Integrating virtual worlds with Learning Management Systems: the MULTIS approach

Leonel Morgado
INESC TEC and Universidade Aberta
Coimbra, Portugal
leonel.morgado@uab.pt

Álvaro Almeida, Andreas Vilela, Bruno Pires, Márcio Cardoso
UTAD, Universidade de Trás-os-Montes e Alto Douro
Vila Real, Portugal, Country

Hugo Paredes, Benjamim Fonseca, Paulo Martins
INESC TEC and UTAD, Universidade de Trás-os-Montes e Alto Douro
Vila Real, Portugal
{hparedes,benjaf,pmartins}@utad.pt

Filipe Peixinho, Arnaldo Santos
Altice Labs
Aveiro, Portugal
{filipe-peixinho,arnaldo}@alticelabs.com

Abstract— **Learning Management Systems (LMS) provide minimal support for educational use of virtual worlds. Integration efforts assume the educators are inside the virtual world, providing hooks to services in the external LMS, to setup and manage virtual world activities. We present the inverse approach, enabling educators to setup and manage virtual world activities using the traditional LMS Web interface as an integral part of the overall educational activities of a course. In our approach, the LMS enables the teacher/trainer to setup, control, track, and store virtual world activities and its elements. It is the result of a joint effort by academic and corporate teams, implemented in the Formare LMS for OpenSimulator and Second Life Grid virtual world platforms. We explain how the Multis architecture can be used for integration, with concrete cases, an approach that can be implemented in other LMS and virtual world platforms, to overcome the limitations of existing systems for organizational management of e-learning activities.**

Keywords- LMS; virtual worlds; integration; OpenSim; SL

I. Introduction

Lack of integration between carrying out and setting up/managing educational activities in virtual worlds was identified as a factor hampering the widespread deployment of virtual worlds in education and training [1]. To set up and manage the activities in these environments, teacher/trainers must deal with an encumbering variety of administrative and technical tasks: login credentials managed separately; trainee/student tracking setup in the virtual world done by the teacher/trainer, and linked back to the learning management system (LMS); object repositories need to be managed in the virtual world by teachers/trainers with no connection with other learning materials stored in the LMS; the list goes on and on.

Our perspective is that this status of virtual worlds use in education and training sees each teacher/trainer as an island, isolated from modern organizational information systems and support services. That is, each teacher/trainer needs to technically set up the virtual world space and its activities and then sort out any connection plug-ins with other systems. Either that or at least worry about having a technical team to do all these tasks. And technical teams within education/training institutions may see that as an offshoot: even for technicians, virtual world management is not as streamlined as one would expect from a core operational information system.

Our approach inverts this perspective: teachers/trainers should be able to specify and supervise educational activities in virtual worlds as a seamless part of their overall teaching-learning plan, without the need of custom technical interventions. Just as they do, in fact, in traditional e-learning platforms: their concern should be with the educational/training content and dynamics, not with the technical/computational issues. To realize this perspective, we conducted a software engineering research effort, using as a prototype the integration of Second Life Grid (SLG) [2] and OpenSimulator (OpenSim) [3] platforms into a corporate-oriented LMS system, Formare [4].

In this paper, we demonstrate the use of the software architecture named Multis [5], which we developed for this purpose, by providing operational details. The architecture and prototype were developed for SLG/OpenSim virtual worlds and the Formare LMS, but the overall approach holds the potential for expanding as a generic approach to integrate virtual worlds/serious games with LMS platforms in general.

II. Related work

A. Integration requirements

While the literature provides extensive documentation of virtual world use in education and training [6], typically only the actual virtual world activities are reported. There are some integration efforts, summarized in the next section, but few actual accounts or surveys of integration requirements with other educational systems, including LMS [7]. We recently determined a set of such requirements, by conducting content and thematic analysis of documents

produced during four years of cooperation between academia and industry (e-learning provider). The results were 39 requirements and 54 sub-requirements, each supported by individual instances of documentation that originated it, and organized under 9 categories and several subcategories (ibid.). In Table I we present these categories, for the convenience of the reader.

