QA and Release Management for VR Platforms: A Meta Quest Update Case Study

Komal Jasani

QA Engineering Lead
Union City, California USA
komal_jasani@yahoo.com

Abstract

This research investigates how Meta Quest implements quality assurance and release management procedures for its VR platform updates. The paper analyzes VR development difficulties, including hardware compatibility issues, performance enhancement efforts, and tests for virtual reality immersion. This investigation analyzes established techniques that help both automated and manual testing and the system for managing launches and user input collection to boost program reliability. An analysis examines the consequences of recurring firmware and software releases for the VR system while discussing strategies to track bugs and improve regression testing efficiency and system deployment methods. The research explores Meta Quest update processes to deliver knowledge about enhancing VR platform release management operations, which guarantee smooth, high-quality user journeys across an evolving digital framework.

Keywords: Quality Assurance, Release Management, Virtual Reality, Meta Quest, Software Updates

I. INTRODUCTION

QA and RM are crucial for Virtual Reality applications since VR apps must meet specific requirements and deliver user-friendliness, stability, and peak performance. This remains vital as more advanced versions of VR platforms are developed and released to find and eliminate software glitches, achieve faster and more efficient rendering of scenes, and ensure compatibility across different structures. The Meta Quest platform, one of the most popular VR ecosystems, has a software update that can add new features, increase the speed and stability of the applications, and change the service's security settings. Coordinating these updates requires a systematic release management process to avoid much disruption time, compatibility with previous versions, and user difficulty.

This paper explains how QA and release management have been executed in Meta Quest updates, particularly about the difficulties of providing performance guarantees when delivering new features. It covers areas like automated testing tools, real-time performance monitoring, and the integration of users' feedback into update cycles. Through the examination of Meta Quest's approach in this regard, this study aims to identify the best practices together with trends present in the context of effective VR software maintenance, which would be helpful in the case comparative analysis of further and potential improvement in the efficiency of the QA practices applied on the example of VR ecosystems

II. OVERVIEW OF VIRTUAL REALITY (VR) PLATFORMS AND THEIR GROWING MARKET

Through rapid growth, VR technology has become mainstream and now helps different sectors, such as the gaming business, healthcare, and training. VR platforms deliver complete 3D environments through virtual sets that users access by wearing headsets supported by hardware like controllers and tracking devices.

A. Key VR Platforms in the Market

VR technology comes from a few top brands that design distinct features and user options:
The Meta Quest (formerly Oculus Quest) provides exceptional quality wireless standalone VR experiences for comprehensive virtual realities. The Meta Quest stands out in consumer VR due to its low cost, intuitive set-up, and great selection of VR games and apps.PlayStation VR is built as a VR system that brings high-quality games to Sony PlayStation consoles through its tailored experiences. The PSVR 2 system with PlayStation 5 integration delivers superior haptic effects and eye-tracking technology to users

.The HTC Vive provides top-tier VR performance that benefits businesses in educational and industrial applications.The Valve Index provides PC gaming specialists and developers the highest-quality virtual reality experience through its precise tracking.The Pico VR headsets and other standalone devices serve as Meta Quest competitors by offering standalone VR experiences for consumers and companies.

B. Growth of the VR Market

The VR industry grew steadily due to improvements in technology and experience content and its wider use across different business fields.The worldwide VR market is expected to grow past $50 billion during the next decade because customers want VR more and businesses use it better, plus improved technology keeps developing.People prefer VR video games like Beat Saber and Resident Evil 4 while enjoying Half-Life Alyx. Various online entertainment services and virtual concert performances now make VR technology more relevant in entertainment.

Businesses use VR technology to teach their employees, improve team collaboration, and enhance professional skills, mainly in medical, military, and engineering fields. Patients engage in advanced medical training, while students benefit from VR learning and mental healthcare treatment at school.The release of Meta Quest 3, alongside PlayStation VR 2 and Apple Vision Pro, makes VR equipment more advanced and affordable for users.

