The history of software engineering as the basis for computer games in the visual novel genre

Yehor Yu. Brilov[1], Andrii M. Striuk[1,2,3]

[1]Kryvyi Rih National University, 11 Vitalii Matusevych Str., Kryvyi Rih, 50027, Ukraine
[2]Kryvyi Rih State Pedagogical University, 54 Universytetskyi Ave., Kryvyi Rih, 50086, Ukraine
[3]Academy of Cognitive and Natural Sciences, 54 Universytetskyi Ave., Kryvyi Rih, 50086, Ukraine

Abstract

The aim of the study is to design and implement a computer game on the history of software engineering. Analysis of similar developments has shown the feasibility of implementing the game in the visual novel genre. The plot structure of the game has been designed to interactively tell the story of the software crisis and the 1968 Software Engineering conference. Based on the report on this conference, the key characters of the game were identified. The theses of their speeches were used to construct the game dialogues. Based on real photos taken during the conference, visual locations and character images were created using artificial intelligence. The use of the Ren'Py engine made it possible to create a cross-platform application that can be used in a web environment and on Windows, Linux, iOS, and Android operating systems. The developed product is aimed at a wide range of users who are interested in the historical aspect of the software industry, and can also be used as an illustrative supplement to the course "Fundamentals of Software Engineering".

Keywords

Software Engineering, visual novel, Software Engineering conference, Ren'Py engine

1. Introduction

Software engineering is one of the youngest branches of engineering, which began to develop actively in the second half of the 20th century. Its emergence was humanity's response to new challenges associated with the rapid development of computer technology and the need to create more complex, large-scale, and reliable software systems. The history of this field is closely intertwined with the software crisis that arose in the 1960s. The concept of "software engineering" was first mentioned at an international NATO conference in 1968. The aim of the conference was to develop new principles, methods, and technologies that would ensure a higher quality, more reliable, and more organized process of creating software products. At this point, the theoretical basis of the discipline began to take shape: standardization of development processes, implementation of program life cycle models, development of approaches to testing and quality assurance, and creation of project management tools.

The report on the 1968 Software Engineering Conference [1] is a valuable document not only from a historical point of view, but also for training future programmers, as it outlines the fundamental concepts that form the basis of the modern software industry. However, studying conference proceedings without understanding the historical context can be quite tedious. This is especially true for students who are just getting started in this field. Therefore, we set ourselves the goal of revealing the history of the emergence of software engineering in the form of a visual novel-style computer game. Thanks to this format, the history of software engineering will become not just a chronology of events, but a lively interactive adventure.

CS&SE@SW 2025: 8th Workshop for Young Scientists in Computer Science & Software Engineering, November 21, 2025, Kryvyi Rih, Ukraine
https://doi.org/10.55056/cssesw2025/paper37
excreman1@gmail.com (Y. Yu. Brilov); andrey.n.stryuk@gmail.com (A. M. Striuk)
http://mpz.knu.edu.ua/andrij-stryuk/ (A. M. Striuk)
0000-0001-9240-1976 (A. M. Striuk)

2. Experience of using visual novels in education

Camingue et al. [2] in their article "What is a Visual Novel?", based on an analysis of previous academic definitions and a corpus of 54 games, formulate a unified definition of a visual novel: "is a digital narrative focused game that requires interactions where the player must be able to impact the story world or the story's progression". The authors also note that a significant number of the visual novels considered are education-oriented and have the characteristics of interactive textbooks. The educational potential of visual novels has been repeatedly emphasized by various researchers. In particular, Himes [3] examines the experience of using visual novels to study literary works, while Øygardslia et al. [4] highlight the principles of pedagogical design for games in the visual novel genre. The article "Development of Visual Novel Games as Learning Media for the History of Indonesia's Independence" [5] describes research on the development and evaluation of the feasibility of a visual novel on Android as an educational medium for studying the history of Indonesia's independence. There are also other striking examples of visual novels, such as Attentat 1942, Venti Mesi, and Golden Threads, which introduce historical events through interactive storytelling. This proves once again that visual novels, thanks to their unique combination of interactivity, graphics, and deep plot, can be an effective and interesting tool in the educational process.

3. Game narrative design

So, with the goal of creating an interactive narrative about the formation of software engineering, we moved on to designing the plot structure of the game. Three acts were identified in the game's plot:

Act 1 – *Introduction.* Introduction to the issues of software development, identification of the main manifestations of the software crisis.