TABLE I. INTEGRATION REQUIREMENTS – CATEGORIES[a]

Cat.	Subct.	Description
C1	-	Privacy of training sessions
C2	-	Record and replay behaviors of actors and other elements
''	C2.1	Recording the full events of a 3D session or generic 3D space
''	C2.1.1	Recording the behaviors of other elements
''	C2.2	Replaying the full events of a 3D session
''	C2.2.1	Replay the events in 3D
''	C2.2.2	Replay the events in 2D
C3	-	Support for virtual world content development
''	C3.1	3D space features manageable independently
''	C3.2	Support for at least 31 concurrent users
C4	-	Automated support for Administration
''	C4.1	Automated support for the administrative flow
''	C4.1.1	Tools/methods to track deployment & user adoption
''	C4.2	Federated authentication, LMS/virtual world platforms
''	C4.2.1	LMS users may use preexistent SL/OpenSim usernames
C5	-	Automated support for trainers and trainees
''	C5.1	Specific-purpose applications to support trainers and trainees
''	C5.2	Trainer should have control over trainee's audio
''	C5.3	Orientation support for trainees
''	C5.4	Ability to manage access to interaction with 3D objects
''	C5.5	Alternative avatar appearance identification features
''	C5.6	Support for training about the use of virtual worlds
C6	-	Access to the LMS data and services in the 3D space
C7	-	Integration of virtual world data in the LMS
''	C7.1	LMS accepts choreographies provided by trainees or trainers
''	C7.2	LMS accepts 3D models provided by trainees or trainers
''	C7.3	Ability to annotate the raw data from a session recording
C8	-	LMS must be the source of control and management over educational activities in virtual worlds
C9	-	Alternatives for voice communication in the 3D platform

a. Source: [7].

B. Integration efforts

Possibly the best-known effort for integration of SL/OpenSim virtual worlds and LMS is the SLOODLE project [8], which employed Moodle [9] as the focus LMS platform, and has over 10 years of development, providing a large array of integration features and tools [10]. But there have also been other, short-lived projects. For instance, the BbSL project, developed in 2008-2009, aimed to "manage, administrate and facilitate any hybrid Second Life / Blackboard Learn instructional experience" [11], or a special-purpose LMS called Vushi, whose website was active between 2010-2012 (acc. Internet Archive Wayback Machine, [12]), but some of its features and operation can still be seen in its YouTube channel, which remains available at the time of writing of this paper [13].

Other efforts focus on specific issues, rather than overall integration. For instance, tracking attendance [14], orchestrating avatar choreographies [15], or responding to control requests issued from virtual world scripts [16]. In this regard, there are similarities with the field of remote physical laboratories, where efforts have been made to orchestrate collaboration [17] and current concerns include federating authentication and conducting laboratory tasks under control of the LMS [18].

Both SLOODLE and the shorter-lived BbSL and Vushi projects have in common a perspective centered on the virtual worlds, not on the LMS: the trainer needs to set up tools and features within the virtual world and these can then be linked to the LMS, to access the information stored there. For example, these are SLOODLE's instructions for using its presenter tool [19]:

"*To setup the presenter:*
1. *Create a Presenter Activity on your moodle website*
2. *Create your slideshow, by adding images, or webpages, or videos as links in the bottom section*
3. *once complete, rez a presenter in Second Life*
4. *Click on it, to authenticate it with your moodle website*
5. *Once its* (sic) *been authenticated, click on the presenter again, to download the saved configuration*
6. *If the land is owned by a group, you must deed the presenter to that group*
7. *Once the Presenter has been deeded, it will automatically (…) after 5 minutes. If you don't want to wait 5 minutes for it to check if it has been deeded or not, you can just touch the Presenter for it to jump to the next step.*
8. *Now your presenter is ready.*
9. *Press play on your media settings.*
10. *You should now be able to see your presentation*"

All steps between 3 and 7 are in effect technical setup issues of the virtual world platform that the trainer should not have to deal with, since they take up time and effort. This perspective on integration sees the virtual world platform as the central point for managing the educational activities, with

the LMS being a secondary external service, hence we call it virtual-world-centric. We propose adopting an LMS-centric perspective, where the LMS is central point for managing the educational activities, of which the virtual world is but one location. Following this perspective, a presentation activity, once created on the LMS, should be ready and available for the trainer/teacher and trainees/students to use within the virtual world. This is a reflection of several requirements' categories of Table I, such as categories C5-C8.

