Unified Modular Architectural and Software Model for Educational XR Systems

Miroslav GALABOV
Computer Systems and Technologies Department,
"St. Cyril and St. Methodius" University of Veliko Tarnovo, Veliko Tarnovo, Bulgaria

Abstract—**This article presents the development and validation of a unified architectural and software model for educational Extended Reality (XR) systems, which integrates the technological, pedagogical, and interface-related aspects of virtual and augmented reality into a coherent, modular, and interoperable framework. The proposed model addresses key limitations of existing XR solutions in education—such as platform dependency, limited scalability, and insufficient pedagogical grounding—through the use of WebXR technologies and open standards. The architecture is designed as a multi-layered structure that ensures cross-platform access, real-time interactivity, adaptive levels of immersion, and seamless integration with learning management systems. A conceptual model is formulated that explicitly links architectural design decisions to instructional objectives and establishes the principle of pedagogically determined design of XR learning environments. The applicability of the proposed model is evaluated through the design, implementation, and functional testing of a prototype WebXR-based educational system for engineering education. The results demonstrate that the unified architecture provides a sustainable, scalable, and pedagogically grounded foundation for the effective integration of XR technologies in educational contexts.**

Keywords—Extended reality; virtual reality; augmented reality; educational XR systems; software architecture; WebXR; immersive learning; modular architecture; interactive simulations

I. INTRODUCTION

Virtual reality (VR) and augmented reality (AR) are transforming traditional models of learning by providing learners with opportunities to explore complex concepts through immersive experiences. Instead of passively receiving information, students can actively manipulate and observe learning objects within a three-dimensional environment. These approaches contribute to deeper understanding, increased motivation, and the development of practical skills [5, 7, 15, 20].

Virtual reality represents a computer-generated simulated environment that enables active user participation through immersion and interaction within a three-dimensional space [25, 27]. It is part of the broader family of extended realities (XR), which also includes augmented reality (AR) and mixed reality (MR) [2, 3, 21, 26].

In education, XR is regarded as a tool for creating immersive learning experiences that enhance engagement, improve comprehension, and support personalized learning [14, 19, 22, 23].

The semantic representation of 3D objects, implemented through ontologies and standardized descriptors, enables not only more efficient organization and retrieval of educational resources but also automated adaptation of content to individual learner needs. In the work of Georgieva-Trifonova and Galabov [9], a conceptual framework is proposed in which 3D models are described using semantic web technologies such as OWL and RDF, making it possible to link geometric characteristics with educational context. The authors demonstrate that by developing an ontology for educational 3D objects, contextual interpretation of learning content can be effectively supported.

A. Relevance of the Problem

Over the past decade, virtual and augmented reality (VR/AR) technologies have steadily penetrated a wide range of industrial and scientific domains. In the educational context, the adoption of XR technologies creates conditions for the development of a new generation of software systems that integrate 3D visualization, interactivity, and sensory interfaces into the learning process [8]. This trend necessitates the design of stable and scalable architectures capable of ensuring reliable real-time performance and compatibility with diverse hardware configurations.

Existing educational XR solutions are often constrained by high hardware requirements, as well as by the lack of unified software interfaces and optimized formats for 3D content. Furthermore, the integration of XR into educational environments requires effective communication among system modules, including tracking algorithms, haptic feedback mechanisms, scene generators, and user interaction components.

The present article addresses this technological gap through the development of an architecture and a prototype of an educational XR system built on a modern development platform (WebXR), supporting adaptive interfaces and true scalability. The study examines in-depth issues related to the selection of file formats for scenes and objects (glTF, OBJ, FBX), challenges in synchronization and motion tracking, and the impact of hardware configuration on the quality of user experience [30].

Given the rapidly increasing significance of cloud computing and web-based XR solutions, the proposed development also includes an analysis of integration possibilities with cloud services, content storage, and remote access to XR applications. The employed technologies adhere to principles of modularity, portability, and compliance with open standards, making the proposed prototype applicable in both local and distributed learning environments.

The relevance of the problem arises not only from the rapid advancement of XR technologies but also from the need for clearly defined solutions for their sustainable integration into education. In this context, the present study aims to bridge the gap between theoretical developments and the practical implementation of effective, scalable, and accessible XR systems.

B. Objective of the Study

The primary objective of this article is to develop and scientifically substantiate a unified architectural and software model for educational XR (Extended Reality) systems that integrates the technological, pedagogical, and interface-related aspects of virtual and augmented reality, and to validate its applicability through the development and experimental evaluation of a prototype XR system in a real educational environment.

