*Journal of Universal Computer Science, vol. 22, no. 2 (2016), 271-297*
*submitted: 2/1/15, accepted: 29/1/16, appeared: 1/2/16 © J.UCS*

# A Bot Spooler Architecture to Integrate Virtual Worlds with E-learning Management Systems for Corporate Training

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

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

**Álvaro Almeida**
(UTAD – Universidade de Trás-os-Montes e Alto Douro, Vila Real, Portugal
alvaro_tiago_almeida@live.com.pt)

**Andreas Vilela, Filipe Peixinho, Arnaldo Santos**
(Altice Labs, Aveiro, Portugal
andreasvilela@gmail.com, filipe-peixinho@alticelabs.com, arnaldo@alticelabs.com)

**Abstract:** Joining efforts of academic and corporate teams, we developed an integration architecture – MULTIS – that enables corporate e-learning managers to use a Learning Management System (LMS) for management of educational activities in virtual worlds. This architecture was then implemented for the Formare LMS. In this paper we present this architecture and concretizations of its implementation for the Second Life Grid/OpenSimulator virtual world platforms. Current systems are focused on activities managed by individual trainers, rather than groups of trainers and large numbers of trainees: they focus on providing the LMS with information about educational activities taking place in a virtual world and/or being able to access within the virtual world some of the information stored in the LMS, and disregard the streamlining of activity setup and data collection in multi-trainer contexts, among other administrative issues. This architecture aims to overcome the limitations of existing systems for organizational management of corporate e-learning activities.

**Keywords:** Learning management systems, Standards and interoperability, Learning environments, Games, Computer Uses in Education, Software Architectures
**Categories:** D.2.11, K.3.1

## 1 Introduction

Training for employees of large corporations requires consistent delivery of standard content and interactions between a large number of trainers and trainees. Typically, teams of trainers are trained and managed with the support of trainer-focused manuals, guidelines and scripts. Thus, e-learning management systems (LMS) that focus on corporate training include management features so that training managers can ensure adequate consistency. For this purpose, such systems enable supervision and tracking of content delivery and interactions between large numbers of trainers and trainees. This is the case of systems such as Blackboard Learn [Blackboard, n.d.], Cornerstone LMS [Cornerstone, n.d.], Oracle iLearning [Oracle, n.d.], Saba [Saba, n.d.], and Formare [AlticeLabs, n.d.], the latter being system used in this work.

Between 2009 and 2012, joint teams at UTAD university and PT Inovação (PTIn), the company which developed the Formare LMS (PT Inovação is now called Altice Labs) cooperated to expand the range of available interaction modes in corporate e-learning management systems, integrating them with three-dimensional (3D) virtual world platforms, as described further ahead.

This integration effort was conducted from the perspective of the providers and managers of corporate e-learning. Rather than changing the typical experience of trainers and trainees during virtual world activities proper, the core concern of these professionals is that the activities are manageable across large numbers of trainers and trainees. I.e., that the deployment of activities, content, and features does not depend on individual initiatives, almost handcrafted, for tasks such as regulating access, providing the specific content stipulated for each session, enabling monitoring features, and others (more details on requirements are provided on section 5).

Thus, in this paper, we present the technological approach developed by the joint UTAD/Altice Labs teams, with the goal of enabling management of virtual world activities in corporate e-training systems, across large numbers of trainers and trainees. It consists of a software architecture developed to enable LMS-based corporate training systems to coordinate and manage 3D virtual world activities. Just as for traditional Web-based or conferencing-based activities, the architecture aims to enable supervision and tracking of content delivery and interactions between large numbers of trainers and trainees. The core idea behind this architecture is that since users access virtual worlds using a piece of client software, any server-based virtual world platform is sure to provide interfaces for client software to connect. Hence, even for virtual world platforms without a public application programming interface, an LMS can still connect to the virtual world platform as an automated user – known as a “bot” – and this bot metaphor can be used as an abstraction to integrate systems.

We start with a brief background on the educational use of virtual worlds, and then present the concept of virtual world platforms and their nature as autonomous software systems, which can be hosted alongside an LMS or externally. We proceed with a sample of corporate e-training management requirements that illustrate specific integration problems. We follow this with a presentation of existing approaches for integrations of virtual worlds with e-learning systems and their limitations. Subsequently, the developed architecture is presented. We conclude with several examples on how that architecture has been used in actual integration of virtual world activities with a corporate LMS: Altice Lab's Formare. This architecture may also hold the promise for integration of virtual worlds in any information system, but such claim cannot be put forth at this moment, since it was developed from LMS integration requirements and validated via LMS providers only.

## 2 Use of Virtual Worlds for Education and Training

The concept of virtual worlds has been used in various forms for decades, intertwined with terms such as “virtual environments” and “virtual reality.” However, currently it is commonly used to denote multi-user computational environments representing a space, within which users find themselves represented by visual avatars, as defined in detail by Morgado et al. [Morgado, 10]. While also used for text-only virtual spaces, the concept is most commonly used nowadays to denote massive multiuser online role-playing games such as World of Warcraft, and also social-oriented software platforms such as Second Life. In virtual worlds, the avatar is the metaphor for interaction with the contents of the virtual space and with other users. Like on social web spaces such as Facebook, there is awareness of user presence, but unlike social Web spaces, in virtual worlds the avatar and his/her/its spatial location is part of the interaction metaphor and context. If one removes the multi-user capability requirement and focuses only on the physical presence of an avatar in a virtual space, any game or simulation where the user controls an avatar or character would be a virtual world. However, such software falls outside the scope of this paper: being single-user it does not enable synchronous interactions between trainers and trainees.

The use of virtual worlds for training and education in general is well documented in literature, both for text-only platforms known as MUDs or MOOs (see the 1998 book by Haynes & Holmevik [Haynes, 1998] for a nice overview) and for the current generation of 3D virtual world platforms, such as Second Life, OpenSimulator, Activeworlds, Project Wonderland, and others. Current research on educational use of virtual worlds has been prolific, and for a panorama of the field we recommend starting with the scoping study of Sara de Freitas [de Freitas, 08] and the review conducted by Hew & Cheung [Hew, 10], and complement them with more recent perspectives such as the decade-long review of their use in health care [Ghanbarzadeh, 14] and a recent overview of current technological challenges they face towards widespread adoption [Morgado, 15]. Projects funded by the European Commission, such as MUVEnation [Pérez-Garcia, 09], VITA [Rodrigues, 09] or Euroversity [Motteram, 14] have also been instrumental in providing guidelines, recommendations, and activities for using virtual worlds for education and training.

In brief, educational and training approaches to leverage the affordances of these platforms aim to enhance the range of synchronous activities already conducted in chatrooms and videoconferencing [Cruz, 15]. They also seek to expand that range over to types of activities that were only possible in face-to-face training or in special-purpose multi-user serious games. E.g. use of the spatial distribution of avatars in a virtual auditorium to provide clear visual distinctions between which avatars are – at each particular time – presenting, and which are the audience; or the obvious option of enabling sets of trainees to conduct online role-playing activities or experience full-fledged interactive multi-user serious games (e.g., [Hudson, 09]; [Cohen, 12]).

However, from the perspective of corporate e-learning, current platforms lack management support for courses provided by large numbers of trainers to large numbers of trainees. For instance, if several training classes need to take place at the same time, then replicas of the intended classroom with identical features need to be available for each class. Consequently, trainers and trainees need to readily know the location of their intended classroom, and be ensured that only enrolled class members are present, to avoid likely disruptions should hundreds of people simply wander around the virtual classroom locations: whereas from a trainers and trainees perspective an unintended participation is only one occurrence, from the perspective of a provider or manager of corporate e-learning, all such disruptions imply an unmanageable quality of service.

## 3 Virtual World Platforms as Autonomous Software Systems

Currently, organizations have a large selection of technology for development and deployment of virtual world solutions. A key distinction is between developing from scratch or by leveraging a virtual world platform. To develop from scratch, creating a customized world, organizations typically use one of many available code libraries or game engines. This is, for instance, the common approach in many pieces of software referred to as serious games. Such an approach is resource-intensive, and developed for specific courseware, rather than for producing generic e-learning platforms. Our focus is on the second alternative: when organizations use of pieces of software that provide ready-to-use multi-user virtual spaces, i.e., virtual world platforms. Using a virtual world platform, the tasks for developing and deploying a virtual world focus on the creation of visual content (such as objects and scripted interactions), user access (network setup and user credentials), and activity management (organization of interactions, of the virtual space, etc.). Basic technical requirements, such as networking, rendering, messaging, and others, are provided by the virtual world platform, rather than having to be implemented. Notable examples of such platforms include Second Life Grid and OpenSimulator (both are described in detail by Sequeira [Sequeira, 13]), Open Wonderland [Kaplan, 11], Sirikata [Cheslack-Postava, 12], and OpenCobalt [Virtual Worlds Group, n.d.].