Our approach aimed to enable the LMS to conduct within a virtual world platform the kind of set up actions that trainers typically have to do themselves when using approaches such as SLOODLE. This could be achieved by custom development, but that would couple the LMS code with the code of the virtual world platform. That would tie the LMS to a customized version of a virtual world platform, rather than enable it to keep up with the ever-changing diversity of platforms that have emerged and are emerging regularly. Therefore, our approach, dubbed Multis (the name of the project where it was created), took advantage of the fact that virtual worlds as a core feature enable remote users to login and interact with the world. While in game-oriented worlds this interaction can be quite limited, in creation-oriented worlds such as those supported by SLG or OpenSim platforms it can achieve most of the necessary setup tasks. Instead of laying these tasks on users, as in SLOODLE, BbSL, Vushi or other systems, the Multis software architecture enables LMS systems to spool a pool of automated clients, known as 'bots', to perform the tasks a user would. Trainers/trainees use the LMS interface and the LMS can use this Multis bot-spooling approach to conduct any associated setup tasks. This software architecture, particularly its bot-spooling approach, was presented in an earlier paper [5], and is summarized in section III.A.

III.   THE MULTIS APPROACH

A.   *Overall architecture*

As mentioned above, the bot-spooling approach used in the Multis architecture aims to allow LMS systems to be integrated with a large diversity of virtual world platforms and serious games, without requiring custom development on the virtual world side. That is, to enable integration even if they do not provide application programming interfaces (APIs) or other services for external systems. As we put forward in our seminal paper on the Multis architecture, "any online virtual world platform needs to provide login systems for clients. Thus, an LMS system can log into the virtual world platform (…) using automated clients" [11], typically known as 'bots'. In order to avoid bottlenecking systems integration through a single bot, we then introduced the bot-spooling approach.

Fig 1 presents the Multis architecture. The general LMS functions are represented in the "LMS logic" module. When a virtual world task needs to be performed, this module sends a request to the "Bot scheduler" module, which chooses an appropriate bot. The request is then converted into bot commands, issued to the "Bot logic" module, which keeps track of running code threads and open connections with the virtual world for controlling the 'bot' avatars, represented in the figure as the "Avatar/Bot" module inside the virtual world server. The bot logic module then carries out the commands with the bot chosen by the scheduler module.

While bots enable the Multis architecture to circumvent the lack of an API, the architecture can also take advantage of the existence of such an API or some other level of interconnection services provided by the virtual world platforms. Some platforms even allow users or administrators to provide code add-ons that can communicate with external systems. For instance, in SLG and OpenSim, end-users can provide scripts for virtual world elements with Web-based communications capabilities. For this reason, the MULTIS architecture foresees interfaces both for bot-mediated actions and for other forms of interconnection (Fig. 1). The following section explains how to use this architecture.

