# EVALUATION OF OPENSIMULATOR EXTENSIBILITY BY DESIGNING COLLABORATIVE AND ADAPTIVE 3D LEARNING OBJECT

Livia ŞTEFAN[1], Florica MOLDOVEANU[2], Alin MOLDOVEANU[3]

*This paper reports research results from design and implementation of several collaborative and adaptive learning objects within a 3D experimental educational simulator on OpenSimulator platform. Our methodology was to identify and apply different extensibility mechanisms of OpenSimulator that can support a flexible design and an extensible model required by an open and adaptive educational simulator. The learning objects were experimented with a small user base using different collaborative scenarios and will be further integrated into the "3DUPB" virtual campus in a Massive MultiUser Online context. The paper also discusses findings and presents conclusions and perspectives of the research.*

**Keywords**: 3D learning environments, 3D learning objects, collaborative learning, adaptive learning environments, virtual campus, OpenSimulator

## 1. Introduction

An online 3D virtual campus is a complex 3D simulator which features architectural components (e.g. buildings, rooms, facilities) and services which implement functionalities similar to those from the real life counter-part. It can be ubiquitously accessed by any user having an internet connection, via a specialized client application (i.e. viewer). The user registers with an account name and password and enters the virtual world represented by its avatar.

Several universities provide 3D presence in Second Life or OpenSimulator [2], [3], [4], as research or educational platforms.

The "3DUPB" [5] is an ongoing research project of the Computer Graphics and Virtual Reality group from the Department of Computer Science and Engineering, Faculty of Automatic Control and Computers, University Politehnica of Bucharest (UPB), which has the purpose to implement a virtual

[1] University POLITEHNICA of Bucharest; Institute for Computers ITC Bucharest, e-mail: livia.stefan@itc.ro
[2] Faculty of Computer Science, University POLITEHNICA of Bucharest, e-mail: florica.moldoveanu@cs.pub.ro
[3] Faculty of Computer Science, University POLITEHNICA of Bucharest, e-mail: alin.moldoveanu@cs.pub.ro

clone of the UPB campus, as a Massive Multiplayers Online (MMO) service-based platform on OpenSimulator (OS). Because it is intended to support distance learning with a full range of university activities and inter-university communication, i.e. to provide a large scale virtual reality platform, several challenges are put in front of this project [6][7][8].

The "3DUPB" virtual campus will complement and enhance the face-to-face teaching and learning, i.e. support a blended-learning paradigm based on modern technologies and innovative solutions [9]. This is the reason why a flexible approach is necessary for the accomplishment of the technical challenges (add, adapt and connect with external services, mixing the virtual medium with real life information) and the educational requirements (collaborative teaching and learning, communication and social interaction, assessment) [10].

By taking advantage of the rich graphical environment and of the programmatic capabilities, both the 3D components and the functionalities can be extended to create a flexible environment, able to dynamically accommodate as much as possible teachers' and students' educational and administrative needs, which in real life are difficult to fulfill or unsupported.

A research topic for 3D virtual environments is to provide instruments for data collection and evaluate indicators concerning the quality of the environment, e.g. number of visits, in-world retention time, learning outcome. The data can be further employed for provision of adaptive services, as a response to learners' needs and as a method to increase users' retainment in the virtual environment [11].

The present research presents evaluation results of OpenSimulator's capabilities that support concepts of extensibility, flexibility and adaptivity of 3D learning environments. For the purposes of this research a 3D educational simulator (ES) named "3DCampSim" [12] comprising several functional, collaborative and adaptive learning spaces and objects (LOBJs) was implemented and experimented.

The paper presents a background of the research work, a summary of the extensibility mechanisms of the OS, the 3DCampSim's design objectives, functional and implementation details of the LOBJs, an educational scenario and a discussion of the research results, conclusions and perspectives of the research.

## 2. Background of the research work

Linden Lab Second Life (SL), There or Active Worlds are commercial, closed and proprietary metaverse engines or 3D multi-user virtual environments (3DMUVEs). The OpenSource Metaverse Project [13], The Open Cobalt [14], OpenSimulator (OS) [15], OpenWonderland [16] or Virtual World Framework

[17] are open-source metaverse frameworks. OpenWonderland is an open source Java toolkit for creating collaborative 3D virtual worlds.

