Proceedings of the 18th Conference on Computer Science and Intelligence Systems pp. 961–965
DOI: 10.15439/2023F2375
ISSN 2300-5963 ACSIS, Vol. 35

Future and Backward Exploration of XR Environments

Jakub Flotyński, Paweł Sobociński, Michał Śliwicki, Mikołaj Maik
Department of Information Technology, Poznań University of Economics and Business
Email: [jakub.flotynski, pawel.sobocinski, michal.sliwicki, mikolaj.maik]@ue.poznan.pl

***Abstract*—XR environments are successfully used in various domains, such as medicine, education, training, and industry. Such environments contain domain knowledge expressed through content and users' behavior. However, current approaches to XR creation lack possibility of exploration of the knowledge included in the environments and expressed by the users' behavior. In this paper, we propose a method of knowledge exploration in XR environments, enabling the analysis of past and potential behavior of users and objects, with queries and automated reasoning. This solution aims to enhance knowledge dissemination using XR.**

I. Introduction

XR SYSTEMS' rising popularity stems from their potential across domains and their ability to offer immersive experiences. However, existing systems often overlook event and interaction analysis within virtual environments, leading to a loss of valuable information. To enhance user experience, there's a crucial need to focus on exploring and analyzing XR system data for hidden insights.

Actions and interactions within XR environments can be explored through semantic queries and automated reasoning, particularly beneficial in employee training systems. This exploration offers valuable insights into users' behavior across different timeframes.

The knowledge gained from exploration is vital for monitoring, analyzing, and controlling XR environments, as well as understanding users' skills, experiences, interests, and preferences. Domain-specific terminology helps specialists, the primary users of XR environments, take full advantage of behavior exploration.

This paper introduces a new method for exploring XR environments, encompassing both forward and backward exploration through semantic queries on the XR environment's knowledge-based representation. The proposed methods were applied to represent the behavior of the virtual environment in an industrial XR training system developed for Amica S.A, Poland's leading house appliances manufacturer, to train employees on specialized industrial devices.

The paper's structure is as follows: Section II presents an overview of the current state of exploring virtual environments. Then, Section III explains the methods used to describe virtual content. Section IV elaborates on exploration methods with examples and an overview of exploration for the XR training system. Finally, Section V concludes the paper and suggests potential areas for future research.

II. Related work

Numerous approaches model 3D content behavior using ontologies and semantic web standards. One notable approach, as discussed in Pellens et al. [1], [2], [3], introduces temporal operators for expressing both primitive and complex behaviors. They also offer a graphical tool to model complex behavior using diagrams, encoding it within X3D scenes [4].

In another approach, De Troyer et al. [5] combine primitive actions like move, turn, and rotate to represent complex behavior in a user-friendly manner. This approach enables end users to specify complex behavior without extensive knowledge of 3D graphics and animation.

Krieg-Brückner et al. [6] introduce a tool using semantic concepts, services, and hybrid automata to describe 3D content behavior. The tool consists of a client component based on a 3D content presentation tool (e.g., XML3D browser) and a server component with various services for content selection and configuration. Additionally, an extra module manages intelligent avatars and their perception of the scene.

Chmiel et al. [7] proposed XSD-based semantic metadata schemes for 3D object interactivity, specifying events, conditions, and actions. CL ontologies [8], [9] represent multi-user virtual environments and avatars, defining geometry, space, animation, and behavior of 3D content. They include semantic counterparts to widely used formats like VRML and X3D. Environmental objects are described by attributes like translation, rotation, and scale, while avatars have names, statuses, user interfaces (UIs), and behavior defined through code bases

In recent research, there has been a notable focus on humans and their interactions within virtual reality, particularly in creating ontologies for experimental purposes. One such recent work is the Virtual Human-Building Interaction Experimentation Ontology (VHBIEO) proposed by Chokwitthaya et al. [10]. The primary objective of this solution is to establish a standardized approach for conducting experiments related to human-building interactions within virtual reality environments. In a similar vein, Heitmayer et al. [11] have introduced another ontology focusing on the human-centred analysis and design of virtual reality conferencing. The main goals of this ontology include enhancing user experience, facilitating research on VR conferencing (particularly in the realms of psychology and behavior), and enabling the sharing of research findings among the scientific community.

