POLITECNICO DI TORINO
Repository ISTITUZIONALE

How to Measure Game Testing: a Survey of Coverage Metrics

Original
How to Measure Game Testing: a Survey of Coverage Metrics / Coppola, R., Fulcini, T., Manzi, S., Strada, F.. - ELETTRONICO. - (2024), pp. 15-19. (GAS '24: ACM/IEEE 8th International Workshop on Games and Software Engineering Lisbon (PT) 14 April 2024) [10.1145/3643658.3643920].

Availability:
This version is available at: 11583/2986572 since: 2024-09-26T12:48:09Z

Publisher:
ACM/IEEE

Published
DOI:10.1145/3643658.3643920

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

11 September 2026

How to Measure Game Testing: a Survey of Coverage Metrics

Riccardo Coppola, Tommaso Fulcini, Serenella Manzi, Francesco Strada
first.last@polito.it
Politecnico di Torino
Turin, Italy

ABSTRACT

As the complexity of video games continues to evolve, so does the importance of effective game testing methodologies. To this end, automated game testing has emerged as a pivotal approach to ensure the quality and functionality of modern games. This paper presents a comprehensive survey of metrics utilized in automated game testing, aiming to provide a deeper understanding of the parameters used to evaluate the robustness and performance of video game testing activities. Through the conduction of a Systematic Literature Review and the Open and Axial coding approaches, the results of the study provide a taxonomy of 26 different metrics for video game testing assessment, grouped in six higher-level categories. The elicited set of metrics can serve as a tool for game testers, researchers and tool developers to evaluate testing approaches and techniques and enable comparability of research results.

CCS CONCEPTS

• **Software and its engineering → Software testing and debugging**; **Formal software verification**.

KEYWORDS

Videogame, Testing, Automation, Systematic Literature Review, Software Metrics

ACM Reference Format:
Riccardo Coppola, Tommaso Fulcini, Serenella Manzi, Francesco Strada. 2024. How to Measure Game Testing: a Survey of Coverage Metrics. In *2024 ACM/IEEE 8th International Workshop on Games and Software Engineering (GAS '24), April 14, 2024, Lisbon, Portugal*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3643658.3643920

1 INTRODUCTION

During the last decades, the game industry has become one of the most profitable of the entire software environment, reaching – according to recent statistics – a revenue of US $249.60bn in 2023 [1]. Modern games have reached enormous budgets and team sizes and may require several years to be eventually released. At the same time, games are well-known for being frequently plagued by day-one bugs, in some cases in a quantity sufficient to completely hinder the economic success of the release and seriously impact the credibility of renowned software houses [30].

[1] https://bit.ly/472DrQZ

This work is licensed under a Creative Commons Attribution International 4.0 License.
GAS '24 , April 14, 2024, Lisbon, Portugal
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0561-8/24/04
https://doi.org/10.1145/3643658.3643920

Software testing is a vast collection of disciplines finalized at the verification and validation of software artefacts of varying nature and to be deployed on different platforms. A plethora of methodologies to design, execute and measure software testing have been defined by Software Engineering literature, and efforts have been performed in several domains to systematize the measures that can be leveraged to evaluate the quality of a testing process [10]. The automation of these processes is considered a fundamental practice to ensure software quality in any domain.

As highlighted by many secondary studies in the literature, there is still little evidence about the application of automated testing techniques to video games, and no tentative systematization has been provided for coverage metrics used to assess testing activities quantitatively. The definition of coverage models is of fundamental importance in research to enable evaluation of existing techniques and tools and to provide comparable results between studies in the sector.

The present paper aims to categorize, in the form of a taxonomy, the coverage metrics proposed or used in game testing literature. The results of the present study are the preliminary outcomes of a larger-scale Systematic Literature Review covering various facets of the software testing process when applied to games.

The remainder of the manuscript is organized as follows: Section 2 reports a background about the techniques and tools described in current game testing literature; Section 3 briefly describes the literature review process employed for the collection of primary sources and the definition of the taxonomy; Section 4 reports the taxonomy and provides pointers to existing literature for each of the coverage metric defined; Section 5 concludes the paper and provides future research directions.

2 BACKGROUND AND RELATED WORK

Due to the relative novelty of the field of automated and systematic game testing, few studies are available in the literature to systematize the methodologies, tools and means of evaluations of the test activities. In the present section, we report and discuss the main secondary studies that are available in the literature.

