

**FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO**

# **Measuring What Matters: Individual Vs. Team Productivity in Software Development**

**Matilde Faro Martins Castelo Pires**



**Mestrado em Engenharia de Software**

Supervisor: Pavlina Gonçalves

Second Supervisor: Prof. Filipe Correia

July 10, 2026



# **Measuring What Matters: Individual Vs. Team Productivity in Software Development**

**Matilde Faro Martins Castelo Pires**

Mestrado em Engenharia de Software

Approved in oral examination by the committee:

President: Prof. Jorge Melegati

Examiner: Carolin Brandt

Member: Pavlína Gonçalves

July 10, 2026

# Resumo

A avaliação da produtividade é central na gestão de projetos de software, mas continua a ser um dos conceitos mais contestados e ambíguos na engenharia de software. A investigação existente tem produzido enquadramentos teóricos e qualitativos sobre como a produtividade deve ser definida e medida, mas a evidência empírica sobre a forma como os profissionais vivenciam e lidam com a avaliação, particularmente a tensão entre a avaliação individual e a avaliação de equipa, permanece limitada.

Este estudo investiga como gestores e desenvolvedores conceptualizam e avaliam a produtividade a nível individual e de equipa, empregando um desenho misto sequencial exploratório: 7 entrevistas piloto com gestores de engenharia de software, seguidas de um inquérito transversal aplicado a 20 gestores e 36 desenvolvedores. Quatro questões de investigação orientaram o estudo, examinando como as definições de produtividade divergem entre funções, como cada função perceciona o uso e o uso indevido de métricas, em que contextos as métricas são consideradas úteis e fiáveis, e sob que condições a produtividade individual ou de equipa é favorecida.

Os resultados revelam que os gestores definem a produtividade em termos orientados para o output, enquanto os programadores atribuem significativamente maior peso a atividades colaborativas. A maior divergência na utilidade percebida das métricas ocorre entre métricas aplicáveis tanto à análise individual como de equipa, onde os gestores as classificam consistentemente como mais úteis. O conhecimento dos desenvolvedores sobre o propósito da avaliação é o principal preditor do comportamento de otimização em relação aos critérios, enquanto os gestores que reconhecem que as métricas incentivam comportamentos de manipulação de métricas não reduzem o uso das categorias de métricas mais suscetíveis a essa prática. Ambos os gestores e os desenvolvedores favorecem fortemente o comportamento orientado para a equipa em cenários concretos, mas também subestimam a orientação colaborativa entre si. A única exceção consistente à primazia da equipa é a proteção do tempo de foco individual, que ambas as funções priorizam em detrimento da disponibilidade para a equipa.

Estes resultados sugerem que o desalinhamento documentado na literatura é primariamente perceptual em vez de substantivo. A avaliação responsável da produtividade exige não apenas melhores métricas, mas também uma governação mais clara de como as métricas são comunicadas e utilizadas entre cargos, e estruturas que protejam o tempo de foco como uma norma de equipa.

**Palavras-chave:** Produtividade em Engenharia de Software • Produtividade Individual • Productividade em Equipa • Métricas de Produtividade • Uso Indevido de Métricas

# Abstract

Productivity assessment sits at the centre of software project management, yet it remains one of the most contested and ambiguous concepts in software engineering. Existing research has produced frameworks and qualitative accounts of how productivity should be defined and measured, but empirical evidence on how practitioners actually experience and navigate assessment, especially the tension between individual- and team-level assessment, remains limited.

This study investigates how managers and developers conceptualise and assess productivity at individual and team levels, employing an exploratory sequential mixed-methods design: 7 pilot interviews with software engineering managers, followed by a cross-sectional survey administered to 20 managers and 36 developers. Four research questions guided the study, examining how productivity definitions diverge between roles, how each role perceives metric use and misuse, in what contexts metrics are considered useful and reliable, and under what conditions individual or team productivity is favoured.

The findings reveal that managers define productivity in output-oriented terms, while developers place significantly greater weight on collaborative activities. The greatest divergence in perceived metric usefulness occurs among metrics applicable to both individual and team analysis, where managers consistently rate them higher. Developers' awareness of the purpose of assessment is the primary predictor of criteria optimisation behaviour, while managers who acknowledge that metrics incentivise gaming do not reduce their use of the most gameable metric categories. Both roles strongly favour team-oriented behaviour in concrete trade-offs, but both also underestimate the other's collaborative orientation. The only consistent exception to team primacy is protecting individual focus time, which both roles prioritise over team availability.

These findings suggest that the misalignment documented in the literature is primarily perceptual rather than substantive. Responsible productivity assessment requires not just better metrics, but also clearer governance of how metrics are communicated and used across roles, and structures that protect focus time as a team norm.

**Keywords:** Software Engineering Productivity • Individual Productivity • Team Productivity • Productivity Metrics • Metric Misuse

# The United Nations Sustainable Development Goals

The United Nations **SDG!**s (**SDG!**s) provide a global framework for achieving a better and more sustainable future for all. It includes 17 goals (<https://sdgs.un.org/goals>) to address the world's most pressing challenges, including poverty, inequality, climate change, environmental degradation, peace, and justice.

This thesis actively contributes to advancing sustainable and inclusive industrial practices by investigating how productivity is defined, measured, and used in software engineering organisations. By documenting the consequences of metric misuse on developer well-being, collaboration, and long-term value creation, this work provides empirical evidence and practical guidance for aligning productivity assessment with the principles of decent work, economic productivity, and participatory governance, directly supporting specific Sustainable Development Goals.

The specific Sustainable Development Goals mentioned have the following names:

**SDG 8** Promote sustained, inclusive and sustainable economic growth, full and productive employment and decent work for all

**SDG 16** Promote peaceful and inclusive societies for sustainable development, provide access to justice for all and build effective, accountable and inclusive institutions at all levels

| SDG | Target | Contribution                                                                                                                                                                                                                                                                           | Performance Indicators and Metrics                                                                                                                                                                        |
|-----|--------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 8   | 8.2    | Achieve higher levels of economic productivity through diversification, technological upgrading and innovation, including through a focus on high-value added and labour-intensive sectors.                                                                                            | Alignment between productivity metrics and innovation outcomes. Long-term value creation vs short-term output focus. Awareness of the limitations of decontextualised metrics. ...                        |
|     | 8.3    | Promote development-oriented policies that support productive activities, decent job creation, entrepreneurship, creativity and innovation, and encourage the formalization and growth of micro-, small- and medium-sized enterprises, including through access to financial services. | Adoption of transparent metric governance policies. Recognition of collaborative and innovative activities. Reduction in perceived metric misuse. Formalisation of productivity assessment practices. ... |
|     | 8.5    | By 2030, achieve full and productive employment and decent work for all women and men, including for young people and persons with disabilities, and equal pay for work of equal value.                                                                                                | Developer perception of assessment fairness. Protection against metric-driven gaming behaviour. Recognition of "invisible" productive activities. ...                                                     |
|     | 8.8    | Protect labour rights and promote safe and secure working environments for all workers, including migrant workers, in particular women migrants, and those in precarious employment                                                                                                    | Protection of cognitive sovereignty and focus time. Transparency in assessment intent. Reduction in misinterpretation of assessments. ...                                                                 |
| 16  | 16.7   | Ensure responsive, inclusive, participatory and representative decision-making at all levels.                                                                                                                                                                                          | Developer involvement in metric design. Transparency of assessment communication. Reduction in the perception gap between roles. Role alignment in collaborative orientation. ...                         |

# Acknowledgements

This thesis marks the end of a long and challenging journey, one filled with learning, growth, and moments of both uncertainty and accomplishment. The path was not always straightforward, but every step along the way was shaped by the support, encouragement, and guidance of many people, to whom I am deeply grateful.

Firstly, I would like to express my sincere gratitude to my supervisors, Pavlina Gonçalves and Filipe Correia, for their invaluable guidance, thoughtful feedback, and continuous support throughout this research. Their expertise and encouragement were instrumental in shaping this work, and I am deeply grateful for their availability and trust in my process.

To Eduardo, thank you for your patience and encouragement. You have been by my side for the last five years, especially in these last two, and you have made this journey truly more special. Thank you for always supporting me and keeping me motivated, and for being there to remind me that I can do anything I put my mind to. I look forward to many more years of your support and affection.

To the friends I made during this master's degree - Rui, Cláudia, José, Leonor, André, and Nuno - thank you for the companionship, shared struggles, and moments of joy and laughter that made this journey more meaningful. Thank you for always joining me in playing board games, going on little adventures (and ringing bells of abandoned churches) and always being there to listen to me.

To the friends I made during my bachelor's degree - Ricardo, Diogo, and João - thank you for all the fun moments we shared during these last five years, for being there for me during tough times and for always knowing how to make me laugh. For more years of Roblox, Catan, and inside jokes that I still do not always get.

Finally, my deepest appreciation goes to my parents, Sandra and Hugo, and my sister, Madalena. Without your unconditional love and support, this journey would have been much harder. Thank you for always believing in me, even when I didn't, and for supporting my dreams. You all inspire me to be better every day, and I could not have done it without you. I love you very much, and I hope you are proud of me.

Matilde Faro Pires

# Declaration of honour

I declare that this dissertation is my own work and has not previously been submitted or assessed at this or any other institution. References to other authors (statements, ideas, thoughts), as well as the use of published works, strictly comply with the rules of attribution and are duly identified in the text and in the bibliographic references, in accordance with the applicable referencing standards. I am aware that the practice of plagiarism or self-plagiarism constitutes an academic offence, as established in the U.Porto Code of Ethics.

I also declare that I have used Artificial Intelligence tools to complete part(s) of this dissertation, and that this use and its results are duly noted and have been carefully checked for errors, inaccuracies, unfounded attributions or other forms of bias in content and arguments, as well as determining authorship.

I acknowledge the use of the following generative artificial intelligence tools in the development of this thesis. These technologies were utilised under strict supervision, and all AI-generated outputs were critically reviewed and verified for accuracy. I accept full responsibility for the validity and originality of the content and confirm that the use of these tools aligns with all institutional policies and academic integrity standards.

**Generative AI tools used:** Claude, Grammarly, and Deepseek.

**Activities supported by Generative Artificial Intelligence Tools:**<sup>1</sup>

---

<sup>1</sup>This list extends the GAIDeT taxonomy.

**Conceptualization**

- |                                                 |                                               |                                             |
|-------------------------------------------------|-----------------------------------------------|---------------------------------------------|
| <input type="checkbox"/> Idea generation        | <input type="checkbox"/> Defining objectives  | <input type="checkbox"/> Research questions |
| <input type="checkbox"/> Feasibility assessment | <input type="checkbox"/> Hypothesis viability | <input type="checkbox"/> Simulated debate   |

**Literature Review**

- |                                             |                                           |                                          |
|---------------------------------------------|-------------------------------------------|------------------------------------------|
| <input type="checkbox"/> Search & Discovery | <input type="checkbox"/> Summarization    | <input type="checkbox"/> Concept Mapping |
| <input type="checkbox"/> Patterns & Themes  | <input type="checkbox"/> Identifying Gaps | <input type="checkbox"/> Translation     |

**Methodology**

- |                                          |                                      |                                     |
|------------------------------------------|--------------------------------------|-------------------------------------|
| <input type="checkbox"/> Research Design | <input type="checkbox"/> Prototyping | <input type="checkbox"/> Validation |
|------------------------------------------|--------------------------------------|-------------------------------------|

**Software Development and Automation**

- |                                             |                                       |                                                   |
|---------------------------------------------|---------------------------------------|---------------------------------------------------|
| <input type="checkbox"/> Code Generation    | <input type="checkbox"/> Optimisation | <input checked="" type="checkbox"/> Debugging     |
| <input checked="" type="checkbox"/> Scripts | <input type="checkbox"/> Algorithms   | <input checked="" type="checkbox"/> Documentation |

**Data Management**

- |                                              |                                             |                                         |
|----------------------------------------------|---------------------------------------------|-----------------------------------------|
| <input type="checkbox"/> Collection/Scraping | <input type="checkbox"/> Validation/Quality | <input type="checkbox"/> Cleaning       |
| <input type="checkbox"/> Curation            | <input type="checkbox"/> Analysis           | <input type="checkbox"/> Visualisation  |
| <input type="checkbox"/> Reporting Results   | <input type="checkbox"/> Labeling           | <input type="checkbox"/> Synthetic Data |
| <input type="checkbox"/> Anonymisation       | <input type="checkbox"/> Translation        |                                         |

**Visuals**

- |                                           |                                  |                                          |
|-------------------------------------------|----------------------------------|------------------------------------------|
| <input type="checkbox"/> Image Generation | <input type="checkbox"/> Editing | <input type="checkbox"/> Charts/Diagrams |
|-------------------------------------------|----------------------------------|------------------------------------------|

**Writing and Editing**

- |                                           |                                                  |                                            |
|-------------------------------------------|--------------------------------------------------|--------------------------------------------|
| <input type="checkbox"/> Drafting         | <input checked="" type="checkbox"/> Polishing    | <input type="checkbox"/> Summarising       |
| <input type="checkbox"/> Connecting Ideas | <input type="checkbox"/> Tone Adjustment         | <input type="checkbox"/> Translation       |
| <input type="checkbox"/> References       | <input checked="" type="checkbox"/> Proofreading | <input type="checkbox"/> Title Suggestions |

**Ethics**

- |                                               |                                          |                                     |
|-----------------------------------------------|------------------------------------------|-------------------------------------|
| <input type="checkbox"/> Bias Mitigation      | <input type="checkbox"/> Risk Assessment | <input type="checkbox"/> Compliance |
| <input type="checkbox"/> Conflict of Interest |                                          |                                     |

**Critique & Feedback**

- |                                                  |                                                         |                                             |
|--------------------------------------------------|---------------------------------------------------------|---------------------------------------------|
| <input checked="" type="checkbox"/> Q&A/Querying | <input checked="" type="checkbox"/> Troubleshooting     | <input type="checkbox"/> Identifying Limits |
| <input type="checkbox"/> Future Work Recs        | <input type="checkbox"/> Response to Reviewers checking |                                             |

Matilde Faro Martins Castelo Pires




---

*“Imagination is the highest form of research.”*

Albert Einstein

# Contents

|          |                                                                                                                                        |           |
|----------|----------------------------------------------------------------------------------------------------------------------------------------|-----------|
| <b>1</b> | <b>Introduction</b>                                                                                                                    | <b>1</b>  |
| <b>2</b> | <b>Related Work</b>                                                                                                                    | <b>4</b>  |
| 2.1      | Definition and Conceptualisation of Productivity . . . . .                                                                             | 4         |
| 2.2      | Purposes of Assessment . . . . .                                                                                                       | 6         |
| 2.3      | Individual Productivity . . . . .                                                                                                      | 7         |
| 2.4      | Team Productivity . . . . .                                                                                                            | 8         |
| 2.5      | Misalignment Problem . . . . .                                                                                                         | 9         |
| 2.6      | Implications for Research and Practice . . . . .                                                                                       | 9         |
| <b>3</b> | <b>Methodology</b>                                                                                                                     | <b>11</b> |
| 3.1      | Research Questions . . . . .                                                                                                           | 11        |
| 3.2      | Research Philosophy And Approach . . . . .                                                                                             | 12        |
| 3.3      | Research Design Overview . . . . .                                                                                                     | 12        |
| 3.4      | Literature Review . . . . .                                                                                                            | 14        |
| 3.4.1    | Methodology . . . . .                                                                                                                  | 14        |
| 3.4.2    | Key Findings . . . . .                                                                                                                 | 15        |
| 3.4.3    | Informing the Pilot Interviews . . . . .                                                                                               | 17        |
| 3.5      | Pilot Interviews . . . . .                                                                                                             | 18        |
| 3.5.1    | Participant Recruitment . . . . .                                                                                                      | 18        |
| 3.5.2    | Interview Protocol . . . . .                                                                                                           | 18        |
| 3.5.3    | Analysis of Pilot Data . . . . .                                                                                                       | 20        |
| 3.5.4    | Key Findings . . . . .                                                                                                                 | 21        |
| 3.5.5    | Informing the Survey . . . . .                                                                                                         | 22        |
| 3.6      | Survey . . . . .                                                                                                                       | 23        |
| 3.6.1    | Survey Development . . . . .                                                                                                           | 25        |
| 3.6.2    | Survey Distribution and Data Collection . . . . .                                                                                      | 25        |
| 3.6.3    | Analysis of Survey Data . . . . .                                                                                                      | 26        |
| 3.7      | Ethical Considerations . . . . .                                                                                                       | 27        |
| <b>4</b> | <b>Results</b>                                                                                                                         | <b>29</b> |
| 4.1      | Sample Characterisation . . . . .                                                                                                      | 29        |
| 4.2      | RQ1 - How do managers' and developers' perceptions and definitions of productivity diverge in software engineering contexts? . . . . . | 30        |
| 4.3      | RQ2 - How do managers and developers perceive the use and misuse of individual and team productivity metrics? . . . . .                | 31        |
| 4.4      | RQ3 - In what context do developers and managers find individual and team productivity metrics useful and reliable? . . . . .          | 36        |

|          |                                                                                                    |           |
|----------|----------------------------------------------------------------------------------------------------|-----------|
| 4.5      | RQ4 - In what context do developers and managers favour individual or team productivity? . . . . . | 45        |
| <b>5</b> | <b>Discussion</b>                                                                                  | <b>51</b> |
| 5.1      | The Nature of Misalignment: Definitions and Perceptions . . . . .                                  | 51        |
| 5.2      | The Belief-Behaviour Inconsistency . . . . .                                                       | 52        |
| 5.3      | The Focus Time Exception: Where Individual Productivity Wins . . . . .                             | 54        |
| 5.4      | What Metrics Are Actually Used For: Dual-Level Utility and Its Risks . . . . .                     | 55        |
| 5.5      | Implications for Research and Practice . . . . .                                                   | 55        |
| <b>6</b> | <b>Threats to Validity</b>                                                                         | <b>57</b> |
| 6.1      | Internal Validity . . . . .                                                                        | 57        |
| 6.2      | External Validity . . . . .                                                                        | 57        |
| 6.3      | Construct Validity . . . . .                                                                       | 58        |
| 6.4      | Conclusion Validity . . . . .                                                                      | 58        |
| <b>7</b> | <b>Conclusion</b>                                                                                  | <b>59</b> |
|          | <b>References</b>                                                                                  | <b>62</b> |
| <b>A</b> | <b>Summary of Included Papers</b>                                                                  | <b>64</b> |
| <b>B</b> | <b>Interview Guide</b>                                                                             | <b>67</b> |
| <b>C</b> | <b>Survey Structure</b>                                                                            | <b>69</b> |

# List of Figures

|      |                                                                                                                               |    |
|------|-------------------------------------------------------------------------------------------------------------------------------|----|
| 3.1  | Research Design Overview . . . . .                                                                                            | 13 |
| 3.2  | PRISMA Flow Diagram: the selection process. . . . .                                                                           | 16 |
| 4.1  | Experiences of Participants . . . . .                                                                                         | 29 |
| 4.2  | Immediate Team Sizes . . . . .                                                                                                | 29 |
| 4.3  | Work Setups of Respondents . . . . .                                                                                          | 30 |
| 4.4  | Company Sizes of Participants . . . . .                                                                                       | 30 |
| 4.5  | Definitions of Productivity - Developers vs Managers . . . . .                                                                | 31 |
| 4.6  | Relation between Belief in "Providing metrics incentivises gaming" and Improvement on Manager Criteria . . . . .              | 32 |
| 4.7  | Relation between Awareness and Optimisation . . . . .                                                                         | 33 |
| 4.8  | Relation between Individual/Team Awareness and Optimisation . . . . .                                                         | 33 |
| 4.9  | Correlation between Awareness, Gaming and Behaviour . . . . .                                                                 | 34 |
| 4.10 | Comparison between Belief in "Providing metrics incentivises gaming" and Managers using Output & Efficiency Metrics . . . . . | 34 |
| 4.11 | Misuse-Fairness-Gaming Correlation Triad - Developers . . . . .                                                               | 35 |
| 4.12 | Misinterpretation & Fairness - Full Distribution . . . . .                                                                    | 36 |
| 4.13 | Metric Category Use Rate vs Perceived Usefulness - Managers . . . . .                                                         | 37 |
| 4.14 | Perceived Usefulness of Metric Categories . . . . .                                                                           | 38 |
| 4.15 | Usefulness Rating Distribution per Metric Category - Heatmaps . . . . .                                                       | 39 |
| 4.16 | Mean Communication & Collaboration Usefulness Score by Work Setup and Role . . . . .                                          | 39 |
| 4.17 | Contextualisation Consensus between Roles . . . . .                                                                           | 40 |
| 4.18 | Quantitative vs Contextual Assessment Beliefs . . . . .                                                                       | 40 |
| 4.19 | Quantitative to Contextual Belief Shift by Role . . . . .                                                                     | 41 |
| 4.20 | Context Belief and Useful Ratings - Spearman . . . . .                                                                        | 42 |
| 4.21 | Comparison between "Metrics must be contextualised" belief and metric category use . . . . .                                  | 43 |
| 4.22 | Belief in Contextualisation vs Use of Output & Efficiency metrics . . . . .                                                   | 43 |
| 4.23 | Mean Usefulness Ratings by Context Group . . . . .                                                                            | 44 |
| 4.24 | Individual vs Team Productivity Beliefs . . . . .                                                                             | 46 |
| 4.25 | M1 vs D1 - Actual Views . . . . .                                                                                             | 46 |
| 4.26 | M2 vs D1 & D2 vs M1 - Prediction of Other Role Preferences . . . . .                                                          | 47 |
| 4.27 | Percentage of Respondents Choosing Scenario B per Perspective . . . . .                                                       | 48 |
| 4.28 | Team > Productivity Belief vs Scenario Choices . . . . .                                                                      | 49 |

# List of Tables

|     |                                                                   |    |
|-----|-------------------------------------------------------------------|----|
| 3.1 | Research Questions . . . . .                                      | 12 |
| 3.2 | Research Phases . . . . .                                         | 13 |
| 3.3 | Search Query Keywords Used for the Literature Review . . . . .    | 15 |
| 3.4 | Demographic Information of Pilot Interview Participants . . . . . | 19 |
| 3.5 | Mapping of Interview Topic Areas to Research Questions . . . . .  | 19 |
| 3.6 | Metric Categories, Descriptions, and Level of Analysis . . . . .  | 24 |
| 3.7 | Survey Structure - Categories and Question Distribution . . . . . | 26 |
| A.1 | Summary of Included Studies . . . . .                             | 64 |

# Chapter 1

## Introduction

Productivity assessment sits at the centre of software project management: managers rely on it to allocate resources, identify bottlenecks, and balance delivery goals with developer well-being [6, 11]. Yet, despite its central role in decision-making, productivity in software engineering remains notoriously difficult to define and assess. The collaborative, knowledge-intensive nature of software work resists simplistic quantification, and the metrics most available to managers often fail to capture what actually contributes to successful software development [6].

In software development, misaligned metrics often push individuals to optimise personal output at the expense of collaboration, while team-level measures can hide individual contributions [14]. This suggests a fundamental mismatch: the purposes for which individual metrics are used, e.g. performance evaluation, may directly conflict with the needs of team-based work, e.g. fostering collaboration. This tension can lead to reduced well-being and weakened teamwork, making productivity difficult to assess but critical to get right [14, 17].

Furthermore, software is normally the result of many developers' contributions, and its collaborative nature makes it difficult to isolate an individual's productivity from the productivity of the team as a whole [6, 7]. Consequently, without a coherent and consensual approach to productivity, organisations risk undermining the very goals they seek to achieve [14, 15]. This risk manifests in damaged team dynamics, impaired decision-making, and strategic misalignment. It can also foster internal competition, reduce knowledge sharing, and discourage essential collaborative activities, such as code review or mentoring, because these are often less visible in individual metrics [6, 13, 16].

Managers are left with fragmented or misleading signals, making it difficult to accurately evaluate performance, allocate resources effectively, or identify genuine bottlenecks [3, 14]. The practical utility and reliability of any given metric are therefore context-dependent, varying based on who is using it and for what purpose [7]. When individual, team, and organisational measures point in different directions, it becomes harder to ensure that daily engineering work is consistently driving broader business outcomes [14].

Together, these challenges point to a specific and underexamined gap: while prior work has produced frameworks and qualitative accounts of productivity in software engineering [1, 8, 16],

there is limited cross-role, quantitative evidence on how managers conceptualise productivity, how managers and developers perceive the use and misuse of individual and team metrics, and in what contexts those metrics are considered useful, reliable, or favoured over one another.

The literature suggests that productivity at individual and team levels reflects fundamentally different phenomena: individual productivity is often associated with task completion, personal efficiency, and self-management [7, 8, 16], while team productivity depends more strongly on coordination, communication, shared norms, and collective decision-making [10, 18]. Yet, it remains unclear how these distinct conceptual levels translate into managers' daily needs, or how often and in what contexts different metrics are applied, and potentially misapplied [3]. Without a clear understanding of these differences, organisations risk adopting assessment practices that fail to capture what actually contributes to successful software development.

Addressing this gap is important for several reasons. For organisations, reliance on misleading productivity metrics can result in misallocated resources, strategic misalignment, and impaired decision-making [3, 14]. For teams, metrics that incentivise individual output over collaboration can erode trust, reduce knowledge sharing, and weaken team cohesion [6, 13, 16]. For individual developers, being evaluated on narrow, decontextualised metrics can lead to stress, reduced job satisfaction, and a sense of being undervalued [1, 14, 17]. These consequences stem not only from the metrics themselves but also from the differing ways managers and developers perceive, interpret, and react to them.

Understanding these perceptual differences is therefore critical and central to this study.