The main purpose of 3DMUVEs is to provide a social and constructivist platform [18][19], but recent implementations are seeking ways to integrate these platforms into a larger infrastructure by means of service mashups. A relevant example is the proprietary mashup solution, PHP-based, between SL or OS and the Moodle Learning Management System (LMS), i.e. SLOODLE (Simulation Linked Object Oriented Dynamic Learning Environment) [20][21]. SLOODLE consists of a set of 3D specialized objects synchronized bidirectionally with different Moodle components (e.g. quiz module, grade-book).

The majority of the current 3D educational simulators lacks openness and adaptivity to users' requirements, and do not provide methods for objective measuring of the effectiveness of such environments for enhancing teaching and learning. In the present research such issues are addressed by leveraging the capabilities of the OS platform, an open-source 3D application server built upon SL concepts and protocol. In response to SL's proprietary server code and architecture, OS core developers allow users to control both the server and the client software, i.e. to contribute to OS or to adapt it to create their own virtual world implementation.

## 3. Evaluation of the openness and extensibility of OpenSimulator

From a programmatic point of view, OS is an extensible server-side .NET C# framework. Currently supports a hyper-grid architecture, an extension developed by one of its core contributors [22], which allow the networking of 3D virtual worlds and movement of avatars, among different OS-based virtual worlds, an important step forward to a seamless 3D online metaverse [13].

Two remarkable distributions of OS demonstrate its flexible and extensible architecture: Diva Distro [22], preconfigured and hypergrid-enabled, with a web front-end, and Sim-on-a-stick [23], a slimmed down portable distribution, runnable from a USB stick. OSGrid [24] is a public free OS host grid and also a preconfigured distribution.

The current version (0.8.1), a beta-version, implements almost all of the LSL scripting functions, and also its own scripting language (OSSL) to fill the gap of missing functionality in LSL (e.g. writing to notecards or programmatically teleporting an avatar).

An OS virtual world functional unit is the *Region*, which is "a memory space and behavior simulator which can share its state with observers… a 3-dimensional MUD" [25]. It corresponds to an island of 256x256 square meters. A Region contains a 3D Scene object by means of which it is possible to programmatically keep track of objects and avatars in the scene, and also to dynamically update the scene with new objects. Furthermore, the Region's objects can be manipulated and interfaced with external systems [25].

OS Region Modules (RM) and OS API are two main mechanisms, unique to OS, to provide a "powerful alternative to in-world scripting" [25]. The OS module mechanism is currently supported by the Mono.Addins [26] framework and on the .NET reflective software.

The RM are Dynamic Link Libraries (DLL on Windows platform, .dylib on MAC, .so on Linux), loaded when the server starts-up. These modules contain several methods which are registered with different region's events (entering the region, login of an avatar, chat) [25],[27]. The RM implement the IRegionModule interface and can be shared across all the regions or non-shared, for each case the initialization procedures differ.

The newer versions of OS provide two additional mechanisms related to RM [28]: one is data transmissions between a region module and the scripted world by means of the OSSL script function *modSendCommand()*, and the other is the invocation of functions defined in a RM from a script, using the *modInvoke()* family of functions.

The first case is a method to extend a scripting function: a region module listens for the data sent by *modSendCommand()* by a scripted function, and sends a reply received by the script in the *link_message* event. The second method is a later extension of this functionality which simplifies the process, by allowing the region modules to implement different functions to be called by scripts at run time, provided that the method takes the name of the function.

The OpenMetaverse [13] is an open-source alpha version collection of C#.NET functions for interactions with 3D virtual simulators, employed for development of OS clients and client-side bots.

RESTful interfaces are implemented in OS as a plugin to support interactions with external applications using the REST architecture [29], which abstracts data to resources, and XML for REST requests. REST implementation in OS is insufficiently exploited and checked, according to [10]. An Application Programming Interface (API) that implements the REST protocol is provided for access to the OS resources (Asset and Inventory).

The extensibility mechanisms of OS are summarized in Table 1.