Organizations can either setup their own virtual world using some of these platforms, or rent the virtual world space and service from a third-party that hosts the servers and platforms (for the Second Life Grid platform, renting virtual world space is the only available option). Such third-party virtual world vendors typically manage a common technological platform and rent sections of the virtual world space to specific customers. In any case, the virtual world platform is a distinct service from the LMS platforms employed for corporate e-training and e-learning (Figure 1).

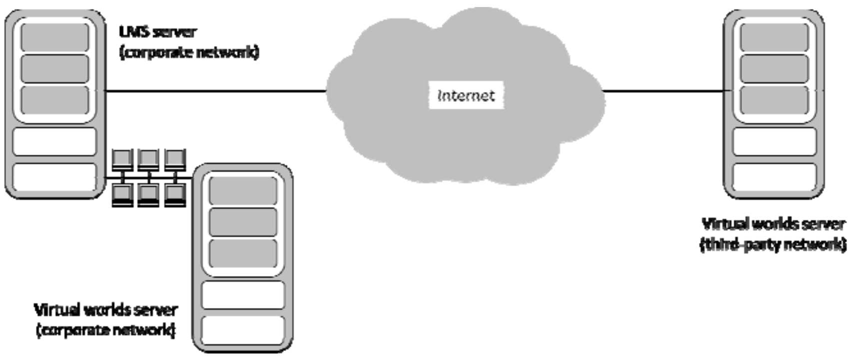



*Figure 1: LMS and virtual world servers as distinct servers.*

Some of the aforementioned virtual world platforms are open-source, which allows organizations to change their code in order to integrate them with external systems. However, this implies that an organization is either locked into the particular version of the platform which was used as a basis for integration, or is from then on committed to maintaining a private forked version of the public open-source code. Also, adapting the virtual world platforms for integration with external systems is not an option when considering hosting of the virtual world on third-party servers (which typically power more than a single customer or more than a single virtual world).

For this reason, the architectural approach detailed in this paper develops the LMS platform to leverage the interfacing features currently available in virtual worlds. Our perspective is that LMS platforms hold the management logic of the e-learning process, and virtual world platforms provide alternative modes of interaction, not the core e-learning management functionality. Under this perspective, managers of e-learning efforts are free to eventually decide to switch virtual world platforms in accordance with novel technology developments, to switch between hosted and rented alternatives or even use a combination of platforms and hosting alternatives.

## 4 Sample requirements for corporate e-training management of virtual world activities

Under the perspective that interactions in 3D virtual worlds are to take place within the wider context of traditional e-learning courses [Antunes, 08], enhancing or expanding the synchronous interactions, since 2009 a team at UTAD, with a background on virtual worlds research, cooperated with the PT Inovação/Altice Labs team behind the Formare LMS design, development, and business focus. Through regular meetings, technological trials and demonstrations, prototyping, and plain trial and error, the two teams developed a series of system requirements for integration of virtual worlds in the Formare LMS and within the context of corporate e-training/e-learning, which is the business focus of the Formare LMS team. This effort resulted on a commercial version of the Formare LMS, dubbed “LMS 3D”, whose first commercial deployment took place at a major corporation in Brazil.

The full list of requirements is large, but from a software engineering perspective the diversity of challenges is smaller, since several requirements end up demanding similar tasks from the software architecture. Therefore, in Table 1 we present a selection of those requirements which are representative of the variety of tasks that the integration architecture needs to support.

These requirements are quite high-level: they have been further subdivided into finer requirements. For instance, requirement R1 implies specific sub-requirements such as that each LMS user is associated with a 3D virtual world user; that virtual rooms can be created on cue; that existing virtual rooms can be tracked and managed; and several further sub-requirements associated with the various features mentioned in the R1 description.

From a workflow perspective, the requirements imply that integration of 3D virtual worlds with an LMS for the context of corporate e-learning and e-training needs to support sequences of operation that can be initiated at either end. That is, while using the Web interface of the LMS, user actions may require automated intervention in the virtual world (for instance, creating a room); and while using the

3D virtual world (under this perspective, also an interface of the LMS), user actions may require automated intervention on the LMS data and/or services (for instance, recording attendance).

| ID | SAMPLE REQUIREMENTS | DESCRIPTION |
|---|---|---|
| R1 | Creation of synchronous 3D sessions | Course managers should be able to use the LMS Web interface to schedule synchronous 3D sessions, specify which LMS users can attend it, whether to use an existing virtual room or create a new one, which features are available (such as slide projectors, chat recording, attendance registry, etc.), and which learning materials are required (presentations, 3D objects, etc.). |
| R2 | Archival of interactive 3D virtual world content | Course managers should be able to allow specific LMS users to provide interactive 3D content for archival. Course managers should subsequently be able to manage this content via the Web interface, in concert with existing methods for management of non-3D content (documents, presentations, images, sounds, videos). |
| R3 | Use in the virtual world of 3D content archived in the LMS | LMS users in the 3D virtual world should be able to access the 3D content stored in the LMS (according to access rights registered in the LMS) and interact with it. |
| R4 | Provision of 3D content to LMS users | Course managers and trainers should be able to specify automated 3D content delivery to specific LMS users, in the virtual world, both prior to a 3D session or during it. |

*Table 1: Sample requirements for integration of 3D virtual worlds in an LMS platform.*

Figure 2 provides a summarized overview of the requirements: users at the Web interface either request actions to be carried out in the virtual world (for instance, creating a virtual classroom) or request that data collection takes place (for instance, tracking attendance during a synchronous session). At the virtual world interface, users either do something that produces data (such as being present or interacting with objects), or something that requires data from the LMS (for instance, request delivery of materials, or an interaction attempt with a virtual object may originate a request for permissions; it can also be something as plain as entering a room, if the system aims to automatically react to that event).


*Figure 2: Abstract use cases perspective underlying the architecture design.*

## 5 Existing Approaches for Virtual World and LMS Integration

Somewhat surprisingly, in spite of all the research attention that virtual worlds have enjoyed over the past years, there have been only a few limited efforts to integrate them with the numerous LMS used today. And those are entirely geared towards Second Life/OpenSimulator platforms. The best known is the open-source SLOODLE project [Kemp, 06], focusing on integration of Second Life or OpenSimulator platforms with Moodle, an open-source LMS [Ferreira, 05] geared towards educational settings (rather than corporate environments). From providers of corporate LMS services, besides the UTAD/Altice Labs effort originating this paper, there is only the BbSL project, integrating Second Life with Blackboard Learn [Werner, n.d.]. There is also an autonomous LMS system, Vushi, specifically aimed at providing some LMS features for Second Life-based teaching [Texas State Technical College, n.d.], hence not a general-purpose, corporate-oriented LMS at all. There are also several smaller efforts, which provide technological solutions for specific integration issues, but without broader system integration concerns (e.g., Madeira et al. [Madeira, 10] proposed a Moodle-based system for automatic attendance registration of Second Life-based classes).

The BbSL project was developed between 2008 and 2009 by Ball State University's Institute for Digital Intermedia Arts (IDIA), under the Blackboard Greenhouse Grant for Virtual Worlds [Blackboard, 08]. It developed a set of tools both virtual and Web-based, allowing users to “manage, administrate and facilitate any hybrid Second Life / Blackboard Learn instructional experience” (Fillwalk, 09, acc. Werner [Werner, n.d.]).

SLOODLE is a project that evolved from its original proposal in 2006, where the goals where to provide Second Life users with access to content stored in a traditional Web-based LMS, Moodle [Kemp, 06]. In this sense, its origins are the opposite of the UTAD/PTIn architecture proposed in this paper: no control or management on part of the training manager, but rather support for independent teachers and sets of students – in fact, this system evolved from a survey of teachers and students (ibid.). SLOODLE has since evolved [Livingstone, 09] to include some support for registering in the LMS data about virtual world activities. For instance, the SLOODLE Tracker module allows the use of Web pages for tracking which virtual world tasks where completed by the students [Callaghan, 09]; and the SLOODLE Prim DropBox module allows using Web pages to track delivery of 3D assignments by students [Farley, 09].