II. THEORETICAL SYNTHESIS AND CONCEPTUAL MODEL FOR THE ARCHITECTURAL DESIGN OF EDUCATIONAL XR SYSTEMS

The theoretical foundations and technological classifications of virtual, augmented, and extended reality examined in this section reveal a considerable diversity of approaches, platforms, and levels of immersion. However, the existing scientific literature lacks a unified conceptual framework that systematically links the degree of immersiveness, architectural decisions, and pedagogical objectives of educational XR systems. This results in fragmented implementations, often optimized for a specific platform or hardware configuration, but difficult to transfer and scale across different educational contexts.

In this sense, the present study is based on the understanding that XR technology in education should not be viewed merely as an isolated visualization environment, but rather as a complex sociotechnical system in which technological, cognitive, and pedagogical components are functionally interconnected. The theoretical analysis indicates that the effectiveness of XR-based learning does not depend directly on the level of technological complexity, but on the alignment between the system's architectural design and the instructional objectives it serves [4, 6].

Based on the reviewed theories of immersiveness, presence, and cognitive load, the principle of pedagogically determined architectural design can be formulated. According to this principle, the selection of the XR platform, visualization type, and interaction mechanisms should be driven by the didactic function of the learning environment, rather than the reverse. This implies a clear distinction between scenarios that require full immersion and unrestricted spatial interaction, and those in which semi-interactive or web-based XR solutions are more appropriate from the perspectives of cognitive efficiency and resource accessibility.

From a theoretical standpoint, the architecture of educational XR systems can be conceptualized as a multi-layered structure comprising:

- Layer for sensory interaction and visualization;
- Layer for interaction logic and simulation;
- Layer for pedagogical scenarios and instructional control;
- Layer for assessment, adaptation, and feedback.

This stratification enables a clear separation of responsibilities between technological and pedagogical components and creates prerequisites for modularity, interoperability, and content reuse. In this context, web-based XR technologies stand out as an architectural solution with strong theoretical potential, as they allow the integration of different levels of immersion within a unified logical structure without dependence on a specific hardware platform.

Based on this theoretical synthesis, a generalized model can be formulated (Fig. 1), according to which sustainable educational XR systems are characterized by architectural adaptability that enables dynamic regulation of immersion, interactivity, and content complexity in accordance with learners' age, cognitive level, and instructional objectives.

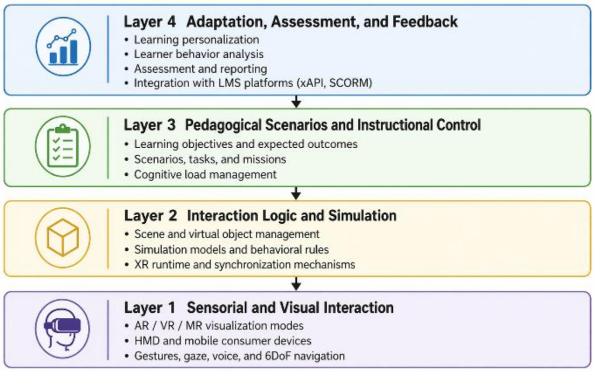

Fig. 1. Conceptual model for the architectural–pedagogical design of educational XR systems.

This model emphasizes the functional relationship between architecture and pedagogy and serves as the theoretical foundation for the proposed unified architectural and software approach.

This model can be formally defined as the principle of pedagogically determined architectural design, according to which architectural decisions related to the level of immersiveness, interaction mechanisms, and system complexity should be derived from instructional objectives, learner characteristics, and cognitive constraints, rather than solely from available technological capabilities.

III. ARCHITECTURAL MODELS OF XR SYSTEMS

Modern educational systems based on virtual reality employ a variety of architectural models tailored to instructional objectives, the number of users, available technical resources, and the required level of interactivity. The choice of architecture has a significant impact on system performance, scalability, and the ability to integrate with existing educational platforms.

Recent studies explore the use of large language models (LLMs) as a means of supporting architectural decision-making through the automatic generation of design rationales and alternative solutions; however, such approaches are complementary in nature and cannot replace the need for clearly defined architectural models in safety-critical and educational XR systems [32].

Client–server model – One of the most commonly used approaches in multi-user virtual environments. In this model, a central server manages the state of the virtual world, synchronizes the actions of individual users, and allocates resources such as positioning, interactions, and real-time events. This enables the simultaneous participation of multiple learners, who can collaborate or compete within a shared environment. A typical example is virtual laboratories with remote access used in engineering education [24].

