Unveiling Cognitive Biases in Software Testing: Insights from a Survey and Controlled Experiment

Eduard Paul Enoiu*, Alexandru Cusmaru**, Jean Malm*
*Mälardalen University, Västerås, Sweden.
**Siemens Mobility, Germany.

Abstract—**Biases are hard-wired behaviours that influence software testers. Understanding how these biases affect testers' everyday behaviour is crucial for developing practical software tools and strategies to help testers avoid the pitfalls of cognitive biases. This research aims to assess the extent to which software testers know the influence of cognitive biases on their work. Our study was conducted in two incremental steps: a survey and a controlled experiment. Firstly, we developed a questionnaire survey designed to reveal the extent of software testers' knowledge about cognitive biases and their awareness of these biases' influence on testing. We contacted software professionals in different environments and gathered valid data from 60 practitioners. The survey results suggest that software professionals are aware of biases, specifically preconceptions such as confirmation bias, fixation, and convenience. Additionally, biases like optimism, ownership, and blissful ignorance were commonly recognized. In line with other research, we observed that software professionals tend to identify more cognitive biases in others than in their judgments and actions, indicating a vulnerability to bias blind spot. To build on these findings, we performed a controlled experiment with 12 participants to investigate the behaviour and biases exhibited by humans when attempting to solve a hypothetical test problem. Through thematic analysis, we identified prevalent biases such as confirmation bias, pattern recognition and overreliance, sunk cost fallacy, and anchoring bias among participants. Additionally, we found that collaborative problem-solving was a prominent feature, often leading to biases like groupthink.**

I. Introduction

Cognitive biases are behaviours that influence testers' and developers' actions and the tools they use. These biases can lead to suboptimal testing results [1], [2], potentially impacting software quality and increasing the risk of undetected defects. While some researchers (e.g., [3], [4]) have found that cognitive biases occur in development and testing tasks in both controlled lab or industrial studies (e.g., confirmation bias [1], selective perception [2], IKEA effect [3], anchoring and adjustment [4] [5]), we still do not know how these biases affect humans performing manual testing or using different tools (e.g., test automation tools) in their everyday behaviour.

Cognitive biases have been extensively studied in various fields, including psychology [6] and sociology [7]. Still, their impact on software engineering, particularly software testing, is an emerging area of research [3], [8]. Cognitive biases in software testing [4] can manifest in various ways, such as confirmation bias, anchoring bias, and overconfidence bias, each affecting the outcomes of testing activities differently.

Confirmation bias [2] is a common issue in software testing, where testers seek or interpret data to confirm their preconceptions. For example, a tester who considers a feature works correctly might only focus on test cases that confirm its expected behaviour, potentially missing defects that appear under specific conditions.

Our research employed a two-step approach, combining a survey and a controlled experiment to investigate biases in software testing. The motivation for this two-step approach was to assess the widespread awareness of cognitive biases among testers through a broad survey and then to explore specific biases and behaviours in a controlled setting.

In the first step, we developed a questionnaire to uncover the extent of software testers' knowledge about cognitive biases and their awareness of how they impact their work. By reaching out to software professionals across various environments, we collected responses from 60 practitioners. The survey revealed that testers are generally aware of confirmation bias, fixation, and convenience. Additionally, they recognized biases such as optimism, ownership, and blissful ignorance. Consistent with existing research [4], [9], [10], our findings indicate that testers are more likely to perceive cognitive biases in others rather than in their judgments, highlighting a bias blind spot. Building on the survey results, the second step involved a controlled experiment with 12 participants. This experiment explored individuals' specific behaviours and biases when solving a hypothetical testing problem. Through thematic analysis, we identified common participant biases, such as confirmation bias, groupthink, pattern recognition and overreliance, sunk cost fallacy, and anchoring bias. Furthermore, collaborative problem-solving emerged as a significant factor, often leading to groupthink bias.

II. Related Work

Cognitive biases in software engineering have been a subject of research since the 1990s, with over 200 biases identified, around 37 of which have been thoroughly investigated and evidenced in the context of software development [11]. These biases have been found to impact various stages of software

[1]Confirmation bias: the tendency to search for, interpret, and recognise information in a way that confirms one's preconceptions.

[2]Selective perception: the tendency to notice and retain information that aligns with one's own beliefs while ignoring information that contradicts them.

[3]IKEA effect: the tendency for people to place a disproportionately high value on results they partially created.

[4]Anchoring and adjustment: the tendency to rely too heavily on an initial piece of information (the "anchor") when making decisions and to adjust insufficiently from that starting point.

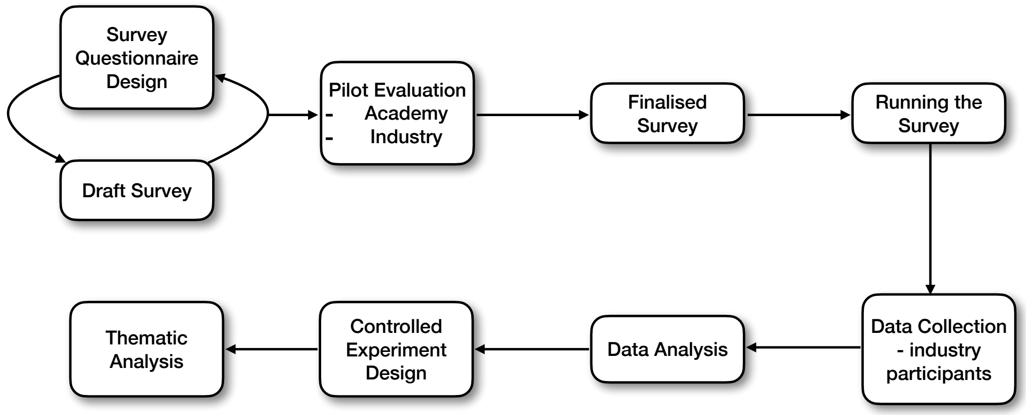


Fig. 1: Overview of the research method used in this study.

development, including requirement gathering, design, coding, and testing, thereby affecting overall software quality.