This study addresses that concern through a mixed-methods design combining pilot interviews with project managers and a cross-role survey for managers and developers. It aims to provide empirically grounded insights that inform more responsible, context-appropriate assessment practices.

To achieve this, the following objectives guide the research:

- Capture how managers and developers define productivity,
- Compare managers' and developers' perceptions of productivity metric use and misuse across individual and team levels,
- Identify the specific contexts in which practitioners consider productivity metrics useful, reliable, and applicable,
- Determine the conditions under which practitioners prioritise team productivity over individual productivity, and vice-versa.

Through pilot interviews and a cross-role survey, this study finds that managers define productivity in output-oriented terms, that developer awareness of assessment purpose is the primary driver of criteria optimisation behaviour, that both roles perceive product-delivery metrics as most useful while consistently underusing individual development and well-being metrics, and that both roles favour team-oriented behaviour in practice while systematically underestimating the other's collaborative orientation. For research, these findings provide quantitative cross-role evidence on

several phenomena previously documented only qualitatively, and identify purpose awareness as a novel predictor of developer behaviour with implications for how transparency in assessment is theorised. For practice, they indicate that responsible productivity assessment requires not just better metrics, but also clearer governance of how metrics are communicated and used across roles.

## Chapter 2

# Related Work

The reviewed literature points to three recurring challenges in productivity assessment: persistent disagreement over how productivity should be defined, recurring tensions between individual and team-level assessment, and widespread concern about metric misuse and its negative consequences.

### 2.1 Definition and Conceptualisation of Productivity

Defining and measuring productivity in software engineering remains a persistent challenge [16]. What productivity *is* and how it should be *measured* are different questions, though they are frequently conflated in the literature [7]. A measurement approach presupposes a definition, yet in software engineering, the definition of productivity remains abstract, which means that many measurement frameworks are built on implicit or partial conceptualisations of the construct they claim to capture [14]. Despite its central role in project management and organisational decision-making, the literature lacks a universally accepted definition [14]. However, productivity is most frequently defined as a relationship between output and input, commonly expressed by the formula  $\text{Productivity} = \text{Output} / \text{Input}$  [1, 11, 14]. In software engineering, productivity cannot be measured by simple output units, as work is intangible, collaborative, and resists straightforward quantification [14].

This distinction between definition and measurement matters because the difficulties encountered in each are different in kind. The definitional challenge is conceptual: there is no consensus on what outcomes count as productive in software work, whether productivity should be understood at the level of the individual, the team, or the organisation, or how quality, speed and value should be weighted against one another [7, 14]. The measurement challenge is technical and practical: even when a definition is adopted, operationalising it requires selecting proxies that may not faithfully represent the underlying construct [1, 18]. Conflating these challenges, by treating a measurement difficulty as if it were a definitional one, or vice versa, produces solutions that are mismatched to the actual problem.

As a result, the literature proposes multiple approaches to conceptualise and measure productivity. Physical size-based metrics, most commonly source lines of code (SLOC), prioritise the size and format of programs over their efficiency and quality. Functional size-based metrics take into account the number of features perceived by the user. Design-based metrics correlate with SLOC but focus on design aspects, such as the number of modules, classes, and related metrics. More recently, value-based metrics assess different aspects of what is produced in a software project, based on added value [1]. However, the software development process itself remains less tangible and less clearly defined than in other domains, making it difficult to determine appropriate outcome measures [7].

The applicability of any of these approaches is highly context-dependent: organisational norms, project phase, team maturity, and product domain all shape what productivity means in practice, and the same metric can carry different meanings across teams or individuals [1, 9, 11, 14]. This variability helps explain why no integrative model for productivity measurement has emerged [10].

Critically, all of the metric categories above focus on outputs or processes. A growing strand of the literature argues that this orientation is incomplete because it omits the human factors that both shape and constitute productive work. Forsgren et al. [6] propose the SPACE framework, which includes satisfaction and well-being as a core dimension of developer productivity, alongside activity, communication and collaboration, efficiency and flow, and performance. Their framework reflects the view that well-being is not merely a consequence of productivity but a precondition for it: developers who are dissatisfied, disengaged, or burned out are less productive, and measurement systems that ignore this dimension produce an incomplete and potentially distorted picture. This is corroborated by Murphy-Hill et al. [9], whose large-scale survey found that the strongest predictors of self-rated productivity are non-technical factors, including job enthusiasm, peer support for new ideas, and receiving useful feedback about job performance, none of which appear in traditional output-based metrics. Storey et al. [17] further argue that job satisfaction and perceived productivity are closely linked, and that well-being should be treated as a leading indicator of engineering performance rather than a peripheral concern. Taken together, these contributions suggest that any adequate conceptualisation of software engineering productivity must account for human factors alongside output-oriented ones.

Managers tend to associate productivity with tasks delivered on time, artefacts that do not require rework, products that meet stakeholders' expectations, and personal qualifications and behaviour [11], often focusing on a single output-oriented factor to quantify productivity and failing to account for developers' work practices. [14] Performance<sup>1</sup> and quality of outcomes are priorities for them. In contrast, developers more often define productivity in terms of activities and task completion, which highlights the misalignment between developers' and managers' views. [16]

Because the factors influencing productivity at different levels are complex and interdependent, and because productivity reflects several dimensions of work shaped by context [6], the

---

<sup>1</sup>Performance is considered to be the outcome of a system or process [6]

literature converges on the view that it cannot be reduced to a single metric [14]. This multidimensionality is acknowledged in principle but inconsistently reflected in practice.

These measurement challenges arise from the absence of a settled definition, but they are not reducible to it. The field struggles to agree on how to *measure* productivity in part because it has not established consensus on what productivity *is* [14], but the two problems require separate solutions. A clearer definition would constrain the space of valid measurement approaches, and better measurement instruments would not resolve the definitional ambiguity on their own. This definitional ambiguity directly leads to criterion problems in measurement: without a clear conceptual boundary, any metric risks either missing core aspects of the construct, criterion deficiency, or capturing irrelevant factors, criterion contamination [10].

Across the literature, several studies highlight complementary challenges in measuring productivity in software development. Some authors argue that productivity measurement should not focus exclusively on the outcomes of developer activities, as the software development process itself plays an important role. However, process-related aspects are difficult to measure due to the variability of their costs and impacts [7]. Other studies emphasise that organisations differ substantially in their working practices and perspectives, making it difficult to define a universal model for productivity measurement [1]. In line with this, the literature reports the absence of an integrative model that clearly specifies the components of productivity as well as a lack of generally agreed-upon definitions of productivity [10]. Despite these challenges and the lack of consensus, measurement remains a central practice in software project management [3]. These gaps are directly relevant to this study: if practitioners themselves lack a shared conceptual foundation for what productivity means, then how they define it in practice and whether those definitions diverge across roles becomes an empirical question rather than a settled one. This motivates the investigation of how managers and developers conceptualise productivity and where their definitions diverge.

## 2.2 Purposes of Assessment

While measurement entails costs, it can contribute to more efficient work when used effectively. Measurement is considered useful when it supports understanding and decision-making, provided that the costs of data collection do not exceed the benefits and that unintended side effects are minimised [3]. Such practices play a role in the management of human resources, which has a significant impact on organisational performance [11].

The literature reports multiple managerial motivations to measure productivity. These include reducing production costs, identifying high- and low-performing individuals or teams, forming teams to meet desired time and quality requirements, and allocating resources [1, 11, 14, 17, 18]. Productivity measures are also used to assess the effectiveness of tools, practices, and interventions, as well as to surface organisational trends [8, 14].

At the organisational level, productivity measurement supports strategic planning, policy making, coordination, and employee motivation, though no single operationalisation is found to be

applicable across all contexts [10]. In contrast, developers draw on productivity data for self-optimisation and transparent communication of progress [6, 7]. Management predominantly employs metrics for strategic decision-making, resource allocation, and performance assessment [1, 8, 11, 17, 18], while developers use them to manage their own focus and communicate status to peers and leadership [6, 7]. Productivity measurement, therefore, serves both systemic organisational goals and individual practitioner needs [3, 6], but these purposes are not always aligned, and applying metrics designed for one level to serve the goals of another carries risk [1].

Despite this, the literature provides limited empirical insight into how managers and developers actually understand and experience the purposes of productivity assessment in practice. It remains unclear whether managers and developers are aware of each other's motivations for using metrics, and whether misalignment in perceived purposes contributes to the tensions in how metrics are designed, applied, and received. Understanding this is necessary to determine whether divergent purposes, rather than the metrics themselves, are a primary driver of misuse and friction in productivity assessment.

## 2.3 Individual Productivity

Developers have reported finding value in tracking their own productivity, as it provides insights into their work and supports time, energy, and daily activity management [6]. In contrast, some studies report that developers frequently express negative perceptions of productivity metrics aimed at identifying the performance of individual engineers due to concerns related to privacy and the risk of misinterpretation [14]. This distinction is consequential: when productivity data is shared with superiors, developers tend to optimise their behaviour toward what is being measured rather than toward genuine performance, whereas data collected for personal use is more likely to reflect reality [7]. This points to a direct link between the purpose and audience of assessment and the behavioural response it elicits, a dynamic central to understanding how individual metrics are experienced in practice.

Quantitative analysis suggests that the strongest predictors of self-rated productivity are non-technical factors. The top factors identified across multiple organisations include job enthusiasm, peer support for new ideas, and receiving useful feedback about job performance [9]. This reinforces that individual productivity is shaped more by psychological and social conditions than by the technical output that conventional metrics capture [9, 16].

The tension between individual productivity and collaborative work is a recurring theme in the literature. Activities central to team effectiveness, such as code review, mentoring, and knowledge sharing, often reduce an individual's measurable output, despite their positive impact on collective outcomes [6, 14]. Conversely, optimising for individual efficiency, for example by minimising interruptions, can reduce a developer's capacity to collaborate [6]. This creates a structural conflict: what registers as productive at the individual level may actively undermine what is productive at the team level. Developers are therefore not simply assessed in isolation, as their individual productivity exists in constant tension with the collaborative demands of software work, and how

they navigate that tension is shaped by what their organisation measures and rewards [6, 14]. The misuse of individual metric compounds this tension: when individual-level data is used for performance evaluation, ranking, or compensation decisions, developers are incentivised to game what is measured, to avoid challenging but visible work that reflects poorly in evaluations, or to withhold help from colleagues to protect their individual numbers [7, 14, 16]. This dynamic illustrates precisely why the conflict between individual and team assessment matters and why understanding the degree to which metric gaming and trust in measurement operate in practice is central.

Taken together, these findings frame individual productivity not as a stable, measurable property of a developer, but as a construct that is inseparable from its social and organisational context [8, 9, 16]. What remains underexplored is how developers themselves perceive and navigate this tension in practice, specifically whether they experience a conflict between individual and team productivity and under what conditions they prioritise one over the other.

## 2.4 Team Productivity

While individual productivity contributes to team outcomes, the relationship between the two levels is not straightforwardly additive. Attributes of individuals within groups influence group activities, meaning that group productivity is shaped by both individual characteristics and collective dynamics [10, 14, 18]. Yet, managers and developers may weigh these contributions differently, a tension that underlies many of the conflicts documented in the literature.

Team productivity depends heavily on communication and coordination. Teams work only as fast as decisions are made, and those decisions are normally made collaboratively, which may require sharing information with colleagues [14].

Team cohesion is another factor frequently associated with team productivity, being commonly defined as the degree to which team members are attracted to one another and wish to remain part of the team. Cohesion, however, does not automatically result in higher productivity. Its relationship with productivity appears to depend on the norms and goals established within the team, such that cohesion is positively associated with productivity when team norms encourage high levels of performance [13].

Beyond team-level dynamics, organisational norms and policies around how projects are managed play a significant role in shaping productivity. Companies often establish norms that influence how efficiently work progresses at both the individual and team levels. However, such norms and policies may also lead to unintended consequences [14]. Misalignment between team and organisational goals may reduce team productivity and lead to the rejection of members who are productive by organisational standards [13].

Team productivity emerges from a complex fusion of individual capabilities, shared cognition, communication quality, and alignment with organisational structures [10, 13]. However, this intricate ecosystem is vulnerable to disruption, particularly when the norms, goals, or measurement systems imposed upon it are misaligned or poorly designed [13, 15]. How managers and developers perceive and prioritise these team-level factors, and whether their perceptions align, is therefore

relevant and underexplored, particularly regarding the contexts in which each role favours team productivity over individual productivity, and vice versa.

## 2.5 Misalignment Problem

The challenges described in the preceding sections converge on a central problem: individual and team productivity not only reflect different phenomena but actively conflict when measured within the same organisational system. Activities essential to team effectiveness and long-term code quality, such as collaboration, mentoring, code review, and knowledge sharing, frequently depress individual productivity metrics despite their positive impact on collective outcomes [6, 14]. On the other hand, optimising for individual efficiency can undermine communication and coordination upon which team performance depends [6].

This tension reveals a fundamental paradox: behaviours that enhance team productivity may appear unproductive when assessed at the individual level [6, 7, 14]. While some studies suggest that team productivity emerges from the aggregation of individual contributions [10, 18], others demonstrate that many of the most valuable activities in collaborative development environments reduce an individual's measurable output [6, 14]. Software development operates simultaneously across multiple levels, each with distinct dynamics and requirements. Individual developers require autonomy and uninterrupted focus, yet teams must also invest in coordination and shared understanding [14].

In addition to the structural conflict, developers' and managers' views on how to define productivity are not aligned. Critically, developers' anticipation of how managers define productivity is not aligned with how managers actually define it, while managers themselves possess more accurate insights into how developers conceptualise productivity [16]. This perceptual gap compounds the structural one: when each role misreads the other's priorities, measurement systems are designed and interpreted on the basis of assumed values rather than actual ones.

When what is measured for an individual conflicts with what benefits the team, the measurement system itself can become a source of friction and counterproductive behaviour. These dynamics raise questions that go beyond the conceptual level: whether practitioners recognise this conflict, whether their trust in productivity metrics is affected by it, and whether metric gaming is a widespread practical reality or a theoretical concern. These are empirical questions that the existing literature, which is predominantly conceptual and framework-oriented, has not resolved.

## 2.6 Implications for Research and Practice

Taken together, the reviewed literature establishes that productivity in software engineering is a contested, context-sensitive construct whose measurement carries significant practical consequences. Yet, despite the breadth of conceptual and framework-oriented work, the field lacks empirical evidence on how these dynamics play out across roles in practice. How managers and

developers perceive and define productivity, how they use and misuse individual and team metrics, and in what contexts each level of assessment is trusted or favoured, remain open questions. Addressing them is necessary to move beyond theoretical concerns about misalignment and toward a grounded understanding of how productivity assessment actually operates, and should be governed, in software engineering organisations.

The preceding results reveal a field struggling with a fundamental tension: the complexity of software engineering work conflicts with the organisational desire for clear, actionable productivity metrics. This discussion synthesises these findings, proposing that effective measurement requires moving beyond the division between individual and team productivity toward an integrated, context-sensitive, and purpose-driven approach.

## Chapter 3

# Methodology

An empirical methodology was employed to investigate how active software managers and developers assess productivity. The study proceeds through three interconnected phases: a literature review to establish existing knowledge and identify gaps, pilot interviews with managers to explore practitioner perspectives, and a structured survey to quantify the prevalence of identified themes across a larger population. The overall design follows an exploratory sequential mixed-methods approach, prioritising practitioners’ perspectives. The replication package for this study can be found at: <https://github.com/matifaro/individual-vs-team-productivity-replication>.

### 3.1 Research Questions

The related work presented in Chapter 2 established that, while the literature convincingly argues for a nuanced, multi-level approach to productivity, practitioner perspectives remain under-represented in existing studies. The persistent disagreement over definitions, widespread concern about metric misuse, and the structural tension between individual and team assessment all point to the need for empirical investigation into how practitioners actually define, assess, and navigate productivity in practice. These insights directly motivate the research questions that guide this study.

This study is guided by four research questions (Table 3.1) that aim to deepen understanding of productivity at both the individual and team levels in software development.

Table 3.1: Research Questions

| RQ  | Research Question                                                                                                      |
|-----|------------------------------------------------------------------------------------------------------------------------|
| RQ1 | How do managers' and developers' perceptions and definitions of productivity diverge in software engineering contexts? |
| RQ2 | How do managers and developers perceive the use and misuse of individual and team productivity metrics?                |
| RQ3 | In what context do developers and managers find individual and team productivity metrics useful and reliable?          |
| RQ4 | In what context do developers and managers favour individual or team productivity?                                     |

## 3.2 Research Philosophy And Approach

This study embraces a constructivist perspective [4], recognising that productivity in software engineering is not a singular objective reality, but a concept shaped by practitioners' experiences, roles, and contexts. This conceptual framework translates into the differing perceptions of productive behaviour held by developers and managers, which may not always align but are valid within their respective contexts.

An exploratory sequential mixed-methods design was adopted. The study proceeded in three phases: a literature review to establish existing knowledge, pilot interviews with managers to explore practitioner perspectives, and a structured survey to quantify the prevalence of identified themes across a larger sample. Each phase informed the next, with the literature review shaping the interview protocol, the interview findings guiding survey development, and the survey testing patterns observed in both prior phases.

The primary research strategy is a cross-sectional survey, which collects data from a sample of software practitioners at a single point in time. This type of survey was chosen due to the descriptive and exploratory nature of the research objectives, which aim to characterise current practitioner perspectives rather than track changes over time.

## 3.3 Research Design Overview

As mentioned in Section 3.2, this study employs an exploratory sequential mixed-methods design, characterised by an initial qualitative phase whose findings directly inform the subsequent quantitative phase. This design was preceded by a systematic literature review that established the existing knowledge base and identified gaps in practitioners' perspectives on software productivity. Figure 3.1 illustrates the research design.

The research was conducted in three sequential phases: a literature review, which informed the design of the pilot interviews with managers, and both of which, together, informed the design of a structured survey. Characteristics, description, purpose, and output of each phase can be found in Table 3.2.

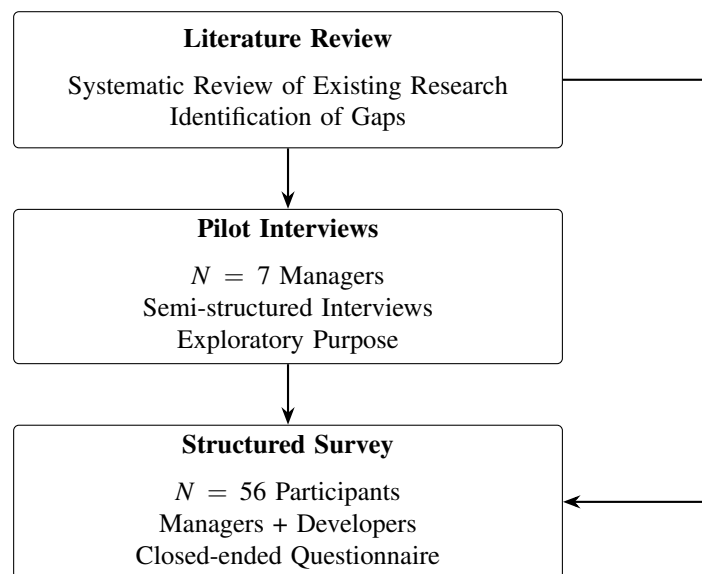


Figure 3.1: Research Design Overview

Table 3.2: Research Phases

|                    | Literature Review                                                                                                        | Pilot Interviews                                                                                                                             | Structured Survey                                                                                                                                                   |
|--------------------|--------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Nature</b>      | Qualitative, Foundational                                                                                                | Qualitative, Exploratory                                                                                                                     | Quantitative, Confirmatory                                                                                                                                          |
| <b>Description</b> | Systematic review of existing research on software productivity, with a focus on individual and team-level perspectives. | Semi-structured interviews with managers.                                                                                                    | Cross-sectional survey administered to both managers and developers.                                                                                                |
| <b>Purpose</b>     | Establish existing knowledge, identify gaps in practitioner perspectives.                                                | Explore managers' perspectives on productivity, specifically their perspective on the relationship between individual and team productivity. | Assess the themes identified in the literature and the interviews across a larger, more diverse population and compare perceptions between managers and developers. |
| <b>Output</b>      | Theoretical foundation and gap identification that informed the interview guide and survey.                              | Thematic findings that informed the development of the survey.                                                                               | Quantitative data on productivity assessment and role-based differences.                                                                                            |

The main purpose of the pilot interviews was to understand how software engineer managers define, assess, and contextualise productivity in their own terms, without imposing predefined categories in the literature. While Chapter 2 identified a lack of shared understanding of productivity, existing research is largely based on developers' perspectives. Managers' perspectives, which are critical for understanding how productivity is evaluated and managed, remain underexplored.

The purpose of the survey was to assess the prevalence and patterns of the themes identified in the pilot interviews across a larger population of software practitioners. While the pilot interviews provided rich, exploratory insights into how managers conceptualise productivity, they involved a small sample (N=7) and did not account for developers' perspectives.

Conducting the literature review first ensured that the interview protocol was grounded in existing knowledge and targeted identified gaps. The pilot interviews were then conducted before finalising the survey to ensure that the survey's design goes beyond literature-based assumptions and includes practitioner perspectives. Conducting the survey before or during the interviews would risk imposing predefined categories on participants without understanding their relevance to them. The chosen sequence prioritises exploration and discovery before assessment, making the sequential order intentional and non-reversible.

Findings from the literature review were used to shape the interview protocol, ensuring that the interviews explored relevant constructs while remaining open to emergent themes. These, together with findings from the pilot interviews, were operationalised into survey items. Recurring themes, terminology, and tensions from the interviews were translated into closed-ended Likert statements, multiple-choice and single-choice questions, and demographic categorisations, ensuring that the survey assessed constructs grounded in practitioner language and experience. These processes are further explained in Section 3.4.3 and Section 3.5.5, respectively.

## 3.4 Literature Review

A systematic literature review was conducted to establish the existing knowledge base on productivity assessment in software engineering, with a particular focus on individual and team-level perspectives. A more extended synthesis is presented in Chapter 2. The review was conducted to answer three guiding questions: (i) what challenges exist in defining and conceptualising productivity in software engineering, (ii) how productivity is assessed at individual and team levels, and what purposes those assessments serve, and (iii) what tensions or misalignments arise between individual and team-level assessments.

### 3.4.1 Methodology

The review followed a systematic process to ensure transparency and reproducibility. Papers were included if they addressed productivity in software engineering at the individual, team, or organisational level; examined productivity assessment, definitions, or influencing factors; investigated developer or manager perspectives; or discussed metric misuse and its consequences. Papers were

excluded if their titles or abstracts were clearly irrelevant, if the full text was unavailable, or if they were published before 2000.

Google Scholar served as the primary source, supplemented by supervisor recommendations and backward reference searching from included papers. Searches were structured around three conceptual components: general productivity in software engineering, level of productivity analysis, and metric use and misuse. The final list of search terms used in the review is summarised in Table 3.3. The search continued until 15 consecutive retrievals yielded no new relevant results. From an initial pool of 50 papers, 14 were selected for in-depth analysis after title, abstract, and full-text screening, as detailed in the PRISMA[12] flow diagram, Figure 3.2.

The 14 included sources span 2001–2022 and include empirical studies, frameworks, position papers, theoretical papers, and books. The sources are listed in Appendix A.

Table 3.3: Search Query Keywords Used for the Literature Review

| Category                                | Search String                                                                                       |
|-----------------------------------------|-----------------------------------------------------------------------------------------------------|
| <b>General Productivity</b>             | "productivity" AND "software engineering" AND "developer"                                           |
| <b>Individual Productivity</b>          | "productivity" AND "software engineering" AND ("individual" OR "developer")                         |
| <b>Team Productivity</b>                | "productivity" AND "software engineering" AND ("team" OR "group")                                   |
| <b>Team and Individual Productivity</b> | "productivity" AND "software engineering" AND ("individual" OR "developer") AND ("team" OR "group") |
| <b>Metrics Use and Misuse</b>           | "productivity" AND "software engineering" AND "metrics" AND ("misuse" OR "use") AND "management"    |

### 3.4.2 Key Findings

The synthesis of the 14 included papers identified five themes describing the central challenges in productivity assessment at individual and team levels in software engineering:

- **Definitional Disagreement:** Productivity lacks a universally accepted definition in software engineering. The literature identifies a persistent divergence between how managers and developers conceptualise it: managers tend to favour output-oriented framings centred on delivery and business value, while developers more often associate productivity with activities and task completion. Human factors, including well-being, job satisfaction, and psychological conditions, are increasingly recognised as constitutive of productive work, rather than merely side effects. This ambiguity creates criterion problems in measurement: without a shared conceptual boundary, metrics risk either missing core aspects of the construct or capturing irrelevant ones.