Local systems – These systems represent fully standalone solutions executed directly on VR devices such as Meta Quest or HTC Vive Focus. They are particularly suitable for individual learning, as they do not require a constant internet connection and allow a high degree of personalization. Applications of this type often include simulators for specific skills, such as surgical training systems, laboratory experiments, or immersive language learning environments. The main advantages of this model are low latency and high interactivity, while its limitations stem from hardware constraints and the lack of shared sessions.

Cloud-based architectures – In this model, VR content and computational processes are hosted in a cloud infrastructure, while access is provided through streaming of video and data to client devices. This enables the use of less powerful end devices—such as laptops, tablets, or even mobile phones—to access high-quality VR experiences. Cloud architectures facilitate centralized content management, updates, and user data analytics, and they allow easy scalability in educational institutions with large numbers of users [31]. Their disadvantages include dependence on a stable internet connection and potential latency issues in lower-quality networks [10].

Regardless of the selected architectural model, every XR system includes a set of core functional modules, including:

- Rendering – responsible for the visualization of the three-dimensional environment, including lighting simulations, textures, and animations;
- Input–output communication – enables interaction with the user through devices such as controllers, voice commands, hands, eyes, or gestures;
- Interaction logic – defines how user actions affect the environment, including object behavior, events, and scenarios;
- Integration with Learning Management Systems (LMS) – through standards such as SCORM, xAPI, and LTI, XR systems can exchange data on learner progress, performance, and activity with platforms such as Moodle, Blackboard, and others [1, 11, 12, 19].

The synergy among these modules ensures smooth and effective educational communication between the learner, the virtual environment, and instructors, while simultaneously meeting contemporary requirements for personalized, scalable, and accessible learning.

A. Critique of Existing Architectural Models of XR Systems

Despite the diversity of architectural approaches used in contemporary VR and XR systems, the analysis indicates that a significant portion of existing solutions suffer from conceptual and practical limitations that render them partially or entirely unsuitable for sustainable application in educational contexts.

First, many of the reviewed systems are monolithic in their architecture, with rendering, interaction logic, user management, and instructional scenarios tightly coupled within a single application. This monolithic design hinders adaptation to diverse instructional objectives, limits component reuse, and complicates integration with external educational platforms.

Second, a large proportion of VR architectures are closed and tightly bound to specific hardware and software ecosystems, resulting in vendor lock-in. This constrains cross-platform deployment, increases implementation and maintenance costs, and creates barriers to the widespread adoption of VR solutions in educational institutions.

An additional issue is that many architectural models were originally designed for entertainment or industrial applications, without explicitly addressing the pedagogical requirements of education. As a consequence, mechanisms for defining learning objectives, instructional scenarios, assessment, and progress tracking are often absent, which limits their effectiveness as comprehensive educational tools.

From a scalability perspective, existing solutions frequently prove difficult to deploy for large numbers of users, particularly in university environments. Centralized client–server models may introduce performance bottlenecks, while local and hardware-dependent systems do not support efficient scaling or remote access. Cloud-based architectures partially mitigate these challenges but introduce new constraints related to latency, network dependency, and operational costs.

In summary, existing architectural models of VR systems demonstrate a lack of a unified, modular, and pedagogically oriented approach that simultaneously ensures accessibility, scalability, interoperability, and integration with educational systems. These limitations motivate the need for the development of a new architectural model capable of overcoming the identified weaknesses and addressing the specific requirements of educational XR environments—a topic examined in detail in the following section of the article.

B. Comparative Analysis with Existing XR Frameworks

In order to further substantiate the identified limitations of existing XR architectural approaches, a comparative analysis was conducted between widely used XR development frameworks and the proposed unified architectural model (Table 1). The comparison focuses on key characteristics relevant to educational XR systems, including platform dependency, integration with learning management systems (LMS), accessibility, pedagogical support, and scalability.

TABLE I. COMPARATIVE ANALYSIS OF XR FRAMEWORKS AND THE PROPOSED MODEL

Feature	Unity XR	Unreal Engine	WebXR (existing)	Proposed Model
Platform dependency	High	High	Medium	Low
LMS integration	Limited	Limited	Limited	Full (xAPI/LTI)
Web-based access	No	No	Yes	Yes (native)
Pedagogical layer	No	No	Partial	Explicit
Scalability	Medium	Medium	High	High

The comparison demonstrates that existing XR frameworks primarily emphasize rendering performance and interaction capabilities, while pedagogical integration is either implicit or absent. This limitation reduces their applicability in structured educational environments, where instructional design, assessment, and learner feedback are essential components.