III. Representation of XR Environment

In order for the XR environment to be appropriate for exploration, it needs a proper representation of interaction and 3D content. This can be created using knowledge representation technologies such as the semantic web (the Resource Description Framework—RDF [12], the RDF Schema—RDFS [13], the Web Ontology Language—OWL [14] and the SPARQL query language [15]) and ontologies.

For the presented methods of exploration, we prepared behavioral semantic models for the representation of activities, called Activity Ontology, as well as the representation of the workflow.

A. Representation of Activities and Features

Activities and features act as domain-specific states that connect the behavior model and the implementations of XR components. Within the Activity Ontology, the Semantic Web approach defines the representation of these activities and properties. This ontology comprises a TBox and an RBox that contain axioms outlining how users' and objects' features and activities are implemented within XR components and environments. The activity ontology is well suited for both procedural and object-oriented XR implementations.

Fig. 1 shows how class in the XR system is mapped to ontology to activity ontology.

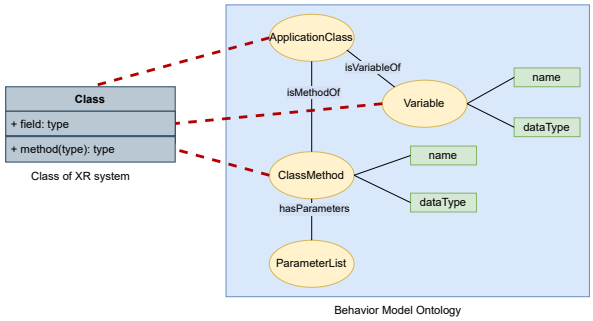

Fig. 1. Mapping XR system class to Activity Ontology

Since both procedural and object-oriented XR implementations rely on functions as the fundamental building blocks of workflow, the activity ontology supports structuring both types of implementations.

The ontology specifies the following classes and attributes associated with code elements:

1) ApplicationClass — is the class of all application classes specified in the code of XR components.
2) Variable — is the class of all variables.
3) ParameterList — is the class of all lists of method parameters.
4) ClassMethod — is the class of all methods specified in the code of XR components

The example of a knowledge base generated using activity ontology is presented in listing 1. The example shows described class *Battery*, which has variable *Charge* and method *ConnectRectifier*.

Listing 1. A fragment of knowledge base describing element of XR system

```
ao:Battery rdf:type owl:NamedIndividual ,
                    ao:ApplicationClass .

ao:ConnectRectifier rdf:type owl:NamedIndividual ,
                    ao:ClassMethod ;
           ao:isMethodOf ao:Battery ;
           ao:datatype "bool"^^rdfs:Datatype ;
           ao:name "connectRectifier"^^xsd:string .

ao:Charge rdf:type owl:NamedIndividual ,
                     ao:Variable ;
             ao:isVariableOf ao:Battery ;
             ao:datatype "integer"^^rdfs:Datatype ;
             ao:name "charge"^^xsd:string .
```

B. Workflow Representation

The workflow representation adapts the behavior model to describe states, events, and time related to the execution of class methods. Hence, it enables the specification of XR components' behavior upon their underlying imperative implementation.

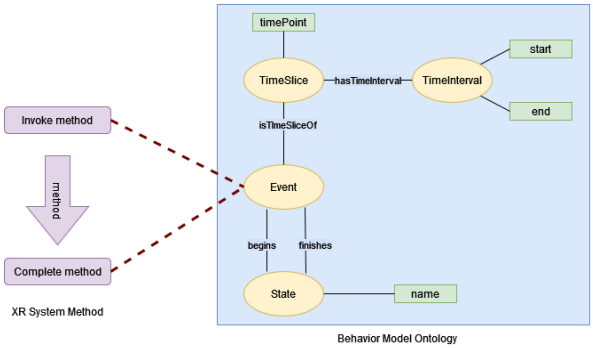

Fig. 2. Mapping the XR system method to fluent ontology

The example of the mapping method of the XR system to behavior model ontology is presented in Fig. 2. An event mapping states that the beginning of an activity is a method invocation or that a method completion is the finish of an activity. Hence, event mappings enclose states of method executions using domain events. Using such mapping, the system can generate a knowledge base, which can be treated as behavioral logs, therefore, can be explored by proper queries. An example of such a knowledge base is presented in Listing 2. The example describes two events that happened while using the system. The first event was "The visual inspection of the battery". That event began *BatteryState1* and finished *BatteryState2*. This event had assigned *TimeSlice*, which gives us information about when and how long the event lasted.