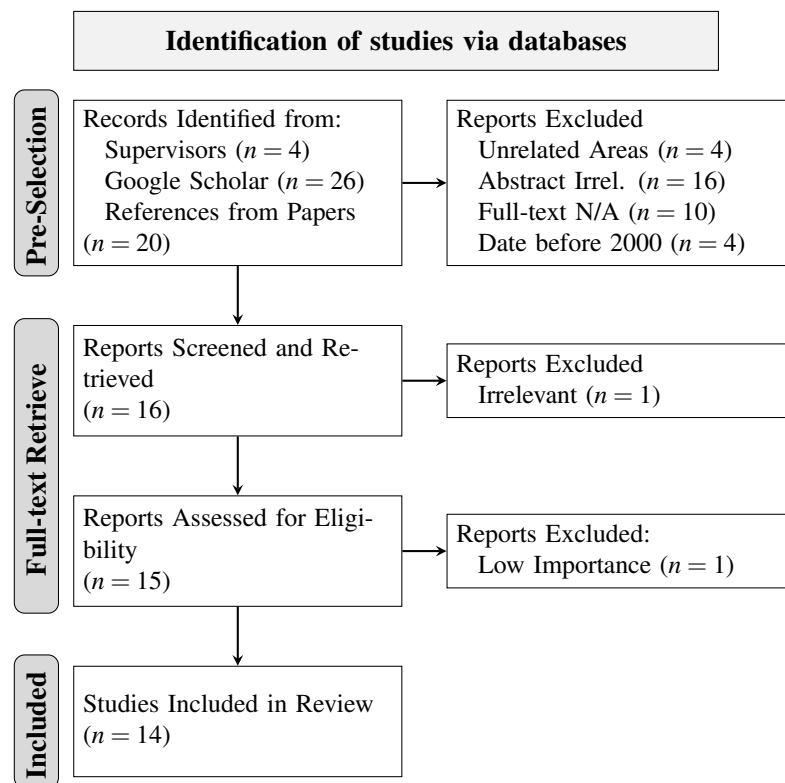


Figure 3.2: PRISMA Flow Diagram: the selection process.

- Divergent Purposes of Productivity Assessment:** The literature documents multiple managerial motivations for productivity assessment, including increasing team and individual effectiveness, resource allocation, and organisational profit. Developers, by contrast, draw on productivity data primarily for personal retrospection and status communication. These purposes are not always aligned, and the literature emphasises that decoupling assessment from its intended purpose is a primary driver of harmful consequences.
- Metric Misuse and Behavioural Consequences:** The literature documents widespread concern about metric misuse, particularly when individual-level metrics are used for performance evaluation, ranking, or compensation decisions. Being assessed prompts developers to adapt their behaviour toward what is measured rather than toward genuine performance, manifesting as gaming, avoidance of challenging work, or withholding help from colleagues. This phenomenon is well captured by Goodhart's Law, which states that when a measure becomes a target, it ceases to be a good measure.
- Individual vs Team Productivity:** Individual and team productivity reflect fundamentally different phenomena. Individual productivity is primarily shaped by psychological and social factors, while team productivity depends on coordination, communication, and shared norms. This creates a structural conflict: activities that are essential to team effectiveness, such as mentoring, code review, and knowledge sharing, frequently depress individual metrics, while optimising for individual efficiency can undermine the coordination on which

team performance depends. The literature establishes this tension conceptually but provides limited empirical insight into how practitioners experience and navigate it.

- **Context Dependency and the Limits of Quantification:** Productivity is context-dependent and cannot be adequately captured by a single metric. Organisational norms, project phase, team maturity, and product domain all shape what productivity means in practice. The literature rejects reductionist approaches and advocates for context-sensitive, multi-dimensional assessment that accounts for the intangible and collaborative nature of software work. Despite this, no integrative model has emerged, and context-sensitive measurement remains aspirational rather than widely practised.

### 3.4.3 Informing the Pilot Interviews

Each of the five themes identified in the literature review was translated into specific interview questions, ensuring coverage of the key constructs while remaining open to emergent perspectives.

The *Definitional Disagreement* finding motivated the inclusion of questions exploring how managers personally conceptualise productivity. For example, the questions *What is productivity for you?* and *Is there a difference for you between individual and team productivity?*. These questions were designed to capture whether managers distinguish between levels and what terminology they use.

The finding regarding *Divergent Purposes of Productivity Assessment* guided the inclusion of questions on assessment practices, such as *Do you assess productivity? Why? For what purposes?*. These questions aim to establish how managers frame the purpose of assessment, whether they distinguish between different intended uses, and how assessment data flows through the organisation.

The *Metric Misuse and Behavioural Consequences* finding informed questions on developer awareness and behaviour. For example, the questions *Does your team know they are being assessed and how? Do you believe that influences their behaviour?*. These questions were designed to capture whether managers have observed gaming behaviour, and whether they perceive a disconnect between their intentions and developer responses.

The finding regarding *Individual vs Team Productivity* prompted extensive questioning about the relationship between the two, including *Is there a difference for you between individual and team productivity?* and *Do you believe there is a conflict between them?*. These questions aimed to capture how managers navigate the tension between levels in practice, which metrics they use for each level, and whether they perceive conflicts between optimising for individual vs team outcomes.

Finally, the *Context Dependency and the Limits of Quantification* finding guided questions on assessment effectiveness and limitations. For example, questions such as *Are there any shortcomings or limitations in how you currently assess productivity?* and *Are there any practices you consider 'good practices' when assessing productivity?*. These questions were designed to

capture how managers contextualise metrics in practice, what strategies they have developed to address limitations, and what they consider responsible or effective assessment.

### 3.5 Pilot Interviews

After the literature review, we conducted semi-structured pilot interviews with software engineering managers. This section describes participant recruitment, the interview protocol, data analysis procedures, key findings and how the findings informed the subsequent survey.

#### 3.5.1 Participant Recruitment

A convenience-based purposive sampling strategy was used to recruit participants who met the inclusion criteria and were accessible to the researcher. Purposive sampling ensures that participants possess the relevant characteristics for the study, while convenience sampling acknowledges the practical constraints of a master's thesis in accessing specialised populations [5]. Given the exploratory nature of this phase and the practical constraints of a master's thesis, access to managers was facilitated through the researcher's and the supervisors' personal and professional networks. The inclusion criteria for participation were: (i) being in an active manager role, such as project manager, engineering manager, team lead, or equivalent; and (ii) working in software development or a closely related context.

The potential participants were invited via email or direct message, with a brief explanation of the study's purpose, time commitment and confidentiality provisions. Recruitment continued until the researcher assessed that the collected data provided sufficient coverage of the research themes within the scope of this exploratory pilot study. Saturation was monitored throughout the data collection process: after each interview, the transcript and notes were reviewed and compared with the themes identified in previous interviews.

In total, 7 managers were recruited and took part in the pilot interviews. Demographic information of participants can be found in Table 3.4. Participants had an average of 6.2 years of management experience, ranging from 1.5 to 15 years. The number of developers managed ranged from 5 to 32, and team locations were remote or hybrid, with 3 and 4 managers, respectively.

#### 3.5.2 Interview Protocol

At the beginning of the interviews, participants were informed that the interviews were semi-structured and encouraged to deviate from the questions if relevant information came to mind. This format allowed participants to introduce concepts not anticipated in the guide, while ensuring consistent coverage of key topics across interviews.

The interview guide can be found in Appendix B and was organised around the following topic areas:

- **Definition of Productivity** - How managers personally define productivity.

Table 3.4: Demographic Information of Pilot Interview Participants

|    | Role                               | Years of Experience | N° Teams | N° Developers | Team Location |
|----|------------------------------------|---------------------|----------|---------------|---------------|
| M1 | Director of Engineering            | 5                   | 4        | 18            | Remote        |
| M2 | Delivery Lead                      | 3                   | 1        | 5             | Hybrid        |
| M3 | Director of Security               | 5                   | 4        | 14            | Remote        |
| M4 | Head of Engineering and Innovation | 1.5                 | 1        | 5             | Hybrid        |
| M5 | Principal Project Manager          | 15                  | 0–30     | 30            | Hybrid        |
| M6 | Engineering Manager                | 4                   | 1        | 6             | Hybrid        |
| M7 | Group Engineering Manager          | 10                  | 5        | 32            | Remote        |

- **Productivity Assessment** - Whether productivity is assessed, why it is assessed, for what purposes, and who uses the resulting data.
- **Individual vs Team Productivity** - Whether managers distinguish between individual and team productivity, how each is assessed, what influences each, and whether conflicts arise between individual and team productivity.
- **Developer Awareness and Behavioural Influence** - Whether developers know they are being assessed, how their awareness influences their behaviour, and whether managers have received feedback from developers about productivity assessment.
- **Good Practices in Productivity Assessment** - What practices managers consider good practices when assessing productivity, and which limitations they have in their current approach.
- **Developer Experience** - Whether developer experience plays a role in how managers think about productivity, how it is incorporated into assessment approaches, and the perceived connection between developer experience and productivity.

The interview guide was designed to address the research questions outlined in Table 3.1. Table 3.5 provides a mapping from each topic area to the corresponding research questions.

Table 3.5: Mapping of Interview Topic Areas to Research Questions

| Topic Area                                    | Research Questions |
|-----------------------------------------------|--------------------|
| Definition of Productivity                    | RQ1, RQ3           |
| Productivity Assessment                       | RQ1                |
| Individual vs. Team Productivity              | RQ2, RQ3, RQ4      |
| Developer Awareness and Behavioural Influence | RQ2                |
| Good Practices in Productivity Assessment     | RQ2, RQ3           |
| Developer Experience                          | RQ1, RQ3           |

Interviews were conducted via Google Meet between 13 February 2026 and 27 March 2026. Each interview lasted approximately 45 minutes. With participant consent, all interviews were audio-recorded on the researcher's phone and transcribed using Jamie AI Note Taker. Transcripts were manually reviewed against the original audio whenever doubts arose. Manuscript notes were taken during each interview to help guide questions.

### 3.5.3 Analysis of Pilot Data

The analysis of the pilot interview transcripts followed a six-phase approach to thematic analysis [2], which provides a systematic and flexible framework for identifying, analysing, and reporting patterns within qualitative data:

1. **Familiarising with the data:** Each interview transcript was read in full to achieve familiarisation with the data. During this reading, individual beliefs and statements from each manager were extracted and written onto physical Post-it notes. To maintain traceability and ensure each manager's contributions could be identified throughout the analysis, each participant was assigned a distinct colour or size of Post-it note. A single statement was written on each note, allowing them to serve as an individual unit of analysis.
2. **Generating Initial Codes:** Following the extraction process, the Post-it notes were systematically reviewed and coded. Each note represented a distinct idea or belief expressed by a participant. Initial codes were assigned to each note based on its content, capturing the core meaning of the statement. This process was applied to all notes, ensuring that relevant data features were systematically identified.
3. **Searching for Themes:** A card sorting technique was conducted to group related codes into broader candidate themes. Post-it notes were manually grouped and aggregated based on content similarity, with notes expressing the same belief or related beliefs clustered together. This aggregation process was iterative, with notes being moved and regrouped as patterns began to emerge. Related codes were combined to form initial candidate themes that captured recurring patterns across the dataset.
4. **Reviewing Themes:** The candidate themes were reviewed against the coded extracts, with themes that were too broad being split, and themes that overlapped being combined. This iterative refinement process ensured that the final themes accurately represented the data.
5. **Defining and Naming Themes:** Once the thematic structure was finalised, each theme was clearly defined and named. This process resulted in a set of nine final themes: individual productivity, team productivity, individual vs team productivity, definition of productivity, assessment challenges and limitations, context and variability, assessment criteria and process, drivers of productivity, and purpose of assessment.
6. **Producing the Report:** To ensure systematic documentation, all aggregated statements were transcribed into a Google Sheet. For each statement, the sheet recorded the thematic

category and which managers had expressed agreement with or had articulated a similar belief, as identified through the colour-coded Post-it system.

### 3.5.4 Key Findings

The pilot interviews revealed a rich set of practitioner perspectives on productivity assessment. The thematic analysis of interview transcripts resulted in a total of nine themes:

- **Definition of Productivity:** Managers conceptualised productivity in several ways. Several managers defined productivity as tied to delivering value, while others linked it to achieving objectives. One manager described productivity as the ability to become more efficient, while another framed it as the relationship between cost and what is delivered. Importantly, one manager noted that the concept of productivity is abstract, a point also noted in the literature. These varied definitions underscore the lack of consensus on what productivity means in practice.
- **Individual Productivity:** Managers described individual productivity as being closely tied to commitment to deliver, autonomy, and the ability to solve critical problems quickly. Specific indicators mentioned included identifying and solving complex architectural problems, reducing technical debt, and helping others.
- **Team Productivity:** Managers consistently prioritised team productivity over individual productivity, with several stating that team productivity is a priority. One manager emphasised that changes within the team normally hinder team productivity, and several highlighted that collaboration is important for a productive team. However, one manager offered a different perspective, describing team productivity as the mean of the productivity of the individuals who compose the team.
- **Individual vs Team Productivity:** A recurring theme was the distinction and tension between individual and team productivity. Several managers stated that individual and team productivity are distinct, with one noting that individual productivity is always looked at keeping team impact in mind. A manager suggested that individual and team productivity are not in conflict if a balance between them is found, while others suggested that individual and team productivity will not be in conflict if individual objectives are aligned with team objectives. One manager further noted that interruptions can decrease productivity at the individual level but are necessary to increase team productivity, illustrating a concrete tension between levels that had already been identified by the literature. One manager highlighted an organisational tension, stating that while they think team productivity is more important, the company prioritises individual productivity.
- **Purposes of Assessment:** The dominant perspective framed assessment as a way to increase the effectiveness of individual developers and teams. One manager stated that productivity is assessed to increase an organisation's profit, while another noted that team and individual productivity are used for the same purpose.

- **Assessment Criteria and Process:** Managers described several aspects of how they assess productivity. One manager stated that productivity is assessed based on the results achieved, the way they are achieved, and how the person develops their work, indicating that process and quality matter alongside outcomes. Managers noted that productivity assessment criteria should be coherent.
- **Context and Variability:** Managers emphasised that productivity assessment is highly context-dependent. One manager stated that productivity and its assessment depend on the individual, the team, or the organisation. Managers stated that there is a big danger in assessing productivity quantitatively without context.
- **Assessment Challenges and Limitations:** Managers acknowledge several difficulties in assessing productivity. Multiple managers stated that productivity is hard to assess, with one adding that it is hard to understand how to assess individual productivity. Managers also stated that assessment, even when quantitative, must always include a qualitative component. One manager noted that a quantitative analysis cannot be linear and can lead to gaming, especially at the individual level.
- **Drivers of Productivity:** Several factors were identified as drivers of productivity. Managers highlighted the importance of autonomy and defining priorities. Several managers noted that people want to feel they are creating something of value to the company, suggesting that purpose and meaning drive productivity. Another manager emphasised that developer experience plays an important role in productivity and that it helps understand what developers need to be productive.

### 3.5.5 Informing the Survey

The findings from the pilot interviews, together with the insights from the literature review found in Section 3.4.2, directly shaped the development of the survey instrument. Recurring themes, specific statements, and findings were translated into specific survey categories, constructs, and items.

The *Definition of Productivity* finding, from both the literature and the interviews, motivated the inclusion of a dedicated survey section on productivity definitions. The definitions of productivity provided by managers, along with those from the literature review, were operationalised as multiple-choice response options, allowing survey respondents to select the definitions that best matched their own perspectives.

The findings regarding *Individual Productivity*, *Team Productivity*, and *Individual vs Team Productivity* guided the development of a survey section regarding the relationship between the two levels of productivity, where beliefs were transformed with Likert statements, as well as a section with scenario trade-offs between individual and team productivity.

The *Purposes of Assessment* finding informed the survey section on productivity assessment practices, with the statements operationalised as Likert-scale items.

The findings regarding *Assessment Criteria and Process*, *Context and Variability*, and *Assessment Challenges and Limitations* guided survey sections on management practices, productivity assessment, and metrics. These insights were translated into Likert-scale items measuring beliefs about contextualisation, the role of quantitative assessment, and positions regarding awareness and misuse.

The *Drivers of Productivity* finding informed survey questions on managerial behaviour and developer experience, with the statements operationalised as Likert items.

Finally, both the literature review and the pilot interviews informed the creation of nine metric categories used in the survey to assess metric use rate and perceived usefulness. These categories were derived from the recurring themes and terminology identified across both sources. Each category was defined in the survey with concrete examples to ensure clarity for respondents. Table 3.6 presents the full list of metric categories, their descriptions as presented on the survey, and whether they are considered individual-level or team-level productivity metrics.

The pilot interviews and literature review not only shaped the content of individual survey items but also provided analytical directions that determined which relationships the survey was designed to examine. Constructs were deliberately included in the instrument to allow these directions to be tested, with specific items operationalising the comparisons motivated by the literature and interviews. Specifically addressing RQ2, the selection of constructs and the analytical comparisons were shaped by three specific propositions emerging from the literature and the pilot interviews. The literature established that metric misuse is most closely associated with individual-level assessment [7, 14], and that developers adapt their behaviour when they know they are being measured, a dynamic captured by Goodhart's Law [7]. The pilot interviews reinforced this, with managers acknowledging that quantitative assessment can lead to gaming, especially at the individual level, and that awareness of being assessed likely influences developer behaviour. Together, these sources suggested three analytical directions. First, that awareness of *why* productivity is assessed would predict criteria optimisation behaviour more strongly than awareness of *how* it is assessed. Second, that individual-level assessment awareness would be more strongly associated with gaming-adjacent behaviour than team-level assessment awareness, given that the literature identifies individual metrics as the primary site of misuse. Third, that perceived misuse, fairness perception, and gaming belief would all be interconnected rather than independent: if a developer perceived that metrics are being misused, this was expected to reduce their sense of assessment fairness and simultaneously increase their belief that measurement incentivises gaming. These propositions were not formally pre-registered hypotheses but served as analytical directions that determined which relationships were examined and how results were interpreted.

### 3.6 Survey

After the pilot interviews, we conducted a closed-ended survey aimed at both software engineering managers and developers. This section describes the survey development, distribution and data collection, and analysis of survey data.

Table 3.6: Metric Categories, Descriptions, and Level of Analysis

| Category                         |  | Description                                                                                                                                                                                        | Individual | Team |
|----------------------------------|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------|------|
| Satisfaction and Wellbeing       |  | Metrics that assess how developers feel about their work, their motivation, and their work environment (e.g., job enthusiasm, perceived productivity).                                             | ✓          |      |
| Communication and Collaboration  |  | Metrics that capture how effectively team members help each other, share information, and coordinate work (e.g., knowledge sharing sessions, team responsiveness, number of times helping others). |            | ✓    |
| Output & Efficiency              |  | Metrics that compare what is produced (output) against the effort or time spent (input) (e.g., features delivered per sprint, story points completed per time unit, lines of code).                | ✓          | ✓    |
| Quality                          |  | Metrics that assess how well software works, how much rework is needed, and whether it meets standards (e.g., rework effort, bug recurrence rate, test pass rate).                                 | ✓          | ✓    |
| Value & Outcomes                 |  | Metrics that assess whether delivered software meets stakeholder expectations and creates business value (e.g., customer satisfaction, business objectives met, feature adoption rate).            | ✓          | ✓    |
| Learning & Growth                |  | Metrics that capture how developers improve their skills, learn new technologies, and evolve in their roles (e.g., mentoring hours, learning goals achieved, role progression).                    | ✓          |      |
| Technical Debt & Maintainability |  | Metrics that assess code quality and the cost of future changes (e.g., technical debt ratio, code complexity, test coverage, number of code smells).                                               | ✓          | ✓    |
| Developer Experience             |  | Metrics that assess how well tools, documentation, and processes support developers in their daily work (e.g., tool satisfaction, documentation quality, onboarding time).                         | ✓          |      |
| Process & Predictability         |  | Metrics that assess how well teams plan, estimate, and deliver according to commitments (e.g., estimation accuracy, on-time delivery rate, planned vs. actual, predictability index).              |            | ✓    |

### 3.6.1 Survey Development

The survey was developed iteratively, informed by the thematic analysis of the pilot interview data and by the literature review, as explained in Section 3.5.5. The survey was developed on INESC TEC's survey platform, which uses LimeSurvey, and can be found in Appendix C.

#### 3.6.1.1 Questionnaire Design

Question types were selected based on the nature of the information being collected, using the most appropriate format for each item. The survey included a mix of Likert-scale and multiple-choice items.

The first question, which was about the respondent's role description (Question 1), distinguished between managers and developers, ensuring each role was presented with questions that used the right wording. Skip logic was implemented to ensure that respondents only saw questions relevant to their role. Some questions and statements were shared, but those who weren't were equivalent. For example, managers were presented with "I assess productivity of individual developers", while developers were presented with "My individual productivity is being assessed in my team".

The language used in the survey was aligned with practitioners' terminology identified in the pilot interviews.

Response options were exhaustive, including 'Does Not Apply' for Likert statements and 'None of the Above' for multiple-choice.

The survey was kept as concise as possible while still addressing all research objectives. An estimated completion time of 10 minutes was communicated to potential respondents.

#### 3.6.1.2 Survey Structure

The survey consisted of 26 questions, some with variants for managers and developers, that were distributed across 7 categories. The survey began with background questions to characterise the sample, followed by sections on productivity definitions, management practices, productivity assessment, individual vs team productivity, and metrics, and concluding with 14 trade-off scenarios comparing individual and team productivity. Table 3.7 presents the full structure.

### 3.6.2 Survey Distribution and Data Collection

The survey was open for data collection from 10 May 2026 to 20 June 2026 and distributed through multiple channels to maximise reach and the diversity of respondents.

The survey was distributed across large online communities where software practitioners congregate. The survey was posted in six software development subreddits, four Discord servers dedicated to software engineering and developer communities, and one additional developer community, Dev.to.

Table 3.7: Survey Structure - Categories and Question Distribution

| Category                         | Description                                                                         | # Questions | RQs           |
|----------------------------------|-------------------------------------------------------------------------------------|-------------|---------------|
| Background                       | Role description, years of experience, team size, work setup, and organisation size | 5           | -             |
| Productivity                     | Productivity definition, satisfaction with self and team productivity               | 2           | RQ1           |
| Management                       | Managerial behaviour toward productivity                                            | 1           | RQ1, RQ3      |
| Productivity Assessment          | Productivity assessment practices                                                   | 1           | RQ1, RQ2, RQ3 |
| Individual vs. Team Productivity | Relationship between individual and team productivity                               | 1           | RQ4           |
| Metrics                          | Metrics currently used and perceived as useful.                                     | 2           | RQ2, RQ3      |
| Individual vs. Team Trade-offs   | Scenarios comparing individual and team productivity.                               | 14          | RQ4           |

Additionally, the survey was shared through personal and professional networks by the researcher and the supervisors. A post containing a brief description of the study, eligibility criteria, and a link to the survey was published on the researcher's LinkedIn profile and subsequently reposted by the researcher's supervisors to reach a broader professional network.

Participants from the pilot interviews were also contacted again to complete the survey and share it with their teams. At the conclusion of the pilot interviews, managers were informed that the interviews would inform a survey, and permission to contact them again for distribution purposes was obtained.

To supplement recruitment, the survey was shared on social media platforms such as Facebook and Instagram, along with a brief explanation of the study and a link to the survey.

### 3.6.3 Analysis of Survey Data

Survey data were analysed using Python scripts. Completed survey responses were exported from LimeSurvey as a CSV file and analysed using Python scripts executed in Visual Studio Code, utilising the `pandas`, `scipy`, `numpy`, and `matplotlib` libraries. For Likert-scale items, responses were coded numerically, with 1 corresponding to Strongly Disagree and 5 to Strongly Agree, to enable quantitative analysis. 'Does Not Apply' responses were excluded from all quantitative analyses and treated as missing data, with the number of respondents selecting this option reported per item where relevant.

Given the limited statistical power associated with 20 managers and 36 developers, effect sizes are reported alongside p-values throughout and are treated as the primary measure of practical significance.

The analysis strategy was designed to accommodate not only the ordinal nature of Likert-scale data but also the small, unequal group sizes and the exploratory objectives of the study.

Descriptive statistics, including frequencies, percentages, means, medians, and standard deviations, were used to summarise responses and demographic characteristics.

For group comparisons, non-parametric tests were employed due to the ordinal nature of Likert-scale data. The Mann-Whitney U test was used to compare the two independent groups of managers and developers, while the Wilcoxon signed-rank test was applied for paired within-subject comparison. For comparisons involving three or more groups, e.g. work setup, the Kruskal-Wallis H test was used, followed by pairwise Mann-Whitney U tests where appropriate.

Pairwise associations between ordinal variables were assessed using Spearman's rank correlation coefficient ( $\rho$ ). Effect sizes are reported as rank-biserial correlation ( $r$ ) for Mann-Whitney U tests and as Spearman's  $\rho$  for correlations.

For binary outcomes, in multiple-choice questions, Fisher's exact test was used to compare proportions between groups.

All statistical tests were two-sided, and statistical significance was set at  $p=0.05$ .

### 3.7 Ethical Considerations

Given that the study involved collecting data from participants through interviews and surveys, ethical principles were observed throughout the research process:

- **Informed Consent:** All participants were provided with clear information about the study's purpose, the nature of their participation, and how their data would be used. For the pilot interviews, a consent form was provided before each interview, and participants were required to confirm their consent before the interview began. For the survey, an introductory page was presented at the beginning of the questionnaire, explaining the study's purpose, estimated completion time, and data-handling procedures.
- **Anonymity and Confidentiality:** To protect participant privacy, all data were anonymised. For pilot interviews, participants were assigned alphanumeric codes, as in Table 3.4, and any identifying information was removed from transcripts. For the survey, no personally identifiable information was collected.
- **Right to Withdraw:** Participants were informed of their right to withdraw from the study at any point. During pilot interviews, participants could stop the interview at any time or request that their data be excluded from analysis. For the survey, because responses were anonymous and no identifying information was collected, participants could withdraw by closing the browser window before submitting.
- **Data Usage and Reporting:** All data collected were used for the purposes of this dissertation. Findings are reported in anonymised form, with no individual participant identifiable.

Any statements from pilot interviews are attributed only by participant code and do not include any contextual information.

- **Data Storage and Security:** All data collected during this study were stored securely. Interview recordings and transcripts were stored on a private Google Drive folder accessible only to the researcher. Survey responses were collected via the LimeSurvey platform and stored on a secure, password-protected account. All data that is not anonymised will be permanently deleted.