In contrast, the proposed model introduces an explicitly defined pedagogical layer, which enables the formalization of instructional scenarios, learning objectives, and evaluation mechanisms. Additionally, the integration with LMS platforms through standardized protocols such as xAPI and LTI supports bidirectional data exchange and facilitates the tracking of learner progress and performance.

Another significant advantage of the proposed approach is the use of WebXR technologies, which reduces platform dependency and enables seamless cross-device accessibility through standard web browsers. This overcomes one of the major constraints of traditional XR frameworks, which are often tightly coupled to specific hardware and software ecosystems.

Based on this analysis, it becomes evident that the proposed unified architectural model addresses key limitations of existing XR frameworks by combining technological flexibility, pedagogical structuring, and interoperability. These characteristics provide a strong foundation for the development of scalable and accessible educational XR systems, as further elaborated in the following section.

C. Principled System Architecture

The proposed architecture follows a design-based research approach, in which theoretical principles derived from pedagogical and cognitive theories are translated into concrete architectural components and validated through iterative prototyping and functional testing in a realistic educational context. Based on the analysis of platforms and architectural approaches for XR systems [18], a unified modular architectural model for an educational XR system implemented via WebXR is proposed. The model aims to provide a technological and pedagogical foundation for the development of XR solutions that integrate virtual and augmented reality within a single architecture and support interactive learning scenarios for the demonstration and practice of engineering concepts (e.g., microcontrollers, sensor circuits, and control logic). The design is oriented toward achieving: 1) high interactivity and low latency in real-time operation, 2) cross-platform access through a standard web browser, and 3) integration with learning platforms and accountability mechanisms [29].

1) Technical specification of architectural layers and their interactions: To overcome the conceptual nature of the architectural presentation and to achieve a higher degree of formalization, this section provides a detailed technical specification of the three main architectural layers—technological, pedagogical, and interface—as well as the mechanisms governing their interactions. Particular emphasis is placed on the application programming interfaces (APIs), data flows, and interoperability protocols that ensure the functioning of the system as an integrated XR environment.

The technological layer encompasses the hardware devices and the execution environment, implemented through the WebXR API. It provides a unified interface to XR device functionalities via standard methods such as requestSession(), getViewerPose(), and requestAnimationFrame(), which manage the creation of XR sessions, retrieval of user position and orientation, and real-time rendering synchronization. Input data at this layer includes spatial coordinates of the head and hands, controller states (buttons and axes), as well as events generated by gestures and voice commands. The output consists of transformation matrices and event streams that are transmitted to the visualization and scene management layer. Communication with higher layers is realized through a continuous rendering loop (frame loop), ensuring low latency and real-time synchronization.

The pedagogical layer implements the instructional logic and includes modules for managing learning scenarios, defining objectives, tracking progress, and evaluating outcomes. This layer operates as a central coordination mechanism that processes events generated by the technological layer and interprets them within the context of specific learning tasks. Data processing is carried out through a scenario engine that manages transitions between instructional states, as well as through simulation modules that model the behavior of virtual objects and processes. Data exchange with external learning management systems is achieved through standardized protocols such as xAPI and LTI, enabling bidirectional communication of learner actions, task completion time, and performance results.

The interface layer provides direct communication between the user and the system through visual, audio, and interactive means. It includes graphical components implemented using HTML/CSS and WebGL, as well as mechanisms for multimodal input, including controllers, gestures, and voice (via the Web Speech API). This layer transforms user actions into events transmitted to the pedagogical layer and visualizes processing results in the form of feedback (visual, auditory, or textual). In this way, a closed interaction loop is established, supporting the cognitive process and user orientation within the XR environment.

Interaction between the layers is implemented through an event-driven architecture, in which input events generated at the hardware level are propagated through the technological layer, processed by the pedagogical layer, and result in state changes that are visualized through the interface layer. The data flow includes transformations of spatial coordinates, logical states of objects, interaction events, and telemetry data for analysis and assessment. To ensure interoperability and extensibility, open web standards such as WebXR, WebGL, and REST-based interfaces are employed, enabling integration with external services and cloud platforms.