Albaghajati and Ahmed have provided an assessment framework for the available approaches, objectives, and means of validation of the testing activities [2]. The authors identify several techniques to perform testing automation for games: (i) search-based approaches, focusing on exploring and analyzing the state space of a game; (ii) goal-directed approaches, utilizing automated agents exploring the games with defined policies, rewards and policies, to explore potential paths through the games; (iii) human-like approaches, focused on imitating the human behaviour, where the agents are optimized to produce results similar to those of schematized human players (i.e., *personas*); (iv) scenario-based approaches, running test sequences based on predefined human-made sequences of actions

or human-requested actions; (v) model-based approaches, which abstract the game's workflow into formal representation and models allowing verification of flow of events or control. Regarding the main objectives of game testing, the authors identify functional correctness (i.e., that the game is behaving as expected according to its requirements), multiplayer stability, performance, visual correctness, game design correctness, game balance and fairness, progression and learnability, and physical correctness. The authors finally discuss the lack of generally applicable validation procedures available in the literature, highlighting the limited applicability of many of the testing approaches to individual domains or game genres and thus encouraging research efforts geared towards higher generalizability of testing activities.

Politowski et al. analyzed the tools for game test automation available in the literature, comparing the evidence from the literature with those collected through a survey with 12 respondents among developers, all having experiences in developing and testing traditional software, and 58% of which with more than four years experience in game development and working full-time in a game company [24]. Their results from the literature survey highlight that most of the testing tools and frameworks available provide and discuss results that are not related to the advancement in functional and user-oriented testing of video games, but are instead more concerned with evaluating the applicability of AI-based approaches to game modelling and exploration. They also underline a general scepticism of the interviewed game developers about using automated agents to test games. They conclude that there is still the need for significant academic and industrial efforts to expand the adoption of testing in video game development processes. In a less recent survey from 2021 [25], Politowski et al. gathered insights from white and grey literature about game testing, concluding that game developers relied almost exclusively upon manual play-testing and the testers' intrinsic knowledge, leading to a widespread lack of automation. Among the main motivations for such scarce evidence of the application of automated testing, the authors identified the low generalizability of available techniques and the need for testing strategies that take into account the particularities of game projects, more than it happens for traditional software in other domains.

Our analysis of existing literature, therefore, encourages the research in methods to evaluate and assess game-testing approaches, justifying a systematic study that can help find common strategies that can be generalized to multiple testing tools or methodologies.

3 METHODOLOGY

The objective of the work is to identify, through a literature review, the most commonly used coverage metrics adopted in game testing and to define a taxonomy for the game testing community.

The taxonomy shall guide testers to identify which metrics are used in research and practice and provide researchers with an instrument to compare different game-testing approaches.

The methodology used in this work can be separated into two steps: (i) literature review, and (ii) formulation of the taxonomy through Open Coding.

3.1 Literature Review

We performed a targeted literature review by applying a search string on a set of scientific literature repositories. To that extent, we applied a subset of Kitchenham's guidelines to conduct Systematic Literature Reviews [15]. As in the guidelines by Garousi et al. [14], we also included relevant grey literature in our search results. The results presented in this manuscript do not include the results of quality assessment, or forward or backward snowballing after the first collection of papers.

Selected Digital Libraries As digital libraries for our search, we selected IEEE Xplore, ACM Digital Library, Science Direct, Springer Link, Google Scholar, and Google Search.

Search String We formulated our search string to include the terms *Game* (or gaming), test*, coverage (or metric*)*. The search string was adapted to fit the syntaxes of the five selected digital libraries.

Inclusion Criteria To gather only sources relevant to our research goals, we defined the following criteria: (i) the source is directly related to the topic of game testing; (ii) the source defines or adopts explicitly metrics or measures for the test execution; (iii) the source is an item of white literature with available full text, or a publicly available Grey Literature item, with publication date between 2012 and 2023; (iv) the source is written in a language that is directly comprehensible by the authors (English or Italian).

3.2 Coding

To define a taxonomy of coverage metrics for game testing, we applied Ralph's guidelines for the construction of taxonomies by following the Grounded Theory approach [26]. We adopted the Straussian definition of the Grounded Theory approach, by utilizing an up-front definition of our research question [32].

The codes for the taxonomy were defined by applying the *Open Coding* technique, i.e., analyzing text data to capture the information of the theory under construction. Open Coding allowed to define the lower-level categories, or codes, of the taxonomy. We therefore formulated a set of common definitions to which we assigned the individual metrics defined or used in the literature sources. The categories are considered mutually exclusive (i.e., one metric in the literature sources can be assigned to only one code).