B.   *The four problem/solution vectors*

Using the MULTIS architecture, we addressed the requirements presented in section II.A under four
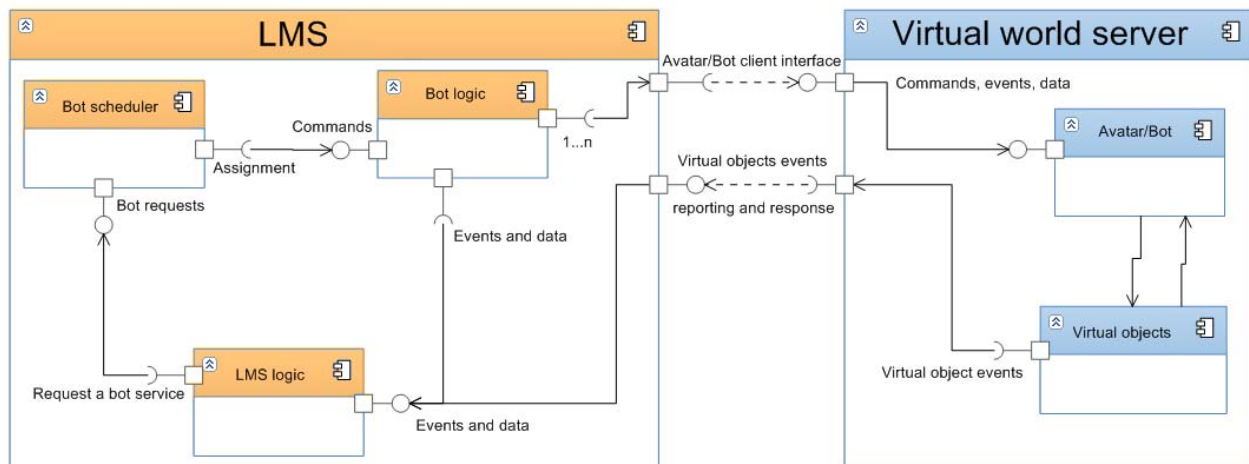

Figure 1.   The MULTIS architecture (from Morgado et al. [5])

problem/solution vectors, which we describe briefly. While the problems are generic, the proposed solutions are based on SLG/OpenSim virtual worlds.

Problem Vector 1: storage of proprietary virtual world content in the LMS. There are quasi-interchangeable formats for content such as 3D models and skeletal poses, but often content is proprietary, such as SLG/OpenSim link sets (i.e., groups of objects), scripted objects, avatar clothing, and more. Solution: we employ automated avatars as data stores for the LMS. The LMS has the credentials for these avatars and controls them programmatically to receive or pickup proprietary formats.

Problem Vector 2: placing LMS-stored virtual world content in the virtual world. Once the proprietary content is within the bot data store, it will need to be placed in the world for users. **Solution:** from problem-solution vector 1, bots act as data stores for the LMS. When necessary, commands are issued to the bot spool as requests, and assigned to a bot, which will log into the virtual world and place the requested content. If necessary, bots being part of the data store exchange content among them programmatically.

Problem Vector 3: receiving virtual world events and data in the LMS. In order to log data, respond to events and in general update the system status, the LMS must receive notification of events and collect data. **Solution:** we deployed a two-pronged approach. When speed of data collection or event reporting isn't critical, we use scripts in objects reporting data and events to Web services in the LMS. These scripts can be part of user interaction objects (placed using the vector 2 solution), part of specific invisible objects (id.) or injected into objects as necessary (see vector 4 solution, ahead). When data collection requires more timely responses, the LMS assigns a data collection request to the bot spool (see vector 2 solution). When virtual world platforms provide APIs, their services can be categorized according to their timeliness and combined with these two approaches.

Problem Vector 4: use the LMS to control the behavior of the virtual world. Following the requirements list [7], there is a plethora of situations were settings need to be adjusted, be it creating/deleting private voice chat groups, changing training room features, changing interaction permissions, resetting tools, and more. **Solution:** the LMS issues the necessary tasks to the bot spool, which then employs the bots to achieve them. This includes adding objects, injecting scripts, issuing private channel parameters and more. A complementary approach when timing is flexible is to have some scripts issue events to LMS Web services keeping the simulation/class state, and decide on necessary outcomes. If these are achievable through parameter passing, the LMS Web services respond with those, and the response can also include commands to be relayed to other scripts. This is the Pinheiro et al. method [16].

IV. USING THE PROBLEM/SOLUTION VECTORS TO IMPLEMENT SPECIFIC FEATURES

The list of requirements [7] is quite extensive, so we selected three cases to clarify how the problem-solution vectors were used to implement specific features, with the core concern of keeping the LMS Web interface at the helm of decision-making and the place where trainers go for control of the virtual world sessions and activities. In Table II, we summarize how each case demonstrates the use of the solution vectors.