The formalization of these interactions is illustrated in Fig. 2, which presents the layered structure, core components, and data flows of the proposed XR system.

The hardware and device layer (A) provides the sensory channels and actuation mechanisms required for immersion and interaction, including XR devices (HMDs, controllers, inertial measurement units, and cameras), as well as audio output and microphones for voice interaction. This layer generates the primary input data for the system—head and hand position and orientation, as well as events originating from controllers, gestures, and/or voice commands.

The XR runtime and device abstraction layer (B) is implemented via WebXR. Its function is to manage XR sessions, reference spaces, and frame synchronization, while providing a unified programming interface to the functionality of different XR devices. Through this layer, support is ensured for both VR and AR modes, as well as a fallback 2D mode in the absence of XR support, which enhances the accessibility and robustness of the system across diverse educational settings [30].

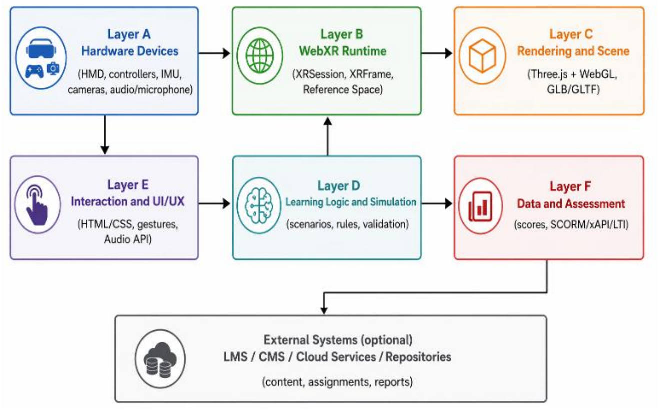

Fig. 2. Block diagram of the unified architectural model of an educational XR system.

The visualization and scene management layer (C) is responsible for real-time rendering of the three-dimensional environment. It employs WebGL as the graphical backend and Three.js as the engine for managing scenes, materials, lighting, and animations. Learning objects are represented through optimized 3D assets in GLB/GLTF format (e.g., models of microcontrollers, LEDs, sensors, etc.). The output of this layer is a visualization synchronized with user movements and optimized to meet requirements for low latency and stable frame rates [16, 17, 29].

A central role in the architecture is played by the learning logic and simulation layer (D), which implements the pedagogical scenario and the functional logic of the learning process. This layer manages the sequence of instructional steps, task objectives, conditions for success or failure, and transitions between system states. It includes a scenario engine, a learning object model, and a simulation module for microcontroller-related actions. Through integration with tools such as Arduino IDE, Tinkercad Circuits, or Firmata, software-based simulation of real engineering processes is enabled, with results fed to the visualization and interface layers in the form of events, states, and indicators.

The interaction and user interface layer (E) provides direct communication between the learner and the system. It includes interface elements (HTML/CSS) for instructions, hints, and control panels, as well as multimodal input through controllers, gestures, and voice. Feedback to the learner is delivered via visual cues, audio notifications, and voice guidance (Web Speech API and Audio API), supporting orientation and aiding the management of cognitive load during interaction within the XR environment [13, 28].

For analysis and assessment purposes, the architecture incorporates a data, tracking, and accountability layer (F), which records learner actions, task completion time, errors made, and hints used. These data enable internal analysis of learning effectiveness and, when required, integration with external learning management systems through standards such as SCORM, xAPI, or LTI [1, 11, 12].

The operation of the system is realized through key data flows, which are presented in Table 2.

IV. SOFTWARE ARCHITECTURE OF THE SYSTEM

In order for the technical components to operate, a complete set of software modules is required to manage the system elements and transmit data from input devices to higher-level logic responsible for the augmented reality visualization system. A schematic representation of the software architecture is shown in Fig. 3. The kernel, together with system libraries and device drivers, forms the core of the operating system. The runtime environment is often included as part of the operating system package, although this is not mandatory. In embedded operating systems, the runtime environment is typically minimal and is used mainly for diagnostics and maintenance, while all input–output interfaces are managed directly by the application software.

