

Supporting Unity Developers with an AI-Powered Asset: Insights from an Exploratory User Study on Multiplayer Game Development

Magdalena Igras-Cybulska^{1,3,*}, Barbara Kolber-Bugajska¹, Rafał Salamon¹, Paweł Babiuch¹, Hubert Jegierski¹, Maciej Jegierski¹, Adrian Łapczyński¹, Alicja Marmon¹, Kamil Kwiatkowski¹, Artur Cybulski¹, Mirosław Płaza², Grzegorz Łukawski², Stanisław Deniziak², Paweł Pięta², Artur Jasiński², Jacek Opalka² and Paweł Węgrzyn^{1,4}

¹EpicVR sp. z o.o., Cracow, Poland

²Kielce University of Technology, Kielce, Poland

³AGH University of Krakow, Cracow, Poland

⁴Jagiellonian University, Cracow, Poland

Abstract

In response to the increasing requirements for advanced tools that facilitate game development, this paper presents the exploratory phase of a research project guided by User-Centered Design (UCD) principles. This phase is dedicated to the conceptualization and preliminary assessment of an AI-enhanced utility, termed MUN (Metaverse Unity Networking). Initiated through a detailed investigation of user needs and preferences, the study employs qualitative research methodologies, including eight in-depth interviews and a focus group discussion with three developers. The development of algorithms for this project leads to the establishment of a recommendation engine aimed at refining the selection process of programming methods and strategies for multiplayer game development. Furthermore, it incorporates mechanisms for the detection and correction of errors within the source code. Subsequent to a sequence of iterative tests (comprising three iterations with a collective participation of 40 users), the utility was subjected to a final usability evaluation involving 32 developers. This document focuses on the initial phase of the project, detailing the primary findings related to user needs and expectations as identified during this exploratory stage.

Keywords

AI, asset, Unity, recommendation engine, user-centered design, Unity developers, multiplayer games development

Proceedings of the 1st International Workshop on Designing and Building Hybrid Human-AI Systems (SYNERGY 2024), Arenzano (Genoa), Italy, June 03, 2024.


*Corresponding author.

✉ magda.igras@gmail.com (M. Igras-Cybulska)

🆔 0000-0001-5621-7901 (M. Igras-Cybulska); 0009-0004-8953-8516 (B. Kolber-Bugajska); 0009-0008-6834-5175 (R. Salamon); 0009-0003-9951-4174 (A. Cybulski); 0000-0001-9728-3630 (M. Płaza); 0000-0002-5262-454X (G. Łukawski); 0000-0002-6812-5227 (S. Deniziak); 0000-0003-1507-4726 (P. Pięta); 0009-0001-5364-5959 (A. Jasiński); 0009-0002-8295-9882 (J. Opalka); 0000-0001-5616-0474 (P. Węgrzyn)



© 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

1. Introduction

In the context of the increasing popularity of multiplayer game development within the Unity environment, particularly for virtual reality (VR) applications as well as continuously expanding complexity and scope of projects, there exists a necessity for tools that can streamline the development process, even for less experienced developers or smaller game studios. These challenges include, but are not limited to, ensuring accurate synchronization of player actions, managing network latency, and optimizing for diverse VR hardware capabilities. To answer these needs, we came up with a MUN (Metaverse Unity Networking) asset [1, 2] that leverages artificial intelligence (AI) to facilitate the development of multiplayer VR games in Unity, thereby addressing identified challenges.

"Assets" refer to any reusable components or resources used in video game development. These can include a wide range of elements such as code libraries, pre-designed graphics, animations, sound effects, music tracks, or even pre-built game level designs. Assets are utilized to streamline the development process, allowing developers to save time and resources by integrating these ready-made elements into their games. This approach speeds up the production cycle and helps to ensure consistency and quality across different parts of the game. In the realm of AI-assisted game development discussed in the article, assets also include tools and modules specifically designed to enhance the functionality and efficiency of the development process, such as AI-driven code generators or behavior modeling systems for non-player characters. The primary repository for assets is the Unity Asset Store, although assets can also be found on the websites of their respective creators.