Act 2 – *Participation in the 1968 conference.* Communication with participants. Discussion of ways to overcome the software crisis. Outlining the basic principles and tasks of software engineering.

Act 3 – *Implementation of acquired knowledge.* The player tries to introduce engineering principles into software development in their company.

In accordance with the general concept, a detailed structure (figure 1) was designed, including details of scenes and locations, as well as characters with whom the player will interact. According to this scheme, the events of the game begin at the fictional American company "Software Services", where our hero is a leading programmer. His team is working on a large software project but faces numerous difficulties. Looking for ways to solve the problems, the company director sends the hero to a Software Engineering conference in Garmisch.

The following events take place at the Sonnenbichl Hotel, where the conference is being held. At the hotel, the player communicates and engages in discussions with conference participants. Unfortunately, meeting all 40 participants would overload the game. Therefore, based on an analysis of the 1968 conference report [1], we selected nine characters whose speeches were the most meaningful and interesting in the context of software engineering.

The characters we chose were:

- *A. J. Perlis.* His presentation highlighted the main problem: the successful design, production, and maintenance of useful software systems, noting that its importance will grow in the future. He emphasized the rapid growth in development complexity, which cannot be overcome without appropriate theory or tools. He also raised critical questions about the education of software engineers and its relationship to computer science.
- *M. D. McIlroy.* His report was one of the key ones, where he put forward the idea of creating a software components industry.