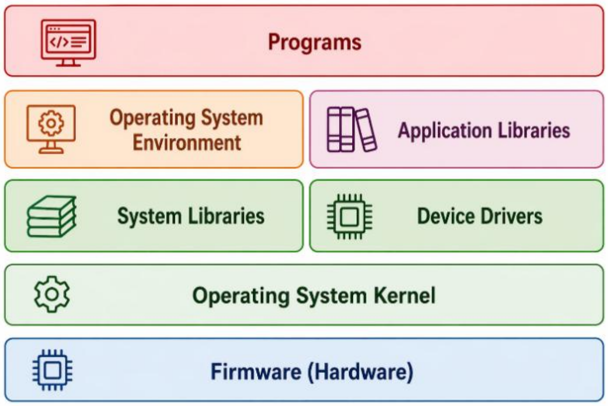

Fig. 3. Software architecture of the system.

At the software level, the system is implemented through several modules that operate jointly to visualize augmented reality content.

The first module is responsible for calibrating the selected camera. It is activated when the device is changed or when recalibration is required. This module guides the user through the calibration process and stores the camera parameters in the device's persistent memory so that they are accessible to the other system modules.

The second module is responsible for real-time tracking of camera motion and for computing the three-dimensional geometry of the observed scene. It utilizes stored camera parameters. Its output includes the position and orientation of the camera in 3D space, as well as extracted data describing the scene geometry.

The third module handles the visualization of the live camera feed and the overlay of augmented content. It uses the camera position and calibration data to create a virtual camera equivalent to the real one, which "views" the AR scene from the same perspective and renders it on the display.

Each of these modules is implemented as a set of libraries and application programming interfaces (APIs), enabling code reuse, modular development, and straightforward extension with new functionalities or applications. This structure also allows independent testing of each system component.

The overall algorithm for visualizing the final output is presented in Fig. 4.

The advantage of the XR approach is that the learner is not isolated from the physical world, as is the case in a conventional VR environment. Instead, the learner can see the real room, their own hands, and surrounding objects, which prevents disorientation and allows virtual devices to be naturally positioned within context (Fig. 5). XR visualization provides a higher degree of realism in the demonstration of sensor technologies, as learners can observe how measurements change in response to real objects placed in front of the sensor.

The prototype is implemented as a web-based XR application using JavaScript and WebGL technologies, executed in a standards-compliant browser environment and tested on Meta Quest 2 devices supporting the WebXR API. The source code for the prototype XR system can be downloaded from GitHub: https://github.com/mgalabov/vr-demo/blob/main/MG_AR.html

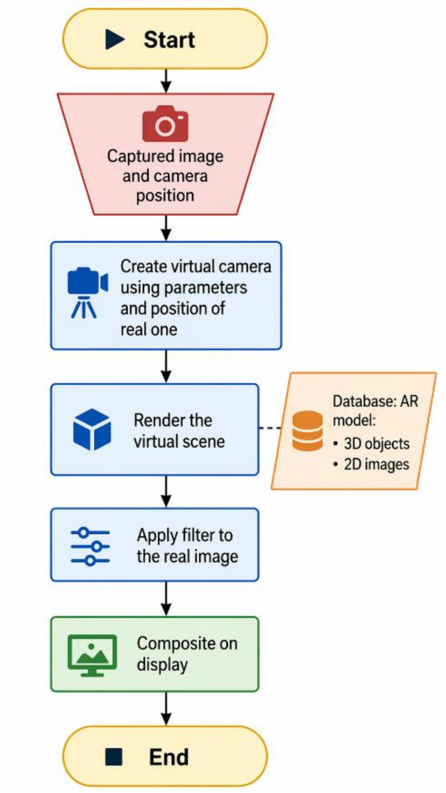

Fig. 4. Augmented reality rendering algorithm.

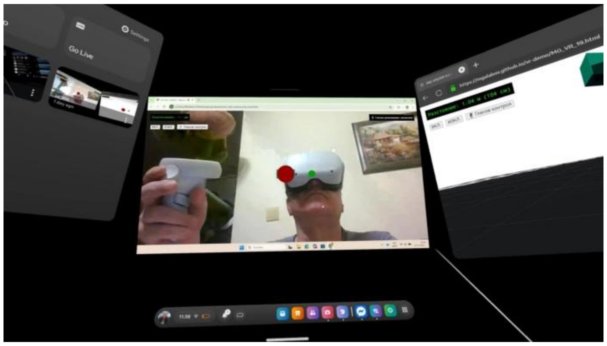

Fig. 5. XR visualization system using Meta Quest 2.