C. Challenges and Future Outlook

Although VR develops quickly, it deals with expensive hardware expenses combined with nausea problems while demanding content of higher quality. New technology advancements are taking the VR industry ahead and it shows strong signs of continued growth. Constant attention toward the metaverse and social VR proves that VR technology will strongly impact how people relate digitally going forward.

III. BACKGROUND ON QA AND RELEASE MANAGEMENT IN VR

QA and Release Management support VR application development by helping users experience the highest-quality virtual reality content. VR applications have different testing demands since their development requires specific techniques for hardware testing and overall performance enhancement.

A. Unique Challenges of QA in VR Applications

The quality assurance process for VR applications needs enhanced testing because it requires analysis of both realistic virtual environments and the interface between hardware and software systems. Some key challenges include:

VR programs must operate effectively on every computer headset model, especially Meta Quest 2 and 3 standalone headsets, PC-tethered systems like Valve Index and HTC Vive, and console versions such as PlayStation VR. Testing needs to happen across multiple devices since they provide unique processing abilities, display quality features, and separate motion sensors.

Having alarming frame rates or a camera moving unexpectedly with a long delay can make users feel sick during VR experiences. The testing process for QA teams includes a detailed evaluation of motion functions linked to frame rate performance and user controls to lower sickness risks.VR applications must deliver a stable frame rate above 72Hz 90Hz to create a seamless display for users. The system requires performance testing to discover where rendering and asset management consume too many resources, and adjustments are needed to fix these problems.

B. VR Differs from other devices

VR differs from other apps because it requires users to interact in space through their hands and controllers while also detecting their movements and vocal cues. Testing controls that work well, plus tracking and physics accuracy makes the testing process difficult to handle.

The effect of three-dimensional spatial audio helps deliver better virtual reality experiences to users. A QA team needs to test sound directional placement, audio balance, and environmental sound effects to build a believable VR sound environment.
VR quality evaluation works differently from testing regular software applications. The SD team tests virtual space immersion through live movement rather than using a monitor, and their physical setup differs from regular SD QA work.

Under VR, Quality Assurance testing covers both useability and gauges user comfort, virtual reality immersion, and mental strain. When users encounter poorly made interfaces, they cannot control them appropriately in VR, which affects their focus and leads to high levels of frustration.

VR applications work with hand tracking controllers, eye movement, and total body movement instead of typical screen input devices. These inputs need constant monitoring because they directly affect the system results.Our quality assurance team must test network connection systems and multiplayer social abilities. QA teams must evaluate how well-planned online sessions flow while testing server performance and real-time display accuracy.Itcarries its specific safety threats because prolonged headsets may cause discomfort and safety hazards. QA teams must find and resolve issues in both VR application accessibility and user safety to serve every customer.

C. The Role of Release Management in Maintaining a Seamless User Experience

VR development release management keeps users' experiences undisturbed by properly delivering planned content updates and fixes. Necessary release management areas include:

Belts need regular updates so users can enjoy faster performance fixes and new product features. Release management tests new product features before use to prevent release problems.New updates should stay compatible with today's computers, and all saved data plus personal settings to keep users happy in the long run.

The release manager conducts A/B testing with customers to test new features before deploying them across all systems. Understanding users' thoughts about the product enables us to enhance and perfect our service.When urgent bugs and security risks arise, platform owners need to release hotfixes promptly because user satisfaction depends on it, and the platform needs protection.Applications for VR need to follow platform rules and industry standards from the Meta Quest Store and PlayStation Store to get accepted for release.

IV. **QA STRATEGIES FOR VR PLATFORMS**

Virtual Reality needs strong Quality Assurance strategies to provide users with an excellent, functioning, and enjoyable experience. The intricate nature of VR technology demands combination testing methods, including automated and manual testing, complete issue monitoring, and user feedback systems. These are the main ways QA helps VR platforms.

A. Automated Testing Approaches

Automated testing plays a vital role in VR development since it helps manage the high testing demands of VR applications. The software quality test checks if the product works appropriately on all VR devices while meeting performance standards. These primary automated testing techniques exist within the testing framework.