TABLE II.   SAMPLE CASES AND SOLUTION VECTORS EMPLOYED

Case	Requirement	Solution Vectors
1	Training session space features specifiable on creation	Solution Vector 1 Solution Vector 2
2	3D objects should have user role-based permissions	Solution Vector 3
3	Recording actors' behaviors as a 3D choreography	Solution Vector 4

A. Case 1: requirement R2f-2 (ibid.), "Training session space features specifiable on creation"

In this case, 'features' can be simple items, such as chairs, or interactive elements, such as voting booths, simulators or games. One such element is a presenter tool for virtual world slideshows, which we will use in this paper for clarity, so that readers can compare it with the operation of the SLOODLE presenter tool, described in section II.A. Figure 2 show two such spaces we created. Both have slideshow panels on the right side.

Figure 2.   Training spaces with simple items and interactive elements

Typically, virtual world presenters employ a pair of interactive objects: a control board and a display. Instead of having the trainer or a support person create them manually in the virtual world, we used solution vector 1 and stored

these objects in the LMS, in the bot data store. When a trainer specifies in the LMS Web interface that a slideshow is required for a session, we use solution vector 2: before the session starts, two requests are issued by the LMS to its bot spool, each one for the placement of each object. The LMS knows the virtual world coordinates of the session space, since it was also created by the LMS when the teacher/trainer requested it in the LMS Web interface. Therefore, to create the presenter, the LMS logic, includes the coordinates in the requests it issues to its bot spool. The LMS bot spool logic then selects the bots, logs them into the virtual world platform, and accomplishes the setup of the slideshow display and control board.

B. Case 2: requirement R2g-6 (ibid.), "3D objects should have user role-based permissions"

In this case, 'permissions' are typically related to the kind of interactions that a participant can have with a 3D object. One such object might be the slideshow control board of the previous case. Suppose it may only be controlled in the virtual world by the trainer or by a student that is currently presenting. Roles are a LMS-specific logic (trainer, student, presenter, etc.), independent from the actual virtual world platform. The trainer edits the roles for a session in the Web interface, and the virtual world behavior must comply.

Figure 3 presents part of such a LMS Web page: each line deals with a different user. We cropped usernames, but the center column (heading: "Perfil", meaning "profile") identifies their roles as Tutor or student ("Aluno") for a course. For a given virtual world session, the Status column allows the trainer to specify each user's role for that session, and likewise for specific objects: Moderator, Participant, or Non-Participant ("Não Participante"). The remaining buttons are for convenience (e.g., "todos moderam", meaning "all can moderate") or for other features not discussed in this paper (e.g., "repor avatar", meaning "reset avatar").

Status	Perfil	..:: CMD 3D::..
Moderador ▼	(Tutor)	repor avatar
Participante ▼	(Aluno)	repor avatar
Não Participa ▼	(Aluno)	repor avatar
todos moderam	ninguém participa	alterar cancelar

Figure 3.   LMS Web page details for managing roles in the virtual space. Translations are provided in the text.

In this case, we used solution vector 3: the interaction script in the slideshow control board, when touched by an avatar, issues an event to a Web service in the LMS, which matches avatar ID with LMS user ID, decides whether that user has the adequate role to use the board and responds accordingly (allowing or disallowing the interaction). The board script then uses the response to ignore the usage attempt or to respond to it.

C. Case 3: requirement R2a (ibid.), "Recording actors' behaviors as a 3D choreography"

In this context, 'actors' are any participants in a virtual world training session or class, regardless of whether they are human-controlled or computer-controlled. And 'choreography', in this case, is the set of actions that participants have performed, including not only their motions, but their conversation, their interaction with virtual elements, and other aspects, such as facial demeanor, body gestures, etc. For instance, supposing one is conducting a training session with a role-play situation, the recording of actors' behaviors as a choreography should enable a later review of the performance and events, supporting reflecting learning approaches per methods such as after-actions review [20].