TABLE II. KEY DATA FLOWS IN A WEB-BASED XR EDUCATIONAL SYSTEM

No.	Data Flow Name	Data Source	Intermediate Layers / Modules	Output / Result	Functional Purpose
1	Real-time Rendering and Synchronization	XR hardware (HMD, position and orientation sensors)	WebXR API → visualization engine (Three.js / Unity Web)	Rendered scene in the HMD	Ensuring smooth visualization, spatial accuracy, and real-time synchronization of the virtual scene with the user's movements
2	Interaction Processing and Instructional Logic	Input events (gestures, controllers, gaze, voice)	Interaction layer → scenario engine / instructional logic	Updated scene and visual/audio feedback	Managing user actions, executing pedagogical scenarios, and maintaining interactivity within the learning process
3	Microcontroller Action Simulation	Learner actions in the virtual environment	Simulation module (virtual microcontroller, logical models)	Simulation results and XR indications	Reproducing the behavior of real electronic components and systems to support hands-on learning without physical hardware
4	Assessment and Reporting	Interaction events and simulation results	Telemetry and assessment module	Reports, progress indicators, and/or LMS integration	Tracking learning activity, evaluating performance, and integrating with external educational platforms

V. CONCLUSION

A significant scientific contribution of this work is the development and validation of a unified architectural and software model for educational XR systems that integrates technological, pedagogical, and interface components into a single, interoperable framework.

A unified modular architectural model of an XR system has been developed, defining the logical dependencies among input–output devices, the user interface, the visualization engine, and the control module for managing real-time interactions. The model presents a generalized approach to XR system design, reveals patterns in the processing of sensory data and their transformation into visual–interactive representations, and serves as a theoretical foundation for the development of scalable XR architectures.

Based on this model, a prototype XR system has been developed with the aim of practically validating the proposed unified architectural model for educational XR environments through real-world implementation and experimental use. The system is designed as an interactive learning environment focused on the demonstration, practice, and analysis of engineering and technological processes, with particular emphasis on microcontroller- and sensor-based systems.

Future work will focus on large-scale empirical validation of the proposed model through controlled educational experiments, quantitative analysis of learning data, and long-term deployment in higher education environments, as well as on extending the architecture with adaptive and artificial intelligence–based instructional components.

REFERENCES

[1] ADL Initiative. Sharable Content Object Reference Model (SCORM) 2004 4th Edition – Overview. Advanced Distributed Learning Initiative. 2009. https://adlnet.gov

[2] Azuma, R. T. A Survey of Augmented Reality. Presence: Teleoperators and Virtual Environments, 6(4), 1997, pp.355–385. https://doi.org/10.1162/pres.1997.6.4.355

[3] Billinghurst, M., Clark, A., Lee, G. A. A Survey of Augmented Reality. Foundations and Trends in Human-Computer Interaction, 8(2–3), 2015, pp.73–272. https://doi.org/10.1561/1100000049

[4] Bowman, D. A., MCmahan, R. P. Virtual Reality: How Much Immersion Is Enough? Computer, 40(7), 2007, pp.36–43. https://doi.org/10.1109/MC.2007.257

[5] Conrad, M., Kablitz, D., Schumann, S. Learning effectiveness of immersive virtual reality in education and training: A systematic review of findings. Computers & Education: X Reality, 2024, 4, 100053. https://doi.org/10.1016/j.cexr.2024.100053

[6] Cummings, J. J., Bailenson, J. N. How immersive is enough? A meta-analysis of the effect of immersive technology on user presence. Media Psychology, 2016, 19(2), 272–309. https://doi.org/10.1080/15213269.2015.1015740

[7] Dede, C. Immersive Interfaces for Engagement and Learning. Science, 323(5910), 2009, pp.66–69. https://doi.org/10.1126/science.1167311

[8] Galabov, M., Samarescu, N. Virtual and augmented reality in learning, Faber publishers, 2025, ISBN:978-619-00-1971-8.

[9] Georgieva-Trifonova, T., Galabov, M. Transforming 3D models to semantic web representation. Romanian Journal of Information Science and Technology, 26(1), 2023, pp. 33–48.