One of the most extensively studied biases in software engineering is confirmation bias, the tendency to search for, interpret, and remember information in a way that confirms one's preconceptions [12]. This bias can lead to significant software quality and testing issues, as highlighted by some studies (e.g., [4], [12]). For instance, Jørgensen and Papatheocharous [12] demonstrated that software managers tend to interpret project data in ways that confirm their preferred contract types, which can lead to misleading conclusions and suboptimal decisions.

Cognitive biases affecting software testing have been less frequently studied, but the existing research [11], [13] indicates that they can significantly disrupt the testing process. For example, confirmation bias can lead testers to focus on confirming that the software works rather than on identifying potential defects [11]. This bias towards positive testing strategies can result in higher defect densities and compromised software quality [13]. Salman [11] emphasized the importance of understanding and mitigating cognitive biases in software quality and testing to improve overall software reliability. This involves recognizing the factors that trigger these biases within organizational contexts and devising strategies to counteract their negative effects.

Despite identifying numerous biases, the research [4] underscores a lack of mitigation strategies, calls for more empirical research to develop debiasing techniques and emphasizes the need for a better comprehensive understanding of cognitive biases in software engineering. This understanding is crucial for improving software engineering practices and outcomes. Our study aims to fill these gaps by providing empirical evidence on the manifestation of biases in testing practice.

III. Method

We outline the research method in this study in Figure 1. It starts with designing a survey questionnaire and a pilot evaluation with academic and industry experts. Based on their feedback, the survey is refined and finalized. The survey is then distributed, and data is collected from industry participants. This data undergoes analysis, which informs the design of a controlled experiment. The experiment is conducted, and the results are analyzed through thematic analysis, identifying common themes and biases observed.

We developed a study to reveal the extent of software practitioners' knowledge about cognitive biases and their awareness of how they influence testing. Unconscious biases are behaviours that influence testing and can lead practitioners in the wrong direction. Understanding how these biases affect testers' everyday actions is crucial for mitigating them and developing more effective tools and strategies to help them avoid cognitive pitfalls. Therefore, we devised a survey and controlled experiment to answer the following research question: *To what extent are practitioners aware that cognitive biases influence their work?* This question guided our research and formed the basis of our investigation.

We contacted software professionals in various environments and gathered valid data from 60 practitioners through 18 open-ended and close-ended questions focused on specific bias categories. Building on the insights from the survey, we conducted a controlled experiment with 12 participants to investigate how biases would impact problem-solving in testing scenarios. In this experiment, we aimed to observe specific behaviours and decisions influenced by cognitive biases during the testing process. Additionally, we aimed to understand how problem-solving might contribute to these biases.

A. Survey Design

We designed the survey to get an overview of participants' perspectives on the influence of biases when involved in testing-related activities based on their experiences. To achieve the aim of this investigation, the survey was developed following the steps shown in Figure 1. We defined the survey goals during this phase and designed the questionnaire through iterations. The survey included closed and open-ended questions, with a Likert scale for the closed-ended ones.

The questionnaire questions were designed to identify and understand the cognitive biases in software engineering (CB1 to CB10 bias categories), as detailed by [3]. As a result, the

Q#	Question
Q1	Which role are you presently working in?
Q2	How long have you been working with testing?
Q3	What kinds or level of testing do you perform?
Q4	Can you summarize what you typically do in your current role (for example typical tasks, duties, etc.)?
Q5	What kind of prior beliefs, expectations, preconceptions, or biases have you encountered (yourself or when working with others) in your current role?
Q6	Prior beliefs, expectations, preconceptions, or biases represent a cause for concern in testing.
Q7	When testing, your own judgments and actions are influenced by prior beliefs, expectations, preconceptions, or biases.
Q8	An experienced tester is less likely to be influenced by prior beliefs, expectations, preconceptions, or biases than a tester with less experience.
Q9	Prior beliefs, expectations, preconceptions, or biases are less of a problem in testing than in other domains of software or system engineering.
Q10	When writing a new test case (either test specification or test implementation), do you usually/often modify/adapt an existing test case or write it from scratch?
Q11	If you had to test a date field (MM/DD/YY) by writing the three most valuable test cases, which input data would you consider?
Q12	Besides your regular test oracles (requirements, user stories, other descriptions), do you use other test approaches to discover undocumented features/behaviours (e.g., exploratory testing, pairwise testing, letting other roles test)?
Q13	When using automated regression, do you usually rely solely on the results of your automated test suite?
Q14	When implementing test cases, do you perform (before or after the implementation) a manual test execution?
Q15	Do you usually re-use the same test values throughout test suites (e.g., the used user is always "test", check-box is always deactivated) even when the usage is not regulated by test oracles (requirements, user stories, other descriptions)?
Q16	If it were to assess the composition of your test suite, which percentage of negative (indirect) test cases does it contain?

TABLE I: Survey Questions

survey consists of 16 questions. Table I summarizes the survey questions. The questions were designed to gather information about the respondents' roles (Q1), experience levels (Q2), and types of testing performed (Q3). They also sought insights into daily tasks (Q4), encountered cognitive biases (Q5, Q6, Q7), and perceptions of how biases impact their testing process. Additional questions explored whether experience mitigates biases (Q8), compared the impact of biases in testing to other domains (Q9), and examined the respondents' approaches to creating new test cases and selecting test scenarios (Q10, Q11). The survey also looked into the use of additional testing approaches (Q12), reliance on automated test suites (Q13), the role of manual testing (Q14), and the consistency of test values used (Q15). Finally, the survey aimed to evaluate the proportion of negative test cases (Q16).

Questions Q1, Q4, and Q5 in Table I explore roles and activities to address preconceptions (CB1 category in [3]). Question Q10 examines ownership bias by asking about modifying tests (CB2). Question Q11 focuses on a specific test (CB3) to reveal fixation bias. Question Q15 investigates the reuse of test values to highlight the resort to default (CB4). Questions Q6 and Q7 address optimism bias by checking the recognition of biases (CB5). Convenience bias is explored in Question Q13 by looking at reliance on automated test results (CB6). Question Q8 examines subconscious actions by questioning the impact of experience on biases (CB7). Blissful ignorance is identified through Question Q14, which looks at manual testing practices (CB8). Question Q12 checks for superficial selection by asking about diverse test approaches (CB9).