1). Performance Benchmarking:

Essential VR system quality assurance needs performance measurements tracked by automatic tools. For quality VR experiences, you need applications that maintain at least 90 updates per second in their frame rate. Technology helps companies check the VR application's frame rate and performance metrics at all times. The testing process captures user habits in different environmental settings to show quality teams where performance issues will occur before launch.Benchmarking tools measure how much power CPUs and GPUs use and their heating impact when users interact with VR applications. Developers use performance tests to improve VR applications and boost performance, especially with Meta Quest 2's limited resources.

2). Compatibility Testing Across VR Hardware:

The system checks how well VR applications work on various VR hardware, including Meta Quest equipment and the Valve Index headset. The system checks if VR applications handle head tracking, hand movement recognition input reaction times, and graphic output properly, no matter what VR system users use.Evidence of VR software compatibility goes through tests designed to work on multiple platforms, including Meta Quest devices that run Android, while PC VR systems rely on Windows. Our testing system checks updates and product releases to avoid platform or operating system compatibility problems that upset users.

B. Manual Testing and User Experience Evaluation

Automation tests fix most of QA, but testers must examine VR user experience elements that automated tools do not match.

1). Motion Sickness and Comfort Assessments:

Motion sickness causes serious problems for people using VR applications. Measurement tools designed for computers cannot detect all elements of motion sickness within virtual reality environments. Real users performing manual tests let developers spot discomfort and illness from visual and motion problems.We put users through distinct actions they may perform in VR (faster movements, turning, and instant transportation) to see whether these activities ease them. Participants help us understand comfort during VR sessions by showing how much time they can use the system before feeling tired or exhausted.

2). Usability Testing with Real Users:

Usability testing examines if users find the VR application simple and easy to use. The testers assess whether the system's navigation works well and how easy it is to interact with virtual items, plus test the user interface's qualities. Because VR requires testers to navigate and interact in complete three-dimensional spaces, unlike regular computer software. Real users test essential workflow execution, reaction to the design elements, and response time of the system controls.Regular users detect essential accessibility problems such as unreachable buttons, challenging-to-read text in VR space, and hard-to-understand user controls for new people. Viable users, through testing, help develop VR systems for users with different abilities.

C. Bug Tracking and Issue Resolution

Strong bug detection and response systems help VR platforms reach high standards since VR programs interact between hardware and software components at many points.Regression Testing and Debugging VR-Specific Issues:

The testing process checks if new system changes preserve all existing features and fix any coding problems. Due to VR's sensitivity, VR developers must rigorously test old features after any new software updates to prevent issues from arising. These tests automate checks for head tracking, motion controls, and user input reactions on basic system operations.Specialized tools help developers inspect VR system data while finding VR hardware problems like controller misconfiguration and tracking errors. Complex VR problems need advanced climate testing methods and detailed analysis of hardware system reports.

V. CASE STUDY ANALYSIS: LESSONS FROM META QUEST UPDATES

The Meta Quest (previously Oculus Quest) is a global leader in VR systems with wireless interface, innovative features, and user-friendly characteristics. Upgrading the Meta Quest platform depends on persistent updates through an organized release management framework to preserve application and platform excellence. We will analyze Meta's past updates to examine the complications they encountered, followed by the quality assurance (QA) and release efficiency strategies and the effects on user interaction and platform stability.

A. Challenges Encountered in Previous Updates

The main difficulty in Meta Quest updates occurs when teams must maintain compatibility between various hardware versions. Developers need to guarantee that software supports older Quest devices while exploiting the capabilities of new model features.

VR applications need substantial system performance in addition to stable operation to keep users in an immersive state. Software updates tend to cause negative performance effects, which result in lower frame rates, graphical problems, and system breakdowns. The bugs caused significant problems during demanding applications with numerous interactions, and users reported poor quality of experience with constant disruptions.
VR systems function most effectively with seamless control because User Interface (UI) bugs remain among the most critical elements in designing interfaces for virtual environments.

New changes to Meta Quest systems caused fundamental problems with user interface elements, including buttons that were out of position. This compromised virtual reality controller communication and the registry of gestures.