1.1. AI-based automation of game development

Automation streamlines development processes but can also foster creativity and innovation by alleviating the burden of repetitive, time-intensive tasks. With the escalation in project complexity, the likelihood of errors proliferates, necessitating the adoption of AI and machine learning algorithms for timely error detection and rectification. The competitive landscape of the gaming industry, with its emphasis on rapid development cycles and superior output quality, benefits markedly from automation's capacity to refine workflows, elevate code quality, and liberate developers to concentrate on advanced design and gameplay elements. Moreover, the increasing sophistication of multiplayer and VR experiences calls for enhanced synchronization, realism, and user experience continuity, highlighting the significance of advanced tools like the MUN asset. Huynh's recent survey [3] delves into AI's contributions to Metaverse development, presenting an in-depth evaluation of AI-driven techniques for virtual world creation and their applicability in various domains. Furthermore, the utilization of AI in software development represents a significant paradigm [4], underscoring its transformative potential across the technological landscape.

1.2. User centered design paradigm in AI-based software production

User-centered design (UCD) plays a pivotal role in the development of digital products, focusing on the needs, preferences, and limitations of end-users at every stage of the design and

development process. By prioritizing user feedback and involving users in the creation cycle, from ideation to testing, UCD ensures that products are not only functional and efficient but also intuitive and satisfying to use. UCD's iterative nature allows for continuous improvements based on real user interactions, making it a crucial strategy in the landscape of digital products development.

The adoption of UCD is useful in the creation of AI-powered products that aim to assist users in their work by providing suggestions and recommendations. UCD ensures that these products are developed with a deep empathy for the user, facilitating a seamless integration into daily workflows and enhancing productivity without adding complexity. Margetis et al. [5] lay the foundation for this exploration, proposing a methodological framework aimed at embedding the human-in-the-loop paradigm within AI development processes. Their work delineates six core concepts, including explainable AI, semantic computing, and UX design for AI, emphasizing the importance of human-centric approaches in the development of AI systems. Building on the necessity of UCD for AI's practical application, Brillowski et al. [6] presented the AI-based decision support systems within traditional craftsmanship. Their mixed-method study underscores the critical role of UCD in ensuring the usability, trust, and acceptance of AI tools, revealing that user-centered design is paramount for the effective integration of AI in decision-making processes. Similarly, Garcia et al. [7] highlight the challenges faced by non-expert users in leveraging machine learning for decision-making, particularly in healthcare. Their development of a user-friendly machine learning platform for cardiologists exemplifies how UCD can bridge the gap between complex AI technologies and end-user needs. Staes et al. [8] presented another medical application of UCD in AI through the design of an interface aimed at communicating machine learning-based prognoses in oncology. The iterative design process, incorporating feedback from oncologists, showcases the necessity of a UCD approach in developing AI tools that are not only functional but also trustworthy and comprehensible to its users. This emphasis on communication and interpretability resonates with the broader objectives of human-centered AI, where the goal is not only to create powerful AI systems but also to make these systems accessible and meaningful to their intended users. Csiszar et al. [9] address the challenges of adopting AI in the manufacturing sector, emphasizing the need for UCD in facilitating the use of AI and machine learning by manufacturing engineers. Their exploration into the barriers of AI adoption by non-expert users underscores a recurring theme in UCD for AI research: the importance of designing AI systems that are intuitive, user-friendly, and tailored to the specific workflows and expertise levels of their users.

In summary, these studies collectively underscore the pivotal role of UCD in the development and adoption of AI technologies across various domains.

1.3. MUN asset

The development of the MUN asset was guided by an extensive examination of user requirements and preferences. This research phase revealed a clear need for an asset that mitigates the complexities associated with the development of multiplayer VR environments. The asset, specifically designed for the configuration of multiplayer settings in VR applications, is supported by a dual-component AI framework. This framework encompasses a recommendation engine aimed at enhancing programming workflows and facilitating code correction, as well

as a behavior analytics system intended to improve the authenticity of interactions between avatars and Non-Player Characters through the application of sophisticated motion atlas and motion recognition technologies. The latter engine's capabilities are detailed in [10], while the comprehensive functionalities of the asset are explored in [11].

This manuscript concentrates on delineating the initial phase of the UCD process—an exploratory study with Unity developers. It is anticipated that the insights regarding developers' needs and expectations gleaned from this study could inform the development of other AI-enhanced tools within the game development sphere.