B. Pilot Evaluation

After designing the survey, we conducted a pilot study with one practitioner and two software testing researchers to verify question clarity and gather suggestions. Based on their feedback, we added brief definitions for terms like prior belief and expectation to ensure consistency in interpretation. We also refined specific questions—changing QA to QA/Test for clarity, adding examples to Question 3, and aligning the response scales for Questions 6 to 10. These changes improved clarity and consistency while maintaining the survey's exploratory nature. With these adjustments, we finalized the survey and conducted our study.

C. Survey Sampling and Data Collection

Our survey targeted software practitioners involved in testing-related activities. We used a non-probabilistic sampling technique, selecting participants based on their expertise rather than availability. To reach this group, we distributed the survey through the Software Center [5] project's email list, which includes experts from over 15 industrial partners. Additionally, we leveraged email lists from several European projects like VeriDevOps [6] and personal contacts within companies in Germany and Sweden, particularly in the Mälardalen area.

We designed the survey using the Microsoft Forms platform and advertised it as an anonymous survey link. The survey was advertised through different practitioner mailing lists and company contacts, focusing on software practitioners involved in testing-related activities.

D. Controlled Experiment Design

Our controlled experiment aimed to study the behaviour and biases of humans attempting to solve a hypothetical test problem. The research was intended to identify the biases experienced, and assessing the impact of bias awareness training on their problem-solving performance.

To address these questions, we defined two independent variables: *the training in bias awareness* (theoretical presentation vs. practical training) and *the configuration of the hypothetical test problem*. The dependent variables included *the identified biases, the steps to resolve the test problem, and other relevant measurements*.

[5]https://www.software-center.se/
[6]https://sites.mdu.se/veridevops

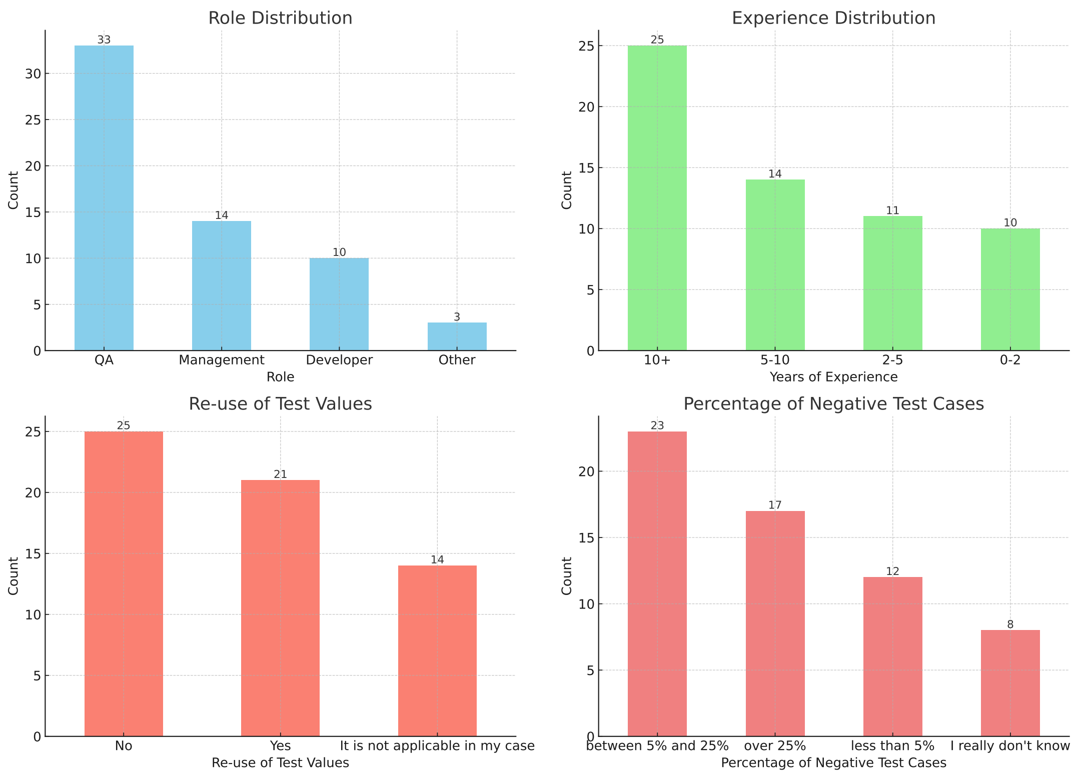


Fig. 2: An overview of the survey results: roles, experience, test value reuse, and negative test case distribution.

The experiment consisted of two phases during a two-hour session: training and game phases. The control group was given training through a presentation on bias awareness. In contrast, the experimental group had an additional practical training session inspired by the *three numbers in ascending order rule* proposed by Wason [14]. It was adapted from the original by 1) running it on software and 2) accepting numbers in hexadecimal form as valid input.

After the training, the groups then played the following game: Given a group of players and an instructor, players are provided with a set of dice varying in colour, shape, material and how numbers are represented[7]. The game rules are: 1) Select five dice, 2) Roll five dice, 3) Present the resulting roll to the instructor, who responds with a number, and 4) All other rules may be broken. The player's goal is to figure out how the instructor arrived at that number, with no other information provided. Due to time constraints, participants were informed they had a limited set of rolls to attempt before the game ended. The relational rule applied during the dice game is as follows: If no dice in the presented set contain any dots, the result is an error. Otherwise, the sum of all dots surrounding a centrally placed dot on the rolled face is the result. This is illustrated in Figure 3, where the numbers below the dice reflect its contribution to the summed number.