Listing 2. A fragment of knowledge base describing event

```
fo:Event1 rdf:type owl:NamedIndividual ,
                   fo:Event ;
          fo:begins fo:BatteryState1 ;
          fo:finishes fo:BatteryState2 ;
          fo:name "Visual␣inspection␣of␣battery"^^xsd:
              string .

fo:TimeSlice1 rdf:type owl:NamedIndividual ,
                     fo:TimeSlice ;
             fo:hasTimeInterval fo:TimeInterval1 ;
             fo:isTimeSliceOf fo:Event2 .
```

```
fo:TimeInterval1 rdf:type owl:NamedIndividual ,
                          fo:TimeInterval ;
                 fo:end "2023-05-20T10:03:12"^^xsd:dateTime
                      ;
                 fo:start "2023-05-20T10:04:22"^^xsd:
                      dateTime .
```

IV. Exploration

In this section, we propose different types of knowledge exploration and visualization within explorable XR environments, utilizing the models proposed in the previous section. We classify these exploration types based on the target periods, distinguishing between simulation with forward and backward exploration.

Simulation and forward exploration facilitate the process of reasoning and querying potential events and states in the XR environment. These events and states may remain undetermined, contingent upon the occurrence or absence of other events and states within the environment. Engaging in simulation and forward exploration necessitates a composed XR environment, which does not necessarily need to be compiled and run. Consequently, simulation and forward exploration can be initiated promptly after composing the environment.

Backward exploration enables the process of reasoning, querying, and visualizing past and present events and states that have been logged in behavior records. To engage in backward exploration, the XR environment must be executed, and behavior logs need to be generated.

Queries play a crucial role in acquiring knowledge about explorable XR environments. Nevertheless, despite their possible support in query languages such as SPARQL, our focus does not revolve around query result presentation operations like result limitation, sorting, or data aggregation. Instead, our primary emphasis lies in utilizing queries to gain deep insights into the properties and behaviors of the XR environment, facilitating exploration and understanding.

By applying these techniques for knowledge exploration and visualization, we elevate our comprehension of the dynamic nature of XR environments and empower users to interact effectively, reason, and query within these environments. These approaches serve as invaluable tools for researchers and developers, enabling them to delve into the potential of explorable XR environments and propel innovation in this field.

We used new exploration methods to query the behavior of users and 3D objects inside a virtual environment for the industrial worker training XR system, which allows trainees to learn how to act safely in an industrial setting. The training scenario implemented in the system focuses on safe work with a forklift. It was developed using resources from Amica S.A., a major producer of household equipment in Poland.

A. Simulation with forward exploration

Forward exploration facilitates reasoning and queries about the potential behaviors of users and objects within explorable XR environments. It encompasses various states and events associated with features and activities, encompassing autonomous actions and interactions among users, objects, and their interplay. Crucially, forward exploration is intimately intertwined with the simulation of environmental behavior, which aims to fulfil conditions necessary for events and states. As a result, simulation precedes forward exploration, encompassing both aspects within simulation queries. Importantly, since simulation and forward exploration revolve around potential events and states, they necessitate a workflow specification but do not mandate that the XR environment be actively running.

Simulation queries to an environment specification enable forward exploration of the environment without running it. Therefore, they must specify the conditions for which the exploration is accomplished. The illustrative simulation queries presented in this section assume that the delay between an event and another following event is equal to 0.001, and no exceptions are thrown during the execution of methods:

$Delay$ =0.001 $\bigwedge exception(executed(Method, ExecutionID), null)$.

To allow forward exploration, the following elements of the XR system were mapped to semantic representation using Activity Ontology:

Classes:

1) Trainee (class representing trainee),
2) Forklift (class representing forklift),
3) Battery (class representing battery),
4) Rectifier (class representing rectifier).