In this regard, all existing systems (Vushi, BbSL, and SLOODLE) have a focus on individual trainers or trainees, rather than organizations: they focus on providing e-learning systems with information about educational activities taking place in a virtual world and/or being able to access within the virtual world some of the information stored in the e-learning system. They do not provide support for LMS-centred control and management: trainers and learners need to access specific locations and objects inside a virtual world in order to get or setup virtual objects and tools necessary for a task, rather than have them preset or delivered automatically; trainers and learners need to setup their own virtual world accounts and then associate them with the LMS, rather than have management support for their account on part of the e-learning provider. That is, while these existing systems provide support for individual teachers and trainers to use virtual worlds in connection with an LMS, they do not provide support for organizational management of corporate e-learning activities. Taking the sample requirements presented in Table 1, as an example, requirements R1 and R2 are not met by either Vushi, BbSL or SLOODLE. R3 can be accomplished by these systems, if necessary. And R4 is only partly met: while Vushi, BbSL and SLOODLE allow users to automate content delivery, their approach is virtual-world centric, not LMS-centric. That is, a training manager/coordinator cannot use the LMS Web interface to manage content across courses, training modules, and sessions, and easily specify the distribution of 3D content to specific users.

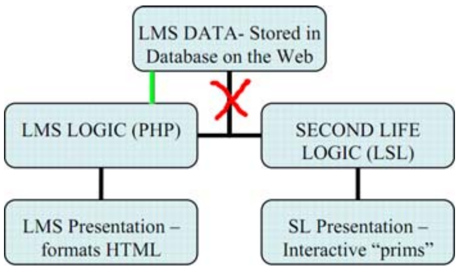



*Figure 3: SLOODLE architecture [Kemp, 06], corrected by us to match its actual operation (crossed link removed, light green link on the left added).*

Figure 3, which details the SLOODLE architecture, further clarifies this individual-oriented perspective. In the SLOODLE architecture, the virtual world logic (in this particular case, Second Life logic) is an interface to the LMS data, allowing virtual world events to report or query LMS data through the LMS logic. But what if the triggering event is not a virtual world event, but an LMS-originated event? For instance, a request by a training coordinator; a timer-triggered action; or some other event caused by the overall management logic? The SLOODLE architecture does not take into account LMS-originated events. For that, it would be necessary for the virtual world platform to be able to accept incoming requests of the LMS logic.

To use virtual worlds as enhanced synchronous modes of interaction for corporate training, we sought to develop an LMS-centric (or organizational-centric) architecture. From requirements such as those presented in the previous section, our focus was on an architecture where communication could be initiated at either the LMS logic or the virtual world logic, and also an architecture that could be employed for different virtual world platforms, rather than be restricted to Second Life.

As an inspiration for a software architecture that would purvey this LMS-centric perspective required for integrating virtual world in corporate e-learning systems, we looked at two kinds of systems which bear some similarities with the current topic. Namely, we looked at integration in e-learning systems of remotely-operated or simulation-based laboratories for educational purposes, and of training simulations.

Parallels with virtual worlds can be readily made regarding remotely-operated laboratories for educational purposes, since some of these systems allow interactive experiments, where a student is remotely operating a laboratory server. And the integration of these remote labs into e-learning systems has been receiving some attention from the technology-enhanced learning community (e.g., [Rapuano, 06]). Indeed this line of research is LMS-centric: the LMS is the basis for the learning activities, performed on a different system. In particular, some remote labs systems now enable synchronous cooperation for conducting remote experiments, further increasing the parallelism with virtual worlds (e.g., [Bochicchio, 09]). E.g., Richter et al. [Richter, 10] proposed virtual worlds as collaborative environments for collaborative remote laboratories. It is thus not surprising that research on remote labs integration in LMS systems has parallels with integration efforts for virtual worlds.

Often research deals with the resolution of specific issues, without broader system integration concerns: for instance, whereas Madeira et al. [Madeira, 10] tracked virtual world class attendance in Moodle, Ferreira & Cardoso [Ferreira, 05] booked laboratory equipment, also in Moodle. Where ontologies and protocols have been developing to orchestrate and share virtual world content (see Table 2, ahead, and the recent proposal by Silva et al. [Silva, 14]), ontologies have also been used to describe remote labs interfaces and orchestrate collaboration [Jailly, 11]. This is emphasized by Bochicchio & Longo [Bochicchio, 10]: they describe the integration with an LMS of a laboratory server for remote interaction with an electron microscope. While ignoring the lab server's collaboration features, they report concerns with striking similarity to the requirements we mentioned: namely, need for "adoption of a single sign-on technique to authenticate and authorize Moodle users to interact with the lab equipment" (ibid., p. 313) and the need to "exchange messages between the LMS and the CRL [(Collaborative Remote Lab)] runtime environment, in order to perform the tasks made in the LMS with effects in the CRL (like a lab reservation) and vice-versa" (ibid.). They have not yet provided an architecture to address these concerns, but the concerns have been echoing in more recent papers on integration of remote laboratories with LMS systems [Al-Khanjari, 15]. Indeed both communities are coming across similar issues and may benefit from following each other's approaches.

Regarding the integration of training simulations in e-learning systems (e.g., [Ribeiro, 11]), a key distinctive factor is that computer simulation systems have now benefited for over 10 years of a standard architecture enabling their distributed interoperability: HLA, described in IEEE standard 1516 [Symington, 00]. This architecture enables communication and data exchange between different simulators, albeit control services have been lacking [Jie, 13]. Several researchers have proposed methods for integration of simulation systems and e-learning systems, leveraging the HLA architecture. These approaches rely on a well-known standard for integrating interactive learning objects in e-learning systems, called SCORM [Advanced Distributed Learning, 09]. The most recent proposals for LMS integration of simulations are based on creating SCORM-compliant learning objects with a module that federates with HLA-compliant simulators (e.g., [de Penning, 08]; [Jiménez, 08]). In these approaches, when a user accesses the SCORM learning object in the LMS, he/she is in fact communicating with the simulator server as a distributed part of it.

While HLA-compliant simulation systems are now a reality, standardization work for virtual world interoperability has not achieved similar status. At least 8 working groups or organizations have ongoing conflicting proposals of interoperability standards of virtual worlds (Table 2). The MPEG-V (ISO/IEC 23005-1:2011) standard may eventually arise as the equivalent of HLA for virtual world platforms, but so far no virtual world platform is compliant; a situation we can only hope changes in the future – either with this or with some other standard [Morgado, 09].

Still, the SCORM-based approach of these architectures does not entirely solve the requirements of corporate e-training. While adequate for the actual participation of users in simulations, it does not tackle, for instance, requirement R1 from Table 1, or any other where a training manager needs to setup or preconfigure the simulation server. This derides from a basic distinction between simulation servers and virtual world servers. While the former provide a ready-made scenario with predetermined interaction rules (possibly with some parameterization), the latter provide the more basic services of virtual content hosting, user-to-user communication, and user-to-user/user-to-object interaction. That is, virtual world platforms have the potential to provide users with the ability to make their own scenarios and content, to determine their own interactions. Some game-oriented worlds restrict this, but the potential is there. Hence the ambition of allowing an external system (such as an LMS) to streamline and manage this level of scenario setup and configuration. Requirement R1 is but an example of this.

The architecture devised by UTAD and PTIn, which we named MULTIS, aims to provide a solution for this ambition of integration of virtual world activities in the management features of corporate e-learning systems. It looks at the connection of the LMS logic with the virtual world logic, and provides a solution for the limitations pointed out in our description of the SLOODLE architecture and for those pointed out above for HLA-based architectures.