This setup allowed us to study the problem-solving abilities of humans in a testing context and identify various biases

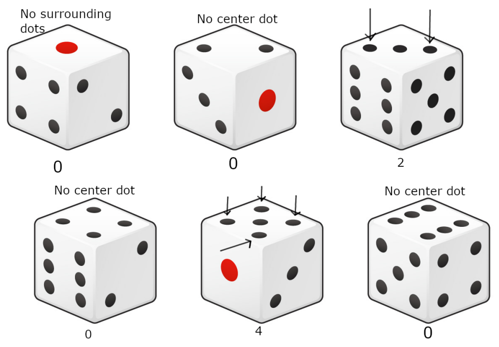


Fig. 3: Example of how dice are scored using normal 6-sided dice. Image is adapted from https://en.ac-illust.com/clip-art/25727029/dice-roll--1-to-6.

that emerged during the problem-solving process. Participants were encouraged to write or verbalise their thoughts during the sessions to capture the participants' thought processes while they worked on their tasks. Video and audio recordings were made of each session, which were later analyzed using verbal protocol analysis [15] and thematic analysis [16] to identify problem-solving processes and biases. The game, which seems simple, required participants to constantly question their assumptions and think analytically, mirroring the thought processes essential in software testing. This method helped highlight the biases and thought patterns that participants relied on while solving testing problems.

[7]e.g., as numerals or a set of dots

IV. Survey Results

The following section outlines some of the survey's primary results. Figure 2 provides an overview of these results.

A. Demographics

We report some results related to the demographics of our participants. 65% of the participants in our study worked with testing for at least five years, and we had participants working both with development and testing as well as managers. 55% of the participants worked with quality assurance and testing. Also, we asked participants about their typical testing duties and the types of testing they perform. Even if the number of participants can be considered relatively small, as shown in Figure 2, we have a diverse crowd of 60 practitioners performing different testing tasks such as manual testing, testing supported by test automation, test analytics, and test team management. These participants perform various types of testing, such as unit-level testing, regression test selection, stress testing, usability testing, and exploratory testing.

B. Overall Results

Our results suggest that professionals are indeed aware of biases. Specifically, they are aware of preconceptions such as confirmation bias, fixation, and convenience. In addition, optimism, ownership, and blissful ignorance were other common biases. In common with other research, we observed that people tend to identify more cognitive biases in others than in their judgments and actions, indicating a vulnerability to bias blind spot. Our results suggest that 87% of participants believe biases, such as prior beliefs, expectations, and preconceptions, are a cause of concern in testing. Furthermore, 70% recognize that these biases influence their judgments and actions during testing. However, awareness of biases alone does not necessarily equip individuals with the knowledge or strategies to mitigate them. Research indicates that biases can persist even when we know them, suggesting some might be inescapable [17]. Despite this, creating an environment that encourages diverse perspectives and random directions can help reduce the risk of becoming stuck on the wrong track [18].

When analyzing the open-ended questions using thematic analysis, we categorized the prior beliefs, expectations, preconceptions, or biases encountered into two main classes (as shown in Table II): statements about the "*victims*" of certain biases and preconceptions about testing. This categorization allowed us to identify specific patterns and understand the common biases and misconceptions present among testers.

In the first category (*Victims of Biases* in Table II), we observed several instances where testers were victims of specific biases. Confirmation bias and happy-path testing were prevalent, where testers tended to focus on scenarios that confirmed their existing beliefs or the expected functionality. The *"what you see is all there is"* bias was also common, leading testers to rely only on visible information without considering other possibilities. There was a strong emphasis on "*automating as much as you can*", reflecting a bias towards automation at the expense of manual testing. Another common bias was assuming they knew what the customer needed without sufficient user feedback, leading to potential misalignment with actual requirements. Testers also exhibited toolless approaches, relying on their judgment without effectively leveraging available tools. The "*narrow framing effect*" was evident, with testers focusing narrowly on specific aspects of testing rather than considering a broader perspective.

The second category focused on preconceptions about testing itself (*Preconceptions About Testing* in Table II). A notable preconception was that testing does not add value, undermining the importance of thorough testing in the software development lifecycle. Another misconception was equating testing with mere checking, reducing testers' role to verify predefined conditions rather than exploring and uncovering unknown issues. There was a belief that testing is for programmers who cannot code, devaluing the specialized skills and knowledge that testers bring. Many held the view that testing is solely the responsibility of testers, ignoring the collaborative nature of quality assurance in software development. Testing was often perceived as boring and easy, failing to recognize the complexity and critical thinking involved. Some believe that testing increases quality, which, while true, oversimplifies the multifaceted contributions of testing. Lastly, manual testing was seen as having no value, reflecting a bias towards automation and undervaluing the insights gained from manual testing practices.

This categorization highlights the diverse biases and preconceptions influencing testers' behaviour and perceptions. Understanding these patterns is crucial for addressing and mitigating cognitive biases in testing, ultimately leading to more effective and comprehensive testing strategies.

C. Ownership Bias

Ownership bias occurs when test practitioners give too much weight to artefacts already created or created by themselves. The IKEA effect is a special case where people place too much value on things they partially create. Our results suggest that 60% of the participants prefer to modify and adapt existing test cases, 59% do this through the reuse of test values, while the remaining participants write them from scratch. This indicates that a considerable number of people may have a preference for existing or self-created test artefacts. While there is a belief that this behaviour might influence less experienced testers, the data indicates that even seasoned professionals exhibit preferences that align with ownership bias, such as modifying existing test cases and reusing familiar test values. While the results are interesting, they are somewhat inconclusive.