2. Exploratory phase

An exploratory study is a type of research aimed at identifying and deeply understanding a problem, phenomenon, or the context of a product being designed, proving invaluable before product development begins, when there is a need to gather foundational knowledge, outline the project context, or at the early stage of the design process.

2.1. In-depth interviews

In the initial phase of our exploratory research, individual interviews were conducted to gain a comprehensive understanding of the challenges and obstacles faced by VR developers during the development of multiplayer projects. The primary aim was to explore the various pain points encountered by developers in this niche, investigate potential development trajectories for an asset designed to assist in multiplayer project execution, and understand current practices in asset interaction among VR developers. This inquiry was motivated by the necessity to bridge identified gaps in the multiplayer development process.

The research questions guiding this phase centered around the specific problems VR developers face in multiplayer settings, the stages at which they require the most support, and strategies for streamlining multiplayer development workflows. Additionally, the study sought to ascertain developers' expectations from a supportive asset, their preferred modes of interaction with such tools, including the potential use of voice assistants, and their current practices concerning asset discovery, acquisition, installation, learning, and utilization.

The interviewees were Unity developers—eight men, aged 18 to 40. Purposeful sampling was used to ensure diversity in participant profiles, considering their experience with Unity, number of completed projects, and experience with multiplayer modes. Regarding their professional tenure in the video game industry, three participants had less than a year, three had between one to three years, one had four to six years, and one had over six years of experience. As for the number of Unity assets they have utilized, three participants had used between 1 and 10 assets, one had used between 11 and 20 assets, and four had used over 20 assets. Two of the VR developers had experience in computer networking, and four respondents had international experience. The interviews were conducted remotely via Discord and lasted approximately an hour each.

Key findings:

- Respondents would most like to see asset support at the beginning of the project (network

architecture) and during the testing phase. The testing phase is the most problematic when working on a multiplayer project. Additionally, respondents often mentioned problems related to object ownership, creating network architecture, and synchronization between players.

- Respondents imagine that at the start of a project, the asset would ask them about a few issues to tailor its operations to their preferences.
- Full control over the asset is mandatory. Respondents cannot imagine the asset taking any actions without asking for their consent first.
- The asset should help but also leave room for the users' creativity.
- With the user's prior consent, the asset could automate monotonous tasks requiring speed and infallibility.
- A voice assistant is not a welcomed enhancement. All respondents stated that they would not like to use it.
- Respondents would most like to be able to adjust the way of communication with the asset to their individual needs.

Functionalities. In the realm of asset automation, respondents are clear about their preferences: they are willing to fully entrust the asset with tasks that are devoid of creativity but require speed and repetition, but only after giving their prior approval. This calls for a tool designed to identify and memorize such repetitive tasks—those that involve simple actions like finding and copying values—and subsequently inquire if the developer wishes for these to be automated. Moreover, the asset should support the recording of macros, allowing developers to directly specify actions for automation. Crucially, alongside these automation capabilities, it's imperative that the tool preserves a space for developers to implement their own creative solutions, underscoring a desire for a balance between efficiency and personal input in the development process.

Prediction and optimization. Participants expressed a desire for a tool that not only oversees their work but also proactively suggests potential solutions to encountered challenges. They envision a tool capable of analyzing their work in real time, adept at predicting the functions they intend to use, and providing nuanced feedback on their choices. Instead of offering binary judgments, the asset would present the effectiveness of a solution in percentage terms, suggest optimizations, and facilitate navigation to relevant documentation sections. It's crucial for this tool to prevent developers from reaching impasses that could force them to revert extensive portions of their work. Additionally, integrating a search engine for solutions, akin to the familiarity and utility of Stack Overflow, along with a manual refresh feature, ensures that the tool remains up-to-date with the project's current state.

Communication between the asset and its users is a delicate balance, with respondents favoring non-intrusive interactions that respect their workflow and autonomy. They advocate for clear communication channels that transparently convey errors, suggest changes, and provide feedback, allowing users the discretion to accept or reject proposed alterations. The ideal tool operates as a supportive assistant rather than an authoritative figure, understanding user needs through direct input and acknowledging the potential validity of developers' perspectives. Subtle notification methods for suggesting corrections, separating community-driven advice from programmed responses, and prioritizing visual over auditory alerts are preferred. Such

a tool would not only enhance productivity but also enrich the development environment by valuing the contributions and discretion of its users, marking a significant step forward in collaborative project development.