The taxonomy is built incrementally, adding new codes every time a new metric definition or usage in the reviewed literature did not fit the available pool of codes.

After applying the Open Coding procedure, Axial Coding was applied to build categories of codes. As defined in the Straussian Grounded Theory, Axial Coding is the process of understanding how codes and related concepts are linked to one another, to identify a structure in the taxonomy and define levels in it. Axial Coding was applied by performing two passes (by all the authors) over all the codes of the taxonomy and defining themes (i.e., higher-level categories) of metrics.

4 RESULTS

The application of the search string led to the collection of 65 sources; the number was reduced to 25 (22 white and 3 grey literature sources) after duplicate removal and application of the inclusion criteria. For the formulation of metrics and measures for game testing, we did not include coverage definitions that are typical of

unit testing (e.g., line, branch, path coverage). The rationale behind the exclusion is that these metrics are horizontally applicable to any software domain and therefore can be implicitly computed for every testing procedure applied to the source code of games.

By analyzing the manuscripts, and the application of the coding procedure, 26 codes (i.e., metric definitions) were defined. All the metrics definitions are reported in Table 1.

The application of axial coding led to the identification of 6 categories of metrics. We describe below the categories of metrics found:

- **UI** : metrics related to the capability of the test cases to efficiently cover the user interface of game applications. The application of such metrics is limited to the games providing an actual Graphical User Interface to the users that can be decomposed into atomic elements (widgets, screens or pages). The UI category is not specific to videogame testing but can be applied to any software provided with a Graphical User Interface.
- **Gameplay** : metrics related to the coverage of the core functionality of the game applications and the verification of its gameplay characteristics. These metrics are tightly related to the structure of the gameplay (e.g., its subdivision in levels, or the possible gameplay paths that can be performed by the agents), and the genre of the gameplay (e.g., the presence of possible hazardous elements or challenges faced by the players).
- **Multimedia** : metrics related to the coverage of different multimedia elements of the game applications, regarding the rendering of sound, video, and application of physics. Because of their nature, these metrics constitute a set of verification measures that can be applied to any category of game applications.
- **Operability and UX** : metrics related to the evaluation of the experience perceived by the users when playing the games. When the game testing activities are automated, these metrics are typically evaluated as the result of the utilization of automated agents performing gameplay sessions by mimicking different types of playstyles (i.e., *user personas*).
- **Performance** : metrics related to the quantitative assessment of performance measures while the test sequences are executed against the game application. Performance measures are declined in literature into several different performance aspects. The performance category is not specific to videogame testing but can be applied to any kind of resource-intensive or multimedia software.
- **Reliability** : metrics related to the measurement of the presence of issues during the execution of the game. Issue tracking is a widespread activity in the test of applications in any domain, and is typically decomposed into several separate metrics according to the level of severity of the encountered issues during the test sequences. The reliability category is not specific to videogame testing but can be applied to any software domain.

5 DISCUSSION

In this preliminary analysis of metrics used to assess game testing, we identified six main categories of measurements common to several studies in the literature of the field.

Albeit the study is preliminary in nature and we foresee the extension of the current set of metrics (and categories) through the inclusion of additional white and grey literature items through backward and forward snowballing, the coded taxonomy can already be helpful in the identification of existing trends in automated game testing.

Firstly, as opposed to traditional application testing, automated test activities for games are hardly based on oracles and on predetermined outputs to be verified. Our literature led us to conclude that most of the conducted testing is based on the utilization of *implicit* oracles, i.e., there is no explicit pass or fail condition for test cases except for the possible verification of bugs or crashes at run time.

A second important emerging aspect, also highlighted by the existing secondary study, is that the state of the art of game testing is exploratory in nature. Automated game testing activities largely privilege the utilization of autonomous agents performing explorations according to heuristics (e.g., random exploration, or persona-based exploration) rather than following pre-defined test scripts and sequences. This inherently exploratory nature of game testing is reflected by many of the metrics that were found in the literature, especially the gameplay and operability metrics related to the provided coverage for available in-game choices and playstyles.

Finally, a crucial aspect of game testing activities is the strong dependency of the used methodologies – and, by consequence, of the used measurements – on the platform and genre of the tested game. It is therefore unlikely to devise a fully generalizable taxonomy of metrics that are applicable to any type of game available in the literature. Of the three taxonomy categories that we derived, only *Performance*, *Reliability* and *Multimedia* metrics have no generalizability limitations.