D. Congruency and Confirmation Bias

We also asked participants how much they use other approaches to discover undocumented behaviours besides their regular test oracles, addressing congruency and confirmation biases; 62% of the participants use other solution space exploration techniques, while 8% do not use or have oracles. Only a minority, 25% of the participants, rely solely on regular test oracles. There was no significant influence of experience on

Category	Bias/Preconception	Description
Victims of Biases	Confirmation Bias	Testers focus on scenarios that confirm the expected functionality.
	Happy-path Testing	Testers tend to test only the expected positive scenarios.
	"What You See Is All There Is"	Reliance only on visible information, neglecting hidden or less obvious issues.
	Bias Toward Automation	Strong emphasis on automating as much as possible, potentially neglecting the value of manual testing. *"Write automation scripts in Selenium/Java, based on existing test cases."*
	Assumed Customer Knowledge	Testers assume they know what the customer needs without sufficient user feedback. *"Always letting the project requirement drill down into detail with who and what will be impacted."*
	Tool-less Approaches	Testers rely on their judgment without effectively leveraging available tools. *"Defect fixes and implementation of user stories, without relying on specific testing tools."*
	Narrow Framing Effect	Focus on specific aspects of testing without considering a broader perspective. *"Specify, design, and implement test cases in accordance with the requirements, often focusing on narrow aspects."*
Preconceptions About Testing	Testing Does Not Add Value	Belief that testing is not valuable in the software development lifecycle.
	Testing Equals Checking	Misconception that testing is only about verifying predefined conditions.
	Testing Is for Non-coders	Belief that testing is for programmers who cannot code.
	Sole Responsibility of Testers	View that testing is only the responsibility of testers, not a collaborative effort.
	Testing Is Boring and Easy	Perception that testing lacks complexity and critical thinking.
	Testing Increases Quality	Oversimplification that testing alone increases quality without recognizing its multifaceted contributions.
	Manual Testing Has No Value	Bias towards automation, undervaluing the insights from manual testing.

TABLE II: Summary of biases and preconceptions in testing experienced by participants.

the responses to this question. Examples of techniques participants use to mitigate predefined oracles include *exploratory testing, pair testing, heuristic testing, strategy model testing, customer feedback, letting other roles test, and domain-specific testing.*

Additionally, related to confirmation bias, we asked participants to evaluate the test suites they create and how they design such test suites in terms of negative test cases (i.e., testing in ways for which the system was not intended to be used). As shown in Figure 2, 28% of participants estimated their test suites contain over 25% negative test cases, while 58% reported less than 25%. Interestingly, 72% of quality assurance practitioners and testers fell into the latter category. Conversely, many participants in other roles, such as management and developers, estimated their test suites to contain more than 25% negative test cases.

Our results show that less experienced participants are more inclined to estimate their test suites contain more positive (fewer negative) test cases. Participants may be more inclined to create test cases that confirm the system works as intended rather than testing for unexpected uses or edge cases. There is also a noticeable discrepancy between different groups regarding the proportion of negative test cases, which could reflect varying levels of understanding, priorities, or approaches to testing within organizations.

E. Automation Bias

We also examined automation bias, which is the over-reliance on automated aids and decision support systems in testing. Therefore, we asked our participants if they rely solely on their automated test suite results when using automated regressions. The results showed that 40% of the participants rely exclusively on the outcomes of their automated tests, indicating a significant level of trust in these test automation systems. In addition, another 40% do not rely solely on automated test suites while the remaining 20% of the participants do not use automated regression test suites at all, highlighting a segment that either lacks access to or chooses not to employ these automated tools. Furthermore, we asked whether participants performed manual test executions either before or after the implementation of test scripts. An overwhelming 76% of respondents confirmed that they do perform such manual tests, suggesting a widespread practice of validating automated results with manual verification.

These findings suggest practitioners try to balance over-reliance on test automation, but we do not fully understand how biases affect its use. Investigating the motivations and experiences could reveal gaps in current strategies and improve practices. Future research should explore these aspects through qualitative or longitudinal studies to understand the interaction between automation, manual testing, and test effectiveness.

V. Controlled Experiment Results

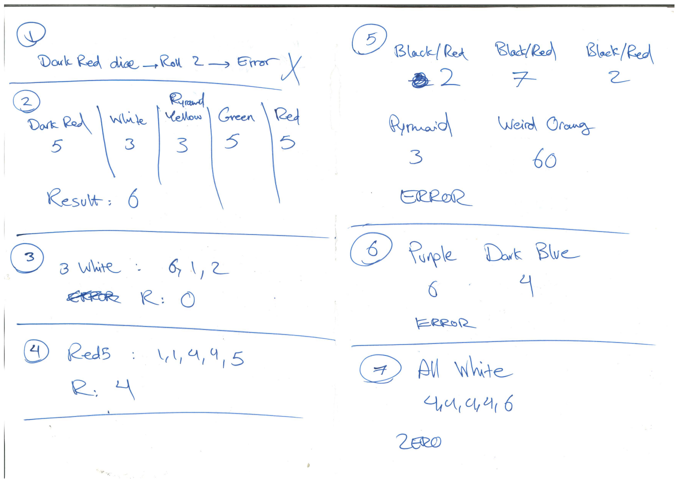
In this section, we detail the observations from the controlled experiment. Our analysis highlights the presence of various cognitive biases in the testing process, including confirmation bias, groupthink, pattern recognition and over-reliance, sunk cost fallacy, and anchoring bias. Significant themes include collaborative problem-solving, risk aversion, resource optimization, memory biases, and overconfidence.

Although detailed demographic data were considered beyond the scope of this study, the 12 participants were university-affiliated individuals either pursuing or having completed a doctoral degree in fields related to software engineering. The participants' ages ranged from 27 to 45.

Participants' notes from the experiment abstracted crucial information needed to identify the true relational rule, focusing only on some properties of the inputs and results. For example, Figure 4 shows participants noted the colour, shape, and numbers rolled for the inputs. This parallels software testing, where

Theme	**Description**
Confirmation Bias	Preference for information that confirms pre-existing beliefs or hypotheses, often disregarding contradictory evidence during testing.
Groupthink	Tendency to conform to the majority opinion in testing teams leads to consensus without thoroughly exploring alternative testing strategies.
Pattern Recognition	Strong inclination to find and rely on patterns in test results, often overlooking alternative explanations or simpler solutions.
Sunk Cost Fallacy	Reluctance to abandon a testing approach that has already consumed time and effort despite contradictory test results.
Anchoring Bias	Significant influence of initial test results on subsequent judgments, limiting the exploration of other testing possibilities.
Collaborative Problem-Solving	Emphasis on building on each other's ideas within testing teams highlighting both good collaboration and the pitfalls of consensus.
Risk Aversion	Cautious approach to selecting test strategies to minimize errors, potentially missing more informative test cases.
Efficiency Seeking	Mindfulness of limited testing resources, aiming to maximize information gained from each test case.
Memory Biases	Influence of faulty recollection of previous test results on the interpretation of current test outcomes and formulation of new test hypotheses.
Learning Through Trial and Error	Fundamental approach to learning and problem-solving in testing through hypothesizing, testing, and adjusting based on outcomes.
Overconfidence	Overestimation of understanding or skills in testing, leading to undue certainty in test strategies despite limited evidence.