Methods:

1) startForkliftInspection (method of Trainee class),
2) finishForkliftInspection (method of Trainee class),
3) chargeBattery(method of Rectifier class),
4) insertBattery(method of Forklift class),
5) showChargingState(method of Rectifier class),
6) plugIn(method of Rectifier class)

Additional predicates were also used, allowing for the representation of temporal entities:

time(event,tp) — predicate that is true for a given event and a time point if and only if the event occurs at the time point.

holds(event,ti) — predicate that is true for a given event and a time interval if and only if the fluent is true within the time interval

Using such prepared knowledge representation, we are able to create the following queries:

1. How long will it take to check the visual state of a forklift?

$time(startForkliftInspection(Trainee, Forklift)), TP_{start}) \bigwedge time(finishForkLiftInspection, TP_{end}) \bigwedge Lenght = TP_{end} - TP_{start}$

It determines the time points of starting and finishing a forklift inspection and calculates the inspection length. (The Fig. 3 presents how a user inspects the forklift in XR system.) The query result is the following:

$TP_{start} = 10,\ TP_{end} = 22,\ Lenght = 12$

2. What will happen after a trainee finishes charging the battery?

$holds(chargeBattery(Trainee, Battery), TI_1) \bigwedge holds(Action, TI_2) \bigwedge after(T_1, T_2)$

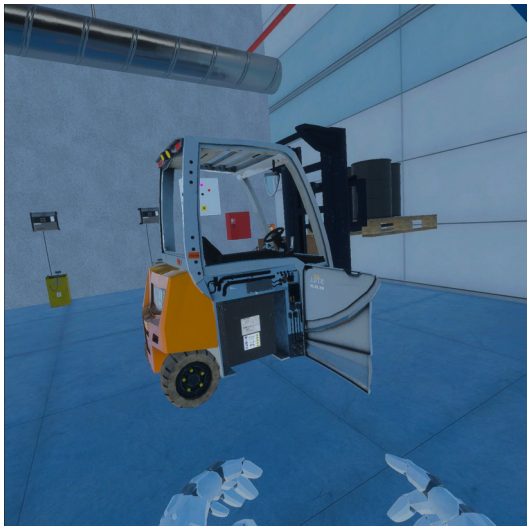

Fig. 3. A user inspects the forklift

It searches for the charging battery event and its time interval and time interval and action that happened later by using the after predicate, which compares time intervals. The query result is the following:

$Action = insertBattery(Trainee, Battery))$

3. What are the possible states (color of light) of rectifier after plugging in the battery?

$holds(showChargingState(lightColor), T_1) \bigwedge holds(plugIn(Battery), TI2) \bigwedge after(T_1, T_2)$

It searches for the event describing showing the charging state of the rectifier that happened after plugging in the battery. The possible answers are:

$lightColor = red,\ lightColor = yellow,\ lightColor = green$

B. Backward exploration

Backward exploration allows for the analysis and querying of activities that took place during the operation of an explorable XR environment.

Unlike forward exploration, which relies on simulating the environment's behavior, backward exploration uses logged activities. This eliminates the need for environmental behavior simulation. Furthermore, the inclusion of temporal statements with visual descriptors in behavior logs enables the visualization of past activities.

Queries specifically designed for backward exploration are referred to as exploration queries. The output of an exploration query is defined similarly to a simulation query. The behavior logs are structured based on RDF, enabling the utilization of the SPARQL language for conducting backward exploration. The example of behavior logs in the form of knowledge-base was presented in Listing 2.

a) Query 1: Which events had happened before the forklift was turned on?

```
SELECT  ?eventName
WHERE { ?event rdf:type fo:Event  .
        ?timeSlice fo:isTimeSliceOf ?event .
        ?event fo:name ?eventName .
        { ?timeSlice fo:timePoint ?time . }
        UNION{
        ?timeSlice fo:hasTimeInterval ?timeInterval .
        ?timeInterval fo:start ?time . }
        ?PushButtonEvent fo:name "Turning on the forklift
             "^^<http://www.w3.org/2001/XMLSchema#string> .
        ?PushButtonTimeSlice fo:isTimeSliceOf ?
             PushButtonEvent .
        ?PushButtonTimeSlice fo:timePoint ?pushButtonTime .
FILTER (  ?time < ?pushButtonTime)}
ORDER BY ?time
```

The first query provides information about what happened in the scene before the trainee pushed the press button, which activated the forklift. The query searches for events and their assigned time slices that happened before the event with the name "Turning on the forklift". The action of turning on the forklift presents Fig. 4.