| Group and URL | Comments |
|---|---|
| HTML5 + WebGL http://www.khronos.org/registry/webgl/specs/latest/ | The WebGL standard was developed by the Khronos Group consortium. It specifies an application programming interface (API) for 3D rendering on the Web, as a context for the HTML5 "canvas" element. |
| IEEE VW Standard Working Group http://archive.is/www.metaversestandards.org | This group aimed to develop "Metaverse Standards", with addressing, interfaces, and inter-component communication (devices, servers, objects, scripts, avatars). Focus: interoperability of avatars and objects. |
| Immersive Education Initiative http://www.immersiveeducation.org | This association announced in 2010 the creation of a free open file format for 3D meshes, aiming at allowing the sharing of 3D models across virtual worlds. This initiative is particularly relevant because it is supported by some of the main current virtual world platforms: realXtend, Open Wonderland, OpenSimulator, Open Cobalt, and Sirikata. |
| MPEG-V (ISO/IEC 23005-1:2011) http://mpeg.chiariglione.org/standards/mpeg-v | ISO/IEC standard focused on interoperability of avatars and virtual objects, but also on control links with the physical world (e.g., so that an avatar's face can be controlled from sensors determining the behavior of the physical face of the user). |
| MXP – Metaverse eXchange Protocol http://archive.is/http://www.bubblecloud.org/ | A protocol for interlinking and federating servers and identities, with sharing of 3D objects, to support virtual worlds with a continuous 3D space, with avatars that can travel transparently across server borders. |
| VWRAP – Virtual World Region Agent Protocol https://datatracker.ietf.org/wg/vwrap/charter/ | Group at the Internet Engineering Task Force, with similar goals as the MXP. Originated on efforts by Linden Lab and IBM for creation of a standard based on current client-server protocols used by Second Life and OpenSimulator. It is currently stagnant and considering whether to close down or integrate its efforts with other identity-federation efforts. |
| X3D (ISO/IEC 19775-1:2008) http://www.web3d.org/ | ISO/IEC standard for defining 3D interactive models integrated with multimedia data. Evolution of the well-known VRML modeling language. Focuses on virtual world content, in terms of visuals and interactivity, and the visual navigation across that content. It does not address server interconnection, identity federation or connection with physical systems. |
| Zelestra https://web.archive.org/web/20141222181452/http://www.zelestra.org/ | A private company about which little information is available. It stated the production of standards with similar goals to those of MXP, without addressing 3D model representation. |

*Table 2: Groups working on standards for virtual world interoperability. This table expands the list compiled by the IEEE VW Standard Working Group in 2010 – vd. the URL of that group, in the table.*

# 6 The MULTIS Architecture

## 6.1 General description

An underlying idea of this architecture is that any online virtual world platform needs to provide login systems for clients. Thus, an LMS system can log into the virtual world platform, even if it doesn't provide an application programming interface (API) for external systems, using automated clients. The virtual world platform responds to these automated logins by generating matching avatars, which are provided with data about the virtual world environment and other participants, and interact with those participants and the environment. In common virtual world lingo, such automated clients are typically referred to as "bots".

This bot-centred approach allows LMS systems to be integrated with a greater range of virtual world platforms – particularly given that the overwhelming majority of virtual world platforms don't provide any API for interfacing with external systems. However, it faces different constraints. A first issue is that a single bot cannot provide all the services that LMS integration requires, since it experiences the virtual world through an embodied perspective. For instance, the data it receives is limited to the virtual geographic area where it is located at a given moment; if data collection is required from several different locations simultaneously, using a single bot would require constant relocations for alternate data collection from each location. But most seriously, since a bot is treated by the virtual world server as any other avatar, actions upon the environment cannot be expected to yield a notification or acknowledgment upon completion: on instances, the bot must monitor the environment to confirm the completion of any actions it has requested upon the environment. For instance, if the bot requested the creation of a virtual chair, it needs to monitor the environment for confirmation of the creation of that chair.

To overcome this potential performance bottleneck, our MULTIS architecture employs scheduling and spooling concepts, common in other fields of computing, such as process management and printing systems. In short, the architecture proposes that e-learning systems include a pool of bots for interfacing with virtual world platforms, and schedules tasks between these bots, in order to circumvent performance bottlenecks and location-related constraints. When virtual world platforms include modular features for interfacing with external systems (i.e., features that do not require specific changes to the underlying code of the virtual world server), the architecture also accepts those for communication. This is the common case of using end-user scripts in objects, but since some different implementations of this possibility exist in virtual worlds, we will refer to this as "code add-ons".

Figure 4 presents the overall MULTIS architecture. On the left side, the section titled "LMS" represents the Learning Management System and its relevant components. These are impacted in terms of code changes to LMS systems that wish to include the MULTIS architecture. On the right side, the section titled "Virtual world server" represents the virtual world platform and its relevant components. These are only presented for illustration purposes, since the MULTIS architecture for LMS systems foresees no code impact on the side of the virtual world platform.

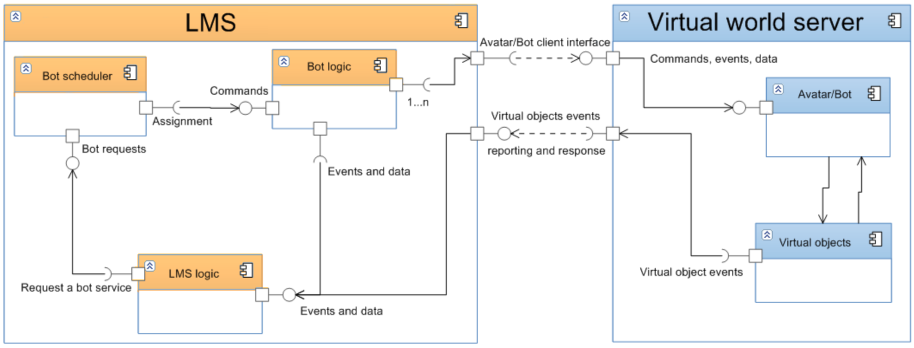



*Figure 4: The MULTIS architecture.*

In this architecture, user-initiated actions can take place either within the virtual world or on the traditional LMS Web-based interface (and any derivatives, such as desktop or mobile interfaces). Actions initiated in the traditional LMS Web-based interface, are here collapsed into the component “LMS logic”, since the actual details vary among different systems. Actions initiated within the virtual world generate data and events that may be of interest to the LMS, either via actions of externally-controlled agents (“avatars”), or via code controlling the behaviour of virtual objects. Both sources of data and events are here represented by the “Virtual objects” component, within the “Virtual world server” block.

The remaining blocks are the core functionality of the MULTIS bot-spooling architecture: in order to enable the LMS to act upon the virtual world, the LMS "Bot logic" component is in charge of logging into the virtual world platform with the automated clients (bots). The avatars generated by the virtual world platform in response to these logins are represented in the right-side block by the “Avatar/Bot” component. The selection of which bot to use to fulfil requests issued by the LMS logic, and any waiting and task management that are necessary when no bots be available, take place within the “Bot scheduler” component.

## 6.2 Operation

To clarify the operation of systems using this architecture, in Table 3 (split into three parts) we explain its operation for the requirements as summarized in Figure 2: Web user requests action; Web user requests data collection; virtual world user produces data; virtual world user requests data.

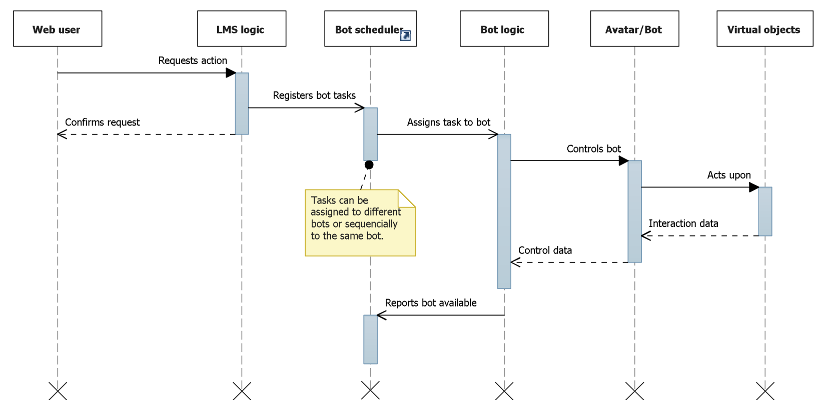
| **Web user requests action** |
|---|
|   |
| In this case, the user requests an action upon the Web interface. The LMS logic converts that action request into bot-specific tasks, which are registered in the bot scheduler. The bot scheduler then selects one or more available bots, assigning the tasks to those bots (if there are more tasks than available bots, the scheduler will assign tasks sequentially, as bots complete prior tasks). Then, for each bot, the bot logic logs it into the virtual world (if necessary), and executes the control actions necessary to achieve the requested tasks. In the virtual world server, the avatar/bot actually carries out the tasks, and (if necessary) acts upon the virtual world environment (objects, environment, other avatars). Once the task is complete, the bot logic reports to the bot scheduler that this bot is now available.<br>One should note that some action requests may be persistent, i.e., the bot is required to perform the requested tasks until requested to stop. In such a case, the bot is only reported as available after that stop request is issued and processed. |

*Table 3, part 1/3: Sample requirements for integration of 3D virtual worlds in an LMS platform.*

