Sistema de Linguagem Dinâmica para a Criação e o Processamento de Rotinas Interativas e Procedurais Sobre Cenários de Ambientes de Realidade Virtual Imersivos

Angel R. Ferreira[1], Alexandre C. Silva[1]

[1]IF Goiano – campus Morrinhos
Morrinhos – GO, Brazil

angel.rodrigues@estudante.ifgoiano.edu.br,
alexandre.silva@ifgoiano.edu.br

Abstract. *Virtual Reality (VR) applications have been increasingly adopted in industry practices, and this is established in a perspective for the future. One of the main pillars of VR applications is the operation of interactive features. However, for interaction in VR environments, it is necessary, the designer and/or programmer, to face some challenges that are present in the authoring process. These challenges as problems that can be solved or minimized under VR application design methods. However, it was proposed the development of a language, which could possibly optimize some aspects in the process of implementing interactive features for VR applications.*

Resumo. *As aplicações de Realidade Virtual (RV) vem sendo cada vez mais adotadas nas práticas da indústria, e isso se estabelece em uma perspectiva para o futuro. Um dos principais pilares das aplicações de RV se trata do funcionamento de recursos interativos. No entanto, para haver interação em ambientes de RV, é necessário, o designer e/ou programador, enfrentar alguns desafios que estão presentes no processo autoral. Esses problemas podem ser resolvidos ou minimizados com base em métodos de projeto de aplicação de RV. Contudo, foi proposto o desenvolvimento de uma linguagem, que poderia otimizar alguns aspectos no processo de implementação de recursos interativos para aplicações de RV.*

1. Introdução

A tecnologia da Realidade Virtual (RV) vem sendo amplamente adotada nos processos e práticas do setor industrial. Em Velev e Zlateva (2017), situa-se um momento em que o cenário aprova sua incursão, com o destaque de sua influência para a indústria. Nesse viés, a popularidade de aplicações de RV vem crescendo com o passar dos anos, se tornando uma grande tendência para a indústria e para o futuro.

Segundo Chevaillier *et al.* (2012), como um argumento necessário, apesar dos avanços recentes no campo da engenharia de software, o desenvolvimento de aplicações de RV ainda é considerado como uma tarefa difícil, seja nos pontos de vista teórico e técnico. Dessa forma, a criação de ambientes de RV têm seus custos, e isso sucumbe a trivialidade, envolvendo aspectos da capacidade profissional projetista, complexidade de processamento computacional, e o aproveitamento e a adaptação de recursos. Nesse contexto, muitos esforços estão voltados ao objetivo de facilitar o processo do desenvolvimento de recursos para RV; visando a necessidade do desacoplamento do

projeto, seja em software como em hardware, e o aproveitamento de recursos prontos, eliminando a necessidade do desenvolvimento do zero (CHEVAILLIER *et al.*, 2012).

Visto que a implementação de recursos interativos habilita um comportamento essencial desejado numa aplicação de RV (BLOM; BECKHAUS, 2007), este trabalho toma preocupações quanto a otimização do processo autoral de tal finalidade, voltada ao setor industrial (comumente associado a cenários de treinamento e operação).

Portanto, este trabalho tem como objetivo, o desenvolvimento de um sistema de recurso de linguagem dinâmica de alto nível para a criação de cenários (procedurais) de ambientes de RV, baseados em rotinas de ações interativas. Para atender o desenvolvimento da proposta, faz-se o estudo de caso da bibliografia correlata e das propostas de otimização encontradas; e, uma implementação e validação prototípica e conceitual do sistema e recursos da proposta. Possivelmente os métodos da linguagem otimizariam o processo autoral e o processamento da sua finalidade aplicativa, flexibilizando ainda a adaptação de recursos interativos do projeto de aplicação de RV.

2. Bibliografia Correlata

Nesta seção, serão apresentadas algumas referências da literatura que se propõem a otimização do projeto de aplicações de RV. As propostas estudam o desenvolvimento de modelos de projeto. Comumente associados ao *core* da aplicação e implementações como, interfaces de programação abstrata, *frameworks* e/ou arquiteturas de *software*, e até mesmo *wokflow* (fluxo de trabalho) otimizados para equipes e processos.