Control. Respondents would like to have full control over what the asset does. They do not want it to make changes to the project without their approval. They prefer the tool not to be intrusive or interfere with their work.

2.2. Focus group interview

In the next study, we conducted a focused group interview, known as a focus group (FGI), which is a qualitative research method where a small group of participants engages in a structured discussion, sharing their opinions and experiences. This approach was enhanced with workshop methods to stimulate the participants' creativity, allowing us to gather a broad spectrum of user behavior patterns, experiences, perspectives, and needs. The methodology enabled us to uncover insights into the characteristics of the best and worst assets in game development, explore possibilities for automating the creation of multiplayer games, understand how assets should manage errors and suggest changes, and find ways to simplify the testing of multiplayer games.

The focus group session, engaging three participants previously involved in the interviews, spanned nearly two hours and was structured to catalyze meaningful discussions. The online session initiated with (1) introduction of the objectives, methods and tools, such as Mural, to be employed. Subsequently, participants were invited to choose a Dixit card (2) they believed symbolically resonated with the process of developing multiplayer games, thereby instilling a creative ambiance for the session. Then the main discussion (3) was performed, which employed creative problem-solving strategies to both delineate the attributes of the least effective asset for multiplayer game development and, inversely, to conceptualize an ideal asset for the same purpose. The conversation advanced with participants addressing a series of "How might we...?" inquiries (4). These inquiries were designed to span a broad spectrum of topics relevant to game development, including the automation of creation processes, the management of errors, the proposition of modifications, and the intricacies of game testing. Through this approach, a holistic perspective on the specific needs and hurdles faced by developers in the realm of multiplayer game development was achieved.

Key findings:

- **Automation fields.** During the study, participants offered a range of ideas for automating the creation of multiplayer games, highlighting the need for both efficiency and flexibility in development tools. They suggested the preparation of ready-made segments, such as server setup, hosting solutions, and matchmaking functionalities, that could be easily integrated into games. These segments would come with carefully considered default settings and offer the flexibility for easy customization to suit specific project needs. Additionally, the creation of a classification system for various types of design problems and challenges was proposed, complete with optimal, optional solutions. Each problem or challenge could be assigned a unique code, making it easily searchable in the documentation, with the code linking directly to the relevant section for quick reference.

- **Visual design** of procedures was also seen as a valuable tool, enabling developers to conceptually and practically map out the flow of multiplayer interactions without the need for extensive coding. Moreover, the idea of creating to-do lists tailored to specific project challenges was mentioned. These lists could serve as step-by-step guides. Addressing the challenge of multiplayer game testing could provide a competitive edge, potentially through the development of bots to simulate users connecting to servers and participating in games, along with tools for visually designing test procedures.
- **The recommendation mode.** Developers believe that suggestions for changes should be presented in multiple places for maximum efficiency and accessibility: contextually next to the code where the suggestion is applicable and also on a separate list within the application's suggestions panel. These suggestions should accommodate the individual working styles of developers and their unique approaches to project challenges. This personalized approach should stem from machine learning algorithms that not only learn from the behaviors of the user base as a whole but also adapt to the preferences and habits of individual developers. Recommendations should be provided in the form of an interactive tutorial within the asset itself. Such a tutorial would guide users through the subsequent stages of development, offering a hands-on learning experience. Additionally, the documentation should include a step-by-step guide to creating multiplayer games, aiding developers in understanding the process and what to expect next. It's also suggested that the tool should be able to inform users about their current stage in the game development process and provide insights into how much work remains. This approach not only helps in planning and managing the workload but also in setting realistic expectations, making the process smoother and more manageable.
- **Smooth error handling** is another key expectation from developers; informing users about the asset's error management approach from the outset is important to ensure that error handling doesn't interrupt work and isn't entirely automated.
- **Updates.** Balancing the approach to asset updates is critical for user satisfaction. Developers desire necessary updates for smooth operation but also value control and fear changes. Finding a middle ground involves seeking user consent for updates, informing them of the benefits and risks, and ensuring compatibility with the engine version.

