# VMXR: a EUD Environment for Virtual Merchandizing in XR

Alessandro Menale[2], Jacopo Mereu[1], Carlo Nuvole[2], Luigi Pannuti[2], Emanuele Mario Spano[2] and Lucio Davide Spano[1,*]

[1]*University of Cagliari, Dept. of Mathematics and Computer Science, Via Ospedale 72, 09124, Cagliari, Italy*
[2]*Techedge S.p.A., Via Caldera 21, Building B2, 20153, Milano, Italy*

**Abstract**
This paper presents the current development state of VMXR, a Proof of Concept (PoC) environment allowing people without programming experience to create and configure product showcases in a Virtual and eXtended reality setting. The aim of the PoC is to identify proper metaphors and workflows for supporting showcase designers in creating interactions with the virtual product representation or enhancing the physical environment through additional information and media.

**Keywords**
eXtended Reality, End-User Development, Configuration, Natural Language, Rules, Event-Condition-Actions

## 1. Introduction

The availability in the past few years of consumer-targeted hardware for immersive Virtual and eXtended Reality experiences (VR and XR) attracted the attention of different media and software companies, which started investing effort in producing content and experiences for these modalities. However, while content created by teams including professional developers and 3D artists is steadily growing and it will grow in the next years [1], it is still challenging to open the creation of XR content to non-tech professionals. Current commercial tools contain scene builders or inspectors, allowing non-experts to position static contents and activate simple trigger-based interactions (e.g., showing or hiding information overlays) [2, 3, 4]. Supporting end users in defining more complex interactions in virtual environments is still an open research question. Recent advances in this field have been limited to environments based on 360°videos [5, 6], or they are mainly targeted to developers requiring to reduce the burden of build-text-fix cycle [7]. This paper summarises the advances in developing VMXR (Virtual Merchandising in eXtended Reality), a Proof of Concept (PoC) environment for supporting end users in creating XR experiences dedicated to product showcases. The PoC applies the method described in [8] to a relevant field for the Italian industry, establishing a workflow among developers and content designers. The tool allows them to create and update the experiences by inserting and

*IS-EUD 2023: 9th International Symposium on End-User Development, 6-8 June 2023, Cagliari, Italy*
*Corresponding author.
davide.spano@unica.it (L. D. Spano)
0000-0001-7106-0463 (L. D. Spano)

configuring objects that define domain-relevant actions. They define the interactive behaviour of these objects through Event-Condition-Action (ECA) rules.

## 2. Related Work

End users can create VR environments with the help of various commercial and research tools. Most of these programs can only derive static scenes or overlay multimedia elements. For instance, Spoke [9], supports the composition of 3D models through user-friendly tools. However, these tools only allow static scene configuration, while proper XR experiences require interactive features beyond exploration and animations.

Tools specifically designed for end users exist. XOOM [10] allows creating web-based immersive VR applications but specifically focused on the cultural heritage domain. More general tools like VR GREP [11] have a broader application scope, but the available interactions are limited to the navigation and reactions to button activations. More recently, tools like FlowMatic [7] provided an immersive authoring tool that allows programmers to specify reactions to discrete events (e.g., user actions and system timers). VREUD [12] defines a simplified authoring environment for novice VR developers, while X-Reality exploits a dataflow graphical environment for creating applications exploiting both virtual contents and physical devices. These approaches focus on rapid prototyping or simplified definitions of common interactions. ECARules4All [8], the work that inspired our tool, provides a more general approach, which leverages meta-design for reducing the development complexity. End users exploit configurable templates for creating XR experiences, instead of developing the whole environments from scratch. Professional developers create a simplified programming interface based on natural language rules to support such configuration and the definition of complex interactive behaviours.

## 3. VMXR Overview and Workflow

The proposed solution is similar to Content Management Systems for the web: users without technical knowledge start from a template XR environment containing dummy content, which they adapt by adding meaningful content and configuring its behaviour. Different users can use the same template, but they will end up defining different experiences. In our case, end users start from predefined environments representing virtual shops. These predefined environments are the correspondent of web templates in our solution. Experienced developers and/or 3D modellers create them and provide means for their configuration. In VMXR, this process consists in adding 3D objects from a predefined list of possible categories and defining the interactions they support. More in detail, configuring a template means:

- Changing settings related to objects or features already present in the scene (e.g. material and position of walls);
- Inserting objects not present in the template, from a predefined object list;
- Insert multimedia material (images, text, video);
- Define interactions with objects (e.g. what happens when a user grabs a product).