Os trabalhos Richard *et al.* (2021) e Thies *et al.* (2019) propõem *frameworks* para a otimização dos processos autorais de aplicações de RV, voltadas ao treinamento e/ou operação no cenário industrial, sobre conceitos de generalização e aproveitamento de recursos implementados, que viabilizassem a geração de aplicações sem grandes esforços ou dependência do desenvolvedor de software, concomitantemente facilitando o acesso do especialista técnico do setor industrial a modificação ou implementação de cenários de aplicações. Ambos os trabalhos implementam uma interface de representação gráfica orientada a dados integrada ao processamento de *framework* de aplicação.

Em Richard *et al.* (2021), foi explorada essencialmente a demonstração de sua arquitetura de *framework* (denominada *INTERVALES*), baseando-se em modelagens UML, destacando a representação de módulos de funcionamento, estabelecendo modelos de dados – para a descrição e representação de cenários (objetivos da aplicação) e estados de cenário (registro da interação) – além de todo o fluxo de trabalho destinado à criação de cenários de aplicações de ambiente de RV; concretizando todo o ecossistema proposto.

Em uma perspectiva complementar, Thies *et al.* (2019) explora a necessidade de um compilador no processo autoral, aquém da arquitetura do *framework* de aplicação. Embora que o desenvolvimento do *framework* ainda tenha sido essencial, numa proposta como essa é possível notar, não só apenas um *framework* e/ou modelo de dados (arquitetura e projeto), mas também, um compilador (geração de programa) destinado à otimização do processo autoral de aplicações de RV.

O trabalho Cavallo & Forbes (2019) identifica a necessidade de ferramentas de autoria eficazes, bem como uma carência por padrões de desenvolvimento bem definidos. Apresenta uma ferramenta de autoria de RV, denominada *CAVE-AR*. Propondo um

workflow, pelo qual descreve uma otimização do custo de projeto entre as etapas de autoria e rotinas de testes, sobre a implantação/geração de cenários da aplicação.

Conforme a Tabela 1, fundamentada às referências, é possível relacionar a importância da arquitetura de projeto com base em um *framework* de aplicação e a necessidade de um *workflow* associado à criação das aplicações de RV. Além de que, com uma estrutura baseada em linguagem formal e projeto de *framework*, consegue-se otimizar o aproveitamento de recursos e todo o *workflow* necessário. Vislumbre o interessante passo contraponto em Thies *et al.* (2019), com proposta de compilador, quando até o momento, ao seu ver, não havia um entendimento da otimização comparável na literatura, e essa suposição se estende neste trabalho, exceto pela recorrência correlata, nota-se que nenhuma das propostas tem o objetivo de compiladores para "linguagens de programação". Logo, surge o interesse da proposta, em potencial correlato, explorar o campo da computação em linguagem de programação de alto nível (interface).

Tabela 1. Relação das propostas de otimização nas referências correlatas.

Trabalhos	Framework	Workflow	Compilador	Interface
Richard *et al.* (2021)	Sim	Sim	Não	Orientada a Dados
Thies *et al.* (2019)	Sim	Sim	Sim	Orientada a Dados
Cavallo & Forbes (2019)	-	Sim	Não	-

3. Arquitetura, Implementação e Funcionamento

Nessa seção, aborda-se brevemente uma concepção da proposta, como uma arquitetura conceitual; junto a implementação e validação de uma relação prototípica de compilador para uma linguagem de programação elaborada. Todo o processo é voltado à geração de aplicações de RV, utilizando o motor de ambiente 3D, Unity (https://unity.com/).

Sob conceito da arquitetura apresentada (relacione a Figura 1.a). Uma linguagem de programação de alto nível concebida e previamente denominada por "Linguagem_RV", realiza a implementação de código sobre uma gramática formal elaborada. Um compilador desenvolvido para a linguagem, implementa etapas de analisadores sobre um programa fonte; e com o processamento de tradução dirigida a sintaxe, utilizando a ferramenta *ANTLR* (https://www.antlr.org/) para estruturação sintática e processos semânticos, consegue gerar um programa objeto da aplicação no objeto da linguagem, *C#* (a mesma que a do compilador – híbrido sobre linguagem de alto-para-alto nível). Noutro ponto do processo autoral, a interface de *scripts* do *framework* de aplicação, integrando o ambiente do motor *Unity*, gerencia a montagem e execução da aplicação no ambiente da ferramenta (sem necessitar gerar um *build*).