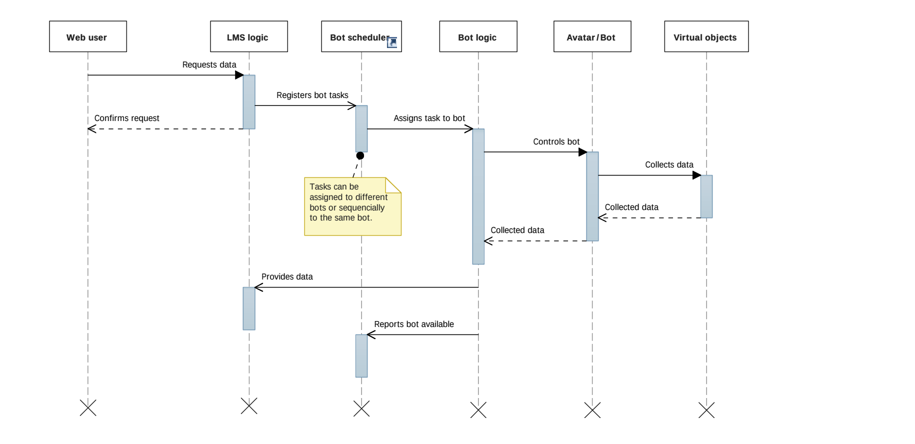
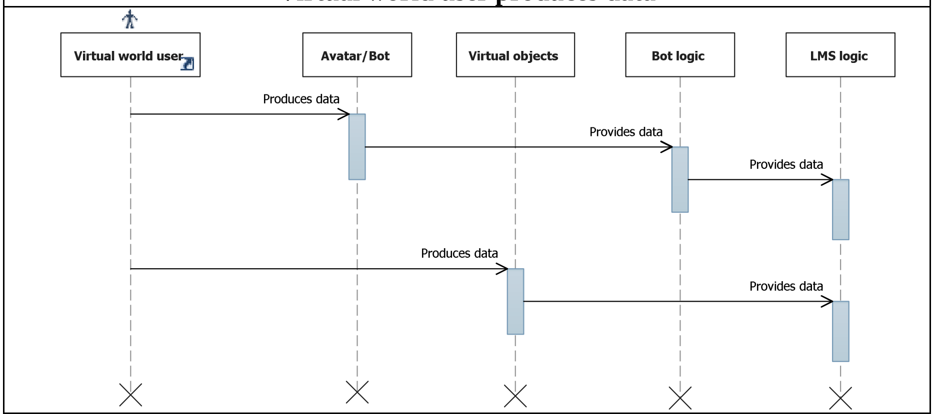
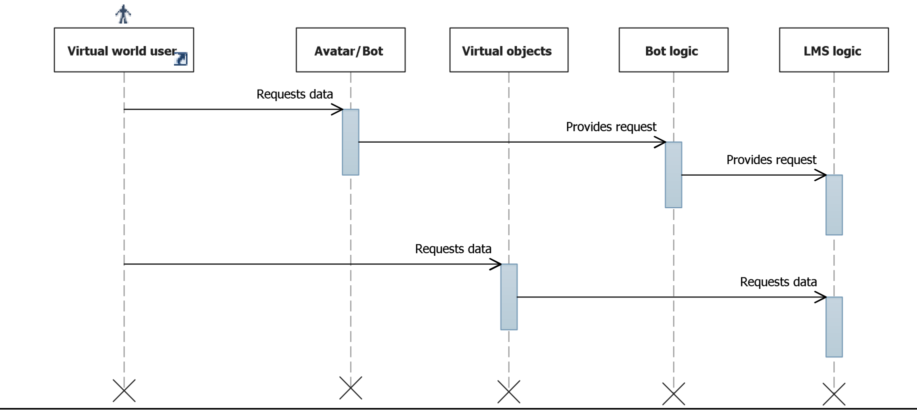
| Web user requests data |
|---|
|   |
| When a Web user requests that data are collected, the process is similar, with the major distinction being that the collected data need to flow upstream until they reaches the LMS logic. One should note that, just as in the previous case, data collection may be persistent. I.e., the data collection and their provision to the LMS logic may be done continually until a stop request is issued. In such a case, the bot is only reported as available after that stop request is issued and processed. |
| **Virtual world user produces data** |
|   |
| The data produced by virtual world users are either observed/collected via a bot of the MULTIS architecture or (if the virtual world platform allows it) via code add-ons to the virtual world platform (e.g., scripts). From an architectural point of view, the only difference is that bots collect data and pass them to their bot logic which further sends them upstream to the LMS logic, whereas data originating from code add-ons can be sent directly to the LMS logic (via HTTP requests or some other network communication method that is available for those add-ons). |

*Table 3, part 2/3: Sample requirements for integration of 3D virtual worlds in an LMS platform.*

**Virtual world user requests data**




Data requests from users follow a sequence that is identical to that for data production, with requests being provided either to a bot or to a virtual object. One should notice that the diagram above does not include any response to the data request. This is intentional: while in the simplest cases there can indeed be a direct response, in the most generic case that response may have to be performed as a sequence of actions, in which case we are in a sequence identical to the first diagram ("Web user requests action").

*Table 3, part 3/3: Sample requirements for integration of 3D virtual worlds in an LMS platform.*

### 6.3 Example of operation for the sample requirements

In Table 1 we presented some sample requirements for virtual world integration into corporate LMS systems. For further clarification of the architecture's operation, we will now explain how those cases would unroll (vd. Table 4, split into three parts, starting with R2 for pagination reasons).

| **R2 – Archival of interactive 3D virtual world content** | |
|---|---|
| For users to provide to interactive 3D content for archival, the LMS system needs to accommodate this form of data (and possibly code). | When a virtual world platform accepts external 3D content, it is simply a new data format for storage in the LMS logic. Some platforms don't allow that, and all content is generated within the virtual world itself. And if they do accept external content, mostly exclude interactive content. But typically avatars have "possessions", including interactive content. Bots of the MULTIS architecture can thus be used as storage interfaces for virtual world interactive content. |

*Table 4, part 1/3: Architecture operation for the sample requirements of Table 1.*

| **R1 – Creation of synchronous 3D sessions** | |
|---|---|
| To schedule a synchronous 3D session: the LMS system needs to allow course managers to:<br>• specify which LMS users can attend it;<br>• whether to use an existing virtual room or create a new one;<br>• which features are available;<br>• which learning materials are required. | This requirement involves new information, and the matching data are stored in the “LMS logic” component. This includes the need to have avatar identification associated with each user – akin to having other personal data – in order for the system to be able to control access and attendance. To implement access control, it needs to employ the “Web user request data” sequence. Virtual world attendance will originate the “Virtual world user produces data” sequence, and in case it is necessary to intervene to impede access of some virtual world user, the LMS system will use the “Web user requests action” sequence. |
| | If a new virtual room is to be created, the LMS will use the sequence “Web user requests action” for creating the room. |
| | If a feature needs to be available, it can involve creation of virtual world content or provision of parameters to code add-ons running within the virtual world. In both cases, the sequence “Web user requests actions” can be used. However, in case of parameter provision to code add-ons, in case they are regularly polling the LMS system for them, this can also be achieved via the “Virtual world user requests data” sequence. |
| | Learning materials such as presentations, text, voice, 3D objects or even choreographies of virtual actors, may involve creation of virtual world content or parameter-passing, as in the case above. But in the more complex cases, such as experiencing LMS-controlled simulations or choreographies, all sequences may be necessary, since data are required from both ends and actions may need to be generated in response to those data. |

*Table 4, part 2/3: Architecture operation for the sample requirements of Table 1.*

| **R3 – Use in the virtual world of 3D content archived in the LMS** | |
|---|---|
| For users of the virtual world to be able to access 3D content stored in the LMS and interact with it, it is necessary for that content to be placed in the virtual world. | Following the case for R2, above, the solution is similar: whether the content is simply stored in the LMS as a new data format, or inside the virtual world but available to the LMS as "possessions" of its bots, the content can then be placed within the virtual world using the "Web user requests action" sequence. |
| **R4 – Provision of 3D content to LMS users** | |
| To automatically deliver 3D content to specific virtual world users, the LMS system needs to transfer content to those user's avatars. | Content provision to avatars in virtual worlds typically can be done by other avatars. When the platform supports code add-ons, usually those add-ons can also fulfil this task. In both cases, the content needs to enter the virtual world first, as explained above for case R3. Its transfer can then take place from LMS bot to users' avatars, or – after being placed within reach of a code add-on – via provision of parameters to the code add-on. Both cases can be achieved via the "Web user requests action" sequence. |

*Table 4, part 3/3: Architecture operation for the sample requirements of Table 1.*

## 7 Sample Application of the MULTIS Architecture to Integrate Second Life Grid and OpenSimulator Virtual Worlds in an LMS: Formare

### 7.1 Architecture concepts applied to Second Life Grid/OpenSimulator platforms

We have implemented this architecture in the Formare LMS for the specific case of the Second Life Grid/OpenSimulator platforms, including the four requirements listed in Table 1 (R1-R4) and detailed in Table 4, plus several others. Presenting their implementation in detail is beyond the scope of this paper, but we believe that providing some examples can render more concrete the abstract description of the MULTIS concepts, and contribute to the reader's understanding.