TABLE III: Main Themes related to Cognitive Biases in a Testing Scenario.


1 Dark Red dice → Roll 2 → Error X

2 Dark Red 5 | White 3 | Pyramid Yellow 3 | Green 5 | Red 5
Result: 6

3 3 White : 6, 1, 2
~~ERROR~~ R: 0

4 Red5 : 1, 1, 4, 4, 5
R: 4

5 Black/Red 2 Black/Red 7 Black/Red 2
Pyrmaid 3 Weird Orang 60
ERROR

6 Purple 6 Dark Blue 4
ERROR

7 All White
4, 4, 4, 4, 6
ZERO

Fig. 4: Sample of participants' notes taken during the session.

testers may not capture all relevant system state information during execution, hindering issue understanding.

By analyzing the transcript of each session and the notes taken during and after each session, we identified a set of emergent themes related to biases in testing. In Table III, we outline the main biases observed in the controlled experiment.

A. Confirmation Bias

Participants showed a preference for information that confirmed their pre-existing beliefs or hypotheses. Despite contrary evidence, they often argued for hypotheses and tended to confirm initial hypotheses about the game's rules. For example, when a player suggested using floating numbers and the test did not match the pattern, it reinforced the belief that only integers are part of the pattern. Additionally, statements like " *Yeah, it must pass. It should pass.*" indicates a preconceived idea that specific inputs should pass, focusing on confirming beliefs rather than considering cases where they might not.

Moreover, participants interpreted new data to support their initial hypotheses. There was a clear indication that participants were fixated on initial ideas, such as the hypothesis that dice rolls involving certain numbers should produce zero. Statements like, "*So if it is a six it's immediately zero...*" illustrate this tendency to cling to initial beliefs and disregard contradictory outcomes.

The bias also manifested in the way participants selected and interpreted test cases. For instance, when participants noted that a six-sided die (d6) consistently gave a particular result, they began to discard other options, focusing only on confirming the behaviour associated with d6s. This led to an overemphasis on confirming their expectations about the die behaviour rather than exploring other potential patterns or rules. Furthermore, the participants often sought validation for their preconceived notions, asking for confirmation from the moderator or their colleagues. For example, in one session, a participant sought to confirm that certain inputs would produce expected results by stating, " *Can we just ask [the instructor] for the correct answer?"*. Overall, these instances underscore the nature of confirmation bias in the participants' approach.

B. Groupthink

Groupthink was observed as participants conformed to the majority opinion. Participants quickly agreed on dice characteristics without much questioning and tended to seek quick consensus. This bias led to consensus without thoroughly exploring alternative testing ideas, potentially hindering optimal problem-solving. Throughout the sessions, the tendency to conform to group opinions was evident. When a participant suggested a hypothesis, others quickly agreed without examining or challenging the test case. For example, when one participant proposed that a particular die type might be key to solving the problem, the group quickly accepted this hypothesis. The group focused on testing it, often disregarding other possibilities.

This rush to agreement often led to poorly thought-out decisions. Participants built upon each other's test cases in several instances without pausing to consider whether those suggestions were based on sound logic or sufficient evidence. Moreover, the influence of dominant voices in the group was notable. Certain participants, who appeared more confident or assertive, often convinced the group's decisions, leading to a situation where their test cases were implemented without adequate reasons.

C. Pattern Recognition and Overreliance

There was a strong inclination to find and rely on patterns. Participants focused on numerical sequences and dice types, demonstrating a natural tendency to seek order and predictability. However, this over-reliance on perceived patterns often led to overlooking alternative explanations or more straightforward solutions, contributing to confirmation bias. For instance, they frequently hypothesized that certain sums or specific dice configurations would consistently produce predictable outcomes.

The participants' fixation on perceived patterns often resulted in the neglect of other potential variables or simpler rules that could explain the game's mechanics. For example, when faced with a consistent outcome from using a particular die type, participants would repeatedly test similar configurations to validate their initial pattern-based hypotheses. This led to repetitive testing and diverted their attention from exploring other viable hypotheses that did not fit the established pattern.

Searching for patterns sometimes led participants to overcomplicate the problem-solving process. Instead of considering more straightforward explanations, they would devise complex theories based on specific testing patterns they believed to have identified. This tendency to complicate rather than simplify further impeded their understanding of the game's rules. Additionally, participants often assumed that the presence of a pattern implied a higher level of complexity in the game's design. This assumption made them overestimate the problem's difficulty and overlook more straightforward test cases.

D. Sunk Cost Fallacy

Participants were reluctant to abandon a line of thinking they had invested time in despite contradictory test results. They persisted with repeated dice rolls and continued testing of similar hypotheses, reflecting the sunk cost fallacy where participants persisted with unproductive strategies due to prior investments.

Throughout the sessions, participants were unwilling to let go of hypotheses and strategies they had spent considerable time and effort developing. Even when faced with evidence, they often continued exploring the same lines of thinking. For example, participants frequently repeated dice rolls and retested similar hypotheses, hoping to obtain test results that would eventually confirm their initial assumptions. This was evident when they continually tested the identical numerical sequences or dice types despite receiving results that should have prompted them to reconsider their approach.

Moreover, the group dynamics reinforced the sunk cost fallacy. Participants encouraged each other to stick with familiar strategies, making it even more challenging for individuals to break away from the established approach and consider alternative test cases.

E. Anchoring Bias

Initial test cases set the stage for future assumptions, and early dice roll results disproportionately influenced subsequent strategies and interpretations. Anchoring on initial ideas limited exploring other possibilities, such as when participants anchored on mapping dice values to ASCII characters. Once this idea was proposed, participants explored this mapping extensively, allowing it to dominate their thinking and strategy development.