While we can find consistent solutions supporting the first three points in the literature (see Section 2), for the fourth point, we applied the general solution proposed in [8], which provides the end users with an effective but understandable mechanism for defining the environment behaviour. We use Event-Condition-Action rules expressed in natural language, following this pattern:

```
When <Event> (if <Condition>)? then (<Action>)*.
```

In this schema, an *event* is a user-input notification or the successful execution of an action. User inputs include pointing, selecting, grabbing and releasing objects in the environment. In addition, it is also possible to specify proximity interactions according to the user's position in the environment (e.g., when the user is close to a table). An *action* is a high-level feature supported by a given category of objects. For instance, lights support actions such as turn-on or turn-off, and screens can display images, play or stop video playback. There are also general actions associated with all types of objects, such as moving them to an absolute position or relative to other objects (i.e., up, below, left, right etc.). *Conditions* allow checking the state of an object before executing actions. They may be simple (containing only one check) or composite (including more than one simple condition in and/or). Usually, rules do not have conditions since end users seldom use them in their definitions [13, 14], but we included them in the language to increase its expressiveness.

VMXR distinguishes four roles. The *developer* (Dev) defines the set of objects that a given environment template includes and implements the high-level actions they support. The *environment configurator* (EC) instead creates the floor plan of the virtual shop and inserts fixed elements of furniture. This supports the requirement of different producers to keep a consistent design in all their shops (either real or virtual) for brand identity. Instead, an *experience configurator* (XC) creates his/her own version of a given shop for inserting elements related to a product campaign. This means inserting the target products and the dedicated furniture elements for showcasing them, together with media elements providing additional information or enhancing the experience. In addition, an XC defines the interactions in the environment through the ECA rules. Finally, the *user* is the final consumer of the experience.

Such a role organization corresponds to the meta-design approach defined in [15, 16, 17]. *Dev*s are at the meta-design level, including professional developers using Turing-Machine equivalent languages for defining the core aspects of the system. *EC* and *XC* are at the design level: they are domain experts using less expressive power participating in the software design. Finally, *user*s are at the use level since they experience the interaction and basically are not aware of the shared process between developers and domain experts leading the creation of the experience.

The workflow supported by VMXR for creating XR showcases consists of four steps:

1. **High-level object implementation**, including the identification and implementation of the objects required by EC and XC.
2. **Configuration of the environment**. The EC defines the room floor plan through a web interface and inserts fixed furnishing elements. VMXR allows inserting only objects defined by Devs at Step 1, which are, in general, useful for creating more than one virtual shop. The result of this step is a reusable template.

