11762. doi:10.18653/v1/2023.acl-long.656

[15] Ariel Flint, Luca Maria Aiello, Romualdo Pastor-Satorras, and Andrea Baronchelli. 2025. Group size effects and collective misalignment in LLM multi-agent systems. arXiv:2510.22422 [cs.MA] https://arxiv.org/abs/2510.22422

[16] Isabel O. Gallegos, Ryan A. Rossi, Joe Barrow, Md Mehrab Tanjim, Sungchul Kim, Franck Dernoncourt, Tong Yu, Ruiyi Zhang, and Nesreen K. Ahmed. 2024. Bias and Fairness in Large Language Models: A Survey. *Computational Linguistics* 50, 3 (Sept. 2024), 1097–1179. doi:10.1162/coli_a_00524

[17] Manuel B. Garcia. 2025. Teaching and learning computer programming using ChatGPT: A rapid review of literature amid the rise of generative AI technologies. *Education and Information Technologies* 30, 12 (2025), 16721–16745. doi:10.1007/s10639-025-13452-5

[18] GitHub, Inc. 2025. GitHub Copilot: Your AI Pair Programmer. https://github.com/features/copilot. Accessed: April 10, 2026.

[19] Diego Gosmar and Deborah A. Dahl. 2025. Sentinel Agents for Secure and Trustworthy Agentic AI in Multi-Agent Systems. arXiv:2509.14956 [cs.AI] https://arxiv.org/abs/2509.14956

[20] Ahmed E. Hassan, Hao Li, Dayi Lin, Bram Adams, Tse-Hsun Chen, Yutaro Kashiwa, and Dong Qiu. 2025. Agentic Software Engineering: Foundational Pillars and a Research Roadmap. arXiv:2509.06216 [cs.SE] https://arxiv.org/abs/2509.06216

[21] Ahmed E. Hassan, Gustavo A. Oliva, Dayi Lin, Boyuan Chen, and Zhen Ming Jiang. 2024. Towards AI-Native Software Engineering (SE 3.0): A Vision and a Challenge Roadmap. arXiv:2410.06107 [cs.SE] https://arxiv.org/abs/2410.06107

[22] Junda He, Christoph Treude, and David Lo. 2025. LLM-Based Multi-Agent Systems for Software Engineering: Literature Review, Vision, and the Road Ahead. *ACM Trans. Softw. Eng. Methodol.* 34, 5, Article 124 (May 2025), 30 pages. doi:10.1145/3712003

[23] Sirui Hong, Mingchen Zhuge, Jiaqi Chen, Xiawu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin Wang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. 2024. MetaGPT: Meta Programming for A Multi-Agent Collaborative Framework. arXiv:2308.00352 [cs.AI] https://arxiv.org/abs/2308.00352

[24] Xinyi Hou, Yanjie Zhao, Yue Liu, Zhou Yang, Kailong Wang, Li Li, Xiapu Luo, David Lo, John Grundy, and Haoyu Wang. 2024. Large Language Models for Software Engineering: A Systematic Literature Review. arXiv:2308.10620 [cs.SE] https://arxiv.org/abs/2308.10620

[25] Dong Huang, Jie M. Zhang, Qingwen Bu, Xiaofei Xie, Junjie Chen, and Heming Cui. 2025. Bias Testing and Mitigation in LLM-based Code Generation. arXiv:2309.14345 [cs.SE] https://arxiv.org/abs/2309.14345

[26] International Organization for Standardization (ISO) and IEC. 2023. Information technology — Artificial intelligence — Management system. Published December 2023; 51 pages.

[27] Gabriele Cesar Iwashima, Claudia Susie Rodrigues, Claudio Dipolitto, and Geraldo Xexéo. 2025. Factors That Support Grounded Responses in LLM Conversations: A Rapid Review. arXiv:2511.21762 [cs.CL] https://arxiv.org/abs/2511.21762

[28] Haolin Jin, Linghan Huang, Haipeng Cai, Jun Yan, Bo Li, and Huaming Chen. 2025. From LLMs to LLM-based Agents for Software Engineering: A Survey of Current, Challenges and Future. arXiv:2408.02479 [cs.SE] https://arxiv.org/abs/2408.02479

[29] Marcin Kawalerowicz, Marcin Pietranik, and Krzysztof Stępniak. 2026. LLMs as Code Review Agents: A Rapid Review and Experimental Evaluation with Human Expert Judges. In *Computational Collective Intelligence*, Ngoc Thanh Nguyen, Vu Dinh Duc Anh, Adrianna Kozierkiewicz, Sinh Nguyen Van, Manuel Núñez, Jan Treur, and Gottfried Vossen (Eds.). Springer Nature Switzerland, Cham, 346–360.