6 CONCLUSION AND FUTURE WORK

In this paper, a taxonomy of 26 metrics for the assessment of game testing activities has been defined. The definition of the taxonomy entailed the utilization of formal Software Engineering methodologies, i.e. Systematic Literature Reviews and Straussian Grounded Theory.

We deem the definition of a comprehensive taxonomy of research metrics a relevant contribution to literature in the field. The availability of taxonomies for the evaluation of software engineering activities can help increase the comparability of research efforts, and reduce the inconsistencies between studies tackling similar aspects using different vocabularies. Our results can therefore serve as aid for researchers, practitioners and testing tool developers to adopt a common language for the evaluation of testing efforts performed with different approaches.

As our future work, we foresee the completion of the Systematic Literature Review by exploring other dimensions of game testing (e.g., the type of oracles and datasets for testing if any, the types of personas covered by test agents). We also plan to provide a mapping study to identify which metrics are covered by existing tools (either provided by academic literature or by the industry). Finally, we envision the execution of empirical comparative studies to evaluate

Table 1: Definitions of the collected metrics

Category	Metric	Definition	Refs
Functionality - UI	Screen Coverage	Number of covered screens of the game over total number of screens	[22] [16]
	Widget Coverage	Number of covered widgets in the game over total number of widgets	[22] [16]
Gameplay	Interactable objects coverage	Interactable objects triggered by test cases over the total number	[29] [35] [6] [31]
	Interactable NPC coverage	Interactable NPC (Non-Playing Characters) triggered by test cases over the total number	[31]
	Player statistics coverage	Player statistics exercised or modified by the test cases over the total number of player statistics available	[8] [20]
	Level Exploration	Explored percentage of levels in terms of a specified in-game measure (e.g., meters or square meters)	[29] [19]
	Level Coverage	Number of levels completed over the total number of available levels in the game	[5]
	Path coverage	Number of paths covered in a single level over the total number of possible paths in the level	[5]
	Plots coverage	Number of tested dynamic plots (depending on the players' choices) over the total number of available plots	[34]
	Enemies coverage	Number of interacted enemies over the total number of enemies present in a level	[31]
	Environment hazard coverage	Number of the triggered environment hazards over the total number of hazards present in a level	[31]
Multimedia	Animation coverage	Number of tested animations over the total number of animations present in a level/scene	[22]
	Sound effects coverage	Number of played sounds over the total number of sounds present in a level/scene	[22]
	Speech/dialogues coverage	Number of played speeches or dialogues over the total number of speeches and dialogues present in a level/scene	[22]
Operability and UX	Difficulty	Evaluation of the difficulty of a level or scene, in terms of time to complete the level, number of failed attempts, size of the level, etc.	[17], [3]
	Number of attempts	Minimum, maximum, average number of attempts required by test agents to complete a level	[29]
	Time to complete a level	Minimum, maximum, average amount of time necessary to complete a level	[29]
	Playstyle coverage	Number of different playstyles (or "personas") applied by automated agents over the total number of playstyles available	[6][31][12]
	Fun factor value	Measurement of the player's enjoyment of the tested levels or scenes (in terms of arousal, stress, boredom, etc)	[4][13]
Performance	Memory Usage	Usage of memory during the execution of the test cases	[23]
	FPS value	Maximum, minimum or average FPS (frames per second) rate during the execution of the test cases	[18] [9] [36]
	CPU usage	CPU usage, measured in time needed to render a frame or in power consumption per frame	[11] [21] [1]
	GPU usage	GPU usage measured in time needed to render a frame or in power consumption per frame	[11] [21] [1]
	Battery usage	Consumption of battery during the execution of the test cases	[1]
Reliability	Number of bugs	Number of bugs discovered during the execution of test cases. A bug is a minor functional, behavioural or graphical issue in the execution of the game, not resulting to an unexpected closure of the software.	[23] [7] [33] [27]
	Number of crashes	Number of crashes triggered during the execution of the test cases. A crash is a critical issue in the execution of the game, resulting to an unexpected closure of the software.	[23]

different game testing methodologies and approaches based on a selection of metrics of our taxonomy.

The conducted analysis of the primary studies also allowed to identify several current research gaps in the game testing research field. The main gaps that we identify are the following: (i) the papers have a focus in bug finding but are less centered around UX; (ii) most of the testing activities are conducted at system level with little to no application of unit and integration testing techniques; (iii) some multimedial aspects, e.g. audio testing, are less object of verification than others. Researchers in the field may find these gaps as starting points for future research.