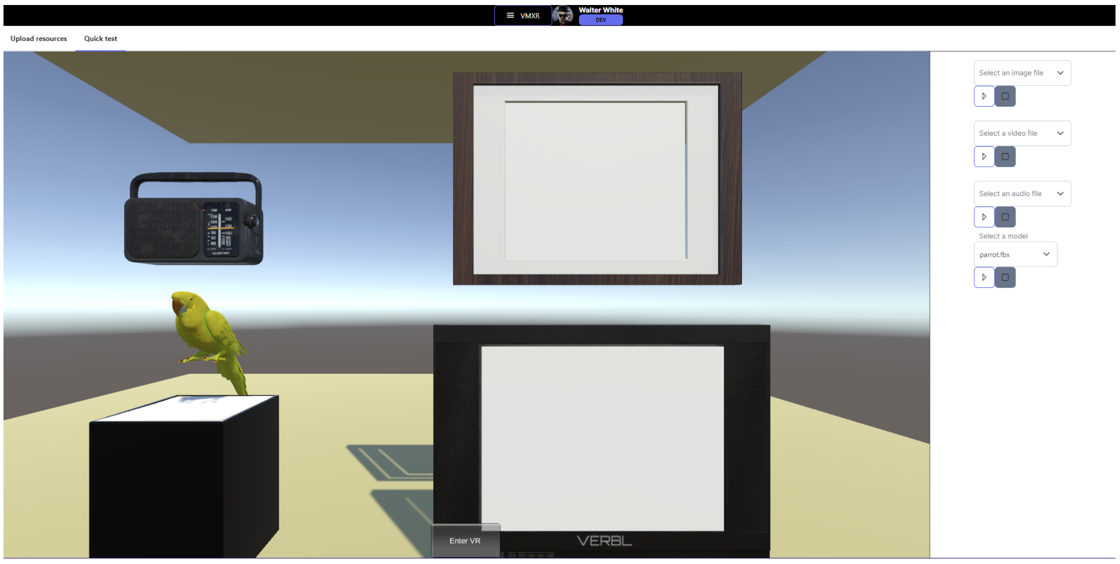


**Figure 1:** Four sample objects uploaded by a Dev for supporting EC and XC in creating the XR environment. Each one is associated to one or more Unity components providing high-level actions (e.g., play sound for the radio, play video for the screen etc.)

3. **Experience configuration**. The XC selects a template from those available after Step 2 and inserts additional objects from those defined in Step 1. Moreover, s/he creates the rules to specify the interaction. The result of this step is an executable interactive XR environment.
4. **Interaction with the environment**. The final user (e.g. a customer) puts on the visor and visits the configured environment by interacting with the virtual objects present.

## 4. VMXR Walktrough

This section shows the current state of the PoC development. We exploit web-based technologies for supporting the authoring, while the final experience is a native build of the XR environment for different platforms of a Unity3D-based application.

The workflow starts from a Dev that uploads the 3D models of the assets that may be used for creating the XR experiences, including pieces of furniture, screens, counter displays etc., together with the virtual models of the products. Besides the assets for managing the shop appearance, the Dev implements the code (i.e., Unity scripts) required for executing the high-level actions available for the XC. In the example in Figure 1, shows the interface Dev use for uploading 3D models into the set of available objects. Through that interface, they can set the properties and assign the required components (Unity scripts) they developed for implementing the high-level actions.

After creating the required assets, the EC furnishes the different shop categories the brand manages. S/he starts by defining relevant floor plans, by using the interface in Figure 2, which allows the creation of walls, doors and windows. After that, s/he can position the different fixed

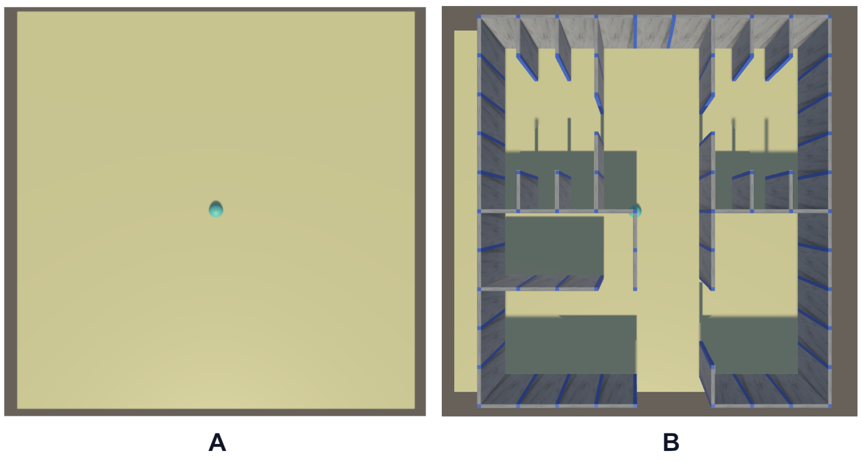


**Figure 2:** A sample floor design by an EC. The design starts with an empty view (A). The EC draws the walls by activating the corresponding tool and drawing them on the floor plane using the mouse (B).