[30] Rana Nameer Hussain Khan, Dawood Wasif, Jin-Hee Cho, and Ali Butt. 2025. Multi-Agent Code-Orchestrated Generation for Reliable Infrastructure-as-Code. arXiv:2510.03902 [cs.SE] https://arxiv.org/abs/2510.03902

[31] Hadas Kotek, Rikker Dockum, and David Sun. 2023. Gender bias and stereotypes in Large Language Models. In *Proceedings of The ACM Collective Intelligence Conference (CI '23)*. ACM, 12–24. doi:10.1145/3582269.3615599

[32] Hao Li, Haoxiang Zhang, and Ahmed E. Hassan. 2025. The Rise of AI Teammates in Software Engineering (SE) 3.0: How Autonomous Coding Agents Are

Reshaping Software Engineering. arXiv:2507.15003 [cs.SE] https://arxiv.org/abs/2507.15003

[33] Feng Lin, Dong Jae Kim, and Tse-Husn Chen. 2024. SOEN-101: Code Generation by Emulating Software Process Models Using Large Language Model Agents. arXiv:2403.15852 [cs.SE] https://arxiv.org/abs/2403.15852

[34] Lin Ling, Fazle Rabbi, Song Wang, and Jinqiu Yang. 2025. Bias Unveiled: Investigating Social Bias in LLM-Generated Code. arXiv:2411.10351 [cs.SE] https://arxiv.org/abs/2411.10351

[35] Yan Liu, Xiaokang Chen, Yan Gao, Zhe Su, Fengji Zhang, Daoguang Zan, Jian-Guang Lou, Pin-Yu Chen, and Tsung-Yi Ho. 2023. Uncovering and Quantifying Social Biases in Code Generation. arXiv:2305.15377 [cs.CL] https://arxiv.org/abs/2305.15377

[36] Jens Lünstedt and Tim Schlippe. 2025. Mitigating Bias in Large Language Models Leveraging Multi-Agent Scenarios. In *2025 7th International Conference on Natural Language Processing (ICNLP)*. 14–18. doi:10.1109/ICNLP65360.2025.11108428

[37] Deep Mehta, Kartik Rawool, Subodh Gujar, and Bowen Xu. 2023. Automated DevOps Pipeline Generation for Code Repositories using Large Language Models. arXiv:2312.13225 [cs.SE] https://arxiv.org/abs/2312.13225

[38] Imran Mirza, Cole Huang, Ishwara Vasista, Rohan Patil, Asli Akalin, Sean O'Brien, and Kevin Zhu. 2025. MALIBU Benchmark: Multi-Agent LLM Implicit Bias Uncovered. arXiv:2507.01019 [cs.CL] https://arxiv.org/abs/2507.01019

[39] n8n. 2025. n8n — AI Workflow Automation Platform & Tools. https://n8n.io/. Accessed: April 10, 2026.

[40] Minh Huynh Nguyen, Thang Phan Chau, Phong X. Nguyen, and Nghi D. Q. Bui. 2024. AgileCoder: Dynamic Collaborative Agents for Software Development based on Agile Methodology. arXiv:2406.11912 [cs.SE] https://arxiv.org/abs/2406.11912

[41] Thi-Nhung Nguyen, Linhao Luo, Thuy-Trang Vu, and Dinh Phung. 2025. The Social Cost of Intelligence: Emergence, Propagation, and Amplification of Stereotypical Bias in Multi-Agent Systems. arXiv:2510.10943 [cs.MA] https://arxiv.org/abs/2510.10943

[42] OpenAI. 2025. Introducing AgentKit. https://openai.com/index/introducing-agentkit/. Accessed: April 10, 2026.

[43] Marc Oriol, Quim Motger, Jordi Marco, and Xavier Franch. 2025. Multi-Agent Debate Strategies to Enhance Requirements Engineering with Large Language Models . In *2025 IEEE 33rd International Requirements Engineering Conference (RE)*. IEEE Computer Society, Los Alamitos, CA, USA, 527–534. doi:10.1109/RE63999.2025.00063

[44] Alicia Parrish, Angelica Chen, Nikita Nangia, Vishakh Padmakumar, Jason Phang, Jana Thompson, Phu Mon Htut, and Samuel R. Bowman. 2022. BBQ: A Hand-Built Bias Benchmark for Question Answering. arXiv:2110.08193 [cs.CL] https://arxiv.org/abs/2110.08193

[45] Federica Pepe, Vittoria Nardone, Antonio Mastropaolo, Gabriele Bavota, Gerardo Canfora, and Massimiliano Di Penta. 2024. How do Hugging Face Models Document Datasets, Bias, and Licenses? An Empirical Study. In *Proceedings of the 32nd IEEE/ACM International Conference on Program Comprehension* (Lisbon, Portugal) *(ICPC '24)*. Association for Computing Machinery, New York, NY, USA, 370–381. doi:10.1145/3643916.3644412

[46] Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, Juyuan Xu, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2024. ChatDev: Communicative Agents for Software Development. arXiv:2307.07924 [cs.SE] https://arxiv.org/abs/2307.07924

[47] Shaina Raza, Ranjan Sapkota, Manoj Karkee, and Christos Emmanouilidis. 2025. TRiSM for Agentic AI: A Review of Trust, Risk, and Security Management in LLM-based Agentic Multi-Agent Systems. arXiv:2506.04133 [cs.AI] https://arxiv.org/abs/2506.04133

[48] Replit. 2025. Replit — Build apps and sites with AI. https://replit.com/. Accessed: April 10, 2026.