## Chapter 4

# Results

This chapter presents the findings of the empirical research. The chapter begins with a characterisation of the survey sample, providing an overview of the participants' demographic and professional backgrounds. The remainder of the chapter is organised by research question, with each section presenting the findings that address one of the four research questions that guide this study, Table 3.1. Findings integrate qualitative insights from the pilot interviews and quantitative patterns from the survey to provide comprehensive responses to each question.

### 4.1 Sample Characterisation

A total of 56 complete survey responses were analysed. The sample consisted of 20 managers (35.7%) and 36 developers (64.3%).

Regarding professional experience, the majority of respondents had 6-10 years, with a range of 1-32 years. Figure 4.1 presents the distribution of participants' professional experience in years. Figure 4.2 shows the distribution of participants' immediate team size, with results ranging from 1-5 to 20+ people.

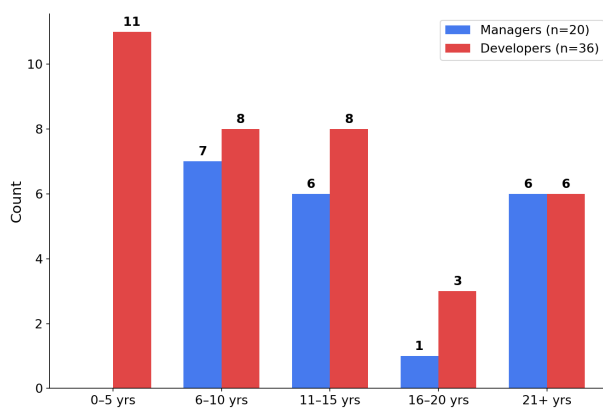


Figure 4.1: Experiences of Participants

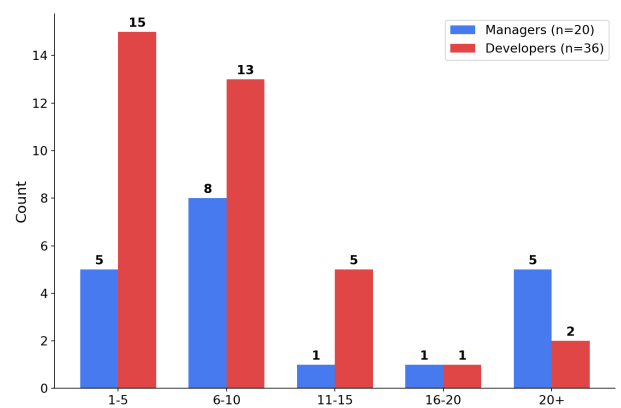


Figure 4.2: Immediate Team Sizes

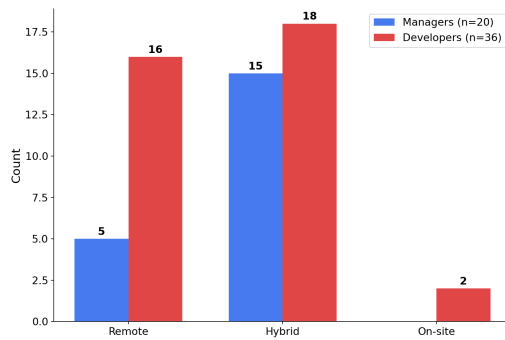


Figure 4.3: Work Setups of Respondents



Figure 4.4: Company Sizes of Participants

Figure 4.3 presents the distribution of participants' work setup. The majority of respondents worked in hybrid arrangements, with 37.5% working remotely, 58.9% working in a hybrid arrangement, and 3.6% working on-site. Regarding company size, Figure 4.4 presents the distribution of participants' company size.

The pilot interviews were conducted with a separate sample of 7 managers, and their demographics are presented in Table 3.4 (page 19).

## 4.2 RQ1 - How do managers' and developers' perceptions and definitions of productivity diverge in software engineering contexts?

To address RQ1, the survey examined how managers and developers define productive behaviour and compared their respective perspectives.

Pilot interview findings provided initial context about how managers conceptualise productivity. M1, M3, and M4 defined productivity as connected to delivering value, while M2, M3, and M5 saw it as achieving objectives. M1 also described productivity as the ability to become more efficient, but added that the concept of productivity is abstract. M3 further elaborated that productivity represents the relationship between cost and what is delivered, meaning the value produced relative to resources invested. These qualitative insights are reflected in the survey findings.

Figure 4.5 shows the percentage of managers and developers selecting each productivity definition. Both groups share *Delivering Business Value* as their most frequently selected definition, with managers at 85% and developers at 77.8%. However, beyond this common ground, substantial divergence emerges. Managers prioritise output-oriented definitions more strongly, with 80% selecting *Reaching Business Objectives*, 65% selecting *Delivering Tasks on Time*, and 65% selecting *Reducing Effort and Increasing Output*. By contrast, developers selected these definitions at considerably lower rates: 55.6%, 52.8%, and 58.3%, respectively. Conversely, developers were more likely than managers to select *Producing Quality Artefacts* (75% vs 55%), *Transforming Ideas into Software* (69.4% vs 60%), and most notably *Helping Others on the Team* (75% vs 40%). This last category represents the only statistically significant difference between the two groups, with  $p=0.020$ .

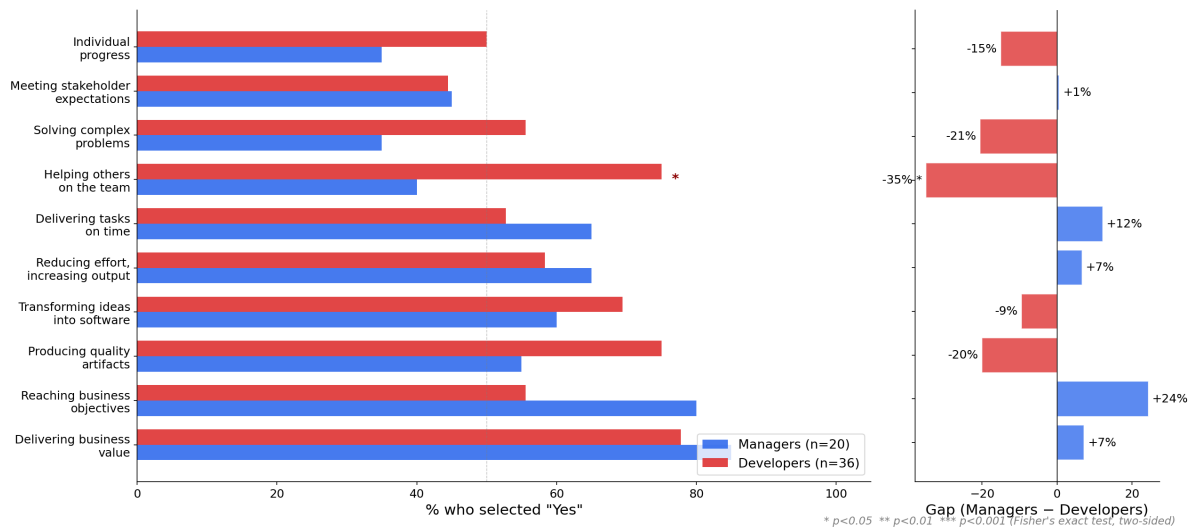


Figure 4.5: Definitions of Productivity - Developers vs Managers

In summary, while both managers and developers converge on delivering business value as a central definition of productivity, managers emphasise output-oriented and objective-driven definitions, whereas developers place significantly greater weight on quality, ideation, and collaborative dimensions. This means that activities developers consider productive, such as helping colleagues, producing quality artefacts, and transforming ideas into working software, may be undervalued or invisible in the assessment frameworks managers apply, which are oriented toward business objectives and delivery efficiency.

### 4.3 RQ2 - How do managers and developers perceive the use and misuse of individual and team productivity metrics?

To address RQ2, the survey examined developer awareness of productivity assessment, developer optimisation toward manager criteria, perceptions of metric misuse and fairness, and whether managers adjust their metric usage in response to gaming risks.

The pilot interviews revealed that managers are acutely aware of the risks associated with quantitative assessment and gaming behaviour. M4 stated that productivity is hard to assess, adding further that it is difficult to understand how to assess individual productivity. M1 and M4 agreed that a quantitative analysis cannot be linear and can lead to gaming, especially at the individual level. M6 stated that they aim to use quantitative assessment as little as possible to prevent developers from gaming the system.

Beginning with the predictors of criteria optimisation behaviour, Figure 4.6 presents a scatter plot that examines whether the belief that metrics incentivise gaming relates to criteria optimisation behaviour. The correlation is near-zero and non-significant ( $r=-0.03$ ,  $p=0.86$ ,  $n=30$ ), and has a flat trend slope of  $-0.07$ . Believing that metrics incentivise gaming, therefore, had no meaningful relationship with actually optimising toward criteria.

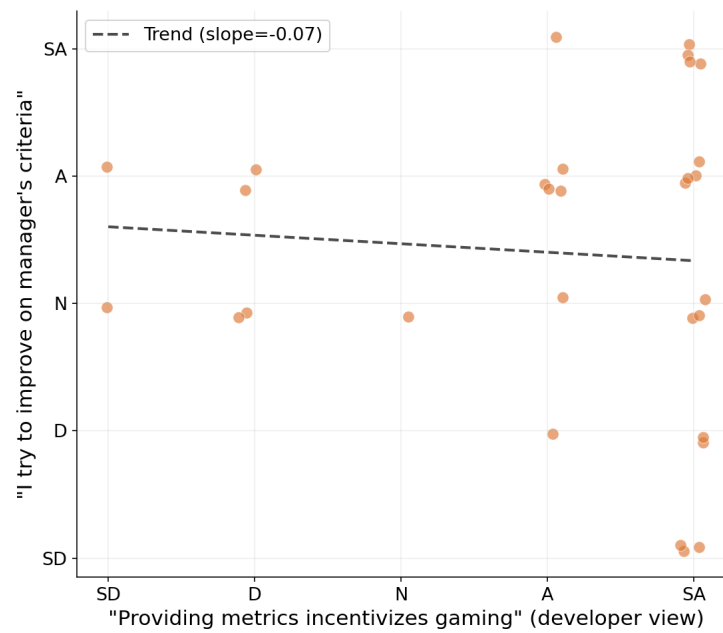


Figure 4.6: Relation between Belief in "Providing metrics incentivises gaming" and Improvement on Manager Criteria

Figure 4.7 presents scatter plots comparing awareness of how the manager assesses productivity (on left panel) and awareness of why productivity is assessed (on right panel) as predictors of criteria optimisation. Both are significant, with the HOW awareness yielding  $r=0.45$ ,  $p=0.013$ ,  $n=29$ , and slope=0.38, and the WHY awareness yielding  $r=0.58$ ,  $p=0.001$ ,  $n=27$ , and slope=0.61. The WHY panel shows a cleaner upward pattern with less unexplained variance, while the HOW panel shows more dispersion, including several developers who are fully aware of how assessment works but still do not adapt their behaviour. The slope difference indicates that a one-unit increase in purpose awareness produces a 60.5% larger increase in criteria optimisation than the equivalent increase in method awareness. These findings confirm a positive, statistically significant relationship between both types of awareness and criteria optimisation. However, the stronger correlation and steeper slope of WHY awareness suggest that understanding the underlying purpose of productivity assessment is a more powerful driver of behavioural adaptation than merely knowing the measurement method.

Regarding individual and team assessment awareness and their influence on optimisation, Figure 4.8 presents two scatter plots. The left panel examines whether awareness that individual productivity is being assessed incentivises improvement on criteria. The correlation is significant ( $r=0.48$ ,  $p=0.010$ ) and has a slope of 0.46. Developers who are aware that their individual productivity is being assessed are considerably more likely to optimise toward manager criteria. The right panel examines whether awareness that team productivity is being assessed predicts criteria optimisation. This correlation is not significant ( $r=0.23$ ,  $p=0.087$ ) and has a slope of 0.37. This suggests that individual assessment awareness has a stronger, statistically significant relationship with gaming-adjacent behaviour, whereas team assessment awareness shows a weaker,

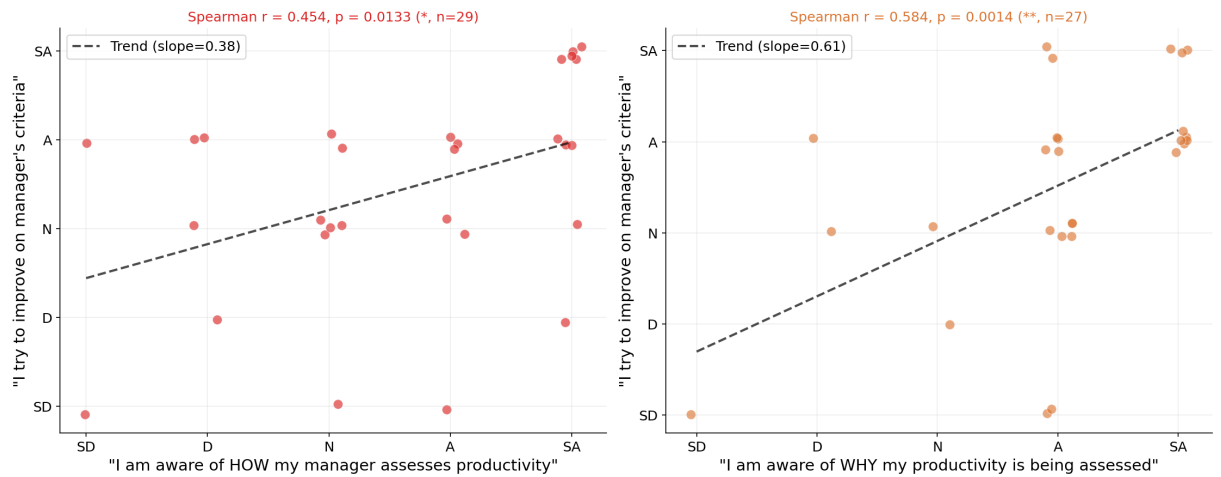


Figure 4.7: Relation between Awareness and Optimisation

non-significant trend in the same direction. This indicates that metrics misuse is more closely associated with individual-level assessment than with team-level assessment.

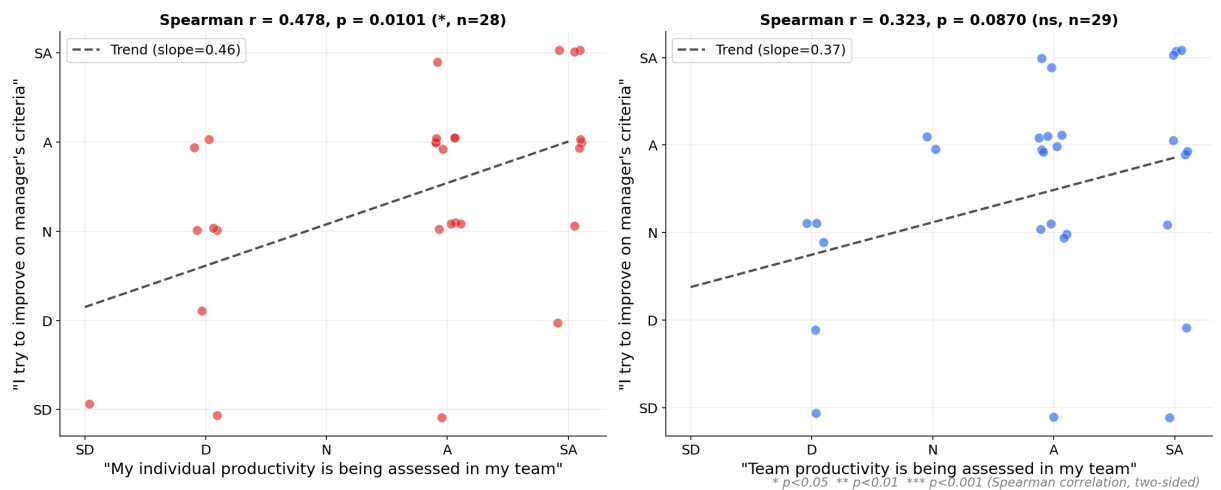


Figure 4.8: Relation between Individual/Team Awareness and Optimisation

Examining cross-perspective patterns, Figure 4.9 presents a summary of who believes metrics incentivise gaming and what they do. The two statistically significant findings are that developer awareness of being assessed at the individual level correlates positively with criteria improvement behaviour ( $r=0.48$ ,  $p=0.010$ ) and that developer awareness of the purposes of that assessment also correlates positively with criteria improvement behaviour ( $r=0.58$ ,  $p=0.001$ ). All other correlations are small and non-significant, with developer perceptions of misuse, developer perceptions of fairness, and developer beliefs that metrics incentivise gaming showing no meaningful relationship with criteria improvement behaviour. On the manager side, gaming belief shows a positive but non-significant association with breadth of metric use ( $r=0.39$ ,  $n=20$ ). The pattern across the chart is therefore consistent: awareness is a driver of criteria optimisation, while attitudes toward the assessment system are not.

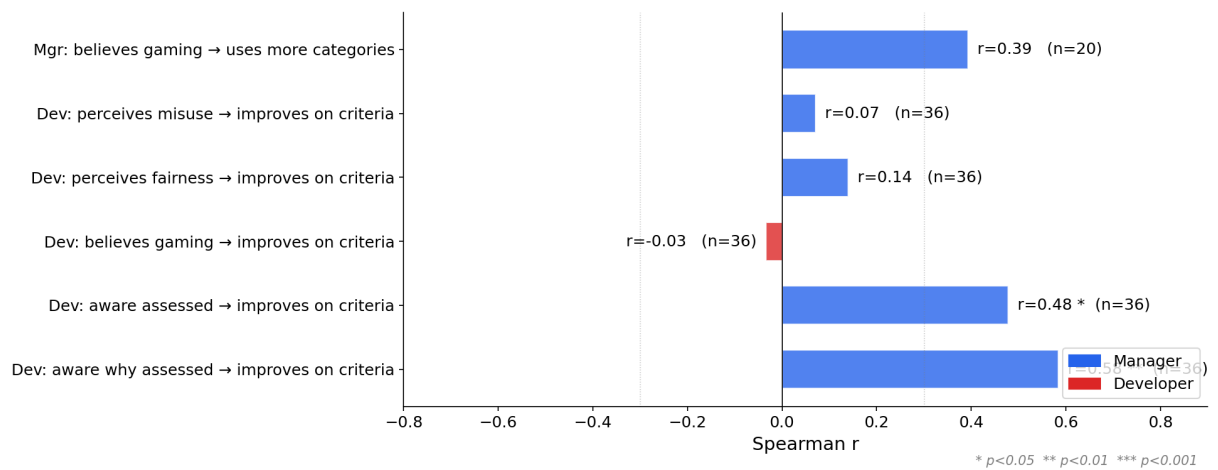


Figure 4.9: Correlation between Awareness, Gaming and Behaviour

Moving to whether managers adjust their behaviour in response to gaming risk, Figure 4.10 examines whether managers who acknowledge that metrics incentivise gaming change their metric use behaviour. The percentage of managers at each belief level regarding whether metrics incentivise gaming behaviour who still use *Output & Efficiency* metrics is 100% for managers who *Disagree* ( $n=1$ ), 100% for managers who are *Neutral* ( $n=3$ ), 100% for managers who *Agree* ( $n=8$ ), and 86% for managers who *Strongly Agree* ( $n=7$ ). Acknowledging gaming risk, therefore, produces no meaningful change in the use of the *Output & Efficiency* metric category, the most gameable metric category, because it is most susceptible to gaming behaviours, e.g., inflating lines of code or rushing task completion.

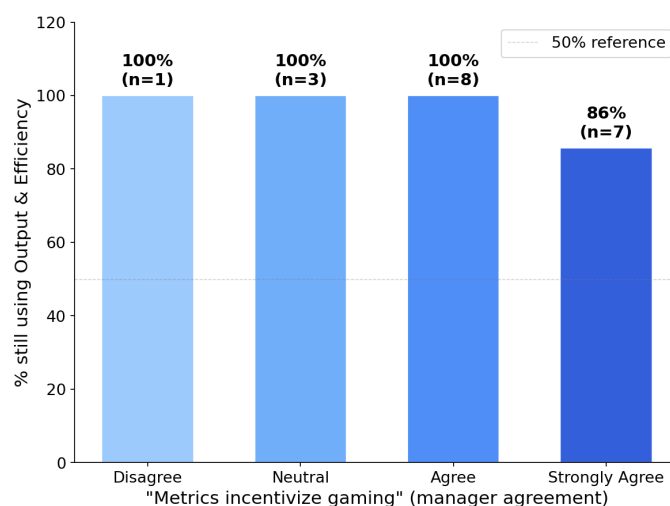


Figure 4.10: Comparison between Belief in "Providing metrics incentivises gaming" and Managers using Output &amp; Efficiency Metrics

Regarding the interconnection among perceived misuse, fairness, and gaming belief among developers, Figure 4.11 presents three scatter plots that examine their relationships. The left panel

shows a significant negative correlation between perceived manager misuse and assessment fairness ( $r=-0.52$ ,  $p=0.004$ ) and a slope of  $-0.52$ , meaning that developers who more strongly perceive misuse rate assessment as considerably less fair. The centre panel shows a significant positive correlation between perceived misuse and gaming belief ( $r=0.43$ ,  $p=0.015$ ) and a slope of  $0.52$ , meaning that developers who perceive misuse more strongly also believe that metrics incentivise gaming. The right panel shows the relationship between fairness perception and gaming belief, which is non-significant ( $r=-0.19$ ,  $p=0.495$ ), and a slope of  $-0.25$ . Fairness perception and gaming belief, therefore, operate independently of each other, while perceived misuse connects to both, as it is associated with lower fairness ratings and higher gaming belief simultaneously, but fairness and gaming belief are not directly linked.

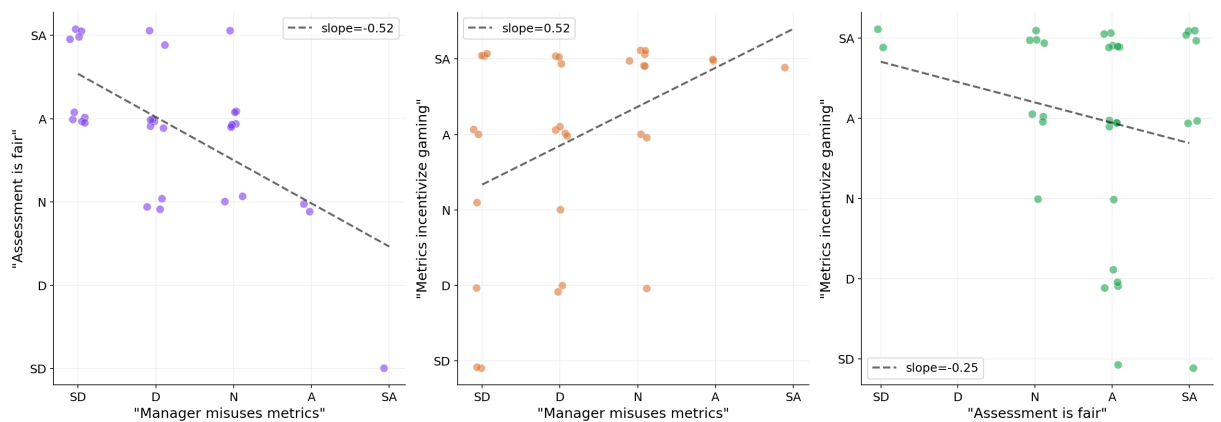


Figure 4.11: Misuse-Fairness-Gaming Correlation Triad - Developers

Finally, Figure 4.12 presents the full Likert distribution for four items across both roles using a diverging bar chart. Among managers, 55% report that their assessment of productivity has been misinterpreted in the past, with 40% *Agree* and 15% *Strongly Agree*, and with 20% *Neutral* and 25% *Disagree*. A further 45% of managers agree that it is difficult to know what constitutes fair productivity assessment, with 30% *Agree* and 15% *Strongly Agree*, and with 25% *Neutral* and 30% disagreeing across two levels. Among developers, only 8% agree that their manager misuses productivity metrics, with 6% *Agree* and 4% *Strongly Agree*, and with 28% *Neutral* and 61% disagreeing, with 32% *Disagree* and 29% *Strongly Disagree*. Assessment fairness receives considerably more positive ratings: 69% of developers agree that their manager's assessment is fair, with 48% *Agree* and 21% *Strongly Agree*, and 24% *Neutral*, with only 6% selecting *Strongly Disagree*. Taken together, these four distributions describe a situation in which the majority of managers have experienced their assessments being misinterpreted and nearly half find it difficult to calibrate fairness, yet most developers do not perceive active misuse, and the majority consider assessment fair. This asymmetry aligns with M4's observation that productivity is hard to assess and that it is difficult to understand how to assess individual productivity. Managers are aware of these challenges, but this awareness does not translate into visible misuse from the developer perspective.