Moreover, we discovered that developers are skeptical about new assets and their promotional materials due to past experiences where assets failed to deliver on their promises. To gain their trust, it's crucial to ensure transparent communication and clearly demonstrate the tasks the tool can perform. Additionally, developers fear the time investment required to learn new assets, so enhancing their initial experience with the tool can improve overall perceptions. Incorporating a wizard or assistant to guide users through their first steps and providing effective customer support are vital. The lack of clear and convenient documentation is a significant issue for developers; even if an asset offers a wide range of attractive functionalities, the absence of helpful documentation can negatively impact user satisfaction. Thus, creating accessible documentation in various formats, such as video, contextual help, and textual classification of functions and problems, is essential.

3. The solution and its evaluation

MUN asset was developed in an agile process, including three iterations of testing with external users. The final version is demonstrated in [11] and the tool is available at [2]. To the best of our knowledge, the MUN asset is the first approach to automating multiplayer development in VR using AI.

The MUN solution simplifies the introduction of multiplayer features in VR games by partly automating the process, which speeds up development and reduces costs. It includes a motion library that minimizes the amount of data that needs to be sent back and forth for multiplayer synchronization and makes VR interactions appear more realistic. The main functions of the MUN asset are organized into several components. The Client Application enables developers to create multiplayer functionalities, while the Service for User and Server Management stores important data and helps in communicating with the Server Manager. The Server Manager is responsible for managing servers specific to each device, and the MasterServer serves as the central connection point for users. Core features include connecting to the MasterServer, creating and managing virtual rooms, overseeing user activities, handling objects in the network, synchronizing character movements, supporting Remote Procedure Calls (RPC) which allow functions to execute across a network, and providing a voice chat feature for players to communicate verbally [11].

We applied AI in MUN asset to support developers in two fields development:

1. Central to the MUN asset is its AI-based recommendation system. At the core of the MUN asset is an AI-based recommendation system, devised to streamline the game development process by automating key tasks. This system extends recommendations for optimizing programming practices and supports error detection and correction within the application's source code. Expected to considerably decrease development duration, reduce coding inaccuracies, and enable developers to devote greater effort to the creative dimensions of game creation, the recommendation engine leverages a machine learning model. This model, which has been trained on a corpus of 2.7 million lines of code, possesses the capability to propose subsequent programming maneuvers and rectify coding mistakes. During evaluation trials, the system exhibited accuracy surpassing 70%, thereby enhancing the programming workflow through customized suggestions and corrections. This bespoke methodology not only augments developer productivity by minimizing errors but also customizes the development trajectory to align with the distinct programming styles of individual developers. Moreover, when assessed against a compendium comprising 1,005 generalized coding errors and their solutions, in addition to 279 errors specific to VR development, the system's error detection and correction functionality recorded an accuracy of 91.05% utilizing the LightGBM algorithm.
2. The automatic movement detection and prediction feature, integral to the MUN asset, is facilitated by a custom-developed motion atlas designed specifically for VR implementations. This feature employs sophisticated machine learning methodologies to analyze the position and rotation of crucial body points, achieving motion recognition with a weighted F-score approaching 98%. Trained on a dataset comprising 56 unique movements, the system exhibits exceptional precision in identifying user actions, thereby substantially