ACKNOWLEDGEMENT

This study was carried out within the "EndGame - Improving End-to-End Testing of Web and Mobile Apps through Gamification" project (2022PCCMLF) – funded by European Union – Next Generation EU within the PRIN 2022 program (D.D.104 - 02/02/2022 Ministero dell'Università e della Ricerca). This manuscript reflects only the authors' views and opinions and the Ministry cannot be considered responsible for them. The authors thank the support provided by the student Mattia Riola with the execution of the literature collection steps, conducted as part of his M.Sc. thesis [28].

REFERENCES

[1] [n. d.]. Game Performance Metrics that Matter: Guide to Interpretation and Action. https://blog.gamebench.net/game-performance-metrics-that-matter. Accessed: 2023-12-04.

[2] Aghyad Mohammad Albaghajati and Moataz Aly Kamaleldin Ahmed. 2020. Video game automated testing approaches: An assessment framework. *IEEE transactions on games* (2020).

[3] Gustavo Andrade, Geber Ramalho, Hugo Santana, and Vincent Corruble. 2005. Automatic computer game balancing: a reinforcement learning approach. In *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*. 1111–1112.

[4] Saba Gholizadeh Ansari, ISWB Prasetya, Mehdi Dastani, Frank Dignum, and Gabriele Keller. 2021. An Appraisal Transition System for Event-Driven Emotions in Agent-Based Player Experience Testing. In *International Workshop on Engineering Multi-Agent Systems*. Springer, 156–174.

[5] Sinan Arıyürek. 2022. AUTOMATED VIDEO GAME TESTING USING REINFORCEMENT LEARNING AGENTS. (2022).

[6] Oleguer Canal Anton. 2021. Automatic game-testing with personality: Multi-task reinforcement learning for automatic game-testing.

[7] Rodrigo Casamayor, Lorena Arcega, Francisca Pérez, and Carlos Cetina. 2022. Bug localization in game software engineering: evolving simulations to locate bugs in software models of video games. In *Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems*. 356–366.

[8] Sven Charleer, Francisco Gutiérrez, Kathrin Gerling, and Katrien Verbert. 2018. Towards an open standard for gameplay metrics. In *Proceedings of the 2018 Annual Symposium on Computer-Human Interaction in Play Companion Extended Abstracts*. 399–406.

[9] Mark Claypool and Kajal Claypool. 2009. Perspectives, frame rates and resolutions: it's all in the game. In *Proceedings of the 4th International Conference on Foundations of Digital Games*. 42–49.

[10] Riccardo Coppola and Emil Alégroth. 2022. A taxonomy of metrics for GUI-based testing research: A systematic literature review. *Information and Software Technology* (2022), 107062.

[11] Benedikt Dietich, Nadja Peters, Sangyoung Park, and Samarjit Chakraborty. 2017. Estimating the limits of CPU power management for mobile games. In *2017 IEEE International Conference on Computer Design (ICCD)*. IEEE, 1–8.

[12] Pedro M Fernandes, Jonathan Jørgensen, and Niels NTG Poldervaart. 2021. Adapting procedural content generation to player personas through evolution. In *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 01–09.

[13] Pedro M Fernandes, Manuel Lopes, and Rui Prada. 2021. Agents for automated user experience testing. In *2021 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*. IEEE, 247–253.

[14] Vahid Garousi, Michael Felderer, and Mika V Mäntylä. 2016. The need for multivocal literature reviews in software engineering: complementing systematic literature reviews with grey literature. In *Proceedings of the 20th international conference on evaluation and assessment in software engineering*. 1–6.

[15] Barbara A Kitchenham. 2012. Systematic review in software engineering: where we are and where we should be going. In *Proceedings of the 2nd international workshop on Evidential assessment of software technologies*. 1–2.

[16] Xudong Li, Dajun Zhou, Like Zhang, and Yanqing Jing. 2020. Human-like UI Automation through Automatic Exploration. In *Proceedings of the 2020 2nd International Conference on Big Data and Artificial Intelligence*. 47–53.

[17] Simon Liu, Li Chaoran, Li Yue, Ma Heng, Hou Xiao, Shen Yiming, Wang Licong, Chen Ze, Guo Xianghao, Lu Hengtong, et al. 2019. Automatic generation of tower defense levels using PCG. In *Proceedings of the 14th International Conference on the Foundations of Digital Games*. 1–9.