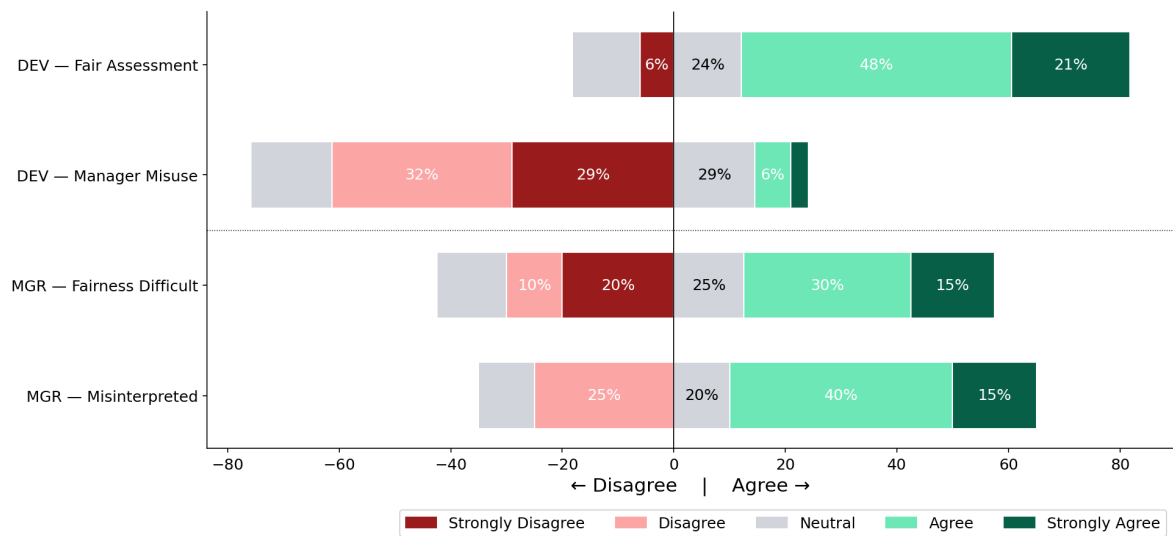


Figure 4.12: Misinterpretation &amp; Fairness - Full Distribution

In summary, these findings reveal several key patterns regarding the frequency of use and misuse of individual and team productivity. Developer awareness, particularly of the assessment's purpose, is the primary driver of criteria optimisation behaviour, whereas beliefs about gaming, fairness, or misuse do not meaningfully predict it. Crucially, individual assessment awareness has a stronger relationship with gaming-adjacent behaviour than team assessment awareness. Purpose awareness produces a substantially larger effect on criteria optimisation than method awareness. Additionally, managers who acknowledge gaming risk do not meaningfully reduce their use of the most gameable metric category, suggesting a disconnect between awareness and practice. Among developers, perceived misuse is associated with both lower fairness ratings and higher gaming beliefs, but fairness and gaming beliefs are not directly linked. Other than that, managers report frequent misinterpretation and difficulty calibrating fairness, yet developers perceive little active misuse and rate assessment as predominantly fair. This suggests that while managers are aware of assessment challenges, they do not manifest as visible misuse from the developer's perspective. Together, these findings indicate that awareness is a driver of behavioural optimisation, while managers' acknowledgement of gaming risk does not translate into reduced use of output-oriented metrics.

#### 4.4 RQ3 - In what context do developers and managers find individual and team productivity metrics useful and reliable?

To address RQ3, the survey examined managers' metric usage rates, perceived usefulness across the metric categories in Table 3.6, role-based differences, work setup influences, and the role of contextualisation in shaping perceptions of reliability.

The pilot interviews provided initial context on what managers consider important when assessing individuals and teams. M4 emphasised that individual productivity is best captured by

metrics such as the ability to solve critical problems quickly, identify and resolve complex architectural problems, reduce technical debt, and help others. M6 further noted that assessment practices depend on the work setup.

Beginning with manager usage patterns, Figure 4.13 presents manager use rates and perceived usefulness scores for all nine metric categories. Among these, the individual-level productivity metric categories, *Satisfaction & Wellbeing* (55%), *Learning & Growth* (40%), and *Developer Experience* (45%), and the team-level productivity metric categories, *Communication & Collaboration* (55%) and *Process & Predictability* (50%), are all used by fewer than 60% of managers, despite receiving moderate usefulness scores, ranging from 3.4 to 4.0, and 4.0 to 4.2, respectively. The four most-used categories, *Output & Efficiency* (90%), *Quality* (80%), *Value & Outcomes* (80%), and *Technical Debt & Maintainability* (75%), are all product-delivery oriented, and they can all be used at both individual and team levels. These categories also receive the highest usefulness scores, ranging from 4.4 to 4.8, out of 5.

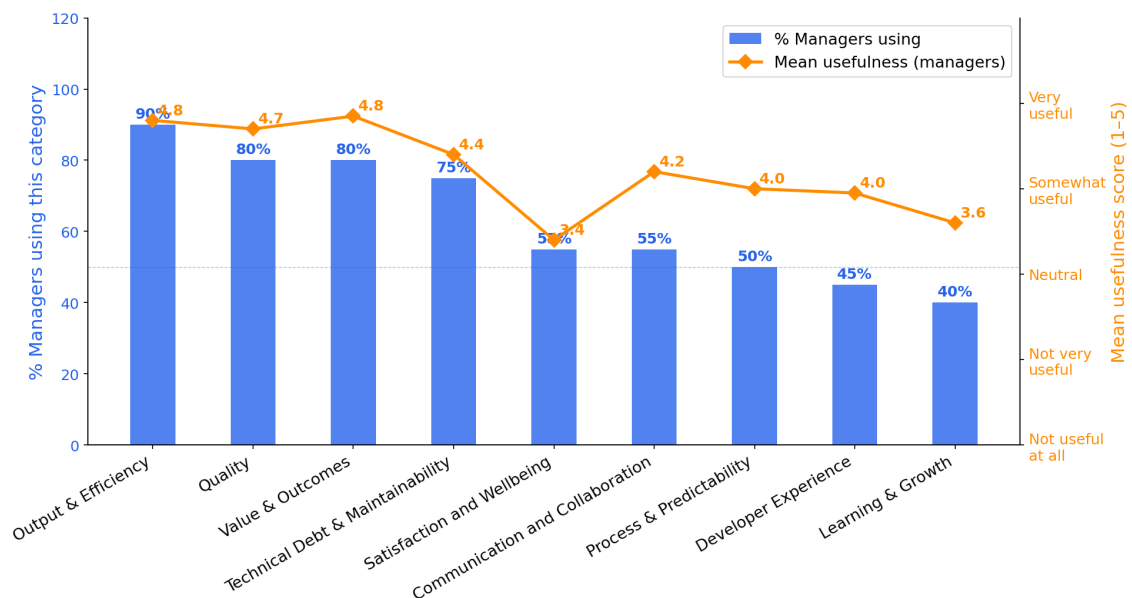


Figure 4.13: Metric Category Use Rate vs Perceived Usefulness - Managers

Comparing managers' and developers' perspectives, Figure 4.14 shows the perceived usefulness of each metric category by role, expressed as the percentage of respondents who rated each category as Somewhat Useful or Very Useful. Four categories reach significance: *Output & Efficiency* ( $p=0.008$ ), *Value & Outcomes* ( $p=0.003$ ), *Quality* ( $p=0.026$ ), and *Technical Debt & Maintainability* ( $p=0.029$ ). Notably, all four significant categories are metrics that can be applied at both the individual and team levels, whereas no metrics that were exclusively individual- or team-level reached significance. In all four significant categories, managers rate usefulness higher than developers, with percentage-point gaps ranging from 0.48 to 0.94. This suggests that the greatest divergence in perceived usefulness between managers and developers occurs for metrics that serve dual purposes, rather than for those specific to a single level of analysis.

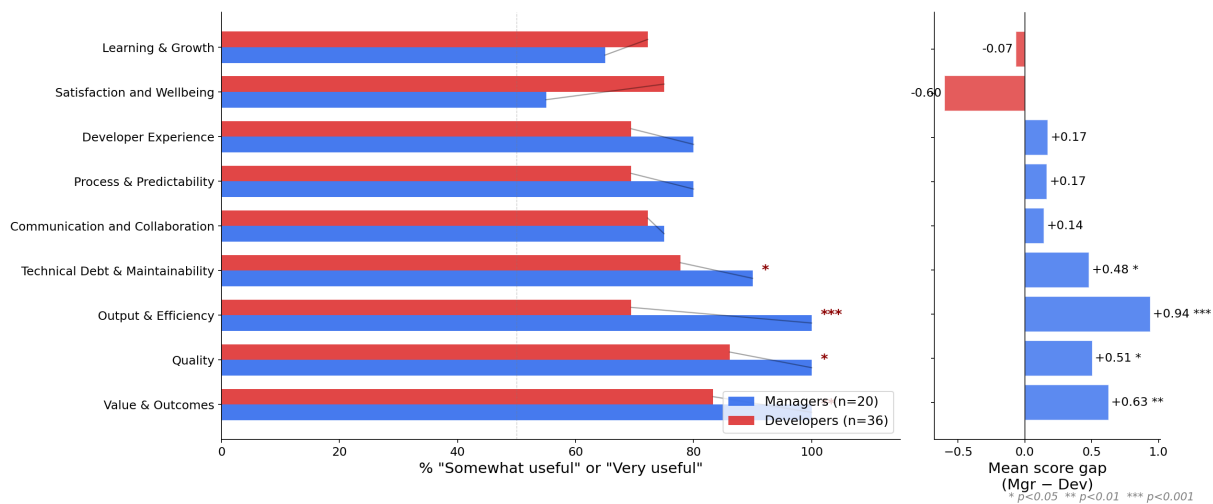


Figure 4.14: Perceived Usefulness of Metric Categories

Moving to the full distribution of responses, Figure 4.15 presents the full Likert-level distribution of usefulness ratings per category for each role. Manager responses are predominantly positive, with *Value & Outcomes* (85%), *Output & Efficiency* (80%), and *Quality* (70%) each receiving clear majorities at *Very Useful*. No managers selected *Not Useful At All* for five of the nine categories, and no manager rated *Output & Efficiency*, *Quality*, or *Value & Outcomes* below *Somewhat Useful*. Developer responses, by contrast, are more distributed and cautious: no category exceeds 53% at *Very Useful*, and every category receives responses across at least four usefulness levels. Developers show consistent concentration at *Somewhat Useful* rather than *Very Useful*, with notable proportions selecting *Not Very Useful* or *Not Useful at All* for categories where managers show no such responses, e.g. 17% of developers rate *Output & Efficiency* as *Neutral*, 8% as *Not Very Useful* and 6% as *Not Useful at All*, compared to 0% of managers at either level. This means that metrics managers consider clearly valuable are experienced by a non-trivial proportion of developers as only moderately useful or even unhelpful. This asymmetry suggests that the same metric can convey meaningfully different perceived validity depending on the role of the person being assessed, and points to a structural gap between the information managers derive from these metrics and the legitimacy developers attribute to them.

Considering the influence of work setup, Figure 4.16 presents mean usefulness scores for *Communication & Collaboration* metrics broken down by work setup and role. All populated cells fall within the range of 4.00 to 4.50, corresponding to the *Somewhat Useful* to *Very Useful* band. Remote managers (n=5, mean=4.40) and hybrid managers (n=15, mean=4.13) rate *Communication & Collaboration* metrics at near-identical levels. Remote developers (n=16, mean=4.00) and hybrid developers (n=18, mean=4.06) are similarly close. On-site developers (n=2, mean=4.50) score highest, though the small group size must be noted. No managers appear in the on-site row, reflecting the composition of the manager sample, which consists entirely of remote and hybrid workers. The perceived value of this team-level metric category is stable across work arrangements for both roles, indicating that work setup does not, in principle, influence respondents' perceived

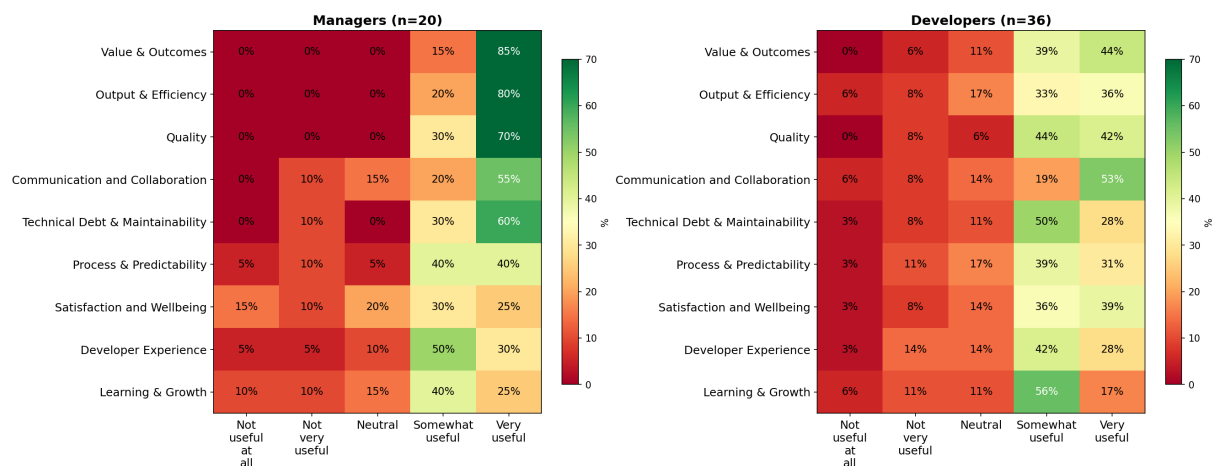


Figure 4.15: Usefulness Rating Distribution per Metric Category - Heatmaps

usefulness of *Communication & Collaboration*.

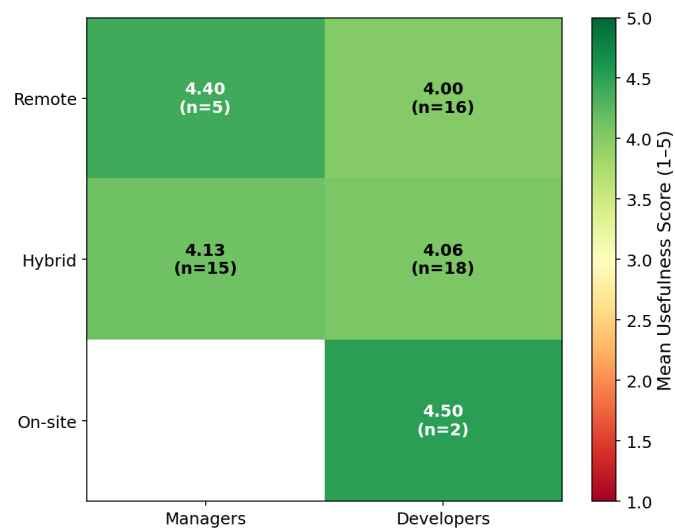


Figure 4.16: Mean Communication & Collaboration Usefulness Score by Work Setup and Role

Turning to contextualisation in productivity assessment, the pilot interviews revealed that managers hold strong and nuanced views. M1, M2 and M5 warned against purely quantitative assessment without context, with M1, M3 and M4 stating that assessment must always have a qualitative side. M7 added that caution is needed when looking at metrics, as they may not reflect the true value, especially at the team level.

Beginning with the overall endorsement of contextualisation, Figure 4.17 presents responses to the statement "Productivity metrics must be contextualized" for both roles. Agreement is near-universal: 95% of managers (30% *Agree* and 65% *Strongly Agree*) and 100% of developers (49% *Agree* and 51% *Strongly Agree*) endorse the statement, with no respondent in either group selecting *Disagree* or *Strongly Disagree*. The boxplot shows that both distributions are compressed toward the upper end of the scale, with manager responses spanning *Neutral* to *Strongly Agree* and

developer responses concentrated entirely at *Agree* and *Strongly Agree*, Mann-Whitney U test not significant.

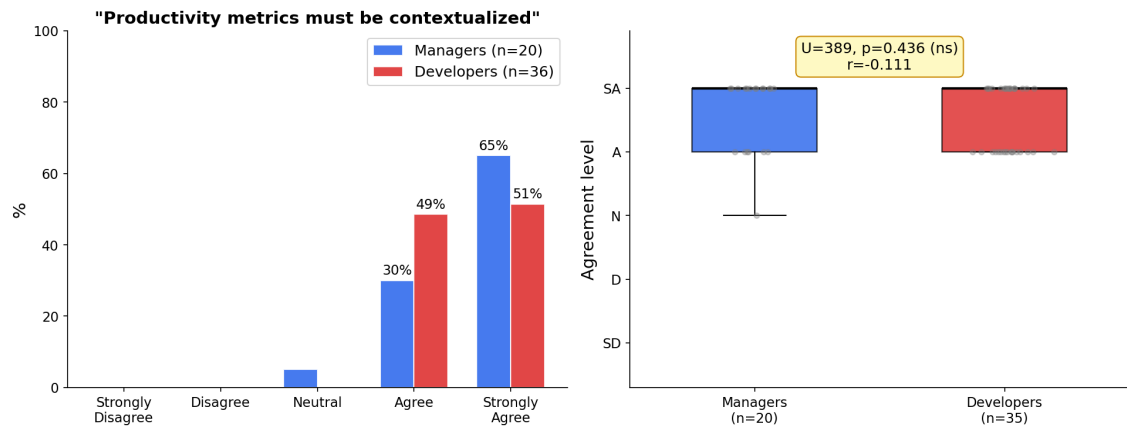


Figure 4.17: Contextualisation Consensus between Roles

Regarding the contrast between contextual and quantitative assessment beliefs, Figure 4.18 presents a diverging bar chart comparing beliefs for the statements "Productivity metrics must be contextualised" (contextual tag) and "Productivity assessment should be quantitative" (quantitative tag). For managers, contextual assessment is endorsed by 95%, with 30% *Agree* and 65% *Strongly Agree*, while quantitative assessment is endorsed by 30% and with 35% disagree, with 35% choosing *Neutral*. For developers, the contextual belief is unanimous, with 49% *Agree* and 51% *Strongly Agree*, while quantitative assessment receives weaker and more dispersed endorsement, with 37% agreement across two levels, 26% *Neutral*, and 37% disagreement across two levels.

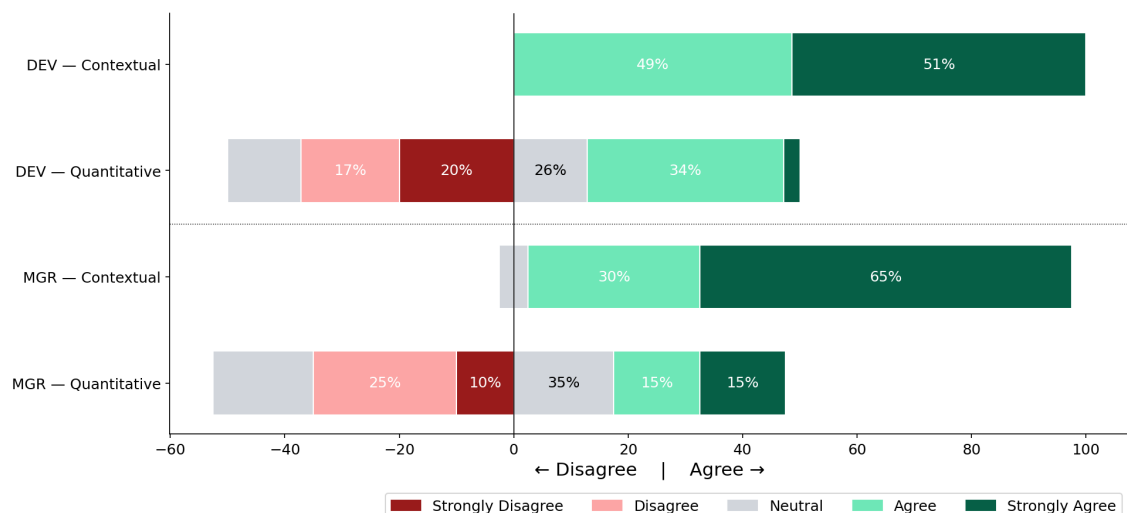


Figure 4.18: Quantitative vs Contextual Assessment Beliefs

Examining the within-person relationship between these two beliefs, Figure 4.19 shows the shift between quantitative and contextual belief scores for each respondent. For managers, the

mean score shifts from 3.00 on the quantitative item to 4.60 on the contextual item, an increase of 1.60 points. For developers, the shift is from 2.58 to 4.53, an increase of 1.95 points. The lines show that almost every respondent shifts upward from their quantitative score to their contextual score. A small number of lines are flat, representing respondents who score identically on both items, and two lines cross downward. The most pronounced shift occurs among respondents who score *Strongly Disagree* or *Disagree* on the quantitative assessment, who shift almost all to *Strongly Agree* on the contextual assessment. The near identical mean shift across both roles indicates that the preference for contextualisation over quantification is consistent at the individual level and is not driven by group-level differences.

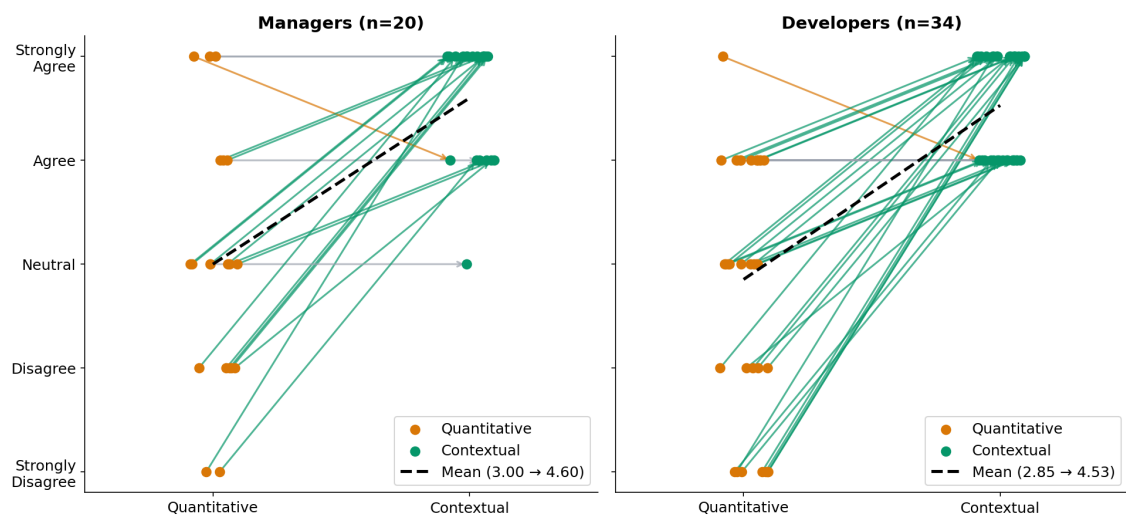


Figure 4.19: Quantitative to Contextual Belief Shift by Role

Regarding the correlation between contextualisation beliefs and perceived metric usefulness, Figure 4.20 presents the Spearman correlations between contextualisation beliefs and usefulness for each of the nine metric categories across all respondents. All correlations fall within the range of -0.14 to +0.21, and none approach significance. The largest positive associations are with *Developer Experience* ( $r=0.21$ ) and *Satisfaction and Wellbeing* ( $r=0.19$ ), both individual-level metric categories, and the largest negative associations are with *Technical Debt & Maintainability* ( $r=-0.14$ ), *Quality* ( $r=-0.07$ ) and *Communication & Collaboration* ( $r=-0.05$ ). The absence of any meaningful correlation across all nine categories indicates that the strength of a respondent's belief in contextualisation is not systematically related to how useful they find any particular metric category.

Moving to the relationship between contextualisation beliefs and metric usage, Figure 4.21 presents the percentage of managers using each metric category, grouped by their level of agreement with the contextualization statement. *Output & Efficiency*, *Quality*, and *Value & Outcomes* are used by 78%-100% of managers across all three belief levels, with little variation by belief strength. The most notable pattern appears in *Learning & Growth*, an individual-level metric category, where usage drops from 70% among those who *Agree* to 20% among those who *Strongly*

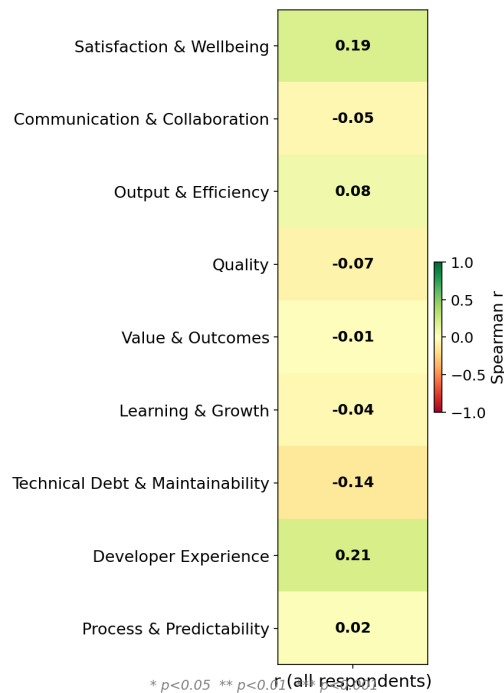


Figure 4.20: Context Belief and Useful Ratings - Spearman

*Agree*. These variations are descriptive, given the small group sizes, particularly the single Neutral managers.

Revealing a further tension within the manager group, Figure 4.22 shows that 19 out of the 20 managers agreed or strongly agreed that metrics must be contextualised, representing 95% of the manager sample. 87.5% still use *Output & Efficiency* as a metric category, making it the most used despite being the least contextual by design. Managers who most endorse contextualisation actually use fewer of the more nuanced categories, such as *Learning & Growth* (39%) and *Developer Experience* (48%), both individual-level metric categories, compared to those who are neutral on contextualisation.

Finally, Figure 4.23 presents mean usefulness scores across all nine metric categories for the Context High group, those who scored *Somewhat Useful* (4) or *Very Useful* (5) on contextualization, and the Context Low group, those who scored *Neutral* (3), *Not Very Useful* (2) or *Not Useful at All* (1). The Context High polygon sits consistently between 3.5 and 4.5 across all categories, forming a broadly regular shape with no category rated substantially higher or lower than the others. The Context Low profile, representing a single respondent, is irregular, with lower scores (3) on *Developer Experience* and *Process & Predictability*. Besides that, and except for *Satisfaction & Wellbeing* and *Learning & Growth*, which both score 4, all other metric categories are rated 5. Given that the Context Low group consists of a single respondent, no inferential conclusions are drawn from this comparison. The shape of the Context High polygon indicates that strong contextualisation believers rate all nine metric categories at broadly similar levels of usefulness, without systematically favouring or disfavouring any particular category.