This can be achieved by employing a combinations of data collection methods. Silva et al.'s approach [15] is striving to make this independent from the LMS and other information systems, by describing these methods in ontologies and creating a managing system separating the LMS from the concerns of data acquisition. But this separation is beyond the scope of the current paper. Here we demonstrate how the LMS can be integrated with specific platforms – OpenSim or SLG – enabling it to operate without end user intervention.

In OpenSim or SLG, choreography data can be collected by a combination of methods: scripts in objects can detect interaction and report it as event to LMS Web services (employing solution vector 3) or even detect the presence and location of user avatars, if timing isn't critical (e.g., for tracking attendance [14]). LMS-controlled bots can collect live data about the behavior of other avatars, such as users, for more time-dependent cases. In both cases, we are employing solution vector 4. Since the LMS manages the session and the virtual space, it knows which objects are present. When the trainer requests the recording of a choreography, the LMS can use both approaches. For instance, it can issue requests to the bot spool to inject an event-reporting script into each object or to use bots to collect live data. When the teacher/trainer requests the choreography recording stops, the LMS again issues the necessary requests to its bot spool: deleting the injected scripts and logging out the data-collection bots.

Notice that the same solution vector 4 can also be used to setup live data collection bots adequately. For instance, "dressing" bots as invisible avatars, or locating them beyond the visual reach of default user cameras (but still within tracking distance) [21]. Or conversely, by dressing them as video-recording operators or as "You are being recorded" signs.

V.   CONCLUSIONS

The provided solution vectors have been devised from research using the OpenSim/Second Life Grid technological ecosystem, and hence bear some dependencies to it. Notwithstanding, the core control aspects of each case take place in the LMS, outside the virtual world and independent from it: knowing the state of session content, participants, roles, enabled features, etc. While other platforms don't provide such a powerful range of interaction capabilities to users (and hence bots), there are often alternatives that can follow the same rationale. For instance, HighFidelity [22] enables remote execution of scripts, rather than inside its

servers; Activeworlds [23] enables developers to link automated clients to collect data and provide interaction. The bot spool approach could, conceivably, be expanded to use a variety of such interaction approaches, abstracting the complexity from the core LMS control processes. That is, expanding to the overall interaction process the proposal by Silva et al. for storing and replaying avatar choreographies across different virtual world platforms [15].

This approach was implemented and tried out in a large-scale training course for a major telecommunications operator in Brazil. While there are no published accounts, we can testify that it was technically sound and operational, but curtailed by the need for massive teacher/trainer and student/trainee training in the use of virtual worlds. Hence we alert the reader for the need to combine the technical solution presented in this paper with a wider educational/training framework for deployment. Field research with end users is needed to improve and refine this architecture and indeed the requirements from which it emerged.

Finally, while the approach was designed for virtual world platforms, it may also hold potential for integration of serious games in LMS. Currently, serious games development is disconnected from LMS in much the same way virtual worlds were, but we hope efforts such as the our own described herein help map a path towards the full integration of not only virtual worlds but also serious games in LMS, and from that enable more widespread use of these technologies in education and training.

REFERENCES

[1] L. Morgado, B. Manjón, and C. Gütl, "Guest editorial: overcoming the technological hurdles facing virtual worlds in education: the road to widespread deployment," Educational Technology & Society, vol. 18 (1), 2015, pp. 1-2.

[2] Second Life Official Site - Virtual Worlds, Avatars, Free 3D Chat. http://secondlife.com

[3] OpenSimulator, http://www.opensimulator.org

[4] Formare, http://www.formare.pt

[5] L. Morgado, H. Paredes, B. Fonseca, P. Martins, Á. Almeida, et al., "A Bot Spooler Architecture to Integrate Virtual Worlds with E-learning Management Systems for Corporate Training," Journal of Universal Computer Science, vol. 22 (2), 2016, pp. 271-297.

[6] R. Ghanbarzadeh, A. Ghapanchi, M. Blumenstein and A. Talaei-Khoei, "A decade of research on the use of three-dimensional virtual worlds in health care: a systematic literature review," Journal of Medical Internet Research, vol. 16 (2), 2014, e47.