[18] Shengmei Liu, Atsuo Kuwahara, James J Scovell, and Mark Claypool. 2023. The Effects of Frame Rate Variation on Game Player Quality of Experience. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. 1–10.

[19] Cong Lu, Raluca Georgescu, and Johan Verwey. 2022. Go-Explore Complex 3D Game Environments for Automated Reachability Testing. *IEEE Transactions on Games* (2022).

[20] Raphaël Marczak, Jasper van Vught, Gareth Schott, and Lennart E Nacke. 2012. Feedback-based gameplay metrics: measuring player experience via automatic visual analysis. In *Proceedings of The 8th Australasian Conference on Interactive Entertainment: Playing the System*. 1–10.

[21] Farouk Messaoudi, Gwendal Simon, and Adlen Ksentini. 2015. Dissecting games engines: The case of Unity3D. In *2015 international workshop on network and systems support for games (NetGames)*. IEEE, 1–6.

[22] Ciprian Paduraru, Miruna Paduraru, and Alin Stefanescu. 2022. RiverGame-a game testing tool using artificial intelligence. In *2022 IEEE Conference on Software Testing, Verification and Validation (ICST)*. IEEE, 422–432.

[23] Johannes Pfau, Jan David Smeddinck, and Rainer Malaka. 2017. Automated game testing with icarus: Intelligent completion of adventure riddles via unsupervised solving. In *Extended abstracts publication of the annual symposium on computer-human interaction in play*. 153–164.

[24] Cristiano Politowski, Yann-Gaël Guéhéneuc, and Fabio Petrillo. 2022. Towards automated video game testing: still a long way to go. In *Proceedings of the 6th International ICSE Workshop on Games and Software Engineering: Engineering Fun, Inspiration, and Motivation*. 37–43.

[25] Cristiano Politowski, Fabio Petrillo, and Yann-Gaël Guéhéneuc. 2021. A survey of video game testing. In *2021 IEEE/ACM International Conference on Automation of Software Test (AST)*. IEEE, 90–99.

[26] Paul Ralph. 2018. Toward methodological guidelines for process theories and taxonomies in software engineering. *IEEE Transactions on Software Engineering* 45, 7 (2018), 712–735.

[27] Geeta Rani, Upasana Pandey, Aniket Anil Wagde, and Vijaypal Singh Dhaka. 2023. A deep reinforcement learning technique for bug detection in video games. *International Journal of Information Technology* 15, 1 (2023), 355–367.

[28] Mattia Riola. 2023. Test automation in video game development: Literature review and Sound testing implementation. (2023).

[29] Samira Shirzadehhajimahmood, ISWB Prasetya, Frank Dignum, Mehdi Dastani, and Gabriele Keller. 2021. Using an agent-based approach for robust automated testing of computer games. In *Proceedings of the 12th International Workshop on Automating TEST Case Design, Selection, and Evaluation*. 1–8.

[30] Piotr Siuda, Dariusz Reguła, Jakub Majewski, and Anna Kwapiszewska. 2023. Broken Promises Marketing. Relations, Communication Strategies, and Ethics of Video Game Journalists and Developers: The Case of Cyberpunk 2077. *Games and Culture* (2023), 15554120231173479.

[31] Samantha Stahlke, Atiya Nova, and Pejman Mirza-Babaei. 2020. Artificial players in the design process: Developing an automated testing tool for game level and world design. In *Proceedings of the Annual Symposium on Computer-Human Interaction in Play*. 267–280.

[32] Johanna C Van Niekerk and JD Roode. 2009. Glaserian and Straussian grounded theory: similar or completely different?. In *Proceedings of the 2009 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists*. 96–103.

[33] Simon Varvaressos, Kim Lavoie, Sébastien Gaboury, and Sylvain Hallé. 2017. Automated bug finding in video games: A case study for runtime monitoring. *Computers in Entertainment (CIE)* 15, 1 (2017), 1–28.

[34] Carolina Veloso and Rui Prada. 2021. Validating the plot of Interactive Narrative games. In *2021 IEEE Conference on Games (CoG)*. IEEE, 01–08.

[35] Xiaoyin Wang. 2022. VRTest: an extensible framework for automatic testing of virtual reality scenes. In *Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Companion Proceedings*. 232–236.

[36] Saman Zadtootaghaj, Steven Schmidt, and Sebastian Möller. 2018. Modeling gaming qoe: Towards the impact of frame rate and bit rate on cloud gaming. In *2018 Tenth international conference on quality of multimedia experience (QoMEX)*. IEEE, 1–6.