*Table 1*

**Extensibility in Opensimulator**

| Description | Mechanism | Requirements | Affected level of functionality |
|---|---|---|---|
| OS API | Extensibility by using the framework | Recompilation of the OS project | Simulator |
| Configuration files / sections | Aggregation or override of sections and files | | Simulator/ Region |
| New OSSL functions | Add-ons | - | OSSL Scripting |
| Application plugins | Implementation of the | - | Simulator |

| | | | |
|---|---|---|---|
| | IApplicationPlugin interface | | |
| Region Modules | Implementation of the INonSharedRegionModule or ISharedRegionModule interface. Registration with region's events | Enabled/disabled in the OS configuration file or in the region module configuration file | Region |
| DataSnapshot [30] | IRegionModule exports data from various parts of the OpenSimulator Scene in a custom-format XML | - | Region for web services mashup |
| modSendCommand()/modInvoke() | Implementation of the INonSharedRegionModule or ISharedRegionModule interface. | Enabled in the OS configuration file | Region |
| HTTP protocol | Data requests in the llHTTPRequest() or llRequestURL() function; data received in the http_response event. | - | Scripted objects. Connection to web pages, web services or data bases |
| XML-RPC | Data requests initiated in other servers | - | Scripted objects. |
| RESTful interfaces | REST handlers for assets, avatars, appearance and inventory | Enabled in the OS configuration file | Access the OS resources from other applications |
| JSonStore module | Mechanism for management of structured data in an LSL script | Enabled in the OS configuration file | Scripted objects. |
| MySQL database, partially by MSSQL | External persistence storage for content other than the 3D objects. Direct database manipulation. | - | Simulator. |
| Various community addon modules [31] | Addon/plugin mechanism | - | Simulator. |
| Various community web interfaces [32][33] | Web interface | - | Simulator. |
| OpenMetaverse | Open-source library | - | Client-side functionalities and bots. |

## 4. The design objectives

For the present research a standalone hyper-grid enabled OS configuration was used to implement the 3DCampSim, consisting in two buildings, several rooms (Fig.1) and specialized 3D collaborative and adaptive LOBJs. The simulator's design also includes spatial orientation indicators and visual clues, heads-up displays (HUD) and a teleport hub.

The main objective was to experiment OS mechanisms for connecting the simulator with external services and to dynamically integrate new regions and/or new functionalities into an existing simulator.

Specific objectives were: a) to implement the behaviour of the LE to support in-world navigation and engagement, both of students and teachers; b) to implement the behavior of the LOBJs, to support collaborative work, as well as to create groups of users, ownership and rights on the LOBJs; c) to perform abstraction of several functionalities (prototyping from a functional point of view) of the LE and LOBJ, e.g. management of users, activities and events, customization, re-purpose and extensibility; d) to trace activities and events, and correlate them with activities in the university's Moodle LMS; e) to collect usage data in an external database (including in the cloud) and to further implement an adaptive behaviour and deliver of the educational content [34].

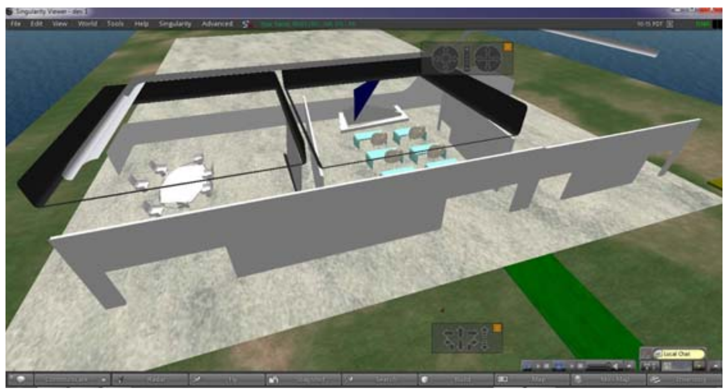

Fig. 1. The collaborative 3D learning environment ("3DCampSim")

### 5. The collaborative learning objects

The collaborative LOBJs (Fig.2) are based on the well-known 3D in-world affordances: media panels, HUDs, notecards, dialog-based menus. They were initially designed bearing in mind the common teaching model for engineering classes, which may consist in a theoretical presentation of the main topic, a practical demonstration, group discussion, project assignment. The design has been further enriched with functionalities that can be mediated by the 3D simulator, such as individual or group work, question and answer sessions, project demonstration. The main LOBJs are described in the Table 2.