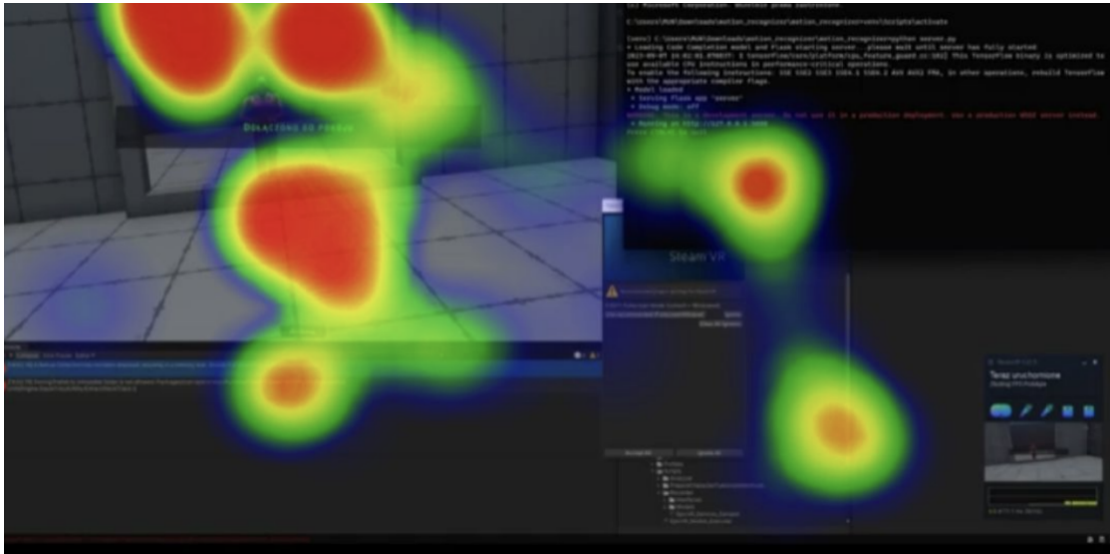


Figure 1: Using eye tracking to create heatmaps over MUN interface during the design and evaluation process. The question that was asked is: where would you look for information about which movement was recognized by the AI? Analysis of the behavior of individual testers showed that 70% looked in the place where this information was provided in the interface, while 30% looked in other places.

enhancing the immersive aspect of multiplayer VR experiences. The predictive capability of this AI engine was assessed through a test set involving 221 players and 1,279 motion sequence attempts, where it demonstrated an accuracy rate of 88.51% in forecasting forthcoming user movements.

Throughout its development, the MUN asset underwent several iterative research stages, encompassing usability studies and user experience (UX) testing involving developers across various levels of expertise, thereby guiding the evolution of the asset. The project progressed through: initial usability studies engaged 8 users, followed by comprehensive UX research with 16 participants, subsequent usability and UX evaluations with an additional 16 individuals, and culminating in a final iteration test involving 32 users. Eye tracking was used to redesign the interface element to better meet the intuition of users (Fig. 1). These stages collectively affirmed the asset’s applicability and value for the intended user demographic.

In the latest iteration of this research, a usability and UX study was conducted with 32 experienced software developers participating as testers. This investigation adopted a mixed-methods framework, amalgamating both quantitative and qualitative research methodologies to elicit a holistic understanding of the asset’s usability and user experience.

The core component of the investigation was a task-based usability test, structured around a scenario divided into three segments. Participants were asked to express their evaluations through a questionnaire that included both closed-ended questions—utilizing the System Usability Scale (SUS) for quantifiable metrics—and open-ended questions for qualitative feedback. The results obtained from the SUS indicated a mean usability score of 60.6, with a standard deviation of 19.3, suggesting specific areas for improvement in the asset’s usability and overall

user experience.

Based on the interviews conducted, respondents identified several key elements of the MUN asset that they found to be most helpful in their work on multiplayer game projects. The most frequently mentioned benefits included ready-made prefabs (pre-constructed elements that developers can use in their games, saving time on building components from scratch), detailed instructions, video materials, and automatic code repair and analysis. Additionally, users appreciated tools for detecting and correcting errors, the ease of creating servers/rooms for players, and integration with Unity, including auxiliary management windows, which facilitates networking players. The RTU server, which significantly reduces the work involved in connecting a player to the server and managing their actions, was particularly well-received. Features such as automatic generation of components for multiplayer games, creating lobbies, and complete, ready-to-use functions that support work, were praised by developers. Despite the diversity of responses, a clear picture emerges of the MUN asset as a tool that can support developers in creating multiplayer games, especially through automation and streamlining the coding process.

4. Discussion and Conclusion

The exploratory research conducted for the development of the MUN asset underscores the critical role of UCD in the creation of AI-based tools for game development. Through in-depth interviews and focus group discussions with experienced Unity developers, our study revealed insights into the complex challenges developers face, highlighting the need for a more intuitive, automated, and user-friendly approach to multiplayer game creation. These interactions provided valuable feedback on the desired functionalities and usability improvements, driving the iterative design process of the MUN asset to better meet the specific needs of the developers community.