These problems created undesirable user reactions and established an unstable condition for platform use.

System or feature changes occurred unintentionally through updates that removed or modified necessary attributes that users depended on, including social features, home screen customization options, and system abilities. The unanticipated changes to systems and features triggered dissatisfaction among users since their choice preferences and expectations failed to be considered.

The extensive worldwide deployment of updates affecting Meta Quest requires substantial time that users must tolerate alongside potential system downtime. System updates stretched over more time until complete user distribution because of insufficient release management practices that caused periodic system slowdowns.

B. Strategies Implemented to Improve QA and Release Efficiency

Meta used strategic initiatives to enhance quality assurance (QA) practices and release management procedures to overcome current difficulties.

Meta used significant funding to create automated tests, which enhanced the Quest ecosystem. The automated testing system executes multiple tests that check VR hardware setups while recording different user inputs under stressful scenarios. The testing method detected bugs in the early stages of the release period, and it focused mainly on device compatibility features and execution speed.

The platform conducted automatic tests dedicated to checking VR-centered technical elements, including framerate stability, 3D graphics functionality, and hand gesture management capabilities. Tests verified the similarities between VR-specific user interface elements, including menus, buttons, and virtual hand-tracking capabilities, across different devices.

Meta established an expanded beta testing system for developers and users through which upcoming features were tested before public release. The approach allowed Meta to collect genuine user input from people who experienced products in realistic situations so they could discover program problems and interface usability issues ahead of complete launches.

Meta monitored system performance through Canary Releases and Gradual Rollouts. This method allowed the company to test updates on a limited group before expanding the deployment to a larger audience. The technique enabled developers to reduce the bugs affecting the complete user population.During updates, Meta created stronger feedback interactions with its user community. Users now had enhanced feedback capabilities todirectly convey their feedback and experience reports to the development team. The team used user performance reports, specifically about hardware performance, as a key reference point to decide which bugs needed immediate fixes.

Meta created specific testing environments to test software performance and hardware compatibility, which duplicated interactions with various hardware platforms. The testing approach served as an essential compatibility evaluation method. Itlets the team confirm new updates fused without glitches across different device models to lower potential post-release defects.After release, the company implemented performance tracking tools to monitor software update real-time activities. Through these observation tools, Meta gained visibility into the functionality of its updates as they spread across different areas and targeted audiences on different device models. The proactive system tracking enabled fast detection of significant system problems before they could reach critical levels.

Meta enlarged its rollback capabilities through enhanced mechanisms to let users rapidly switch back to former stable versions whenever an update failure occurs. Rolling back updates proved essential in preserving platform stability because it solved performance and compatibility problems generated by new updates.

DISCUSSION

It is clear, therefore, that the ability to deliver effective QA and release management of Meta Quest updates reflects the general experiences of high-performance VR ecosystems. The update process indicates automatic and manual testing should be used for updates with less interference or shorter interruption on the site's user interface. Whereas automated testing involves giving tools to help determine system weaknesses, slow or sluggish areas, and compatibility problems, manual testing offers users an experience of how the whole process feels to be a user or client in terms of immersion. This way, the two approaches guarantee VR applications present a stable and low latency performance, besides optimizing the graphical rendering for various hardware platforms.

There are two main concerns regarding managing the releases in VR creation and sustenance. These risks are managed in Meta Quest through its well-designed release pipeline, which uses a beta test and rolls out more updates. This way, developers can build updates based on evidence rather than previous experience and thus have a minimal number of bugs when transitioning between releases of a program. Assessing user feedback is essential in determining the proper QA methods for Meta Quest. The developers can identify the most critical issues that should be brought to the forefront and improved by analyzing the specter of errors, user feedback, and telemetry data. The performance indicators enable monitoring of latency, fluctuations in the frame rate, and rendering, where real-time control can be made to improve the users' experience. Also, the software updates pass through compatibility tests across different VR systems to avoid developing the ecosystem into various platforms that may not be compatible.