[7] L. Morgado, H. Paredes, B. Fonseca, P. Martins, R. Antunes, et al., "Requirements for the use of virtual worlds in corporate training - Perspectives from the post-mortem of a corporate e-learning provider approach of Second Life and OpenSimulator," iLRN 2016 Santa Barbara - Workshop, Short Paper and Poster Proceedings from the Second Immersive Learning Research Network Conference, Santa Barbara, CA. USA. Graz, Austria: Technischen Universität Graz, 2016, pp. 18-29.

[8] J. Kemp and D. Livingstone, "Putting a Second Life «Metaverse» Skin on Learning Management Systems," Proceedings of the First Second Life Education Workshop, Part of the 2006 Second Life Community Convention, August 18th-20th 2006, Fort Mason Centre, San Francisco. Paisley: University of Paisley, 2006, pp. 13-18.

[9] Moodle - Open-source learning platform | Moodle.org, https://moodle.org/

[10] Sloodle User Documents - SLIS Second Life Wiki, https://www.sloodle.org/docs/Sloodle_User_Documents

[11] J. Fillwalk, interview reported by G. Werner in "Online toolset will allow educators to maximize use of Second Life and Blackboard," https://apps.bsu.edu/CommunicationsCenter/Story.aspx?CategoryID= 81&MessageGuid=FB31E866-0086-4089-8728-CF2BDD88549C&OptIn=Y

[12] Internet Archive Wayback Machine, "http://vushi.org," https://web.archive.org/web/20120715000000*/http://vushi.org/

[13] vushination, https://www.youtube.com/user/vushination

[14] A. Madeira, P. Sequeira, L. Morgado, amd L. Gonzaga, "Controlo da Assiduidade em Aulas Efectuadas no Second Life," RISTI - Revista Ibérica de Sistemas e Tecnologias de Informação, vol. 5, 2010, pp. 87-100.

[15] E. Silva, N. Silva, and L. Morgado, "Model-Driven Generation of Multi-user and Multi-domain Choreographies for Staging in Multiple Virtual World Platforms," Model and Data Engineering, 4th International Conference, MEDI 2014, Larnaca, Cyprus, September 24-26, 2014. Proceedings. Cham, Switzerland: Springer International Publishing, 2014, pp. 77-91, doi: 10.1007/978-3-319-11587-0_9.

[16] A. Pinheiro, P. Fernandes, A. Maia, G. Cruz, D. Pedrosa, et al., "Development of a mechanical maintenance training simulator in OpenSimulator for F-16 aircraft engines," Entertainment Computing, vol. 5 (4), 2014, pp. 347-355, doi: 10.1016/j.entcom.2014.06.002.

[17] B. Jailly, C. Gravier, M. Preda, and J. Fayolle, "Interactive mixed reality for collaborative remote laboratories," MTDL '11 Proceedings of the third international ACM workshop on Multimedia technologies for distance learning. ACM, 2011, pp. 1-6, doi: 10.1145/2072598.2072600.

[18] Z. Al-Khanjari and Y. Al-Roshdi, "Developing virtual lab to support the Computer Science Education in Moodle," 2015 12th International Conference on Remote Engineering and Virtual Instrumentation (REV). IEEE, 2015, pp. 186-191, doi: 10.1109/REV.2015.7087290.

[19] SLOODLE Presenter - SLIS Second Life Wiki, https://www.sloodle.org/docs/SLOODLE_Presenter

[20] J. Morrison and L. Meliz, "Foundations of the After Action Review Process," Special Report 42. Washington, DC, USA: United States Army Research Institute for the Behavioral and Social Sciences, 1999.

[21] A. Vilela, M. Cardoso, D. Martins, A., Santos, L. Moreira, et al., "Privacy challenges and methods for virtual classrooms in Second Life Grid and OpenSimulator," 2010 Second International Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES). IEEE, 2010, pp. 167-174, doi: 10.1109/VS-GAMES.2010.30.

[22] High Fidelity. https://highfidelity.io/

[23] ActiveWorlds: Home of the 3D Internet since 1995. https://www.activeworlds.com/