From the perspective of the MULTIS architecture, these two platforms are identical. Significant differences in behaviour are routinely found by developers, but at lower levels of implementation or regarding specific features, not affecting this discussion, that were described in detail by Sequeira [Sequeira, 13]. This is due to the origins of the OpenSimulator platform, which started as a project to create an open-source version of Second Life server software. Initially this was taking place by reverse engineering and analysis of the communication between Second Life client software (known as "viewer") and server software. However, following the 2007 release of the source code for the official Second Life viewer [Linden Research, 07], both of these platforms share a common protocol for communication between virtual world servers and client software, which they refer to as "viewer" – the Second Life Grid Open Grid Protocol, or SLGOGP [Linden Research, 08]. Hence, the subsequent descriptions of the use of the MULTIS architecture apply to both platforms.

In these platforms, avatar control by external systems, as bots, is streamlined. More than just the existence of source code for client software, as mentioned above, there is an open source collection of .Net libraries in C#, libopenmetaverse, which implement the protocol and core functionalities, such as networking and specific data handling [LibOpenMetaverse, n.d.]. These libraries were used to develop our "Bot logic" components. Avatars – including bots – have possessions, called "Inventory", which can be used as LMS stores for virtual world items. In these systems, clients (avatars/the bot logic) can only communicate with the virtual world server, which determines and routes any communication towards other virtual world content, including virtual objects and other avatars.

These platforms support code add-ons, under the form of text scripts written in Linden Scripting Language or LSL [Linden Research, n.d.]. These can be uploaded to the virtual world server by any user (including bots) and are executed by being associated with a virtual world object. I.e., under the adopted metaphor, the scripts are placed "inside" the object. Scripts cannot be associated with terrain, avatars, other scripts, or any other virtual world elements besides virtual world objects. These scripts have significant capabilities for interacting with virtual world content, but for the purpose of the MULTIS architecture, the relevant aspect is that they can issue HTTP requests to external systems. They can also, temporarily, act as HTTP servers.

### 7.2 R1 – Creation of a synchronous 3D session

The Formare LMS already had a "synchronous session" feature using traditional audio-visual conference services with text chat. Workflow features such as session creation, scheduling, enrolment, and others, were already part of the LMS logic. To expand it to cover virtual 3D sessions, we created the concept of "session type", so course administrators could select whether to create an audio-visual conference or a virtual 3D session. This way, the workflow for the course administrator and trainees associated with the virtual 3D session type is identical (alerts, calendars, etc.). New data items were created, and Figure 5 provides a screenshot of the associated Web form (in Portuguese). These include trivial data items such as title, session type, start and end dates and times, and checkboxes for activating desired features or content such as slide projection, chat recording, attendance records, etc. The "Local" radio button group, meaning "Location", allows the administrator to specify where the session will take place: in the pre-existing 3D space assigned to the training course; in a new room to be created (both were in a private OpenSimulator server), or as study tour within the Second Life world. Finally, on the bottom the administrator can control the access to this session: on the left side there is a list of trainees enrolled in the course, which can be moved to the right-side list in order to be included in this session and allowed access to it. To implement access control based on LMS users, not virtual users, we included in the LMS extra data items for the users. Specifically, a list of virtual world identities: the user's identity in Second Life, in the organization's OpenSimulator server, etc.

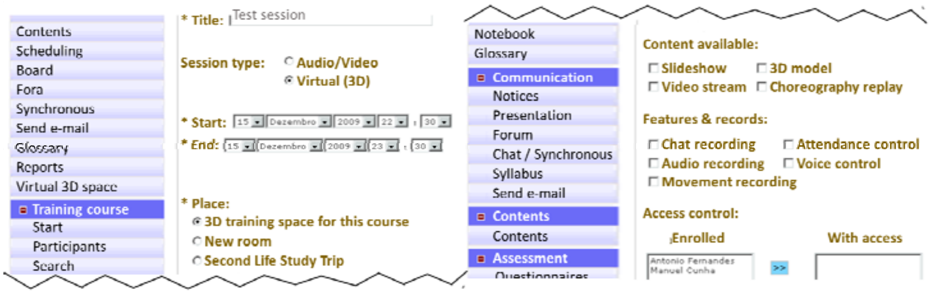



*Figure 5: The new Formare LMS Web form for synchronous session scheduling, with new data for the 3D session (image edited to present English language translation; original in the Portuguese language).*

Depending on the course administrator's choices in this form, several different tasks will have to be performed by the MULTIS components implemented in the Formare LMS. Since we are creating a new session, all of these follow the "Web user requests action" sequence from Table 3. For instance, if the "New room" location is selected, the LMS logic needs to check its list of available locations for creating rooms in its associated virtual world, select one, and issue a "room creation" task to the bot scheduler, with the appropriate location and room details information. Depending on the complexity of the room, the bot scheduler will then assign a task or a series of tasks to an available bot. Its associated bot logic will then login the bot (if necessary) and control it to perform the required task or tasks – in this case, creating the new room, with all required contents, in the specified virtual world location. Another example, also following this sequence, is found in the implementation of several of the features that the administrator can specify with the checkboxes. In our implementation for Second Life and OpenSimulator, most of these were implemented via code add-ons (scripts) contained inside room objects. We could have used for this purpose room objects such as the walls, floor, ceiling, or furniture. However, in order for the design be independent from the system features, we opted to use small objects with specific code add-ons inside, which we could simply create and embed in the floor, hiding them from view. This way, implementing a feature such as "access control" could be done in simple steps: first, a bot task is issued to create the appropriate object, containing the access control code; then another bot task is issued to position the object inside the floor, away from view; finally, another bot task provides the object with configuration data, by communicating with it via a private text chat channel. This data can be the actual required parameters (for instance, the list of avatars with access to the room) or an URL for the code add-on to use for retrieving the required data via HTTP request.

When trainer or trainee follow links or directions to the room of the synchronous session, provided via e-mail or other alert methods of the LMS logic, they find the room ready and setup as requested by the course administrator (Figure 6).


*Figure 6: A ready-to-use training room, created using the MULTIS architecture.*

### 7.3 R2 – Archival of interactive 3D virtual world content

3D content can be generated in the OpenSimulator and Second Life Grid platforms either by import of 3D meshes (which become part of the inventory of the avatar that imports them) or by modification and combination of pre-existing 3D templates, such as cubes, cones, and spheres, which in these platforms as called "prims", short for "primitives" (for more details, see Sequeira [Sequeira, 13]). Regardless, it's only after a 3D item is inside the virtual world that it can be further tuned, by specifying its physical properties, lighting, and other features. To include code add-ons (scripts) in 3D content, the source code for them first needs to be uploaded to the servers and registered as an independent asset (which renders the script part of the inventory of the avatar that uploaded it). Only then a user or bot specify for a 3D object to be associated to a script (as mentioned previously, via the metaphor of "containing" the script). It is via these code add-ons that 3D content in OpenSimulator or Second Life can exhibit behaviours such as switching slides following a trainer's touch on the projection screen, react to other objects via text messaging or proximity/collision detection, or even contact Web servers for detailed behaviour parameters. They can also communicate with a bot via messaging or their properties (including their mere existence), using the bot as a communication channel.

Under this context, 3D meshes and scripts can be readily uploaded by LMS system using the "Web user request action" sequence: the LMS logic request the upload task, the bot scheduler assigns it to a bot, and the bot logic does the actual uploading. The uploaded materials become part of the inventory of that specific bot, and the LMS logic keeps a record of each material and where, in terms of bot's avatar's inventory, those materials are stored. We have implemented a method by where some bots are exclusively devoted to act as stores of materials for use by other bots. Describing it is beyond the scope of this paper, but follows the MULTIS architecture.