Moreover, the influence of early information extended beyond specific hypotheses to the overall approach used by the participants. Initial test successes or failures with certain interpretations of the dice values often set the tone for the group's overall strategy.

F. Collaborative Problem-Solving

Throughout the sessions, participants tended to work together and use the group's collective input. This collaborative approach enabled the sharing of diverse perspectives and ideas, often facilitating the development of creative and comprehensive hypotheses. In addition, nonverbal communication played a significant role in the collaborative process. Good collaboration was noted, but it also highlighted the pitfalls of consensus. Participants mentioned biases, nonverbal communication, and building on each other's ideas.

G. Risk Aversion and Fear of Failure

A theme of risk aversion and fear of failure was identified, with participants showing a cautious approach to selecting dice and formulating hypotheses. This cautiousness reflects a preference for strategies perceived as less likely to lead to errors, potentially missing more test cases.

Throughout the sessions, participants demonstrated a noticeable tendency to avoid taking risks in their problem-solving approach. This risk aversion was evident in their careful selection of test cases and hypotheses, as they often chose options they believed to be safer or more likely to yield predictable results. For instance, by repeatedly testing configurations they believed to be correct, participants inadvertently sought to confirm their initial assumptions rather than explore alternative explanations. As a result, the group's problem-solving process lacked the breadth and creativity that might have emerged from a different approach.

H. *Efficiency Seeking*

Participants were mindful of the limited test cases and attempted to be more efficient. Through verbal reminders such as "*we have a limited number of attempts* " and "*we did not get any new information*", participants aimed to maximize information gained from each test case, reflecting a general bias towards resource optimization and efficiency-seeking.

This focus on efficiency was evident in their strategic selection of test cases. Participants consistently reminded each other of the constraints and made deliberate choices to avoid wasting attempts on redundant or less insightful tests. However, it is important to note that this efficiency-seeking behaviour coexisted with instances of sunk cost fallacy. While participants aimed to be efficient, they sometimes persisted with repeated dice rolls that were perceived to confirm their initial hypotheses. This indicates an approach where an unwillingness to abandon supported test strategies sometimes surpasses the efficiency goal.

I. *Memory Biases*

Memory biases, such as faulty recollection of previous results, affected participants' hypotheses. This bias influenced the interpretation of test outcomes and the formulation of subsequent hypotheses.

Participants occasionally misremembered previous test results throughout the sessions, which led to incorrect assumptions. For example, after discussing prior outcomes, they sometimes inaccurately recall the specifics of those results, leading to misguided strategies based on these faulty memories. Statements like " *If we put 5 dice [answer is a] 6. No, when we did [previous test] we got a zero.* " illustrate how these memory errors impacted their problem-solving process.

J. *Experimentation and Learning Through Trial and Error*

The process of hypothesizing, testing, and adjusting based on outcomes was emphasized as a fundamental approach to learning and problem-solving in testing.

Participants engaged in a cycle of experimentation, where they proposed hypotheses, tested them and then refined their ideas based on the results. However, this trial-and-error method also revealed the influence of various biases. Confirmation bias, for example, often led participants to favour hypotheses that aligned with their initial beliefs. Statements like " *Yeah, it must pass. It should pass.*" highlighted how they sometimes attached to their preconceived notions rather than fully considering the test results.

K. *Overconfidence and Underestimation of Complexity*

Initial attempts and conversations indicated overconfidence in quickly solving the test problem or underestimating its complexity. This theme is relevant to cognitive biases such as the Dunning-Kruger effect [19], where individuals might overestimate their understanding or skills. Despite limited evidence, participants sometimes expressed certainty in their knowledge of the game's rules or outcomes.

From the beginning of the sessions, participants tended to assume that the problem would be easily solvable. This overconfidence was evident in their initial hypotheses and conclusions from early test cases. Statements like "*It must be this* ", made with little supporting evidence, highlighted their belief that they had quickly grasped the game's rules.

L. *Comparison of Theoretical and Practical Training Phases*

Our study also sought to understand how different theoretical and practical methods impacted participants' problem-solving behaviour. In the first session, during the theoretical training phase, participants were familiarised with cognitive biases. In the second session, during the practical training phase, participants experienced biases in action. In both sessions, confirmation bias and anchoring bias were prevalent. Groupthink remained a challenge, with participants often seeking quick consensus.

However, the second session (containing the practical training phase) highlighted some differences. Participants exhibited greater risk aversion, showing caution in selecting test cases to avoid errors. They were mindful of optimizing resources and carefully planning their tests to maximize information gain, a theme less evident in the first session.

Despite these observations, our study's results remain inconclusive. While we identified some patterns and differences between the training methods, more in-depth studies are needed to understand better the impact of theoretical versus practical bias awareness training. Future research should include larger sample sizes and explore additional variables to provide clearer insights into the effectiveness of these training methods.

VI. Comparative Analysis of Survey and Experiment Results

The survey showed that professionals are aware of confirmation bias, focusing on tests that confirm expected functionality. The experiment similarly found that participants sought test cases confirming their beliefs. The sunk cost fallacy was not emphasized in the survey but was noted in the experiment, where participants were reluctant to abandon unproductive strategies. The survey did not highlight overconfidence or underestimation of complexity, but the experiment found participants often overconfident and underestimated testing task complexity.

The controlled experiment provided new insights that were not evident from the survey:

- *Detailed Observation of Biases in Action.* The experiment provided concrete examples of how biases manifest in real-time problem-solving situations, offering a deeper understanding of the dynamics involved.
- *Impact of Training on Bias Mitigation.* By comparing the effects of theoretical versus practical bias awareness training, the experiment revealed differences in participants' problem-solving behaviour, highlighting the need for more in-depth studies on training methods.
- *Complexity of Collaborative Dynamics*. The experiment underscored the dual nature of collaboration, where working together could both aid and hinder problem-solving

in testing due to groupthink and consensus-seeking behaviours.