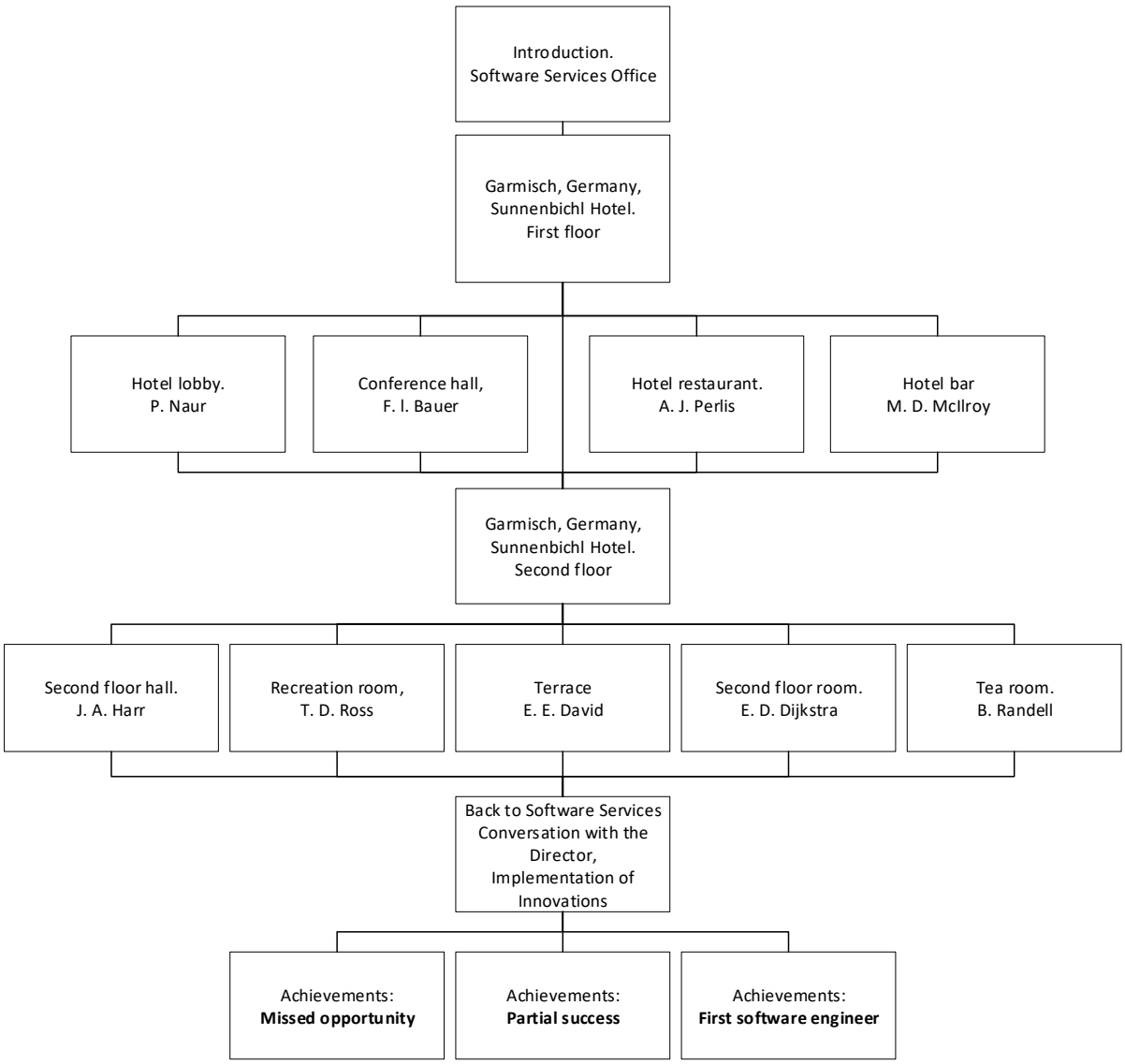

Figure 1: The game's narrative structure.

- *Edsger W. Dijkstra.* He argued that the only way to deal with complexity is to divide the task into a number of steps, each of which should be trivial. He opposed a rigid division between design and production, arguing that product quality cannot be established after the fact and is closely linked to the design process.
- *Peter Naur.* He suggested drawing ideas for solving design problems in architecture and civil engineering, referring to Christopher Alexander's work "Notes on the Synthesis of Form". He emphasized the importance of documentation as a natural part of software development, which should be structured hierarchically, starting with a brief description.
- *Brian Randell* presented a working paper entitled "Towards a Methodology for Computer System Design", in which he discussed top-down and bottom-up approaches. He advocated the integration of simulation into the design process, where a partially designed system is gradually transformed into a real one.
- *E. E. David Jr.* emphasized the enormous costs and labor intensity of large software projects (e.g., OS/360, TSS/360) and the increasing complexity due to numerous non-identical situations. He noted that the computer industry often combines the phases of research, development, and

production, which leads to high risks. He discussed the problem of incompetence in managing large projects.

- *J. A. Harr* described in detail the process of designing real-time software for electronic switching systems, including specification, modularity, interface definition, and testing. Identified common reasons for project delays, including underestimation of time and resources.
- *D. T. Ross* introduced the concept of "plex" as the basic theory for software engineering, which includes data, structure, and algorithm, aimed at fully covering the problem being solved.
- *F. L. Bauer.* He chaired the 1968 conference. It is believed that he was the one who suggested naming the conference "Software Engineering", which was somewhat provocative at the time. He emphasized that software systems are mathematical in nature and that systems should be built on levels and modules that form a mathematical structure.

Through communication with these characters, the player learns the basic ideas behind implementing engineering approaches in software development. After the hero has talked to all the participants and held discussions and debates, he can return to his home company. Back at Software Services, the hero and the director discuss the changes that need to be made to implement engineering approaches to software development. In this dialogue, the player must choose the correct answers from the suggested actions. The player receives points for each correct choice. A total of 10 points can be earned during the answers. Depending on how many points the player has earned, they will receive one of the possible achievements and game endings:

10–8 points – Achievement: "First Engineer-Programmer." The player correctly or almost correctly implements engineering principles in the work of their company's programmers.

5–7 points – Achievement: "Partial Success." The player changes most of their approaches to software development but does not achieve complete success due to half-hearted actions.

0–4 points – Achievement: "Missed Opportunity." The player was inattentive at the conference and failed to implement engineering principles in their company's work.

Thus, the last act actually serves as a test of knowledge of the basic engineering principles in software development, and also provides the player with feedback—whether they understood the main issues discussed at the 1968 conference and the basic principles underlying software engineering.

The main game mechanic of visual novels is dialogue. In our version, dialogues can be divided into three groups:

1) regular dialogues, in which there are choices of responses that influence the opponent's specific answer;
2) extended dialogues, during which it is important to ask all the questions from the list and get all the answers;
3) dialogues with evaluation, during which the player scores points for correct answers.