However, due to the richness of information and configurations that can be associated with a 3D object within the virtual world proper, as described above, it is desirable that the LMS can receive objects from virtual world users. This allows the LMS to store not just 3D models, but also fully interactive objects or set of objects, with physical and lighting properties. We have implemented this via a scripted object, which we called a “content area”. Other virtual world users can deposit objects inside it, to be collected by the LMS using the MULTIS architecture. In Figure 7 we present an instance of its use: the user has opened his inventory pane, selected a virtual object and is dragging it onto the content area object, which became highlighted, indicating that it can accept items. The user can proceed and drop the object into it.

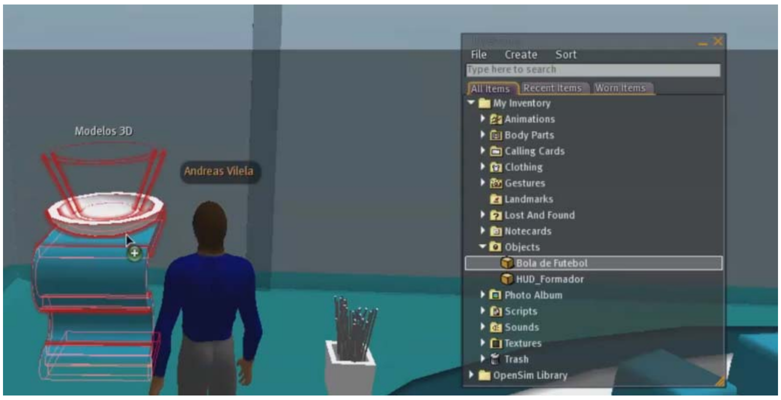



*Figure 7: Avatar of a user depositing content inside the virtual world.*

To make this object part of the LMS, when this drop occurs, the sequence “Virtual world user produces data” is used. The script in the content area issues an HTTP request to the LMS logic, reporting the event and associated data (object details, and avatar that dropped it, for instance). The LMS logic can respond by accepting or rejecting the object (in which case the content area script could delete it or return it to the original avatar). Then the LMS logic employs the “Web user requests data” sequence. It issues an “object collection” request to the bot scheduler, who selects a bot and issues to its bot logic component the series of tasks to collect it: log in (if necessary), travel to the content area locale, and collect the appropriate object from the content area object. Then it reports upstream, so that the LMS logic can record that the object exists and is stored in the inventory of the avatar associated with that bot.

Another alternative implementation – and one necessary in virtual world platforms which don’t support code add-ons for objects – would be to have a bot persistently in the content area, and the virtual world user could then deliver the object directly to the bot, thus doing the entire process without script requirements, but at the expense of maintaining a permanent use of some network resources between the LMS server and the virtual world server.

### 7.4 R3 – Use in the virtual world of 3D content archived in the LMS

This situation is resolved using the procedures presented above for R1 (which addressed the creation of objects inside the virtual world) and for R2 (which addressed the use of bots' inventories as stores for 3D objects and other virtual world assets). In the traditional Web interface, the course administrator can browse the available 3D objects, registered there as described for R2, and specify which are to be used and where. The only additional complexity is that the bot in whose inventory an object is stored may not be the bot which the bot scheduler wants to assign for its creation. In this situation, the bot scheduler needs to assign the tasks to both bots, so that content is transferred between them: the bot logic of the bot where the content is stored is assigned the task of sending that content to the second bot, and the bot logic of the second bot then is assigned the task of accepting that content. Subsequently, the tasks for creating it inside the virtual world can also be assigned to the second bot logic by the bot scheduler, as described above for R1.

### 7.5 R4 – Provision of 3D content to LMS users

Following the descriptions above for R1, R2, and R3, this case is trivial. The course administrator can use the traditional LMS Web interface to browse the available 3D content, registered as described above for R2. To send it to a user's avatar, it simply uses the "Web user requests action" sequence: the LMS logic issues the request "Transfer item to avatar" to the bot scheduler component, which then either assigns that task to the bot logic of the bot whose avatar's inventory holds the item, or transfers content between bots prior to that, as described above for R3.

## 8 Final Thoughts and Future Work

Since this architecture has been implemented in actual corporate systems, we are confident that it will prove feasible for large-scale deployment. Significant issues remain at that level, particularly from an information systems management perspective. Based on our own experience, the most challenging issue for wider deployment is that current virtual world client software for end-users is not geared towards corporate control and support of the trainee's experience. This places extra demands on user support structures, particularly when – as in Second Life or OpenSimulator viewers – the users have significant freedom to interact. For instance, a trainee can be provided with a custom on-screen control for a virtual machine, but current Second Life/OpenSimulator viewers allow him/her to freely remove that on-screen control from view – a complex support situation if the user doesn't know how to replace it and is amidst a synchronous training scenario, alongside other trainees and trainer. We have been developing strategies to monitor this and other situations, but ultimately this serves as an anecdotal note on the amount of evolution that these platforms will still see before widespread corporate adoption for training scenarios is viable. Other virtual world platforms have similar issues, derided from the same non-organizational focus of virtual world client software. Organizations and companies providing LMS software can develop their own versions of virtual world client software, but given that this field is still in flux, such a path will demand significant resources for software development and maintenance of another complex piece of software – which ultimately may need to be abandoned when the field embraces common protocols and features, much like the Web today [Morgado, 09].

On a wider perspective, the development of training services and content based on virtual worlds needs to separate concerns and greater independence between the training content and the virtual world platform where it is provided. The MULTIS architecture provides a step in that direction, by proposing to treat training contexts as integral part of an LMS, pointing towards virtual worlds' integration with the current panorama of information systems, rather than remaining isolated. In the future, we envision that this separation of concerns may enable storage and control of training activities as complex as multi-agent scenarios, independent of the virtual world platform of choice. Small steps are being taken in that direction (e.g., [Silva, 14]; [Fonseca, 11]), and require not just data standards but also software architectures that enable it.

**Acknowledgements**

This work was supported in part by research grants from Portugal Telecom Group, under its innovation plans for 2009 and 2010. The authors wish to thank doctoral student Ricardo Antunes, whose health issues have impeded him from taking part in this research, for the inspiration for linking LMS and virtual worlds.

## References

[Advanced Distributed Learning, 09] Advanced Distributed Learning, SCORM 2004 – 4th Edition, Documentation, Alexandria, VA, 2009, https://www.adlnet.gov/adl-research/scorm/scorm-2004-4th-edition/

[Al-Khanjari, 15] Al-Khanjari, Z., Al-Roshdi, Y., Developing virtual lab to support the Computer Science Education in Moodle, 2015 12th International Conference on Remote Engineering and Virtual Instrumentation (REV), 186-191, IEEE, 2015.

[AlticeLabs, n.d.] Altice Labs, Formare, http://www.formare.pt/

[Antunes, 08] Antunes, R., Morgado, L., Martins, P., Fonseca, B., Managing 3D Virtual Classrooms, Learning Technology, 10 (1), 3-5, 2008.

[Blackboard, 08] Blackboard Inc., Blackboard Announces Ball State University as Inaugural Recipient of Blackboard Greenhouse Grant for Virtual Worlds(TM), Press Release, Washington, DC, Feb. 19, 2008, http://lac.blackboard.com/news-and-events/press-releases/archive.aspx?releaseid=1109469

[Blackboard, n.d.] Blackboard Inc., Blackboard Learn | Learning Management System, http://www.blackboard.com/Platforms/Learn/Overview.aspx

[Bochicchio, 09] Bochicchio, M., Longo, A., Hands-On Remote Labs: Collaborative Web Laboratories as a Case Study for IT Engineering Classes, IEEE Trans. Learning Technologies, 2 (4), 320-330, Oct./Dec. 2009.

[Bochicchio, 10] Bochicchio, M., Longo, A., Extending LMS with Collaborative Remote Lab Features, 2010 IEEE 10th International Conference on Advanced Learning Technologies (ICALT), 310-314, 2010.

[Callaghan, 09] Callaghan, M., McCusker, K., Losada, J.L., Harkin, J., Wilson, S., Engineering Education Island: Teaching Engineering in Virtual Worlds, Italics, 8 (3), 2009.

[Cheslack-Postava, 12] Cheslack-Postava, E., Azim, T., Mistree, B., Horn, D.R., Terrace, J., Levis, P., Freedman, M., A Scalable Server for 3D Metaverses, Proceedings of the 2012 USENIX Annual Technical Conference, USENIX '12, 209-222, 2012.

[Cohen, 12] Cohen, D.C., Sevdalis, N., Patel, V., Taylor, D., Batrick, N., Darzi, A.W., Major Incident Preparation for Acute Hospitals: Current State-of-the-Art, Training Needs Analysis, and the Role of Novel Virtual Worlds Simulation Technologies, The Journal of Emergency Medicine, 43 (6), 1029-1037, 2012.