- *Behavioral Adaptations and Strategy Shifts*. Observations from the experiment showed how participants adapted their testing strategies over time, providing insights into the iterative nature of problem-solving and the persistence of certain biases despite awareness.

VII. Threats to Validity

The controlled experiment examined how testers handle a hypothetical test problem. Despite efforts to create realism in a code-agnostic way, the task's simplicity may not fully capture real-world software testing complexity. Nevertheless, the experiment provided insights into participants' testing-related problem-solving behaviours and biases.

Other potential threats to validity included the influence of the moderator's comments on participants' strategies, which limited the solutions attempted by the participants. This underscores the need to minimize external influences.

Participant selection posed another concern for the survey. Non-probability sampling might introduce selection bias. While diverse practitioners were included, they may not fully represent the broader software testing population. This limitation was recognized, and efforts were made to include participants with varied industrial backgrounds.

Strategies to address these threats included pilot testing the survey and procedures, establishing consistent protocols, and validating findings using multiple data sources, such as survey responses, verbal protocols, and thematic analysis.

VIII. Conclusions

This study explores the impact of cognitive biases on software testing through a survey and a controlled experiment, focusing on confirmation bias, fixation, and optimism bias among testers. While many testers are aware of these biases, they often struggle to identify them in their own work, suggesting the presence of a bias blind spot. The controlled experiment underscored instances of common biases, such as confirmation bias, groupthink, and the sunk cost fallacy. Although collaboration in testing can be beneficial, it sometimes contributes to groupthink and a preference for agreement over considering alternative testing strategies.

The study offers insights into sunk cost fallacy and overconfidence in software testing, which have been less emphasized in previous research on testing-related biases. Moreover, the initial findings suggest that practical bias awareness training could be more beneficial than theoretical training.

IX. Future Work

In our future work, we plan to investigate how training can help mitigate some effects of bias. We plan to create version two of the questionnaire and send it to more participants, which would require the help of the testing community. In addition, we plan to look deeper into the answers to some of the questions related to test design.

X. Acknowledgements

The Software Center project and Vinnova partially funded this work through the SmartDelta project. We want to express our gratitude to Dr Thomas Lindner, Principal Key Expert for Integration, Test, and Validation at Siemens Mobility GmbH, for his valuable comments and insights on the draft of this study.

References

[1] G. Çalıklı and A. B. Bener, "Influence of confirmation biases of developers on software quality: an empirical study," *Software Quality Journal*, vol. 21, pp. 377–416, 2013.

[2] I. Salman, B. Turhan, and S. Vegas, "A controlled experiment on time pressure and confirmation bias in functional software testing," *Empirical Software Engineering*, vol. 24, pp. 1727–1761, 2019.

[3] S. Chattopadhyay, N. Nelson, A. Au, N. Morales, C. Sanchez, R. Pandita, and A. Sarma, "A tale from the trenches: cognitive biases and software development," in *ICSE'20*, 2020.

[4] R. Mohanani, I. Salman, B. Turhan, P. Rodríguez, and P. Ralph, "Cognitive biases in software engineering: a systematic mapping study," *IEEE Transactions on Software Engineering*, vol. 46, no. 12, pp. 1318–1339, 2018.

[5] P. Slovic and A. Tversky, *Judgment under uncertainty: Heuristics and biases*. Cambridge university press, 1982.

[6] M. G. Haselton, D. Nettle, and P. W. Andrews, "The evolution of cognitive bias," *The handbook of evolutionary psychology*, pp. 724–746, 2015.

[7] R. O'Gorman, D. S. Wilson, and R. R. Miller, "An evolved cognitive bias for social norms," *Evolution and Human Behavior*, vol. 29, no. 2, pp. 71–78, 2008.

[8] M. Shepperd, C. Mair, and M. Jørgensen, "An experimental evaluation of a de-biasing intervention for professional software developers," in *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*, 2018, pp. 1510–1517.

[9] M. Petre, "Technical perspective: Exploring cognitive bias' in the wild'," *Communications of the ACM*, vol. 65, no. 4, pp. 114–114, 2022.

[10] M. Swillus and A. Zaidman, "Sentiment overflow in the testing stack: Analyzing software testing posts on stack overflow," *Journal of Systems and Software*, vol. 205, p. 111804, 2023.

[11] I. Salman, "Cognitive biases in software quality and testing," *2016 IEEE/ACM 38th IEEE International Conference on Software Engineering Companion*, 2016.

[12] M. Jørgensen and E. Papatheocharous, "Believing is seeing: Confirmation bias studies in software engineering," *2015 41st Euromicro Conference on Software Engineering and Advanced Applications*, 2015.

[13] G. Calikli and A. Bener, "Empirical analyses of the factors affecting confirmation bias and the effects of confirmation bias on software developer/tester performance," *Proc. 6th Int. Conf. Predictive Models in Software Engineering*, 2010.

[14] P. C. Wason, "On the failure to eliminate hypotheses in a conceptual task," *Quarterly Journal of Experimental Psychology*, vol. 12, no. 3, pp. 129–140, 1960. [Online]. Available: https://doi.org/10.1080/17470216008416717

[15] A. Green, "Verbal protocol analysis." *The psychologist*, 1995.

[16] V. Braun and V. Clarke, *Thematic analysis.* American Psychological Association, 2012.

[17] Y. Okan, F. Blanco, D. Petrova, M. Capra, and J. C. Perales, "Understanding and overcoming biases in judgment and decision-making with real-life consequences," *Frontiers in Psychology*, vol. 13, p. 917896, 2022. [Online]. Available: https://doi.org/10.3389/fpsyg.2022.917896

[18] R. Team, "Educational strategies in the health professions to mitigate cognitive and implicit bias impact on decision making: a scoping review," *BMC Medical Education*, 2022. [Online]. Available: https://bmcmededuc.biomedcentral.com/articles/10.1186/s12909-022-03474-8

[19] D. Dunning, "The dunning–kruger effect: On being ignorant of one's own ignorance," in *Advances in experimental social psychology*. Elsevier, 2011, vol. 44, pp. 247–296.