Para validação da proposta (protótipo), cenários procedurais com atividades de deslocamento no ambiente podem ser gerados; onde o programa fonte descreve as etapas de percursos para o cenário processar e validar sobre a interação do usuário. Observe na Figura 1.b, a configuração e execução de um cenário de percursos. O usuário da aplicação (A) deve realizar os percursos (1-6) para cumprir uma atividade, onde cada percurso delimita uma região de acerto (Figura 1.b – Editor). O processo da aplicação (B) é procedural das interações implementadas e realizadas (Figura 1.b – Aplicação). A princípio, os recursos da linguagem são flexíveis e permitem a rápida implementação e execução de diversos cenários de aplicações de ambiente de RV.

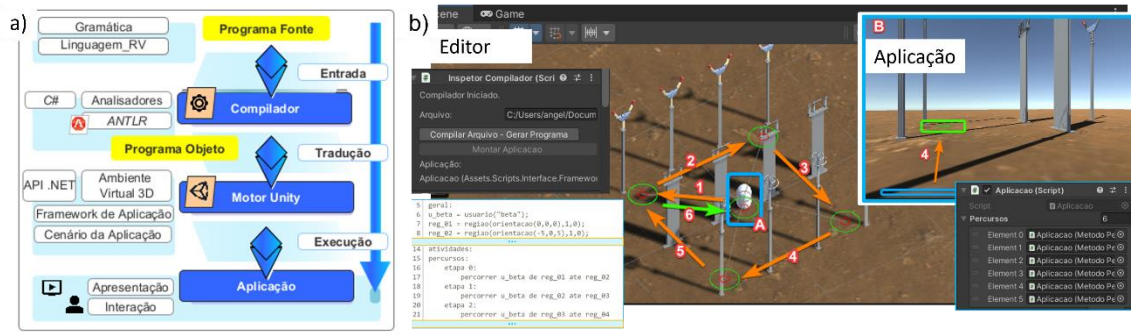

Figura 1. a) arquitetura conceitual; b) validação do protótipo da pesquisa – para o acesso ampliado às imagens, *https://github.com/LordKrampus/SVR2022_WuW.git*.

4. Considerações Finais e Trabalhos Futuros

Contudo, demonstrou-se a capacidade da implementação das interações e o simples procedimento de geração e execução de aplicações de RV, sob o conceito de compilador de linguagem. Para o futuro, pretende-se complementar as estruturas e métodos da linguagem e o motor de aplicação. A finalidade é tornar a implementação de recursos relativamente mais acessível ao usuário comum, mais prática ao designer/programador e mais viável ao setor industrial interessado no uso/implantação da tecnologia de RV.

Agradecimentos

Agradecimentos ao programa de iniciação científica, PIBIC; ao programa de PD&I (Edital nº 19) sob o apoio financeiro da PROPPI (IF Goiano); e, a instituição (IF Goiano – campus Morrinhos) que custeia o recebimento de bolsa de pesquisa. Considerando ainda todo eventual apoio financeiro oriundo da instituição e do programa PIPECTI.

References

Blom, K. J., & Beckhaus, S. (2007). Supporting the creation of dynamic, interactive virtual environments. *Proceedings of the 2007 ACM Symposium on Virtual Reality Software and Technology - VRST '07*, 51.

Cavallo, M., & Forbes, A. G. (2019). CAVE-AR: A VR Authoring System to Interactively Design, Simulate, and Debug Multi-user AR Experiences. *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, 872–873.

Chevaillier, P., Trinh, T.-H., Barange, M., de Loor, P., Devillers, F., Soler, J., & Querrec, R. (2012). Semantic modeling of Virtual Environments using MASCARET. *2012 5th Workshop on Software Engineering and Architectures for Realtime Interactive Systems (SEARIS)*, 1–8.

Richard, K., Havard, V., His, J., & Baudry, D. (2021). INTERVALES: INTERactive Virtual and Augmented framework for industriaL Environment and Scenarios. *Advanced Engineering Informatics*, *50*, 101425.

Thies, L., Strohmeyer, C., Ebert, J., Stamminger, M., & Bauer, F. (2019). Compiling VR/AR Trainings from Business Process Models. *2019 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*, 181–186.

Velev, D., & Zlateva, P. (2017). Virtual Reality Challenges in Education and Training. *International Journal of Learning and Teaching*, *3*(1), 33–37.