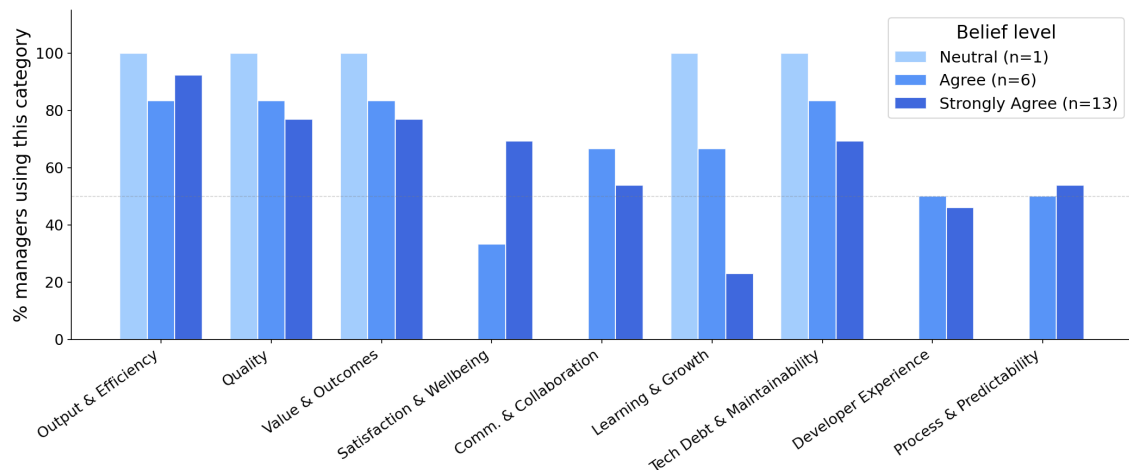


Figure 4.21: Comparison between "Metrics must be contextualised" belief and metric category use

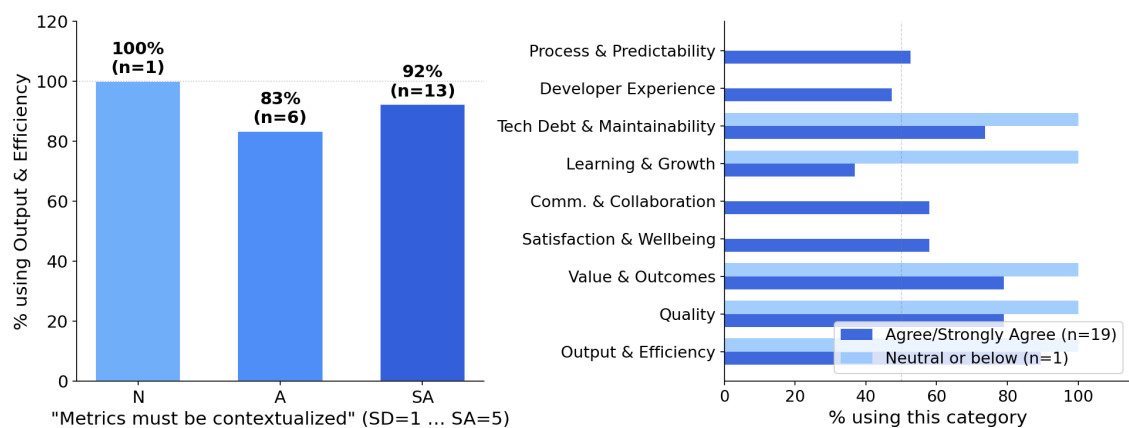


Figure 4.22: Belief in Contextualisation vs Use of Output & Efficiency metrics

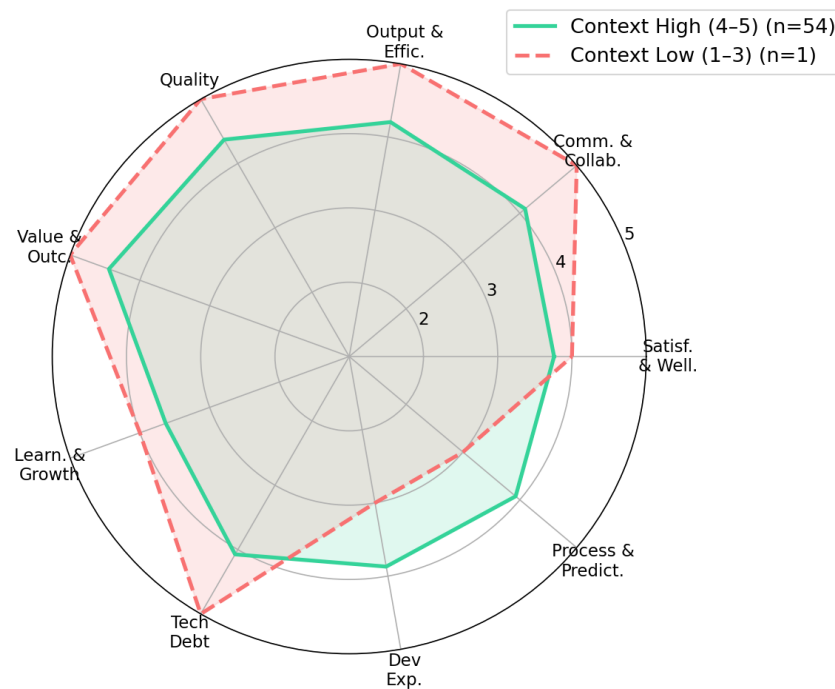


Figure 4.23: Mean Usefulness Ratings by Context Group

In summary, these findings indicate that metrics are perceived as most useful and reliable in contexts where they are tied to product delivery, specifically *Output & Efficiency*, *Quality*, *Value & Outcomes*, and *Technical Debt & Maintainability*, which are metrics that can be applied at both individual and team levels, suggesting that managers prefer metrics that can serve both levels of analysis. Metrics related to individual development and well-being, such as *Learning & Growth*, *Developer Experience*, and *Satisfaction & Wellbeing*, are used less frequently and perceived as less useful. Managers consistently rate metric usefulness higher than developers, and developers show more distributed and cautious ratings, suggesting that reliability is perceived differently across roles. Work setup does not appear to influence perceptions of *Communication & Collaboration* metrics, indicating that usefulness is stable across remote, hybrid, and on-site contexts.

Furthermore, metrics are only considered reliable when interpreted in context. The near-universal endorsement of contextualisation, combined with weak support for quantitative assessment, suggests that practitioners view decontextualised metrics as unreliable. However, a tension exists: despite endorsing contextualisation, managers continue to use decontextualised metrics such as *Output & Efficiency* at high rates, suggesting a gap between belief and practice. Strong contextualisation believers rate all metric categories at broadly similar usefulness levels, without systematic favouritism toward any particular category, indicating that those who prioritise context see value across diverse metric types. Together, these findings suggest that productivity metrics are most useful and reliable in product-delivery contexts and when interpreted qualitatively and contextually, though a gap remains between practitioners' stated beliefs and their actual metric usage.

## 4.5 RQ4 - In what context do developers and managers favour individual or team productivity?

To answer RQ4, the survey examined participants' beliefs about the relationship between individual and team productivity, their preferences in concrete scenario trade-offs, and the accuracy with which each role predicts the other's preferences.

The pilot interviews provided initial context for how managers conceptualise the relationship between individual and team productivity. M1, M5, and M7 stated that team productivity is a priority over individual productivity, and M4 noted that individual productivity should always be considered keeping team impact in mind. M2 offered a different perspective, describing team productivity as the mean of the productivity of the individuals who compose the team, representing a more aggregate view. M1, on the other hand, cautioned that a group of productive individuals may not make a productive team. Managers also noted that conflict between levels can be avoided through goal alignment, as suggested by M1 and M3, or through balance, as stated by M6. M1 further noted that interruptions, while detrimental to individual productivity, may be necessary to support team outcomes. M7 highlighted an organisational tension: while they think team productivity is more important, the company prioritises individual productivity.

Regarding beliefs about individual vs team productivity, Figure 4.24 presents the rate of agreement with five statements about individual and team productivity, answered identically by both managers and developers. The two items with the highest agreement across both roles are "Team productivity and individual productivity should be assessed for different purposes", with managers at 75% and developers at 72.2%, and "Team productivity is more important than individual productivity", with managers at 65% and developers at 80.6%. Both roles, therefore, lean toward team primacy and purpose differentiation, with developers expressing stronger agreement on team importance. Near-consensus is also observed in "It is harder to assess individual productivity over team productivity", where managers sit at 50%, and developers sit at 44.4% agreement, showing genuine uncertainty rather than endorsement. Both roles show low agreement that productive individuals make a productive team, with managers at 45% and developers at 33.3%, indicating scepticism toward the reductionist view that team productivity simply aggregates from individual performance. The most notable divergence is on "There is a conflict between productivity of individuals and of the whole team", where only 10% of managers agree compared to 33.3% of developers, which is significant with  $p = 0.016$ , with a medium effect size,  $r = 0.38$ , and represents the largest substantive disagreement between roles regarding this matter.

Regarding actual-scenario preferences, Figure 4.25 presents the percentage of managers (M1) and developers (D1) who choose Scenario B, which represents team-oriented behaviour, for each of the fourteen scenarios. Both groups show a strong overall preference for team-oriented behaviour, with managers choosing B in 81.1% of scenarios and developers in 74.6%. The largest manager-positive gaps are "Solve Alone vs Team Discussion", with managers at 75% and developers at 52.8%, and "Deepen Expertise vs Broaden Skills", with managers at 80% and developers at 63.9%, but both are non-significant. The only developer-positive gap of a meaningful size is

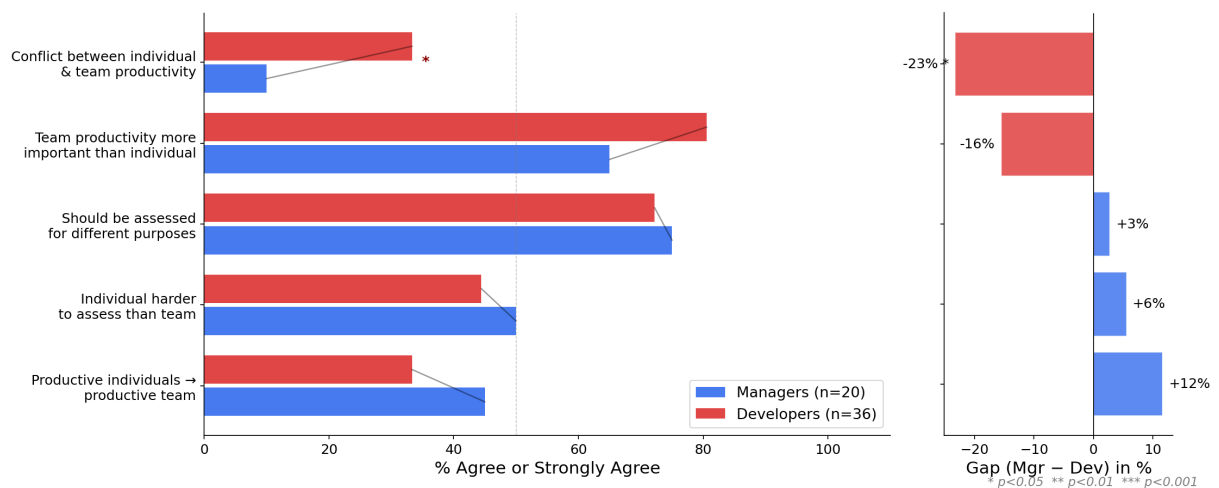


Figure 4.24: Individual vs Team Productivity Beliefs

"Ship Fast vs Refactor", with developers at 75% and managers at 50%. Three scenarios show near-zero gaps and high B preference rates for both roles: "Decline Mentoring vs Mentor Juniors", "Optimistic Estimates vs Realistic Buffer", and "Avoid Feedback vs Honest Feedback". "Calendar Blocks vs Stay Available" is the only scenario where participants did not incline for the team oriented scenario, with only 50% of managers and 33.3% of developers selecting scenario B, the lowest percentages of both roles.

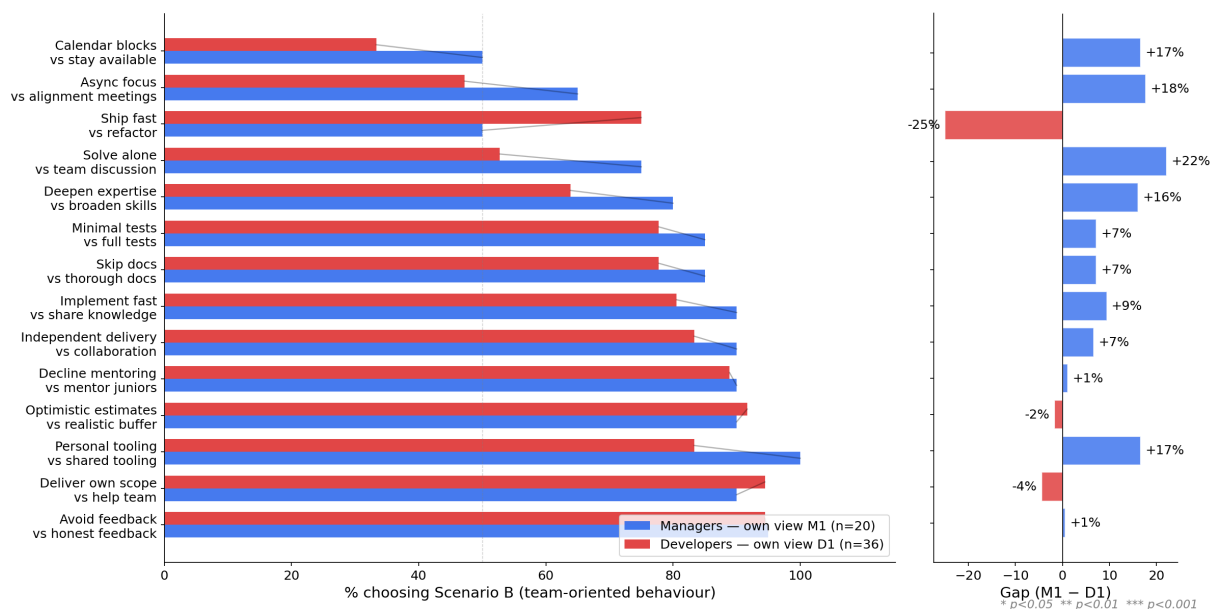


Figure 4.25: M1 vs D1 - Actual Views

Comparing how accurately each role predicts the other's preferences, Figure 4.26 presents the comparison between predictions and real answers. In the left panel, managers' guesses of developers' answers vs developers' views, it is possible to observe that managers' guesses about developer preferences are consistently lower than developers' actual preferences across almost

#### 4.5 RQ4 - In what context do developers and managers favour individual or team productivity?47

all scenarios. The largest underestimations by managers are "Skip Docs vs Thorough Docs", "Independent Delivery vs Collaboration", "Implement Fast vs Share Knowledge", and "Ship Fast vs Refactor". In the right panel, developers' guesses of managers' answers vs managers' views, it is possible to observe that developers similarly underestimate manager team orientation across scenarios, with the largest errors on "Skip Docs vs Thorough Docs", "Implement Fast vs Share Knowledge", "Independent Delivery vs Collaboration" and "Minimal Tests vs Full Tests". Both roles assume the other is less team-oriented than they actually are, and do so with magnitude.

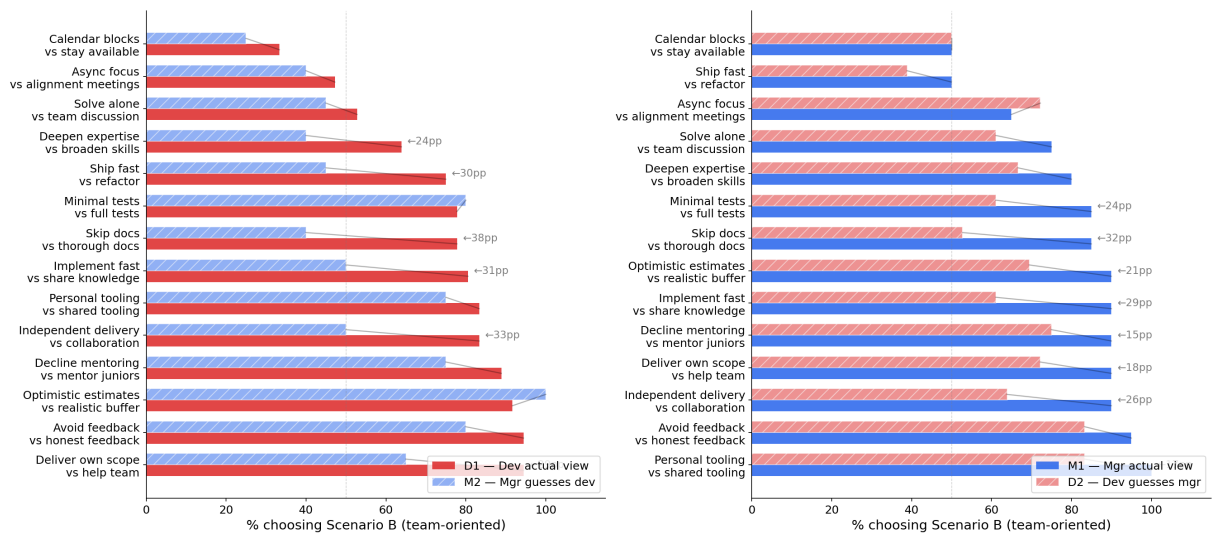


Figure 4.26: M2 vs D1 & D2 vs M1 - Prediction of Other Role Preferences

Examining the full cross-perspective pattern, Figure 4.27 shows the percentage of participants choosing B across all four perspectives for each scenario. The M2 column is consistently the palest, indicating that managers' guesses about developer preferences are the most pessimistic of all four perspectives, with several M2 cells falling into yellow or orange territory while the corresponding D1 values are substantially higher. Three scenarios show high B preference across all four perspectives, being "Avoid Feedback vs Honest feedback" (M1=95%, M2=84%, D1=94%, and D2=85%), "Optimistic Estimates vs Realistic Buffer", (M1=89%, M2=100%, D1=91%, and D2=73%), and "Personal Tooling vs Honest Feedback" (M1=100%, M2=74%, D1=85%, and D2=82%). The scenarios with the greatest cross-perspective variability are "Calendar Blocks vs Stay Available", "Ship Fast vs Refactor", and "Async Focus vs Alignment Meetings".

Finally, Figure 4.28 presents the relationship between agreement with the statement "Team productivity is more important than individual productivity" and team-orientation scores across the fourteen scenarios, expressed as the fraction of scenarios in which each respondent chose scenario B. A Spearman correlation across all respondents showed no significant association ( $r = 0.11$ ,  $p = 0.417$ ). The scatter shows that respondents at every belief level, including those who *Strongly Disagree* with team primacy, consistently score above 0.5 on team orientation in their scenario choices, with most scoring between 0.65 and 0.95 regardless of stated belief. Both managers and developers show the same pattern, with no visible role-based separation in the scatter. The abstract

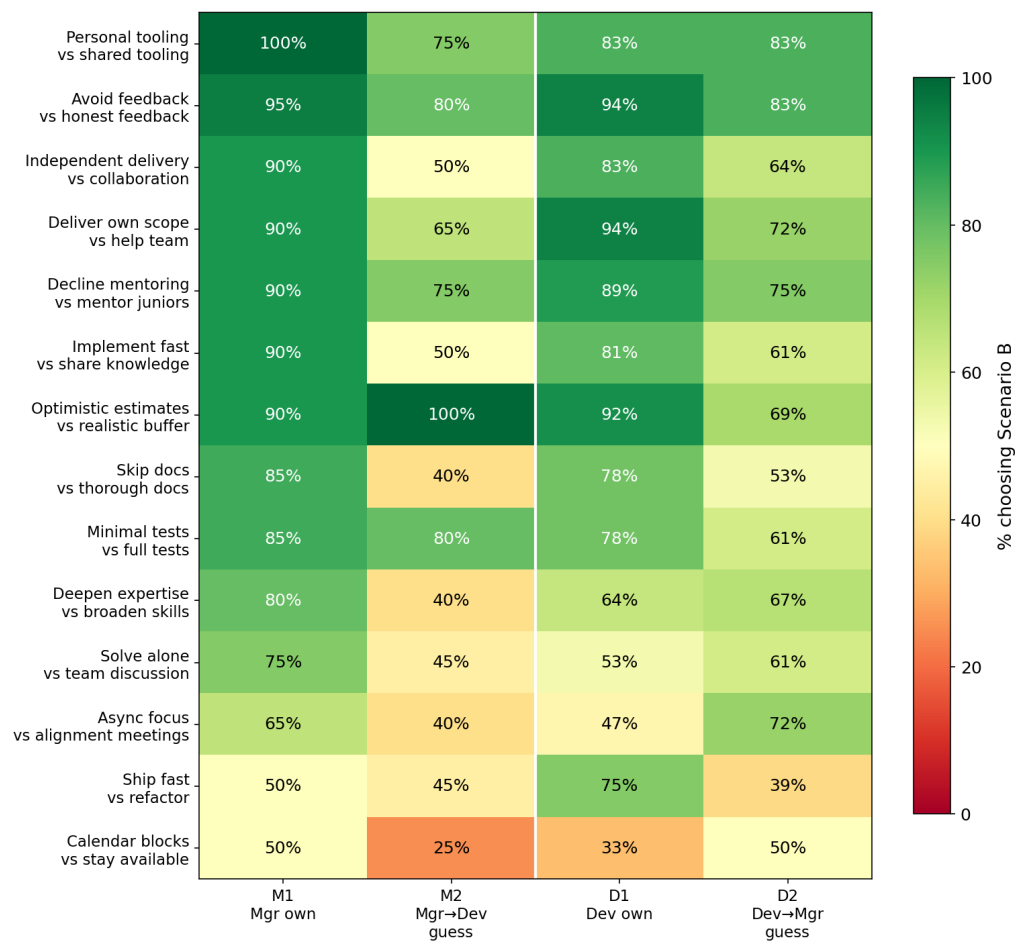


Figure 4.27: Percentage of Respondents Choosing Scenario B per Perspective

#### 4.5 RQ4 - In what context do developers and managers favour individual or team productivity?<sup>49</sup>

belief that team productivity matters more than individual productivity, therefore, does not predict the degree to which respondents choose team-oriented behaviour in concrete situational trade-offs.

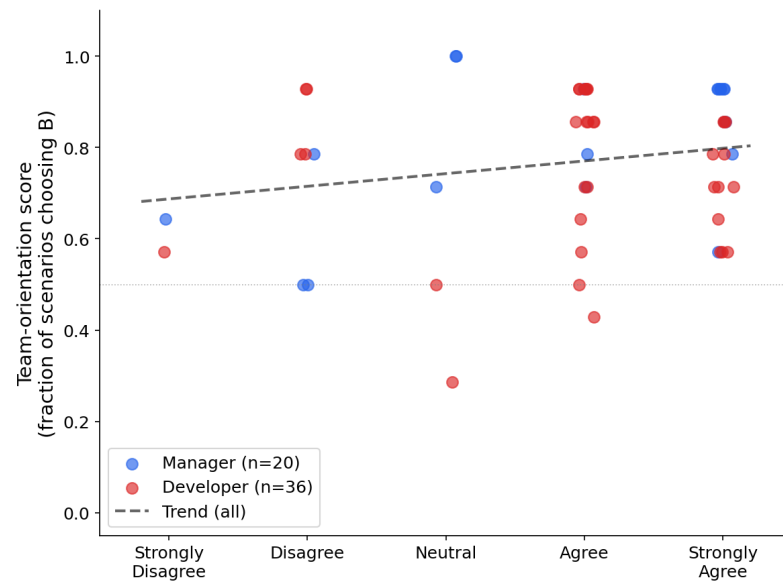


Figure 4.28: Team > Productivity Belief vs Scenario Choices

In summary, the findings reveal that both managers and developers strongly favour team productivity over individual productivity in principle, with developers expressing even stronger agreement on team primacy. A notable divergence emerges in the perception of conflict between individual and team productivity, with developers significantly more likely than managers to perceive tension. In concrete scenario trade-offs, both roles demonstrate a strong preference for team-oriented behaviour, though managers consistently choose team-oriented options more frequently than developers. The only scenario in which neither role favoured the team-oriented option was "Calendar blocks vs Stay Available", where maintaining focused individual coding time was preferred over remaining available for team coordination. This indicates that individual productivity is prioritised specifically when the stakes are uninterrupted cognitive focus, a resource both roles appear to treat as non-negotiable, even within a broadly team-oriented disposition. Both roles systematically underestimate the other's team orientation, with managers producing the most pessimistic predictions. This means that managers are likely designing assessment systems on the basis of an assumed individual orientation in developers that does not reflect developers' actual values, while developers are interpreting managerial practices through an assumed output focus that similarly overstates the gap between their priorities. Furthermore, stated beliefs about team primacy do not predict actual team-oriented behaviour scenarios, suggesting that abstract agreement on team importance does not consistently translate into concrete situational choices. Together, these findings indicate that team productivity is favoured in most contexts, except when individual cognitive focus is required. The tension between individual and team productivity is navigated differently depending on the role: managers underestimate developer collaborative values and may design systems that over-correct toward individual accountability, while developers,

perceiving more conflict than managers acknowledge, may experience those systems as misaligned with their actual priorities.