[10] Ibáñez, M.-B., Delgado-Kloos, C. Augmented Reality for STEM Learning: A Systematic Review. Computers & Education, 123, 2018, pp.109–123. https://doi.org/10.1016/j.compedu.2018.05.002

[11] IEEE. IEEE Standard for Learning Technology—Experience API (xAPI) (IEEE Std 9274.1.1-2023). Institute of Electrical and Electronics Engineers,2023. https://standards.ieee.org

[12] IMS Global Learning Consortium. Learning Tools Interoperability® (LTI®) Core Specification Version 1.3.2022. https://www.imsglobal.org

[13] Jerald, J. The VR Book: Human-Centered Design for Virtual Reality. ACM Books, 2015. https://doi.org/10.1145/2792790

[14] Kamińska, D., Zwoliński, G., Laska-Leśniewicz, A., Raposo, R., Vairinhos, M., Pereira, E., Urem, F., Ljubić Hinić, M., Haamer, R. E., Anbarjafari, G. Augmented Reality: Current and New Trends in Education. Electronics, 2023, 12(16), 3531. https://doi.org/10.3390/electronics12163531

[15] Kavanagh, S., Luxton-Reilly, A., Wuensche, B., Plimmer, B. A systematic review of Virtual Reality in education. Themes in Science and Technology Education, 10(2), 2017, pp.85–119. https://files.eric.ed.gov/fulltext/EJ1165633.pdf

[16] Khronos Group. glTF™ 2.0 Specification.2017. https://www.khronos.org/gltf

[17] Khronos Group. WebGL 2.0 Specification (Version 2.0.0). 2017. https://registry.khronos.org/webgl/specs/2.0.0/

[18] Kouril, M., Maczewski, M. Exploring WebXR: Building immersive experiences on the web. Apress,2021.

[19] Makransky, G., Mayer, R.E. Benefits of Taking a Virtual Field Trip in Immersive Virtual Reality: Evidence for the Immersion Principle in Multimedia Learning. Educ Psychol Rev 34, 2022, pp.1771–1798. https://doi.org/10.1007/s10648-022-09675-4

[20] Makransky, G., Andreasen, N. K., Baceviciute, S., & Mayer, R. E. Immersive Virtual Reality Increases Liking but Not Learning With a Science Simulation and Generative Learning Strategies Promote Learning in Immersive Virtual Reality. Journal of Educational Psychology 113(4), 2021, pp.719-735. https://doi.org/10.1037/edu0000473

[21] Milgram, P., Kishino, F. A taxonomy of mixed reality visual displays. IEICE Transactions on Information and Systems, 77(12), 1994, pp.1321–1329.

[22] Moreno, R., Mayer, R. E. Learning Science in Virtual Reality Multimedia Environments: Role of Methods and Media. Journal of Educational Psychology, 94(3), 2002, pp.598–610. https://doi.org/10.1037/0022-0663.94.3.598

[23] Parong, J., Mayer, R. E. Learning Science in Immersive Virtual Reality. Journal of Educational Psychology, 110, 2018, pp.785–797. https://doi.org/10.1037/edu0000241

[24] Radianti, J., Majchrzak, T. A., Fromm, J., Wohlgenannt, I. A systematic review of immersive virtual reality in higher education. Computers & Education, 147, 103778, 2020. https://doi.org/10.1016/j.compedu.2019.103778

[25] Sanchez-vives, M. V., Slater, M. From presence to consciousness through virtual reality. Nature Reviews Neuroscience, 6(4), 2005, pp.332–339. https://doi.org/10.1038/nrn1651

[26] Skarbez, R., Smith, M., Whitton, M. C. Revisiting Milgram and Kishino's Reality-Virtuality Continuum. Frontiers in Virtual Reality, 2021, 2, Article 647997. https://doi.org/10.3389/frvir.2021.647997

[27] Slater, M., Wilbur, S. A framework for immersive virtual environments (FIVE). Presence: Teleoperators and Virtual Environments, 6(6), 1997, pp.603–616. https://doi.org/10.1162/pres.1997.6.6.603

[28] Sweller, J. Cognitive load during problem solving: Effects on learning. Cognitive Science, 12(2), 1988, pp. 257–285. https://doi.org/10.1207/s15516709cog1202_4

[29] Three.js Contributors. (2024). Three.js Documentation. https://threejs.org

[30] W3C. (2024). WebXR Device API. World Wide Web Consortium. https://www.w3.org/TR/webxr/

[31] Wu, H.-K., Lee, S. W.-Y., Chang, H.-Y., Liang, J.-C. AR in education: Status and challenges. Computers & Education, 62, 2013, pp.41–49. https://doi.org/10.1016/j.compedu.2012.10.024

[32] Zhou, X., LI, R., Liang, P., Zhang, B., Shahin, M., LI, Z., Yang, C. Using LLMs in Generating Design Rationale for Software Architecture Decisions. ACM Transactions on Software Engineering and Methodology, 2025. https://doi.org/10.1145/3785010