The article emphasizes that developers desire a tool that provides personalization, continuous learning, detailed guidance, and precise control over task delegation. The MUN asset is intended to incorporate these expectations through features like a recommendation engine and behavior analytics, which help in automating some tasks and providing personalized assistance based on past interactions and user preferences. The recommendation engine is built to adjust its functions based on accumulated data from user interactions, suggesting a degree of personalized support. Although the article does not specify if the system continues to learn post-deployment, it does mention that the engine uses a machine learning model trained on extensive coding examples to refine its recommendations and error corrections. Significant effort was made to ensure the AI recommendations do not undermine the developer's initiative. The system allows developers to retain full control over accepting or rejecting suggested changes, thus maintaining a balance between benefiting from AI-driven automation and retaining creative freedom.

Developers expressed a desire for tools that could not only simplify repetitive tasks but also offer intelligent recommendations and error corrections tailored to their unique development contexts. This feedback points to a significant opportunity for future work: to refine the AI components of the MUN asset, making them more adaptive and responsive to individual development styles and project requirements. Enhancing the asset's learning algorithms to

incorporate a broader spectrum of developer interactions will be pivotal in achieving this goal. Further studies and evaluations would be required to fully understand the long-term learning and adaptation capabilities of the MUN system.

In conclusion, the findings from our exploratory user research highlight the indispensable role of UCD in the development of AI-powered development tools like the MUN asset. The insights gathered have not only informed the initial development of MUN but also laid the groundwork for ongoing improvements and innovations. Future iterations of MUN will continue to be guided by user feedback, especially when the users' competences in using AI in their work rapidly evolve.

Acknowledgments

This project was supported by the grant *An innovative self-learning tool facilitating the design of a multiplayer mode in VR applications* (POIR.01.01.01-00-1039/21) from the Polish National Centre for Research and Development. The research for this publication has been supported within the Priority Research Area DigiWorld under the Strategic Programme Excellence Initiative at Jagiellonian University.

References

- [1] MUN website, <https://multiverse.epicvr.pl/>, 2024. URL: <https://multiverse.epicvr.pl/>.
- [2] MUN asset, <https://gitlab.com/public-repos9/metaverse-unity-networking>, 2024. URL: <https://gitlab.com/public-repos9/metaverse-unity-networking>.
- [3] T. Huynh-The, Q.-V. Pham, X.-Q. Pham, T. T. Nguyen, Z. Han, D.-S. Kim, Artificial intelligence for the metaverse: A survey, *Engineering Applications of Artificial Intelligence* 117 (2023) 105581.
- [4] I. Ozkaya, The next frontier in software development: Ai-augmented software development processes, *IEEE Software* 40 (2023) 4–9.
- [5] G. Margetis, S. Ntoa, M. Antona, C. Stephanidis, Human-centered design of artificial intelligence, *Handbook of human factors and ergonomics* (2021) 1085–1106.
- [6] F. Brillowski, T. Gries, L. S. Vervier, T. Schemmer, P. M. Brauner, M. C. Ziefle, User centered design and evaluation of an artificial intelligence based process recommender system in textile engineering, *Universitätsbibliothek der RWTH Aachen*, 2022.
- [7] A. García-Holgado, A. Vázquez-Ingelmo, J. Alonso-Sánchez, F. J. García-Peñalvo, R. Therón, J. Sampedro-Gómez, A. Sánchez-Puente, V. Vicente-Palacios, P. I. Dorado-Díaz, P. L. Sánchez, User-centered design approach for a machine learning platform for medical purpose, in: *Iberoamerican Workshop on Human-Computer Interaction*, Springer, 2021, pp. 237–249.
- [8] C. J. Staes, A. C. Beck, G. Chalkidis, C. H. Scheese, T. Taft, J.-W. Guo, M. G. Newman, K. Kawamoto, E. A. Sloss, J. P. McPherson, Design of an interface to communicate artificial intelligence-based prognosis for patients with advanced solid tumors: a user-centered approach, *Journal of the American Medical Informatics Association* 31 (2024) 174–187.

- [9] A. Csiszar, P. Hein, M. Wächter, A. Verl, A. C. Bullinger, Towards a user-centered development process of machine learning applications for manufacturing domain experts, in: 2020 Third International Conference on Artificial Intelligence for Industries (AI4I), IEEE, 2020, pp. 36–39.
- [10] P. Pięta et al., Automated classification of virtual reality user motions using a motion atlas and machine learning approach, under review (2024).
- [11] P. Babiuch et al., MUN: an AI-powered multiplayer networking solution for VR games, in: 2024 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW), IEEE, 2024.