The text content of dialogues is usually the most labor-intensive stage in the development of visual novels. To make them as realistic and meaningful as possible, it would be necessary to carefully process the materials of the 1968 conference, highlight the key ideas expressed by the conference participants, and creatively transform them into a dialogue with the game's protagonist. Previously, such work would have required a team of copywriters. However, now the routine part of the work can be entrusted to large language models. Shields et al. [6] in the article "Generating Together: Lessons Learned from Developing an Educational Visual Novel with AI Collaboration", describes the experience of a team of developers who integrated generative artificial intelligence, in particular large language models (LLMs) and text-to-image models, into the creation of an educational visual novel called Academical 2.0. Researchers note that although AI increases productivity, it still requires significant human oversight to ensure quality and thematic consistency. However, despite its shortcomings, generative technologies

expand the capabilities of creators and can be useful for small teams of developers with limited project funding. Therefore, using the Gemini model and the NotebookLM application, we processed a report from the 1968 Software Engineering conference and generated dialogue options with key conference participants. It should be noted that the resulting texts undoubtedly required creative editing and adaptation to the complex structure of visual novel dialogues. However, the use of a large language model made it possible to significantly reduce the time required to create the text part of the narrative computer game.

4. Generation of scenes and characters

Building visual scenes was a separate challenge. We found a selection of photos taken during the conference in the Internet archives. However, these photos were of rather poor quality, so it was impossible to use them directly to build the game scenes. Therefore, we found modern photos of this hotel and its premises and used artificial intelligence to stylize them to look like the 1960s (figure 2).

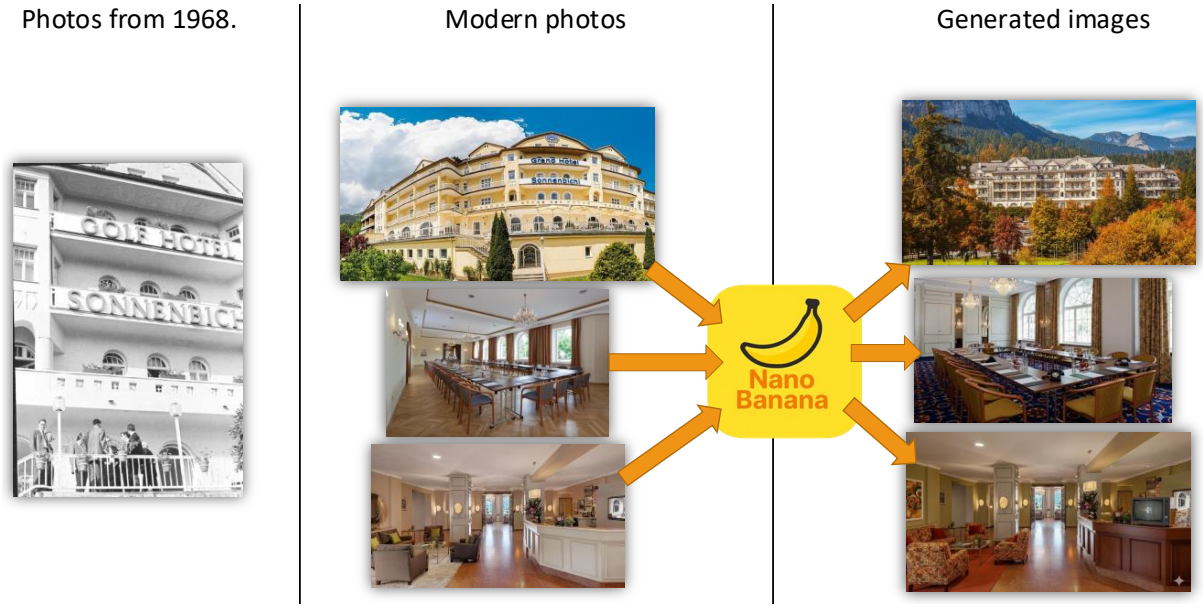

Figure 2: Game scene generation.

We took a similar approach to creating characters by processing photos of real conference participants (figure 3). Thanks to improvements in generative models, in particular Google's Nano Banana model, we were able to not only modernize the characters' photos, but also give each of them unique gestures and emotions without compromising their recognizability. Thus, the game scenes, albeit with a certain degree of convention, became closer to real historical events.

5. Implementation of a visual novel about the origins of software engineering

The next stage of development was choosing a game engine. The most well-known engines used for visual novels are Ren'Py, Unity with corresponding packages for creating novels, TyranoBuilder, Visual Novel Maker, and Godot. For our project, we chose the Ren'Py engine, which is free and combines ease of use with functionality. An additional advantage of Ren'Py is its cross-platform compatibility. Games created with it can be exported to run on various devices, including Windows, macOS, Linux, mobile platforms such as Android and iOS, as well as web applications. This makes it possible to reach a wide audience and ensure high accessibility of the game for different situations.