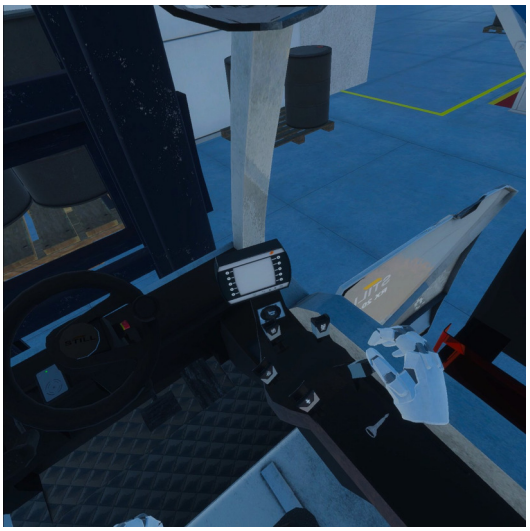

Fig. 4. A user turns on the forklift

b) Query 2: How long the battery was inspected?

```
SELECT ?start ?end
WHERE { ?event fo:name "visually controlling the battery
     "^^<http://www.w3.org/2001/XMLSchema#string> .
        ?timeSlice fo:isTimeSliceOf ?event .
        ?timeSlice fo:hasTimeInterval ?timeInterval .
        ?timeInterval fo:start ?start .
        ?timeInterval fo:end ?end .}
```

The second query gives information about the duration of the visual inspection of the battery. The query searches for the time slice of the event named "visually controlling the battery", then using the time interval object, access the information when the event started and ended. Fig. 5 presents, how user inspects the battery in virtual scene.

c) Query 3: When and What states did the battery transition into?

```
SELECT ?begins  ?stateName ?stateID
WHERE{ ?state rdf:type fo:InstantState .
        ?object fo:hasState ?state .
        ?object fo:name "battery"^^xsd:string .
        ?state fo:name ?stateName .
        ?state fo:id ?stateID .
        ?event fo:begins ?state .
```

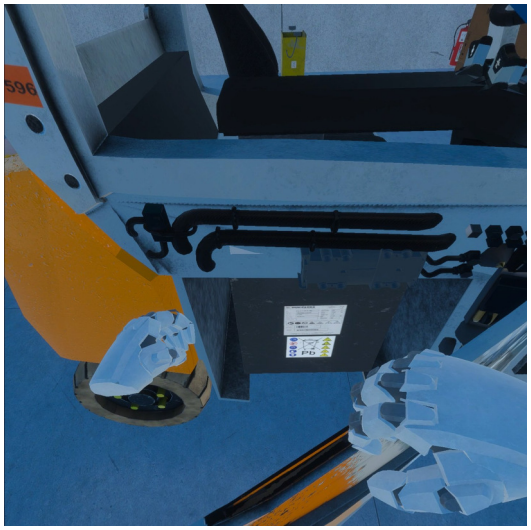

Fig. 5. The user inspects the battery

```
        ?timeSlice fo:isTimeSliceOf ?event .
        ?event fo:name ?eventName .
        { ?timeSlice fo:timePoint ?begins . }
        UNION {
        ?timeSlice fo:hasTimeInterval ?timeInterval.
        ?timeInterval fo:start ?begins . } }
ORDER BY ?begins
```

The last query provides information about what happened sequentially with the battery. The query searches for events that changed the states of the scene object named ”battery”. Then using proper time slices, the query determines the time by which the states are ordered. The results consist of the name and id of the states and the time when they have begun.

V. Conclusions and Future Work

The use of exploration of behavior and 3D content in XR systems can have multiple applications, like learning about users, their experience, preferences, and interests and measuring their skills. This can be beneficial, for example, in virtual training, where we want to maximize the training results.

In this paper, we have proposed methods for forward and backward exploration based on semantic queries. The presented approach uses the knowledge-based representation of the XR environment, which is represented by described ontologies. Moreover, we provided examples of explorations based on the developed XR system for employee training in an industrial environment.