Regarding the discussion presented in the study, I would also like to point out that the impact of new forms of hardware and software technologies as a part of VR on the practice of VR QA is also touched upon. Due to the advent of such features, the QA teams need to design new ways of evaluating these elements' effectiveness and ensure that they are compatible with the previous applications and tools.

FUTURE WORK

Thus, future studies should look for a better solution regarding QA and release management issues that may arise with the evolution of VR platforms. Another area with a lot of room for advancement is the use of AI for test automation to improve bug detection as well as adjust and optimize test settings and determine potential failures before the system is deployed. For instance, by using big data analysis at scale, QA teams can verify specific interaction data, from which a model can be learned when problems occur and maintenance plans are developed. More research is also required to examine the role of blockchain in the release management process to improve trust. Blockchain can include proof of logs of all updates done on the software without the possibility of being altered or changed in any way. This would be especially beneficial to large-scale organizations that are involved in virtual reality-based applications that are heavily regulated to ensure accuracy and dependency on data. The growing complexity of VR hardware and software ecosystems necessitates improvements in cross-platform testing methodologies. Future research works must explore the extensibility of compatibility testing frameworks mainly because of the ability to update VR applications depending on the end-user's hardware specifications that effectively reduce fragmentation.

CONCLUSION

In conclusion, managing QA and release processes for VR platforms, such as Meta Quest, presents unique challenges due to the hardware's complexity, the applications' immersive nature, and the wide range of user expectations. The case study of Meta Quest updates highlights the critical importance of ensuring performance stability, compatibility, and an optimized user experience across different hardware models. Meta has demonstrated the significance of implementing robust strategies for automated testing, beta programs, and real-time monitoring by addressing challenges such as compatibility issues, UI bugs, and performance hiccups.

Furthermore, Meta's focus on improving its QA and release efficiency through gradual rollouts, community-driven feedback, and AI-powered testing reflects the growing importance of technological advancements in the VR ecosystem. These strategies mitigate risks associated with new releases and enable swift resolution of issues, minimizing user disruptions.

As the VR industry continues to evolve, maintaining a proactive and adaptive approach to QA and release management will ensure the long-term success of platforms like Meta Quest. By continuously refining processes and embracing innovative solutions, Meta will continue to provide a stable, enjoyable, and immersive experience to its diverse user base. Ultimately, the lessons learned from Meta Quest's release management practices can serve as a valuable guide for other VR platforms striving to meet the growing demands of the VR community while delivering exceptional user experiences.

REFERENCES

[1]. MS Anwar, A Choi, S Ahmad, K Aurangzeb, et al. (2024). *A Moving Metaverse: QoE challenges and standards requirements for immersive media consumption in autonomous vehicles.* Applied Soft Computing, Elsevier.

[2]. L Basyoni, A Tabassum, K Shaban, et al. (2024). *Navigating Privacy Challenges in the Metaverse: A Comprehensive Examination of Current Technologies and Platforms.* IEEE Internet of Things Journal, IEEE.

[3]. AYC Hou, IJG Lu, C Hill (2022). *Impact of COVID-19 on driving digitalization, innovation and crisis management of higher education and quality assurance?—A Taiwan case study in alignment with …* Higher Education Policy, NCBI.

[4]. K Mok (2005). *The quest for world class university: Quality assurance and international benchmarking in Hong Kong.* Quality Assurance in Education, Emerald.

[5]. T Şimşek, MA Öner, Ö Kunday, GA Olcay (2022). *A journey towards a digital platform business model: A case study in a global tech-company.* Technological Forecasting and Social Change, Elsevier.

[6]. L Ortiz (2022). *Risks of the metaverse: A VRChat study case.* The Journal of Intelligence, Conflict, and Warfare, journals.lib.sfu.ca.

[7]. D Nielsen (2025). *Virtual Migrations and Meta-Innovations: Update Struggle in No Man's Sky.* ACM Games: Research and Practice, ACM.

[8]. C Mayr-Dorn, M Vierhauser, S Bichler, et al. (2021). *Supporting quality assurance with automated process-centric quality constraints checking.* IEEE/ACM International Conference, IEEE Xplore.