## Chapter 5

# Discussion

This study set out to investigate how managers and developers conceptualise and assess productivity at the individual and team levels, and to identify where their perceptions diverge, with practical consequences for how metrics are designed, applied, and received. The results presented across the four research questions reveal a field in which stated beliefs, organisational practices, and perceptions are persistently misaligned between managers and developers: (i) managers define productivity in output-oriented terms, while developers place significantly greater weight on collaborative and well-being dimensions such as helping others and divergence in what each role considers productive work, rather than a disagreement about fairness; (ii) both developers and managers endorse team primacy in principle, yet both also systematically underestimate the other’s collaborative orientation; (iii) contextualisation is near-universally endorsed, yet decontextualised metrics dominate practice, and (iv) awareness of gaming risk is widespread among managers, yet it has no meaningful impact on metric selection. These contradictions are the central empirical findings of this study, and they carry significant implications for how productivity assessment should be understood and designed in software engineering organisations.

### 5.1 The Nature of Misalignment: Definitions and Perceptions

The literature review established misalignment as a central concern in software engineering productivity research. Storey, Houck, and Zimmermann [16] documented misalignment between how developers and managers define productivity, specifically, that developers anticipate more output-oriented managerial definition than managers actually hold, while managers possess more accurate insight into how developers conceptualise productive work, and Forsgren et al. [6] argued that many of the most valuable developer activities, such as mentoring, knowledge sharing, and code review, are systematically invisible in individual metrics.

Regarding definitions of productivity, the results partially support prior work. Managers were substantially more likely than developers to define productivity in terms of business value (85% vs 77.8%), reaching business objectives (80% vs 55.6%), and delivering tasks on time (65% vs 52.8%). These differences are directionally consistent with the output-oriented framing that the

literature attributes to managers [11, 16]. The most statistically significant divergence concerns the definition of productivity as helping others, endorsed by 75% of developers but only 40% of managers. This single finding quantitatively confirms what prior qualitative studies had suggested: collaborative and relational activities are more central to what developers consider productive work than to what managers do [6, 14].

These findings extend prior qualitative work in two specific ways. Hernández-López et al. [8] and Oliveira et al. [11] established that managers favour output-oriented definitions, and Storey et al. [16] documented a conceptual misalignment between roles, but these studies relied on single-role samples or qualitative methods that precluded direct cross-role comparison. This study measures both roles with the same instrument, allowing the divergence to be quantified: the gap on output-oriented definitions such as *Reaching Business Objectives* (80% vs 55.6%) and *Delivering Tasks on Time* (65% vs 52.8%) confirms what the literature attributed to managers, while the 35 percentage-point gap on *Helping Others*, the only definition reaching statistical significance, identifies collaborative activity as the sharpest point of divergence, a finding the prior qualitative literature had not foregrounded.

However, the scenario-based data in RQ4 introduces an important corrective. When asked to make concrete behavioural choices, rather than abstract definitional ones, both managers and developers overwhelmingly favour team-oriented options, with managers choosing team-oriented scenarios in 81.1% of cases and developers in 74.6%. Crucially, both roles also systematically underestimate the other's collaborative orientation, with managers producing the most pessimistic predictions of developer preferences. This pattern suggests that the misalignment documented in the literature is not primarily a conflict of values, but a conflict of perceived values: managers and developers are more alike in their actual orientations than either group believes the other to be. The practical consequence is significant. Interventions designed to resolve values misalignment may be addressing the wrong problem. The more adjustable intervention should be increasing mutual transparency about each role's actual priorities, since the gap between real behaviour and perceived behaviour is larger than the gap between real behaviours.

This distinction also helps clarify the divergence in conflict perception documented in RQ4. Only 10% of managers agree that there is a conflict between individual and team productivity, compared to 33.3% developers. A plausible explanation is that managers, as the architects of assessment systems, may be less likely to experience the friction those systems produce. Developers, who are subject to those systems, may encounter situations in which optimising for what is measured conflicts with what benefits the team, a dynamic extensively documented by Sadowski and Zimmermann [14] and Fritz [7].

## 5.2 The Belief-Behaviour Inconsistency

Across all four research questions, a consistent significant pattern emerges: awareness of a problem does not reliably translate into behaviour change. This gap between stated belief and actual

practice challenges a foundational assumption in the productivity literature: that raising awareness of metric misuse is itself a meaningful corrective.

The asymmetry documented in RQ2 between managers and developers is a useful starting point. 55% of managers report that their productivity assessment has been misinterpreted in the past, and 45% find it difficult to know what constitutes a fair productivity assessment. Yet, only 8% of developers agree that their manager misuses productivity metrics, and 69% rate assessment as fair. This suggests that managers experience the difficulty of calibration, while developers receive the result without the context of how it was reached. This is consistent with M4's pilot interview observation that productivity is hard to assess, and that understanding how to assess individual productivity is genuinely difficult. A difficulty that, from the developer perspective, largely goes unnoticed.

The clearest illustration of the belief-behaviour gap comes from RQ2. Among managers who *Strongly Agree* that providing metrics incentivise gaming ( $n=7$ ), 86% continue to use *Output & Efficiency* metrics, the most gameable category in the study. Among those who *Agree* ( $n=8$ ) or are *Neutral* ( $n=3$ ), the rate is 100%. The rate of *Output & Efficiency* use does not decrease as gaming belief increases. It is also the most used metric category overall (90%) and receives the highest usefulness score from managers (4.8 out of 5). *Output & Efficiency* metrics, even if gameable, are valuable: teams need to produce, and measuring output is a legitimate organisational need. This suggests that managers do not experience acknowledging the risk of a metric and continuing to use it as contradictory, likely because no readily available alternative satisfies the same informational demands.

*Output & Efficiency* metrics are employed because they are legible, comparable, and defensible to organisational stakeholders. A manager who acknowledges their risk may still perceive no alternative that satisfies the same organisational demands. This interpretation is consistent with Aquino Júnior and Meira's [1] observation that no integrative model for productivity assessment exists. The risk, as the literature warns [7, 14], is that without deliberate effort to select metrics that are meaningful rather than merely accessible, organisations may inadvertently anchor their assessment practices on what is easy to measure rather than on what is meaningful. This underscores the need for organisations to actively seek out and adopt metrics that reflect the full range of what constitutes productive work, including the collaborative and well-being dimensions that output-oriented metrics leave out.

A second, complementary account concerns contextualisation. RQ3 reveals near-universal endorsement of the contextualisation of metrics (95% of managers and 100% of developers), yet 87.5% of the managers who endorse contextualisation still use *Output & Efficiency* as a metric category, and use fewer of the more nuanced individual-level categories, such as *Learning & Growth* (39%) and *Developer Experience* (48%). One explanation for this pattern is that managers may treat contextualisation as something they do when interpreting results, rather than as a principle that should guide which metrics they select in the first place. In other words, a manager may choose *Output & Efficiency* metrics and still consider themselves a contextualised assessor, because they plan to interpret the numbers alongside their knowledge of the developer, the sprint,

and the project. The issue is that this approach places the entire burden of contextualisation on individual managerial judgement, rather than building it into the measurement system from the outset.

The RQ2 data introduces a further nuance. Among developers, the primary predictor of criteria optimisation behaviour is not the belief that metrics incentivise gaming, nor a perception of misuse and unfairness. It is awareness, specifically of the purpose for which productivity is being assessed, and, to a lesser extent, of how it is assessed. This finding reframes Goodhart's Law [7] at the individual level. The law is typically described as a systemic phenomenon: once a measure becomes a target, it ceases to be a good measure. The data suggests it operates through a specific psychological mechanism: when developers understand the goal behind the measurement, they optimise toward that goal. If developers understand the purpose of assessment and optimise toward it, the question becomes whether that purpose is well-designed. When metrics reflect meaningful broader goals, optimising toward them is desirable. The risk arises when the metrics being optimised toward are narrow or decontextualised, which is precisely the case with *Output & Efficiency* metrics used in isolation. Opacity, by contrast, is not a solution: withholding the purpose of assessment would undermine the trust and autonomy that Storey et al. [17] identify as critical to developer wellbeing and perceived productivity, and would likely lead to disengagement and a loss of confidence in the assessment process. The more constructive response is to ensure that what developers are aware of, and therefore optimise toward, is worth optimising for.

### 5.3 The Focus Time Exception: Where Individual Productivity Wins

A central argument of the literature is that team productivity should take precedence over individual productivity in collaborative knowledge work environments [6, 14], and the results from this study broadly support that framing. Both managers and developers endorse team primacy in the abstract, and both demonstrate team-oriented preferences across the majority of concrete scenario trade-offs. However, the data reveal one consistent exception.

The "Calendar Blocks vs Stay Available" scenario, in which participants choose between blocking large portions of the calendar for focused individual work vs remaining available for team coordination, is the only scenario in the study where participants did not favour the team-oriented option. Only 50% of managers and 33.3% of developers chose the team-oriented option, remaining available, the lowest rates of both roles across all fourteen scenarios. This finding is consistent with Murphy-Hill et al. [9] large-scale survey identifying uninterrupted focus as among strong predictors of self-rated developer productivity, and with Sadowski and Zimmermann's [14] acknowledgement that minimising interruption and optimising for individual efficiency can conflict with coordination demands of team work. The study presents that practitioners, when forced to choose, prioritise individual focus time over team availability. Developers are willing to sacrifice individual efficiency for team outcomes across a wide range of scenarios, but not when the cost is control over uninterrupted attention. Organisations seeking to support both individual and

team productivity may need to treat protected focus time not as an individual accommodation, but as a structural team norm.

The data in RQ4 also reveal that stated beliefs about team primacy do not predict team-oriented behaviour in concrete scenarios. Respondents who *Strongly Disagree* that team productivity is more important than individual productivity still choose team-oriented scenarios between 65% and 95% of the time. This is an instance of the broader belief-behaviour gap: attitudinal survey items about relative importance may be measuring a different construct from behavioural preferences in situated trade-offs.

## 5.4 What Metrics Are Actually Used For: Dual-Level Utility and Its Risks

The metric usage data in RQ3 surfaces a pattern that lacks quantitative documentation in the literature: the metrics managers use most frequently are not level-specific but dual-level, applicable to both individual and team analysis. By contrast, the metrics used less frequently are those designed for a single level of analysis, whether individual-oriented categories, such as *Satisfaction & Wellbeing*, *Learning & Growth*, and *Developer Experience*, or team-oriented ones such as *Communication & Collaboration* and *Process & Predictability*.

Critically, the four significant differences in perceived usefulness between managers and developers ( $p < 0.05$ ) are all concentrated in the dual-level categories. The greatest divergence in perceived value between roles occurs precisely for metrics that serve both individual and team analysis, rather than for metrics designed for a single level. Managers rate all four significantly higher than developers. This suggests that managers derive additional utility from the dual applicability of these metrics, whereas developers do not. Developers may perceive the same dual applicability as a source of risks, as a metric that can be applied at the individual level can be used for evaluation and comparison, which prior work has identified as a primacy driver of dysfunctional behaviour and concerns about privacy and misinterpretation [7, 14, 16].

The literature argues that organisations should maintain a strict operational separation between individual-level metrics used as personal diagnostic tools and team-level metrics used for systemic evaluation [6, 7, 14]. The present findings show that the metrics managers use most and value most are precisely those that resist clean level separation.

## 5.5 Implications for Research and Practice

Taken together, these findings carry several implications for both research and practice.

For research, the persistence of the belief-behaviour inconsistency demands a theoretical explanation. Future work should investigate why acknowledging a metric's risks does not translate into changing its use, specifically whether this reflects the absence of viable alternatives, the organisational costs of switching away from legible and defensible measures, or a tendency to locate contextualisation in interpretation rather than in metric selection. Experimental or longitudinal

designs that introduce governance mechanisms, e.g. formally restricting the purposes for which *Output & Efficiency* data may be used, would allow researchers to test whether structural constraints change behaviour where awareness alone does not. The finding that purpose awareness drives criteria optimisation more strongly than gaming belief or fairness perception also challenges assumptions about transparency, suggesting that purpose communication should be treated as an independent variable with measurable behavioural consequences. Future research should also distinguish between the purposes for which individual-level metrics and team-level metrics are used, examining how managerial intent differs across these levels and how developers perceive and respond to each. The mutual underestimation of team orientation documented in this study raises the question of whether the misperception is self-reinforcing: if managers assume developers are individually motivated and design assessment systems accordingly, developers may respond by behaving more individually than they otherwise would, thereby confirming the manager's initial assumption. A longitudinal study tracking the relationship between managers' beliefs about developer motivation, the assessment systems they design, and subsequent developer behaviour could test whether this feedback loop operates in practice and whether transparency interventions break it. Finally, the belief-behaviour gap between abstract team primacy beliefs and concrete scenario choices suggests that attitudinal items may systematically diverge from situated preferences. Future studies measuring productivity preferences should therefore prioritise or complement attitudinal survey items with scenario-based instruments, as the two appear to capture meaningfully different constructs.

For practice, organisations should focus on governing how metrics are used rather than which metrics are collected. Metrics such as *Output & Efficiency* can inform team retrospectives and process improvement, but should be carefully analysed when regarding individual performance evaluation or compensation decisions. Organisations should recognise protected focus time as a structural team norm, not as an individual accommodation, and should invest in interpretive practices, such as qualitative peer feedback and performance conversations, that supplement rather than replace quantitative data. Contextualisation should be operationalised as a design principle for assessment systems, not left to individual interpretation after data collection.

## Chapter 6

# Threats to Validity

This study carries several limitations, organised below into internal, external, construct, and conclusion validity [19].

### 6.1 Internal Validity

Internal validity concerns matters that may affect the independent variable in ways that are not under the researcher's control.

*Self-Report Bias.* Both the pilot interviews and the survey relied on self-reported data, which is subject to several biases. Participants may have presented themselves in a favourable light, may not have accurately recalled past events or behaviours, or may have provided responses they believed the researcher expected. To mitigate this, anonymity was assured, and questions were framed in neutral, non-judgmental language. However, these biases cannot be entirely eliminated.

### 6.2 External Validity

External validity concerns factors that limit the ability to generalise the study's results.

*Sample Size and Composition.* The survey sample consisted of 20 managers and 36 developers, a total of 56 respondents. The small sample size, particularly for the manager group, limits the generalisability of the findings and constrains the ability to detect small but meaningful effects. While non-parametric tests such as Mann-Whitney U and Fisher's exact test are robust to small sample sizes, statistical power remains limited. Findings should therefore be interpreted with caution, and replication with larger, more balanced samples is recommended.

*Sampling Strategy.* Participants of the pilot interviews were recruited through convenience sampling, primarily via the researcher's and supervisors' personal and professional networks. This may have introduced selection bias, as participants may share similar organisational, geographical, or cultural characteristics.

### 6.3 Construct Validity

Construct Validity concerns whether the survey items and interview questions adequately measure the theoretical constructs they are intended to capture.

*Scenario Trade-Offs.* The 14 hypothetical scenarios presented in the survey were grounded in real-world tensions described during the pilot interviews and literature. However, choices made in hypothetical scenarios may not perfectly predict behaviour in actual work situations, where multiple competing priorities and contextual factors interact. The scenarios were necessarily simplified compared to real-world decision-making.

### 6.4 Conclusion Validity

Conclusion validity concerns issues that affect the ability to draw the correct conclusion from the observed relationships in the data.

*Statistical Power.* The small sample sizes increase the risk of failing to detect effects that would reach significance with a larger sample. This is most consequential for comparisons where trends are observed but do not reach the significance threshold, which are, therefore, treated as directional rather than conclusive. Non-parametric tests were appropriately used given the ordinal nature of Likert-scale data and the small sample sizes, but they have low power.

## Chapter 7

# Conclusion

This study set out to investigate how productivity is conceptualised and assessed at individual and team levels from the perspective of software engineering practitioners. Its central finding is not that managers and developers disagree on fundamental values, but that they systematically misread each other, and that the assessment systems they use reflect and reinforce that misreading. The paragraphs that follow summarise what was found, what it contributes, and what it implies for research and practice.

Regarding RQ1, the findings reveal that managers define productivity in output-oriented terms, emphasising business value, objectives, efficiency and timeliness, and focusing less on collaborative definitions centred on helping others than developers. Regarding RQ2, developer awareness, particularly of the purpose of assessment, emerged as the primary driver of criteria optimisation behaviour, whereas beliefs about gaming, fairness, or misuse did not meaningfully predict it. Crucially, individual assessment awareness has a stronger relationship with gaming-adjacent behaviour than team assessment awareness, indicating that misuse is more closely associated with individual-level metrics. Managers who acknowledge gaming risk do not meaningfully reduce their use of the most gameable metric category, suggesting a disconnect between awareness and practice. Regarding RQ3, metrics are perceived as most useful and reliable in product-delivery contexts, while individual development and well-being metrics are underused. The near-universal endorsement of contextualisation, combined with weak support for quantitative assessment, suggests that practitioners view decontextualised metrics as unreliable, though a gap remains between stated beliefs and actual metric usage. Regarding RQ4, both managers and developers strongly favour team productivity over individual productivity in principle, with developers expressing even stronger agreement on team primacy. In concrete scenario trade-offs, both roles demonstrate a strong preference for team-oriented behaviour, though managers consistently choose team-oriented options more frequently than developers. Both roles systematically underestimate the other's team orientation, assuming the other is less collaborative than they actually are. A notable exception emerged in the scenario pitting calendar blocking for focused individual work against remaining available for team coordination, the only scenario in which neither role favoured the team-oriented option, suggesting that protecting individual focus time represents a genuine bound-

ary condition for team primacy. Stated beliefs about team primacy do not predict team-oriented behaviour in scenarios, suggesting that abstract agreement on team importance does not consistently translate into concrete situational choices.

This study makes several contributions to both research and practice. From a theoretical perspective, this study makes three specific contributions that extend beyond what prior qualitative work established. First, while prior studies such as Hernández-López et al. [8] and Oliveira et al. [11] had documented qualitatively that managers favour output-oriented productivity definitions, this study provides a cross-role quantitative comparison, revealing that the gap is largest on collaborative definitions. Second, the study identifies purpose awareness, rather than gaming belief, fairness perception, or misuse perception, as the primary predictor of criteria optimisation behaviour among developers. Third, the study reveals that both managers and developers systematically underestimate the other's team orientation, demonstrating that the misalignment documented in the literature is primarily perceptual rather than substantive. From a practical perspective, the findings carry implications that are more specific than prior work has offered. The finding that purpose awareness drives criteria optimisation more strongly than method awareness suggests that how organisations communicate the intent of measurement matters as much as which metrics they select. Managers and developers who understand why a metric is being collected, and what decisions it will and will not inform, are better positioned to engage with it constructively. The finding that both roles systematically underestimate the other's team orientation suggests that the friction organisations experience around productivity assessment may stem less from genuine value conflict and more from mutual misperception.

These findings support evidence-based guidance for designing more context-appropriate and ethically sound productivity assessment practices, discussed in the recommendations below.

The findings of this study should be interpreted in light of some limitations, which are discussed in detail in Chapter 6. These include the small sample size, particularly for the manager group ( $n=20$ ), which limits statistical power and generalisation, the reliance on self-reported data, which is subject to biases, and the cross-sectional nature of the survey design, which precludes causal inference. The hypothetical scenarios, while grounded in real-world tensions, may not perfectly predict actual behaviour in work contexts. Despite these limitations, the study provides valuable insights into practitioner perceptions of productivity assessment.

Based on the findings, several recommendations for practice emerge. First, organisations should treat metric governance as a structural matter rather than a cultural one. The finding that managers who strongly agree that metrics incentivise gaming still use the most gameable metric categories indicates that awareness alone is insufficient. Governance mechanisms that formally distinguish the purposes for which each metric may be used, e.g. specifying that Output & Efficiency data may inform sprint retrospectives but may not be used as the basis for individual performance ranking or compensation decisions, are more likely to reduce misuse than awareness campaigns alone. Second, developers should be involved in the design of productivity assessment practices, as metrics perceived as credible and fair are more likely to be accepted and used constructively. The mutual underestimation of team orientation documented in RQ4 suggests that

structured dialogue between managers and developers about their actual values could reduce the perceptual gap that currently shapes how both roles interpret and react to metric systems. Third, organisations should protect individual focus time as a structural team norm rather than an individual accommodation. The scenario evidence showing that both roles decline team availability when it conflicts with sustained focus work suggests that measurement systems treating responsiveness as a default productivity indicator risk penalising the very behaviour practitioners identify as most critical to individual effectiveness.

Longitudinal studies are needed to establish causal relationships between awareness, governance interventions, and behavioural outcomes. Experimental studies manipulating the degree and specificity of purpose communication would directly test the causal direction of the RQ2 finding: whether making metric purpose explicit to developers reduces criteria optimisation, or whether developers who already optimise are simply more attentive to purpose signals. Larger-scale replication would help test the generalisability of the perceptual findings, particularly the mutual underestimation of team orientation and the dual-level metric preference documented in RQ3, which rested on a manager sample of  $n=20$  and a total sample of  $n=56$ . Finally, future research could examine whether the focus time boundary condition documented in RQ4 (practitioners declining team availability when it conflicts with individual focus) generalises across different team structures, project phases, and work arrangements.

In conclusion, this study reveals that the central challenge in software engineering productivity assessment is not that managers and developers hold irreconcilably different values: the scenario evidence shows both roles behave collaboratively across the overwhelming majority of situational trade-offs. The challenge is that they do not know this about each other. Both roles assume the other is less team-oriented than they actually are. Measurement systems designed on the basis of assumed value conflict may therefore be solving the wrong problem. The more fundamental intervention is transparency: about the limits of what individual-level data can and should inform, and about the actual collaborative orientations that both roles already hold. The findings of this study suggest that responsible productivity assessment in software engineering requires not better metrics alone, but better governance of how metrics are used, communicated, and understood across the roles that both produce and are shaped by them.

# References

- [1] Gibeon Soares de Aquino Júnior and Silvio Romero de Lemos Meira. “Towards effective productivity measurement in software projects”. In: *2009 Fourth International Conference on Software Engineering Advances*. IEEE. 2009, pp. 241–249.
- [2] Virginia Braun and Victoria Clarke. “Using thematic analysis in psychology”. In: *Qualitative Research in Psychology* 3.2 (2006), pp. 77–101.
- [3] Carol Dekkers and Patricia McQuaid. “The dangers of using measurement to (mis) manage: Measuring the software process”. In: *ASQ World Conference on Quality and Improvement Proceedings*. American Society for Quality. 2002, p. 551.
- [4] Norman K Denzin and Yvonna S Lincoln. *The Sage handbook of qualitative research*. sage, 2011.
- [5] Ilker Etikan, Sulaiman Abubakar Musa, Rukayya Sunusi Alkassim, et al. “Comparison of convenience sampling and purposive sampling”. In: *American journal of theoretical and applied statistics* 5.1 (2016), pp. 1–4.
- [6] Nicole Forsgren et al. “The SPACE of Developer Productivity: There’s more to it than you think.” In: *Queue* 19.1 (2021), pp. 20–48.
- [7] Thomas Fritz. “Measuring individual productivity”. In: *Perspectives on Data Science for Software Engineering*. Elsevier, 2016, pp. 67–71.
- [8] Adrián Hernández-López, Ricardo Colomo-Palacios, and Ángel García-Crespo. “Productivity in software engineering: A study of its meanings for practitioners: Understanding the concept under their standpoint”. In: *7th Iberian Conference on Information Systems and Technologies (CISTI 2012)*. IEEE. 2012, pp. 1–6.
- [9] Emerson Murphy-Hill et al. “What predicts software developers’ productivity?” In: *IEEE Transactions on Software Engineering* 47.3 (2019), pp. 582–594.
- [10] Andrew Neal and Beryl Hesketh. “Productivity in organizations”. In: *Handbook of Industrial, Work & Organizational Psychology-Volume 2: Organizational Psychology*. SAGE Publications Ltd, 2001, pp. 7–24.
- [11] Edson Oliveira et al. “Software project managers’ perceptions of productivity factors: findings from a qualitative study”. In: *Proceedings of the 10th ACM/IEEE international symposium on empirical software engineering and measurement*. 2016, pp. 1–6.
- [12] Matthew J Page et al. “The PRISMA 2020 statement: an updated guideline for reporting systematic reviews”. In: *bmj* 372 (2021).
- [13] Malgorzata Pinkowska. “It software project management: impact of team cohesiveness on productivity and performance”. In: *Proceedings of 14th IDIMT* (2006), pp. 75–90.
- [14] Caitlin Sadowski and Thomas Zimmermann. “Rethinking productivity in software engineering”. In: Springer Nature, 2019. Chap. 1, 2, 3, 4, 6, 12.

- [15] Ioannis Stamelos. “Software project management anti-patterns”. In: *Journal of Systems and Software* 83.1 (2010), pp. 52–59.
- [16] Margaret-Anne Storey, Brian Houck, and Thomas Zimmermann. “How developers and managers define and trade productivity for quality”. In: *Proceedings of the 15th International Conference on Cooperative and Human Aspects of Software Engineering*. 2022, pp. 26–35.
- [17] Margaret-Anne Storey et al. “Towards a theory of software developer job satisfaction and perceived productivity”. In: *IEEE Transactions on Software Engineering* 47.10 (2019), pp. 2125–2142.
- [18] Goparaju Sudhakar, Ayesha Farooq, and Sanghamitra Patnaik. “Measuring productivity of software development teams”. In: *Serbian Journal of Management* 7.1 (2012), pp. 65–75.
- [19] Claes Wohlin et al. *Experimentation in software engineering*. Vol. 236. Springer, 2012.

## Appendix A

### Summary of Included Papers

This appendix provides a summary of the 14 studies included in the systematic literature review. The studies span 2001-2022 and represent diverse methodological approaches, including empirical studies, frameworks, position papers, theoretical papers, and books. The table provides an overview of each paper's authors, year, title, methodological approach, and primary focus.