The possible future research directions could focus on developing available inexperienced user plug-ins for simplifying and automating the process of creating semantic queries and visualization of the results.

Acknowledgment

The presented research was funded by the Polish National Center for Research and Development under grant number LIDER/55/0287/L-12/20/NCBR/2021.

References

[1] B. Pellens, O. De Troyer, W. Bille, and F. Kleinermann, “Conceptual modeling of object behavior in a virtual environment,” in *Proceedings of Virtual Concept 2005*. Biarritz, France: Springer-Verlag, 2005, pp. 93–94.

[2] B. Pellens, O. De Troyer, W. Bille, F. Kleinermann, and R. Romero, “An ontology-driven approach for modeling behavior in virtual environments,” in *Proceedings of On the Move to Meaningful Internet Systems 2005: Ontology Mining and Engineering and its Use for Virtual Reality (WOMEUVR 2005) Workshop*, R. Meersman, Z. Tari, and P. Herrero, Eds., no. 3762, Springer-Verlag. Agia Napa, Cyprus: Springer-Verlag, 2005, pp. 1215–1224.

[3] B. Pellens, F. Kleinermann, and O. De Troyer, “A development environment using behavior patterns to facilitate building 3d/vr applications,” in *Proc. of the 6th Australasian Conf. on Int. Entertainment*, ser. IE ’09. ACM, 2009, pp. 8:1–8:8.

[4] B. Pellens, O. De Troyer, and F. Kleinermann, “Codepa: a conceptual design pattern approach to model behavior for x3d worlds,” in *Proceedings of the 13th International Symposium on 3D web technology*, Los Angeles, August 09-10, 2008, pp. 91–99.

[5] O. De Troyer, F. Kleinermann, B. Pellens, and W. Bille, “Conceptual modeling for virtual reality,” in *Tutorials, posters, panels and industrial contributions at the 26th Int. Conference on Conceptual Modeling - ER 2007*, ser. CRPIT, J. Grundy, S. Hartmann, A. H. F. Laender, L. Maciaszek, and J. F. Roddick, Eds., vol. 83. Auckland, New Zealand: ACS, 2007, pp. 3–18.

[6] P. Kapahnke, P. Liedtke, S. Nesbigall, S. Warwas, and M. Klusch, “ISReal: An Open Platform for Semantic-Based 3D Simulations in the 3D Internet,” in *International Semantic Web Conference (2)*, 2010, pp. 161–176.

[7] J. Chmielewski, “Describing interactivity of 3d content.” in *Interactive 3D Multimedia Content*, W. Cellary and K. Walczak, Eds. Springer, 2012, pp. 195–221.

[8] Y. Chu and T. Li, “Using pluggable procedures and ontology to realize semantic virtual environments 2.0,” in *Proceedings of The 7th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry*, ser. VRCAI ’08. New York, NY, USA: ACM, 2008, pp. 27:1–27:6.

[9] Y.-L. Chu and T.-Y. Li, “Realizing semantic virtual environments with ontology and pluggable procedures,” in *Applications of Virtual Reality*, C. S. Lanyi, Ed. Rijeka: IntechOpen, 2012, ch. 9.

[10] C. Chokwitthaya, Y. Zhu, and W. Lu, “Ontology for experimentation of human-building interactions using virtual reality,” *Advanced Engineering Informatics*, vol. 55, p. 101903, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1474034623000319

[11] M. Heitmayer, M. G. Russell, S. Lahlou, and R. D. Pea, “An Ontology for Human-Centered Analysis and Design of Virtual Reality Conferencing,” in *TMS Proceedings 2021*, nov 3 2021, https://tmb.apaopen.org/pub/3rbumwgw.

[12] W3C. (accessed March 24, 2015) Rdf. [Online]. Available: http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/

[13] ——. (accessed March 24, 2015) Rdfs. [Online]. Available: http://www.w3.org/TR/2000/CR-rdf-schema-20000327/

[14] ——. (accessed March 24, 2015) Owl. [Online]. Available: http://www.w3.org/2001/sw/wiki/OWL

[15] ——, “Sparql query language for rdf,” accessed March 24, 2015 2008. [Online]. Available: http://www.w3.org/TR/2008/REC-rdf-sparql-query-20080115/