[9]. F Cabitza (2015). *On a QUESt for a web-based tool promoting knowledge-sharing in medical communities.*Behaviour& Information Technology, Taylor & Francis.

[10]. S Mäkinen, M Leppänen, T Kilamo, AL Mattila, et al. (2016). *Improving the delivery cycle: A multiple-case study of the toolchains in Finnish software-intensive enterprises.* Information and Software Technology, Elsevier.

[11]. M Nayebi (2018). *Analytical Release Management for Mobile Apps.* Prism, University of Calgary.

[12]. A Bueckle, C Qing, S Luley, Y Kumar, et al. (2023). *The HRA Organ Gallery affords immersive superpowers for building and exploring the Human Reference Atlas with virtual reality.* Frontiers in [Science/Technology].

[13]. JA Jary, AF Xaver (2023). *Microsoft's Acquisition of Activision Blizzard: A Power Play to Increase Shareholder Value: The Quest to Increase Shareholder Value.*Search.ProQuest.

[14]. FA Ghansah, DJ Edwards (2024). *Digital Technologies for Quality Assurance in the Construction Industry: Current Trend and Future Research Directions towards Industry 4.0.* Buildings, MDPI.

[15]. JW Fresen (2006). *Quality assurance practice in online (web-supported) learning in higher education: An exploratory study.* University of Pretoria.

[16]. Q Terry, S Keeney (2022). *The Metaverse Handbook: Innovating for the Internet's Next Tectonic Shift.* John Wiley & Sons.

[17]. M Wu, C Chen (2023). *VR Force Feedback Gloves.*DigitalCommons@CalPoly.

[18]. L Leite, G Pinto, F Kon, P Meirelles (2021). *The organization of software teams in the quest for continuous delivery: A grounded theory approach.* Information and Software Technology, Elsevier.

[19]. M Popović, B Atlagić, et al. (2001). *Case study: A maintenance practice used with real-time telecommunications software.* Journal of Software Maintenance, Wiley Online Library.

[20]. AI Koka (2015). *Software quality assurance in Scrum projects: A case study of development processes among Scrum teams in South Africa.* CORE.ac.uk.

[21]. V Chheang, S Narain, G Hooten, R Cerda, B Au, et al. (2024). *Enabling additive manufacturing part inspection of digital twins via collaborative virtual reality.* Scientific Reports, Nature.

[22]. S Sahay, B Nicholson, S Krishna (2003). *Global IT outsourcing: software development across borders.* Books.Google.com.

[23]. P Ayoup, DE Costa, E Shihab (2022). *Achievement unlocked: a case study on gamifying DevOps practices in industry.* Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering.

[24]. J Arnedo-Moreno, KML Cooper, D Lin (2024). *Emerging Advanced Technologies for Game Engineering.* ACM SIGSOFT Software Engineering Notes.

[25]. JM Garon (2022). *Legal implications of a ubiquitous metaverse and a Web3 future.* Marquette Law Review, HeinOnline.

[26]. D Russo, P Ciancarini, T Falasconi, et al. (2018). *A meta-model for information systems quality: A mixed study of the financial sector.* ACM Transactions on Management Information Systems.

[27]. M Uddin, M Obaidat, S Manickam, et al. (2024). *Exploring the convergence of Metaverse, Blockchain, and AI: A comprehensive survey of enabling technologies, applications, challenges, and future directions.* Data Mining and Knowledge Discovery, Wiley Online Library.

[28]. A Shrestha, F Zuo, G Qian, J Rhee (2024). *A Survey and Insights on Modern Game Development Processes for Software Engineering Education.* International Conference on Software Engineering Education, Springer.

[29]. J Pesola (2016). *Implementing Continuous Integration in a Small Company: A Case Study.* Academia.edu.

[30]. M Furmakiewicz, C Liu, A Taylor, I Venger (2024). *Design and evaluation of AI copilots—case studies of retail copilot templates.*arXiv preprint arXiv.org.