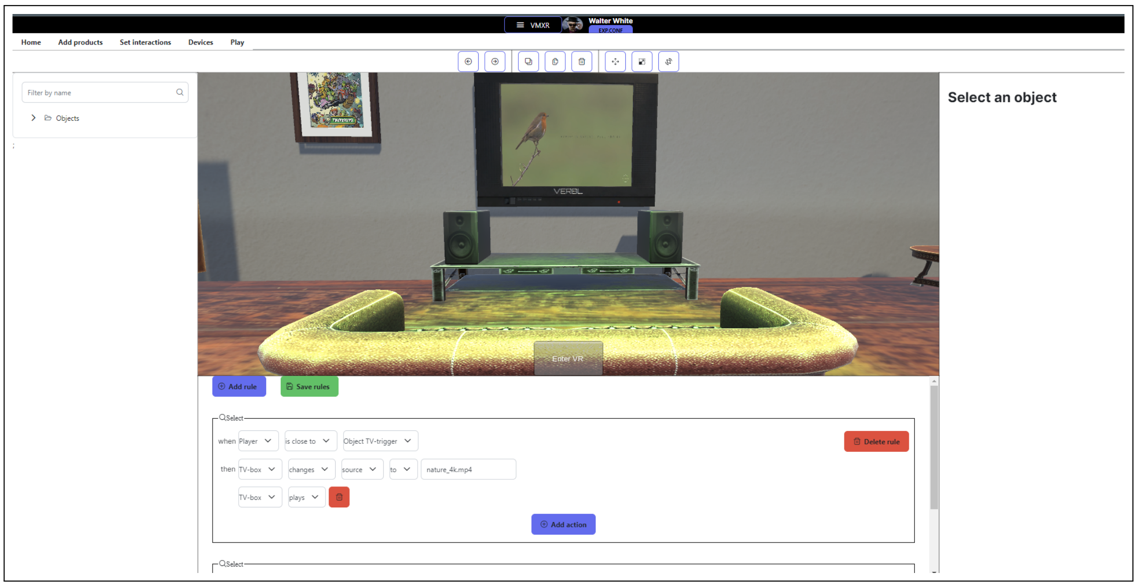
furniture pieces, i.e. those that do not change across different product campaigns. The interface for this task is similar to the one used by an XC for positioning products or removable furniture elements (see Figure 3, top part). For both users, the set of available objects available in the main menu of the authoring environment, grouped into the categories defined by Devs.

Figure 3 shows the interface XCs use for creating the interaction rules. It consists of a constrained editor, allowing the selection of the different parts that constitute an action or an event in the rule language, which are the *subject*, the *verb* and an optional *object*. The tool suggests the available options for each part to avoid syntax errors. In addition, it maintains consistent the rule by resetting all the fields that depend on another one, when the latter changes. For instance, in the event triggering the rule depicted in Figure 3 (the *when*), the *Player* is the subject. The fields containing the verb (*is close to*) and the object (*Object TV-trigger*) depend on the subject selection. So, if the XC changes the subject, the interface resets the verb and the object.

When the experience design is finished, the final user can interact with the immersive environment, experiencing the effects of the ECA rules.

## 5. Conclusion and Future Work

In this paper, we presented the current state of VMXR, a tool allowing end users to create XR showcase experiences. The tool applies meta-design to provide end users with abstractions they can manipulate, involving professional developers in defining the available virtual object types and the associated high-level actions. Rules expressed constrained natural language support defining the interactive behaviour of the environment. In future work, besides completing the technical implementation of the environment, we aim to evaluate and improve it by deploying the solution in a real-world scenario. Further research directions include enhancing the natural language processing techniques used in the tool, by integrating the support provided by the



when Player is close to Object TV-trigger
then TV-box changes source to nature_4k.mp4
TV-box plays

**Figure 3:** The interface for authoring rules in VRMX. The XC navigates the environment in the 3D view, and defines rules through the constrained editor at the bottom of the interface. The left-hand side panel shows a list of the objects in the environment, while the right-hand panel shows the properties of the currently selected object (if any).

latest large generative models.