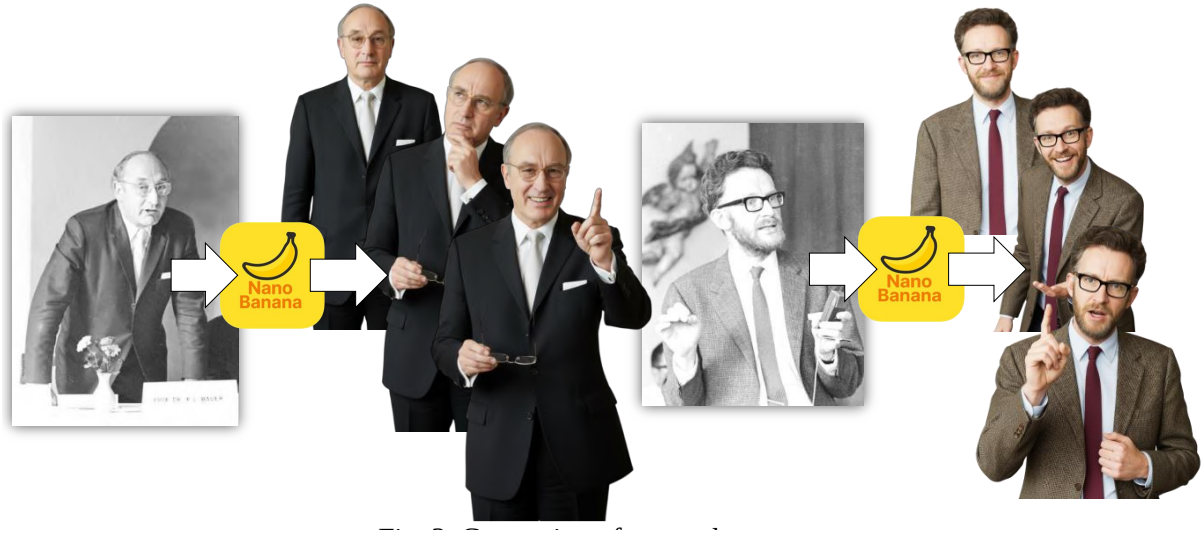

Figure 3: Generation of game characters.

The engine provides basic functionality for building a start page with a main menu (figur 4), game saving and loading functions, display settings, etc.

Figure 4: Game start screen.

The engine also simplifies the creation of visual scenes, providing flexible combinations of backgrounds and characters on them (figure 5).

This gave us the opportunity to focus on implementing the plot structure in the game code and ensuring interactivity using Python scripts.

6. Conclusions

So, we developed a game dedicated to the history of software engineering and based on real historical events. During development, we became convinced that modern artificial intelligence tools can be useful