Table A.1: Summary of Included Studies

| Author(s)                   | Year | Paper Name                                                      | Methodological Approach | Primary Focus                                                                                                                    |
|-----------------------------|------|-----------------------------------------------------------------|-------------------------|----------------------------------------------------------------------------------------------------------------------------------|
| Aquino Júnior and Meira [1] | 2009 | Towards Effective Productivity Measurement in Software Projects | Systematic Review       | Proposing a process for defining organisation-specific productivity measurement models, reviewing productivity metrics evolution |
| Dekkers and McQuaid [3]     | 2002 | The Dangers of Using Measurement to (Mis)manage                 | Critical Analysis       | Dangers and pitfalls of software metrics misuse, demonstrating unintended consequences of poor measurement                       |
| Forsgren et al. [6]         | 2021 | The SPACE of Developer Productivity                             | Framework               | SPACE framework - multidimensional developer productivity framework addressing common myths and measurement challenges           |

Continued on next page

Table A.1 – continued from previous page

| Author(s)                    | Year | Paper Name                                                                                  | Methodological Approach                           | Primary Focus                                                                                                              |
|------------------------------|------|---------------------------------------------------------------------------------------------|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------|
| Fritz [7]                    | 2016 | Measuring Individual Productivity                                                           | Conceptual Analysis                               | Challenges of measuring individual developer productivity, discussing objective vs. subjective measures and Goodhart's Law |
| Hernández-López et al. [8]   | 2012 | Productivity in Software Engineering: A study of its meanings for practitioners             | Exploratory Qualitative Study                     | Understanding productivity definitions from software engineers' and project managers' perspectives                         |
| Murphy-Hill et al. [9]       | 2019 | What predicts Software Developers' Productivity?                                            | Large-scale Survey Study                          | Identifying which factors predict self-rated developer productivity, comparing productivity factors across contexts        |
| Neal and Hesketh [10]        | 2001 | Productivity in Organizations                                                               | Theoretical Review from Organizational Psychology | Group and organisational productivity from work psychology perspective                                                     |
| Oliveira et al. [11]         | 2016 | Software Project Managers' Perceptions of Productivity Factors                              | Qualitative Study                                 | Managers' perceptions of developer productivity, how managers define and identify productive developers                    |
| Pinkowska [13]               | 2006 | IT Software Project Management: Impact of Team Cohesiveness on Productivity and Performance | Literature Analysis                               | Analysing the impact of team cohesiveness on the productivity and performance of software intensive projects               |
| Sadowski and Zimmermann [14] | 2019 | Rethinking Productivity                                                                     | Book                                              | Comprehensive collection examining various aspects of software engineering productivity from multiple perspectives         |
| Stamelos [15]                | 2010 | Software Project Management Anti-patterns                                                   | Literature survey                                 | Exploration of software project management anti-patterns and their formal representation for education and avoidance       |

Continued on next page

Table A.1 – continued from previous page

| Author(s)                         | Year | Paper Name                                                                         | Methodological Approach      | Primary Focus                                                                                                                                                                  |
|-----------------------------------|------|------------------------------------------------------------------------------------|------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Storey, Houck and Zimmermann [16] | 2022 | How Developers and Managers Define and Trade Productivity for Quality              | Exploratory Survey Study     | Investigating how different cohorts define and trade productivity for quality and the level of alignment in their views                                                        |
| Storey et al. [17]                | 2019 | Towards a Theory of Software Developer Job Satisfaction and Perceived Productivity | Theoretical Framework        | Exploring the connection between job satisfaction and perceived productivity, focusing on well-being as a driver of engineering success and a leading indicator of performance |
| Sudhakar, Farooq and Patnaik [18] | 2012 | Measuring Productivity of Software Development Teams                               | Systematic Literature Review | Reviewing methods for measuring productivity of software development teams and identifying affecting factors                                                                   |

## Appendix B

# Interview Guide

This appendix presents the semi-structured interview guide used during the pilot interviews with software engineering managers. The interviews lasted approximately 45 minutes and were conducted prior to the survey to inform its design and contextualise the findings. The guide was structured to explore managers' definitions of productivity, assessment purposes, metric use and misuse, and the tensions between individual and team-level assessment. The semi-structured format allowed for flexibility to probe emerging themes while maintaining consistency across interviews.

**Duration:** ~45 minutes

### Introduction:

- Thank the participant for their participation and introduce the interviewer.
- Explain what the study is about and state the aim of the interview.
- Remind about recording and transcription, data anonymisation, withdrawal, and the expected duration.
- Inform that this is a semi-structured interview.
- Ask if they have any questions before starting.

### Main Questions:

1. What is productivity for you?
2. Do you assess productivity? Why? If you do, for what purposes?
  - Why do you assess productivity?
  - For what purposes is productivity assessed?

- Who uses this data?
3. Does your team know they are being assessed and how? Do you believe that influences their behaviour? (Goodhart's Law)
  4. Do you have any feedback from developers about how they perceive the metrics or productivity assessment?
  5. At which level and how do you assess productivity?
  6. Is there a difference for you between individual and team productivity?
    - What influences each?
    - How is each measured? (metrics)
    - How does that metric work for you?
    - Doesn't that lead to X for you?
    - What metrics are you satisfied/not satisfied with?
    - What are they used for?
    - Is there a conflict between them?
  7. What aspects of your current approach to assessing productivity work well for you?
  8. In what situations do you find productivity assessment or metrics particularly useful or reliable?
    - Do these differ when looking at individual vs. team productivity?
  9. Are there any shortcomings or limitations in how you currently assess productivity?
  10. Are there situations where productivity metrics are less useful or misleading?
    - Do these limitations differ when looking at individual vs. team productivity?
  11. What practices or strategies do you use to deal with these limitations?
  12. Are there any practices you consider good practices when assessing productivity?
  13. Does developer experience play a role in how you think about productivity?
    - How is developer experience incorporated into your current approach to measuring or discussing productivity?
    - Do you also try to collect data about developer experience?
    - How is developer experience connected to productivity?
  14. Is there anything you feel you forgot to mention that may be relevant?

# Appendix C

## Survey Structure

This appendix presents the full survey instrument administered to both managers and developers. The survey was developed based on findings from the pilot interviews and the literature review, and was designed to address the four research questions guiding this study. The survey includes role-based branching: some questions were presented only to managers, others only to developers, and some to both roles. The branching logic is indicated throughout the instrument.

### Survey Structure:

- **Section 1:** Background
- **Section 2:** Productivity
- **Section 3:** Management
- **Section 4:** Productivity Assessment
- **Section 5:** Individual vs Team Productivity
- **Section 6:** Metrics
- **Section 7:** Individual vs Team Productivity Trade-Offs

### Section 1: Background

1. Which description best fits your role?

I manage other software development professionals.

I develop software under a tech lead or manager role.

2. How many years of professional experience do you have?

3. How many people are in your immediate team?

1-5

6-10

11-15

16-20

20+

4. What is the work set up for collaboration with your team members?

On-site

Remote

Hybrid

5. What is the size of your company?

1-10 employees

11-50 employees

51-250 employees

250+ employees

## **Section 2: Productivity**

1. What do you consider as being productive? (Select all that apply)

Reaching business objectives.

Delivering business value.

Individual progress.

Delivering tasks on time.

Producing quality artifacts that need little rework.

Meeting stakeholder expectations.

Transforming ideas into software effectively.

Reducing developer effort while increasing output.

Solving complex problems.

Helping others on the team.

None of the above.

2. Please, rate how satisfied are you with your productivity and the productivity in your team in the last 3 months.

|                                                  | <b>Strongly Disagree</b> | <b>Disagree</b> | <b>Neutral</b> | <b>Agree</b> | <b>Strongly Agree</b> |
|--------------------------------------------------|--------------------------|-----------------|----------------|--------------|-----------------------|
| I am satisfied with my productivity.             |                          |                 |                |              |                       |
| I am satisfied with the productivity of my team. |                          |                 |                |              |                       |

**Section 3: Management***[Branch: Managers only]*

|                                                                                                       | <b>Strongly Disagree</b> | <b>Disagree</b> | <b>Neutral</b> | <b>Agree</b> | <b>Strongly Agree</b> |
|-------------------------------------------------------------------------------------------------------|--------------------------|-----------------|----------------|--------------|-----------------------|
| I assess productivity of individual developers.                                                       |                          |                 |                |              |                       |
| I assess productivity of the team as a whole.                                                         |                          |                 |                |              |                       |
| I assess how do the processes and tools for software development fit what my developers want or need. |                          |                 |                |              |                       |
| My team is aware of how I assess their productivity.                                                  |                          |                 |                |              |                       |
| I am aware of how my team defines productivity.                                                       |                          |                 |                |              |                       |
| My team and I have the same understanding of what it means to be productive.                          |                          |                 |                |              |                       |
| I inform my team of the purposes of their productivity assessment.                                    |                          |                 |                |              |                       |
| I review productivity expectations regularly with my team.                                            |                          |                 |                |              |                       |
| I review productivity expectations regularly with each developer.                                     |                          |                 |                |              |                       |
| I provide timely and actionable productivity feedback.                                                |                          |                 |                |              |                       |

*[Branch: Developers only]*

|                                                                                             | <b>Strongly<br/>Disagree</b> | <b>Disagree</b> | <b>Neutral</b> | <b>Agree</b> | <b>Strongly<br/>Agree</b> |
|---------------------------------------------------------------------------------------------|------------------------------|-----------------|----------------|--------------|---------------------------|
| My individual productivity is being assessed in my team.                                    |                              |                 |                |              |                           |
| Team productivity is being assessed in my team.                                             |                              |                 |                |              |                           |
| The processes and tools for software development I want or need are assessed by my manager. |                              |                 |                |              |                           |
| I am aware of how my manager assesses productivity.                                         |                              |                 |                |              |                           |
| My manager and I share the same definition of productivity.                                 |                              |                 |                |              |                           |
| I am aware of why my productivity is being assessed.                                        |                              |                 |                |              |                           |
| My manager and I review productivity expectations regularly.                                |                              |                 |                |              |                           |
| My manager's productivity feedback is timely and actionable.                                |                              |                 |                |              |                           |

#### **Section 4: Productivity Assessment**

Help us understand your experience with assessing productivity in your team by rating the following statements.

*[Branch: Managers only]*

|                                                                                                                        | <b>Strongly<br/>Disagree</b> | <b>Disagree</b> | <b>Neutral</b> | <b>Agree</b> | <b>Strongly<br/>Agree</b> |
|------------------------------------------------------------------------------------------------------------------------|------------------------------|-----------------|----------------|--------------|---------------------------|
| Developers' productivity should be assessed differently depending on their responsibilities and working style.         |                              |                 |                |              |                           |
| Productivity assessment should be quantitative.                                                                        |                              |                 |                |              |                           |
| Productivity metrics must be contextualized.                                                                           |                              |                 |                |              |                           |
| Providing productivity metrics explicitly incentivizes developers to gamify metrics.                                   |                              |                 |                |              |                           |
| My assessment of productivity has been misinterpreted in the past.                                                     |                              |                 |                |              |                           |
| It is difficult for me to know what is fair productivity assessment.                                                   |                              |                 |                |              |                           |
| Developers' experience with the processes and tools play an important role in their productivity.                      |                              |                 |                |              |                           |
| Productivity is hard to assess.                                                                                        |                              |                 |                |              |                           |
| Productivity is assessed to increase the profit of an organization.                                                    |                              |                 |                |              |                           |
| Productivity is assessed to increase effectiveness of individual developers and teams.                                 |                              |                 |                |              |                           |
| When assessing productivity of individual developers, I take the team context into consideration.                      |                              |                 |                |              |                           |
| When assessing productivity of the team, I take in context the situation of the individual team members at the moment. |                              |                 |                |              |                           |

*[Branch: Developers only]*

|                                                                                                                                    | <b>Strongly<br/>Disagree</b> | <b>Disagree</b> | <b>Neutral</b> | <b>Agree</b> | <b>Strongly<br/>Agree</b> |
|------------------------------------------------------------------------------------------------------------------------------------|------------------------------|-----------------|----------------|--------------|---------------------------|
| Developers' productivity should be assessed differently depending on their responsibilities and working style.                     |                              |                 |                |              |                           |
| Productivity assessment should be quantitative.                                                                                    |                              |                 |                |              |                           |
| Productivity metrics must be contextualized.                                                                                       |                              |                 |                |              |                           |
| Providing productivity metrics explicitly incentivizes developers to gamify metrics.                                               |                              |                 |                |              |                           |
| My manager misuses productivity metrics.                                                                                           |                              |                 |                |              |                           |
| I consider my manager's assessment of my productivity fair.                                                                        |                              |                 |                |              |                           |
| The tools and processes we follow strongly affect how much I can be productive.                                                    |                              |                 |                |              |                           |
| Productivity is hard to assess.                                                                                                    |                              |                 |                |              |                           |
| Productivity is assessed to increase the profit of an organization.                                                                |                              |                 |                |              |                           |
| When assessing productivity of individual developers, my manager takes the team context into consideration.                        |                              |                 |                |              |                           |
| When assessing productivity of the team, my manager takes into context the situation of the individual team members at the moment. |                              |                 |                |              |                           |
| I try to improve my performance in the criteria my manager uses for productivity assessment.                                       |                              |                 |                |              |                           |

### Section 5: Individual vs Team Productivity

This section asks about your opinion regarding the relationship between individual and team productivity.

|                                                                                          | <b>Strongly Disagree</b> | <b>Disagree</b> | <b>Neutral</b> | <b>Agree</b> | <b>Strongly Agree</b> |
|------------------------------------------------------------------------------------------|--------------------------|-----------------|----------------|--------------|-----------------------|
| Team productivity is more important than individual productivity.                        |                          |                 |                |              |                       |
| Team productivity and individual productivity should be assessed for different purposes. |                          |                 |                |              |                       |
| A group of productive individuals makes a productive team.                               |                          |                 |                |              |                       |
| There is a conflict between productivity of individuals and of the whole team.           |                          |                 |                |              |                       |
| It is harder to assess individual productivity over team productivity.                   |                          |                 |                |              |                       |

### Section 6: Metrics

In the last section, we would like to understand how do you work with different types of metrics. Please refer to the following list if the type names are unclear to you.

- **Satisfaction and Wellbeing** - Metrics that assess how developers feel about their work, their motivation, and their work environment. (e.g. job enthusiasm, perceived productivity)
- **Communication and Collaboration** - Metrics that capture how effectively team members help each other, share information, and coordinate work. (e.g. knowledge sharing sessions, team responsiveness, number of times helping others)
- **Output & Efficiency** - Metrics that compare what is produced (output) against the effort or time spent (input). (e.g. features delivered per sprint, story points completed per time unit, lines of code)
- **Quality** - Metrics that assess how well software works, how much rework is needed, and whether it meets standards. (e.g. rework effort, bug recurrence rate, test pass rate)
- **Value & Outcomes** - Metrics that assess whether delivered software meets stakeholder expectations and creates business value. (e.g. customer satisfaction, business objectives met, feature adoption rate)

- **Learning & Growth** - Metrics that capture how developers improve their skills, learn new technologies, and evolve in their roles. (e.g. mentoring hours, learning goals achieved, role progression)
  - **Technical Debt & Maintainability** - Metrics that assess code quality, and the cost of future changes. (e.g. technical debt ratio, code complexity, test coverage, number of code smells)
  - **Developer Experience** - Metrics that assess how well tools, documentation, and processes support developers in their daily work. (e.g. tool satisfaction, documentation quality, onboarding time)
  - **Process & Predictability** - Metrics that assess how well teams plan, estimate, and deliver according to commitments. (e.g. estimation accuracy, on-time delivery rate, planned vs actual, predictability index)
1. For each metric category below, indicate whether you consider metrics in this category to be useful for assessing productivity.

| <b>Metric Category</b>           | <b>Not Useful At All</b> | <b>Not Very Useful</b> | <b>Neutral</b> | <b>Somewhat Useful</b> | <b>Very Useful</b> |
|----------------------------------|--------------------------|------------------------|----------------|------------------------|--------------------|
| Satisfaction and Wellbeing       |                          |                        |                |                        |                    |
| Communication and Collaboration  |                          |                        |                |                        |                    |
| Output & Efficiency              |                          |                        |                |                        |                    |
| Quality                          |                          |                        |                |                        |                    |
| Value & Outcomes                 |                          |                        |                |                        |                    |
| Learning & Growth                |                          |                        |                |                        |                    |
| Technical Debt & Maintainability |                          |                        |                |                        |                    |
| Developer Experience             |                          |                        |                |                        |                    |
| Process & Predictability         |                          |                        |                |                        |                    |

2. *[Branch: Managers only]* For each metric category below, indicate which you use to assess productivity.

Satisfaction and Wellbeing

Communication and Collaboration

Output & Efficiency

Quality

Value & Outcomes

- Learning & Growth
- Technical Debt & Maintainability
- Developer Experience
- Process & Predictability
- None of the above

### Section 7: Individual vs Team Productivity Trade-Offs

Please, help us understand which scenario would you and your team consider more productive.

1. **A)** A developer works independently, avoids interruptions, and consistently delivers their assigned tasks ahead of schedule, but rarely engages in team discussions or collaborative problem-solving.  
**B)** A developer frequently pairs with teammates, joins discussions, and helps unblock others, but their individual task output is slower as a result.

|                                                                                                                  | Scenario A | Scenario B |
|------------------------------------------------------------------------------------------------------------------|------------|------------|
| Which behaviour do you think is more productive?                                                                 |            |            |
| <i>[Branch: Managers only]</i> Which behaviour do you think is more productive from your developers perspective? |            |            |
| <i>[Branch: Developers only]</i> Which behaviour do you think is more productive from your manager perspective?  |            |            |

2. **A)** A developer skips writing detailed documentation and keeps code review comments brief so they can move on to the next task quickly, maintaining a high personal throughput.  
**B)** A developer writes thorough documentation and provides detailed code review feedback, which helps the whole team but reduces the number of tasks they personally complete.

|                                                                                                                  | Scenario A | Scenario B |
|------------------------------------------------------------------------------------------------------------------|------------|------------|
| Which behaviour do you think is more productive?                                                                 |            |            |
| <i>[Branch: Managers only]</i> Which behaviour do you think is more productive from your developers perspective? |            |            |
| <i>[Branch: Developers only]</i> Which behaviour do you think is more productive from your manager perspective?  |            |            |

3. **A)** A developer uses personal scripts, shortcuts, and local tooling that significantly speed up their own workflow, but that teammates can't easily replicate or maintain.  
**B)** A developer invests time in setting up shared tooling and standardized workflows that benefit the whole team, even though the setup takes longer and constrains their own preferences.

|                                                                                                                  | Scenario A | Scenario B |
|------------------------------------------------------------------------------------------------------------------|------------|------------|
| Which behaviour do you think is more productive?                                                                 |            |            |
| <i>[Branch: Managers only]</i> Which behaviour do you think is more productive from your developers perspective? |            |            |
| <i>[Branch: Developers only]</i> Which behaviour do you think is more productive from your manager perspective?  |            |            |

4. **A)** A developer delivers their part of a feature on time by focusing solely on their own scope, even if other parts of the team are struggling and the overall release is at risk.  
**B)** A developer delays finishing their own scope to help struggling teammates, risking their personal deadline but improving the team's chances of a successful, complete release.

|                                                                                                                  | Scenario A | Scenario B |
|------------------------------------------------------------------------------------------------------------------|------------|------------|
| Which behaviour do you think is more productive?                                                                 |            |            |
| <i>[Branch: Managers only]</i> Which behaviour do you think is more productive from your developers perspective? |            |            |
| <i>[Branch: Developers only]</i> Which behaviour do you think is more productive from your manager perspective?  |            |            |

5. **A)** A developer declines mentoring or onboarding responsibilities to remain fully focused on delivering features, keeping their output metrics high.  
**B)** A developer regularly invests time in mentoring junior teammates and supporting onboarding, accepting a reduction in their own delivery speed.

|                                                                                                                  | Scenario A | Scenario B |
|------------------------------------------------------------------------------------------------------------------|------------|------------|
| Which behaviour do you think is more productive?                                                                 |            |            |
| <i>[Branch: Managers only]</i> Which behaviour do you think is more productive from your developers perspective? |            |            |
| <i>[Branch: Developers only]</i> Which behaviour do you think is more productive from your manager perspective?  |            |            |

6. **A)** A developer deepens expertise in their specific domain, becoming highly efficient in their area but creates a dependency that slows the team when they are unavailable.  
**B)** A developer intentionally broadens their skills across the team's stack to reduce dependencies, even though it means being less of an expert in any one area.

|                                                                                                                  | Scenario A | Scenario B |
|------------------------------------------------------------------------------------------------------------------|------------|------------|
| Which behaviour do you think is more productive?                                                                 |            |            |
| <i>[Branch: Managers only]</i> Which behaviour do you think is more productive from your developers perspective? |            |            |
| <i>[Branch: Developers only]</i> Which behaviour do you think is more productive from your manager perspective?  |            |            |

7. **A)** A developer quickly implements solutions based on their own expertise without spending time explaining decisions to the team.  
**B)** A developer spends time explaining design decisions and sharing knowledge with teammates, even if it slows down implementation.

|                                                                                                                  | Scenario A | Scenario B |
|------------------------------------------------------------------------------------------------------------------|------------|------------|
| Which behaviour do you think is more productive?                                                                 |            |            |
| <i>[Branch: Managers only]</i> Which behaviour do you think is more productive from your developers perspective? |            |            |
| <i>[Branch: Developers only]</i> Which behaviour do you think is more productive from your manager perspective?  |            |            |

8. **A)** A developer focuses on delivering features quickly, postponing refactoring and technical improvements.  
**B)** A developer spends time refactoring and improving the codebase, which slows feature delivery but improves maintainability.

|                                                                                                                  | Scenario A | Scenario B |
|------------------------------------------------------------------------------------------------------------------|------------|------------|
| Which behaviour do you think is more productive?                                                                 |            |            |
| <i>[Branch: Managers only]</i> Which behaviour do you think is more productive from your developers perspective? |            |            |
| <i>[Branch: Developers only]</i> Which behaviour do you think is more productive from your manager perspective?  |            |            |

9. **A)** A developer prefers asynchronous communication and minimizes meetings to maximize uninterrupted coding time.
- B)** A developer participates actively in alignment meetings and planning sessions to ensure the team stays coordinated.

|                                                                                                                  | Scenario A | Scenario B |
|------------------------------------------------------------------------------------------------------------------|------------|------------|
| Which behaviour do you think is more productive?                                                                 |            |            |
| <i>[Branch: Managers only]</i> Which behaviour do you think is more productive from your developers perspective? |            |            |
| <i>[Branch: Developers only]</i> Which behaviour do you think is more productive from your manager perspective?  |            |            |

10. **A)** When encountering a complex issue, a developer solves it independently to avoid slowing down others.
- B)** When encountering a complex issue, a developer brings it to the team to discuss possible solutions together.

|                                                                                                                  | Scenario A | Scenario B |
|------------------------------------------------------------------------------------------------------------------|------------|------------|
| Which behaviour do you think is more productive?                                                                 |            |            |
| <i>[Branch: Managers only]</i> Which behaviour do you think is more productive from your developers perspective? |            |            |
| <i>[Branch: Developers only]</i> Which behaviour do you think is more productive from your manager perspective?  |            |            |

11. **A)** A developer blocks off large chunks of their calendar for focused coding, becoming temporarily unreachable to teammates but delivering complex features faster.
- B)** A developer keeps their calendar flexible and remains constantly available to answer questions and unblock teammates, even if it fragments their own focus.

|                                                                                                                  | Scenario A | Scenario B |
|------------------------------------------------------------------------------------------------------------------|------------|------------|
| Which behaviour do you think is more productive?                                                                 |            |            |
| <i>[Branch: Managers only]</i> Which behaviour do you think is more productive from your developers perspective? |            |            |
| <i>[Branch: Developers only]</i> Which behaviour do you think is more productive from your manager perspective?  |            |            |

12. **A)** A developer writes minimal tests to deliver features faster, accepting that bugs might surface later and require team effort to fix.
- B)** A developer writes comprehensive tests that slow initial delivery but reduce bugs and maintenance burden for the whole team.

|                                                                                                                  | Scenario A | Scenario B |
|------------------------------------------------------------------------------------------------------------------|------------|------------|
| Which behaviour do you think is more productive?                                                                 |            |            |
| <i>[Branch: Managers only]</i> Which behaviour do you think is more productive from your developers perspective? |            |            |
| <i>[Branch: Developers only]</i> Which behaviour do you think is more productive from your manager perspective?  |            |            |

13. **A)** A developer provides optimistic estimates to please stakeholders and get projects approved, knowing the team will need to absorb the overrun later.
- B)** A developer pushes for realistic, buffer-inclusive estimates that protect the team from burnout, even when stakeholders pressure for faster delivery.

|                                                                                                                  | Scenario A | Scenario B |
|------------------------------------------------------------------------------------------------------------------|------------|------------|
| Which behaviour do you think is more productive?                                                                 |            |            |
| <i>[Branch: Managers only]</i> Which behaviour do you think is more productive from your developers perspective? |            |            |
| <i>[Branch: Developers only]</i> Which behaviour do you think is more productive from your manager perspective?  |            |            |

14. **A)** A developer avoids giving constructive feedback to teammates to preserve harmony and avoid uncomfortable conversations, maintaining pleasant relationships.
- B)** A developer regularly provides honest, constructive feedback to help teammates improve, even when it creates temporary tension or discomfort.

|                                                                                                                  | Scenario A | Scenario B |
|------------------------------------------------------------------------------------------------------------------|------------|------------|
| Which behaviour do you think is more productive?                                                                 |            |            |
| <i>[Branch: Managers only]</i> Which behaviour do you think is more productive from your developers perspective? |            |            |
| <i>[Branch: Developers only]</i> Which behaviour do you think is more productive from your manager perspective?  |            |            |