[49] Krishna Ronanki. 2025. *Facilitating Trustworthy Human-Agent Collaboration in LLM-based Multi-Agent System oriented Software Engineering*. Association for Computing Machinery, New York, NY, USA, 1333–1337. https://doi.org/10.1145/3696630.3728717

[50] Mark Ryan and Bernd Carsten Stahl. 2020. Artificial intelligence ethics guidelines for developers and users: clarifying their content and normative implications. *Journal of Information, Communication and Ethics in Society* 19, 1 (06 2020), 61–86. arXiv:https://www.emerald.com/jices/article-pdf/19/1/61/1616450/jices-12-2019-0138.pdf doi:10.1108/JICES-12-2019-0138

[51] Manish Sanwal. 2025. Layered Chain-of-Thought Prompting for Multi-Agent LLM Systems: A Comprehensive Approach to Explainable Large Language Models. arXiv:2501.18645 [cs.CL] https://arxiv.org/abs/2501.18645

[52] Tanush Sharanarthi. 2025. Adaptive Multi-Agent AI Framework for Real-Time Energy Optimization and Context-Aware Code Review in Software Development. In *2025 5th International Symposium on Computer Technology and Information Science (ISCTIS)*. 353–358. doi:10.1109/ISCTIS65944.2025.11066037

[53] Ron Solomon, Yarin Yerushalmi Levi, Lior Vaknin, Eran Aizikovich, Amit Baras, Etai Ohana, Amit Giloni, Shamik Bose, Chiara Picardi, Yuval Elovici, and Asaf Shabtai. 2025. LumiMAS: A Comprehensive Framework for Real-Time Monitoring and Enhanced Observability in Multi-Agent Systems. arXiv:2508.12412 [cs.CR] https://arxiv.org/abs/2508.12412

[54] Rao Surapaneni, Miku Jha, Michael Vakoc, and Todd Segal. 2025. Announcing the Agent2Agent Protocol (A2A): A new era of Agent Interoperability. https://developers.googleblog.com/en/a2a-a-new-era-of-agent-interoperability/. Accessed: April 10, 2026.

[55] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 6000–6010.

[56] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems* (New Orleans, LA, USA) *(NIPS '22)*. Curran Associates Inc., Red Hook, NY, USA, Article 1800, 14 pages.

[57] Zhiqiu Xia, Lang Zhu, Bingzhe Li, Feng Chen, Qiannan Li, Chunhua Liao, Feiyi Wang, and Hang Liu. 2025. Analyzing 16,193 LLM Papers for Fun and Profits. arXiv:2504.08619 [cs.DL] https://arxiv.org/abs/2504.08619

[58] Qinghua Xu, Guancheng Wang, Lionel Briand, and Kui Liu. 2025. Hallucination to Consensus: Multi-Agent LLMs for End-to-End Test Generation. arXiv:2506.02943 [cs.SE] https://arxiv.org/abs/2506.02943

[59] Zhenjie Xu, Wenqing Chen, Yi Tang, Xuanying Li, Cheng Hu, Zhixuan Chu, Kui Ren, Zibin Zheng, and Zhichao Lu. 2025. Mitigating Social Bias in Large Language Models: A Multi-Objective Approach Within a Multi-Agent Framework. *Proceedings of the AAAI Conference on Artificial Intelligence* 39, 24 (Apr. 2025), 25579–25587. doi:10.1609/aaai.v39i24.34748

[60] Boyang Yang, Zijian Cai, Feng Liu, Bach Le, Lingming Zhang, Tégawendé F. Bissyandé, Yang Liu, and Haoye Tian. 2025. A Survey of LLM-based Automated Program Repair: Taxonomies, Design Paradigms, and Applications. *ArXiv* abs/2506.23749 (2025). https://api.semanticscholar.org/CorpusID:280010745

[61] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing Reasoning and Acting in Language Models. In *International Conference on Learning Representations (ICLR)*.

[62] Vahid Yazdanpanah, Enrico H. Gerding, Sebastian Stein, Corina Cirstea, M. C. Schraefel, Timothy J. Norman, and Nicholas R. Jennings. 2021. Different Forms of Responsibility in Multiagent Systems: Sociotechnical Characteristics and Requirements. *IEEE Internet Computing* 25, 6 (2021), 15–22. doi:10.1109/MIC.2021.3107334

[63] Zheng Yu, Ziyi Guo, Yuhang Wu, Jiahao Yu, Meng Xu, Dongliang Mu, Yan Chen, and Xinyu Xing. 2025. *PATCHAGENT: a practical program repair agent mimicking human expertise*. USENIX Association, USA.

[64] Zapier Inc. 2025. Zapier: Automate AI Workflows, Agents, and Apps. https://zapier.com/. Accessed: April 10, 2026.

[65] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. arXiv:2306.05685 [cs.CL] https://arxiv.org/abs/2306.05685