[Cornerstone, n.d.] Cornerstone, Learning Management System - LMS Software | Cornerstone, https://www.cornerstoneondemand.com/learning

[Cruz, 15], Cruz, A., Paredes, H., Fonseca, B., Martins, P., Morgado, L., Fitting Three Dimensional Virtual Worlds into CSCW, 2015 IEEE 19th International Conference on Computer Supported Cooperative Work in Design (CSCWD), 419-424, IEEE, 2015.

[de Freitas, 08] de Freitas, S., Serious virtual worlds: A scoping study. Bristol, UK: Joint Information Systems Committee, 2008.

[de Penning, 08] de Penning, L., Boot, E., Kappé, B., Integrating Training Simulations and e-Learning Systems: The SimSCORM platform, The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC) – 2008 (Learn. Train. Win!), 2008.

[Farley, 09] Farley, H., Reusable Learning Designs and Second Life: Issues and Strategies, Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 2009, 4047-4052, 2009.

[Ferreira, 05] Ferreira, J., Cardoso, A., A Moodle extension to book online labs, International Journal of Online Engineering, 1 (2), 2005.

[Fonseca, 11] Fonseca, B., Paredes, H., Rafael, J., Morgado, L., Martins, P., A Software Architecture for Collaborative Training in Virtual Worlds: F-16 airplane engine maintenance, Collaboration and Technology: 17th International Conference, CRIWG 2011, Paraty, Brazil, October 2-7, 2011. Proceedings, 102-109, 2011.

[Ghanbarzadeh, 14] Ghanbarzadeh, R., Ghapanchi, A. H., Blumenstein, M., Talaei-Khoei, A., A decade of research on the use of three-dimensional virtual worlds in health care: a systematic literature review. Journal of Medical Internet Research, 16 (2), e47, 2014.

[Haynes, 98] Haynes, C., Holmevik, J.R. (Eds.), High wired: on the design, use, and theory of educational MOOs. Ann Arbor, MI: University of Michigan Press, 1998.

[Hew, 10] Hew, K.F., Cheung, W.S., Use of three-dimensional (3-D) immersive virtual worlds in K-12 and higher education settings: A review of the research, British Journal of Educational Technology, 41 (1), 33-55, 2010.

[Hudson, 09] Hudson, K., deGast-Kennedy, K., Canadian Border Simulation at Loyalist College, Journal of Virtual Worlds Research, 2 (1), 2009.

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

[Jie, 13] Jie, F., Chen, Y., Li, F., Li, F., A universal simulation control service based on HLA, 2013 Fourth International Conference on Intelligent Control and Information Processing (ICICIP), 792-796, IEEE, 2013.

[Jiménez, 08] Jiménez, A., Rodríguez, J., Alfaro, Z., Architecture for Integration of Simulations with SCORM in E-learning Environments, Proceedings – The 8th IEEE International Conference on Advanced Learning Technologies, Santander, Spain, 1-5 July 2008, ICALT '08, 217-221, 2008.

[Kaplan, 11] Kaplan, J., Yankelovich, N., Open Wonderland: An Extensible Virtual World Architecture, IEEE Internet Computing, 15 (5), 38-45, 2011.

[Kemp, 06] Kemp, J., Livingstone, D., 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, 13-18, 2006.

[LibOpenMetaverse, n.d.] LibOpenMetaverse, libopenmetaverse - libomv - Developer Wiki, http://lib.openmetaverse.co/wiki/Main_Page

[Linden Research, 07] Linden Research, Linden Lab To Open Source Second Life Software, Press Release, San Francisco, CA, 8 Jan. 2007, http://www.lindenlab.com/releases/linden-lab-to-open-source-second-life-software

[Linden Research, 08] Linden Research, SLGOGP Draft 1, Technical Report, San Francisco, CA, Mar. 2008, http://wiki.secondlife.com/wiki/SLGOGP_Draft_1

[Linden Research, n.d.] Linden Research, LSL Portal,
http://wiki.secondlife.com/wiki/LSL_Portal

[Livingstone, 09] Livingstone, D. (ed.), Online Learning In Virtual Environments with SLOODLE, Computing and Information Systems Technical Reports, no. 50 (Dec. 2009), Paisley, Scotland, UK: University of the West of Scotland,
https://www.sloodle.org/downloads/SLOODLE_Eduserv_report_final.pdf

[Madeira, 10] Madeira, A., Sequeira, P., Morgado, L., Gonzaga, L., Controlo da Assiduidade em Aulas Efectuadas no Second Life, RISTI - Revista Ibérica de Sistemas e Tecnologias de Informação, 5, 87-100, 2010.

[Morgado, 09] Morgado, L., Interconnecting virtual worlds, Journal of Virtual Worlds Research, 3 (1), 4-7, 2009.

[Morgado, 10] Morgado, L., Varajão, J., Coelho, D., Rodrigues, C., Sancin, C., Castello, V., The Attributes and Advantages of Virtual Worlds for Real World Training, The Journal of Virtual Worlds and Education, 1 (1), 2010.

[Morgado, 15] Morgado, L., Manjón, B. F., Gütl, C., Guest Editorial: Overcoming the Technological Hurdles Facing Virtual Worlds in Education: The Road to Widespread Deployment, Educational Technology & Society, 18 (1), 1–2, 2015.

[Motteram, 14] Motteram, G., Koenraad, T., Outakoski, H., Jauregi, K., Molka-Danielsen, J., Schneider, C., The Euroversity Good Practice Framework (EGPF) and its application to minority languages and elder learners, CALL Design: Principles and Practice, Proceedings of the 2014 EUROCALL Conference, Groningen, The Netherlands, Research-publishing.net, Dublin, 241-247, 2014.

[Oracle, n.d.] Oracle, Welcome to Oracle iLearning, http://ilearning.oracle.com/ilearn/en/learner/jsp/login.jsp

[Pérez-Garcia, 09] Pérez Garcia, M., MUVEnation: a European peer-to-peer learning programme for teacher training in the use of MUVEs in education, British Journal of Educational Technology, 40 (3), 561-567, 2009.

[Rapuano, 06] Rapuano, S., Zoino, F., A Learning Management System Including Laboratory Experiments on Measurement Instrumentation, IEEE Trans. Instrumentation and Measurement, 55 (5), 1757-1766, 2006.

[Ribeiro, 11] Ribeiro, T., Faria, N., Simões, P., Morgado, L., Rodrigues, P., Leite, L., Integrating a military air traffic control simulator with an LMS, Proceedings - 2011 Third International Conference on Games and Virtual Worlds for Serious Applications - VS-GAMES 2011, 132-135, 2011.

[Richter, 10] Richter, T., Boehringer, D., Jeschke, S., LiLa: A European Project on Networked Experiments, Automation, Communication and Cybernetics in Science and Engineering 2009/2010, Berlin, Germany: Springer, 307-317, 2011.

[Rodrigues, 09] Rodrigues, C., Coelho, D., Morgado, L., Varajão, J., Haidimoschi, A., Doppler, G., Koivusalo, H., Jokinen, P., Velegrakis, G., Sancin, C., Carmenini, V., Virtual Learning for the management of successful SMEs in Europe, Proceedings of the SLACTIONS 2009 International Conference - Life, imagination, and word using metaverse platforms, 165-170, 2009.

[Saba, n.d.] Saba, LMS, Talent Management, and Web Conferencing - Saba, http://www.saba.com/

[Sequeira, 13] Sequeira, L., Morgado, L., Mechanisms of three-dimensional content transfer between the OpenSimulator and the Second Life Grid® platforms, Journal of Gaming & Virtual Worlds, 5 (1), 41-57, 2013.

[Silva, 14] Silva, E., Silva, N., Morgado, L., 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, 77 – 91, Cham, Switzerland, Springer International Publishing, 2014.

[Symington, 00] Symington, S., Morse, K.L., Petty, M., IEEE Standard for Modelling and Simulation (M&S) High Level Architecture (HLA) - Framework and Rules, IEEE Std. 1516-2000, New York, USA: IEEE SISC, 2000.

[Texas State Technical College, n.d.] Texas State Technical College, vushiNation, https://web.archive.org/web/20120629125559/http://www.vushi.org/

[Virtual Worlds Group, n.d.] Virtual Worlds Group, Duke University, About - Open Cobalt Website, http://www.opencobalt.net/about

[Werner, n.d.] Werner, G., 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