## References

[1] Vv.Aa., Virtual Reality Market Research Report, Technical Report, Fortune Business Insights, 2021.
[2] Google, Google poly, 2017. URL: https://poly.google.com/.
[3] F. Games, Fungus, 2020. URL: https://fungusgames.com.
[4] Ottifox, Ottifox, 2018. URL: https://ottifox.com/index.html.
[5] I. Blečić, S. Cuccu, F. A. Fanni, V. Frau, R. Macis, V. Saiu, M. Senis, L. D. Spano, A. Tola, First-person cinematographic videogames: Game model, authoring environment, and potential for creating affection for places, J. Comput. Cult. Herit. 14 (2021). URL: https://doi.org/10.1145/3446977. doi:10.1145/3446977.
[6] F. A. Fanni, M. Senis, A. Tola, F. Murru, M. Romoli, L. D. Spano, I. Blečić, G. A. Trunfio, Pac-pac: End user development of immersive point and click games, in: A. Malizia, S. Valtolina, A. Morch, A. Serrano, A. Stratton (Eds.), End-User Development, Springer International Publishing, Cham, 2019, pp. 225–229.
[7] L. Zhang, S. Oney, FlowMatic: An Immersive Authoring Tool for Creating Interactive Scenes in Virtual Reality, Association for Computing Machinery, New York, NY, USA, 2020, p. 342–353. URL: https://doi.org/10.1145/3379337.3415824.
[8] V. Artizzu, G. Cherchi, D. Fara, V. Frau, R. Macis, L. Pitzalis, A. Tola, I. Blecic, L. D. Spano,

Defining configurable virtual reality templates for end users, Proc. ACM Hum.-Comput. Interact. 6 (2022). URL: https://doi.org/10.1145/3534517. doi:10.1145/3534517.

[9] Mozilla, Mozilla spoke, 2022. URL: https://hubs.mozilla.com/spoke, [Online; accessed 17-February-2022].

[10] F. Garzotto, M. Gelsomini, V. Matarazzo, N. Messina, D. Occhiuto, Xoom: An end-user development tool for web-based wearable immersive virtual tours, in: J. Cabot, R. De Virgilio, R. Torlone (Eds.), Web Engineering, Springer International Publishing, Cham, 2017, pp. 507–519.

[11] T. Zarraonandia, P. Díaz, I. Aedo, A. Montero, Inmersive end user development for virtual reality, in: Proceedings of the International Working Conference on Advanced Visual Interfaces, AVI '16, Association for Computing Machinery, New York, NY, USA, 2016, p. 346–347. URL: https://doi.org/10.1145/2909132.2926067. doi:10.1145/2909132.2926067.

[12] E. Yigitbas, J. Klauke, S. Gottschalk, G. Engels, Vreud - an end-user development tool to simplify the creation of interactive vr scenes, in: 2021 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), 2021, pp. 1–10. doi:10.1109/VL/HCC51201.2021.9576372.

[13] B. Ur, M. Pak Yong Ho, S. Brawner, J. Lee, S. Mennicken, N. Picard, D. Schulze, M. L. Littman, Trigger-action programming in the wild: An analysis of 200,000 ifttt recipes, in: Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems, CHI '16, Association for Computing Machinery, New York, NY, USA, 2016, p. 3227–3231. URL: https://doi.org/10.1145/2858036.2858556. doi:10.1145/2858036.2858556.

[14] B. Ur, E. McManus, M. Pak Yong Ho, M. L. Littman, Practical trigger-action programming in the smart home, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '14, Association for Computing Machinery, New York, NY, USA, 2014, p. 803–812. URL: https://doi.org/10.1145/2556288.2557420. doi:10.1145/2556288.2557420.

[15] C. Ardito, P. Buono, M. F. Costabile, R. Lanzilotti, A. Piccinno, L. Zhu, On the transferability of a meta-design model supporting end-user development, Universal Access in the Information Society 14 (2015) 169–186. URL: https://doi.org/10.1007/s10209-013-0339-7. doi:10.1007/s10209-013-0339-7.

[16] G. Desolda, C. Ardito, M. Matera, Empowering end users to customize their smart environments: Model, composition paradigms, and domain-specific tools, ACM Trans. Comput.-Hum. Interact. 24 (2017). URL: https://doi.org/10.1145/3057859. doi:10.1145/3057859.

[17] G. Fischer, E. Giaccardi, Y. Ye, A. G. Sutcliffe, N. Mehandjiev, Meta-design: A manifesto for end-user development, Commun. ACM 47 (2004) 33–37. URL: https://doi.org/10.1145/1015864.1015884. doi:10.1145/1015864.1015884.