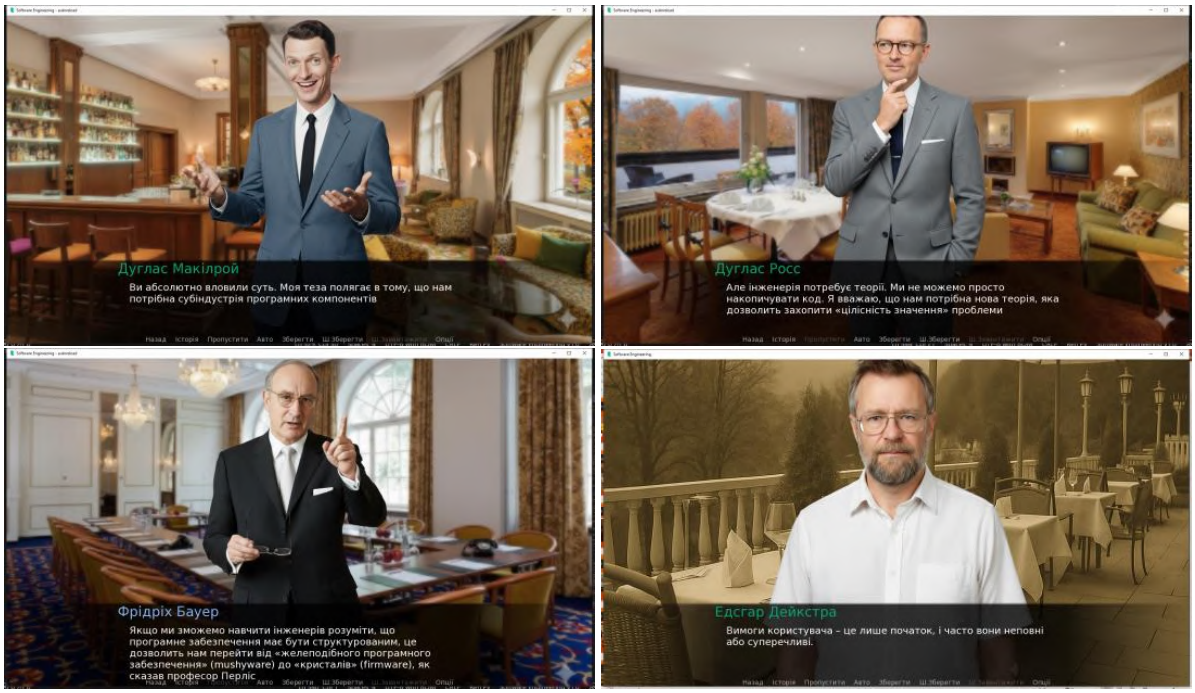

Figure 5: Game scenes.

assistants in developing content for games in the visual novel genre. Generative artificial intelligence was best suited for creating visual content: images of locations and characters. The generated text content required significant reworking and expert verification.

The game, created in an exciting story format, reveals the essence of the software crisis and highlights the 1968 Software Engineering Conference as an attempt by the best specialists of the time to join forces to overcome this crisis. The game allows players not only to learn about key events, but also to immerse themselves in the world of outstanding personalities and their contribution to the development of software engineering. In addition to its entertainment function, the game is planned to be used as an interactive learning tool, a supplement to the course “Fundamentals of Software Engineering”, which will help students consolidate their knowledge through gaming experience.

Author contributions

Conceptualization, Yehor Yu. Brilov and Andrii M. Striuk; methodology, Andrii M. Striuk; software, Yehor Yu. Brilov; validation, Yehor Yu. Brilov and Andrii M. Striuk; investigation, Yehor Yu. Brilov; resources, Andrii M. Striuk; data curation, Yehor Yu. Brilov; writing – original draft preparation, Yehor Yu. Brilov; writing – review and editing, Yehor Yu. Brilov and Andrii M. Striuk; visualization, Yehor Yu. Brilov.

Funding

This research received no external funding.

Conflicts of interest

The authors declare no conflict of interest.

Declaration on Generative AI

The authors have not employed any generative AI tools.

References

[1] B. Randell, NATO Software Engineering Conference: Garmisch, Germany, 7-11 Oct 1968, 2003. URL: http://homepages.cs.ncl.ac.uk/brian.randell/NATO/N1968/index.html.

[2] J. Camingue, E. Carstensdottir, E. F. Melcer, What is a Visual Novel?, Proceedings of the ACM on Human-Computer Interaction 5 (2021) 285. doi:10.1145/3474712.

[3] M. Himes, Visual Novel Based Education in English Literature: A Study on Student Engagement, Press Start 8 (2022). URL: https://press-start.gla.ac.uk/press-start/article/view/206/118.

[4] K. Øygardslia, C. L. Weitze, J. Shin, The Educational Potential of Visual Novel Games: Principles for Design, Replaying Japan 2 (2020) 123–134. URL: https://ritsumei.repo.nii.ac.jp/record/13382/files/rcgs_2_%EF%BE%83%E3%83%BBgardslia.pdf.

[5] M. H. Khoiruddin, Z. H. Z. Bahari, M. S. Kaka, S. Saenpich, Development of Visual Novel Games as Learning Media for the History of Indonesia's Independence, Journal of Educational Technology and Learning Creativity 1 (2023) 33–41. doi:10.37251/jetlc.v1i1.622.

[6] S. Shields, A. Calderwood, S. Johnson-Bey, N. Wardrip-Fruin, M. Mateas, E. Melcer, Generating Together: Lessons Learned from Developing an Educational Visual Novel with AI Collaboration, in: 2024 IEEE Conference on Games (CoG), 2024, pp. 1–8. doi:10.1109/CoG60054.2024.10645553.