*Table 2*

**The collaborative learning objects**

| Object | Functionality | Adaptive functionality/ Collaborative work |
|---|---|---|
| COURSE_PANEL | A notecard customized with the acronym of the course, activity, semester and year | Can be read by student and visitors. Correlated with LMS. |

| | | |
|---|---|---|
| MEDIA_PANEL | Delivery of course materials (PowerPoints, documents, videos, web pages). Load. Backward and forward navigation. | Configurable content. |
| WHITEBOARD | Interactive writing or drawing, e.g. explanations, launch of projects, recommend resources for students. | First line describes the current activity, from the COURSE_PANEL.Comments from notecards. |
| NOTECARD_QA | Question and Answers. | Configurable content. Takes students' questions; teachers answer on a dedicated communication channel. |
| QUIZ_PANEL | Quiz giver. The student receives a feedback; the teachers receive the score. | Configurable from notecards. Stores the scores in the database. |
| EXHIBIT_PANEL | Media panel for exhibition of students projects. | Configurable with links to a website or images |
| LEADERBOARD | Gamification panel; displays students having a score greater than the minimal one. | Retrieves information from the database. |
| MEDIA_STORE | Access to educational resources. | Configurable. Group owned. Gamified according to the students' level retrieved from the database. |
| ANNOUNCEMENTS | Displays real-time information feeds. | Information from external services (e.g. RSS, RESTful webservice). |
| PRESENCE_COUNTER | Displays the number of participants in an ongoing course. | Stores information in the database. |
| LANDMARKS | Sense visitors in a well-defined area of the simulator; teleport destination. | Stores information in the data base. |
| ROLE_INDICATOR | Attachable object indicating the role of the avatar by means of a colored ball. | Retrieves information from the database. |

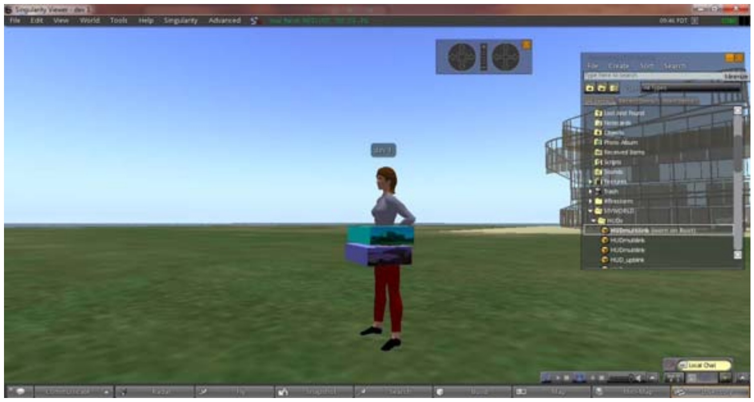

Fig. 2. Learning object for individual access to web sites

## 6. Implementation

The basic behaviour of the majority of the learning objects was implemented with scripting functions, which can describe complex actions, following a software model of an event-based finite-state machine (FSM). The learning objects are composed of several linked primitives which communicate with each others or with other objects.

The RM were created as listeners for MOD functions, which implemented the tasks of database connection and persistent storage of data concernig the learning activities and the usage of the environment. The notecards are used for object's configuration. The dialogues are a more efficient communication tool, especially for multiple choice menus.

**Case study – the collaborative tracking objects:**

A non-shared RM and two methods, *course_presence()* and *landmark_visits()*, were created for **tracking** the environment usage, and registered on the timer event. The collaborative object LANDMARK calls a MOD function from this module for *writing in the database*, having as parameter an acronym of the monitored area or room, taken from a global variable in a script attached to the object. The collaborative object PRESENCE_COUNTER calls a MOD function from this module for *writing in the database*, having the parameters: the avatar UUID and the acronym of the activity, taken from the COURSE_PANEL object, configurable by teachers.

A non-shared RM and a method, *student_quiz(),* were created for **educational** functionalities and registered on the control event (related to mouse and keyboard events). The collaborative object PANEL_QUIZ gives every student a quiz under the form of a notecard which calls a MOD function for *writing in the database*. The parameters are the avatar UUID, the acronym of the activity, of the quiz and the final result (true/false).

A code snippet for both sides (i.e. scripted object and RM) is given below.

(1)

```
//object script
string activity=”Web design”;
default //default state
{
  timer() //the event handler
    {
    modSendCommand("course_presence", activity + “|” + string (llGetKey()));
//module name and strided list of parameters
  }
  link_message(integer sender_num, integer num, string message, key id)
 {
            if (sender_num == -1)
                    llSay (DEBUG_CHANNEL, message); //the response message is
           said on the Debug channel.
   }
}
```

(2)

```
//Region module
//In PostInitialise a callback function is called which treats the script command
public void Initialise(Scene scene, IConfigSource source)
    {
      m_scene = scene;
    }
public void PostInitialise()
   {
     m_commsMod
       = m_scene.RequestModuleInterface<IScriptModuleComms>();
     m_commsMod.OnScriptCommand += course_presence();. //the handler
    }
```

## 7. Experimentation using collaborative scenarios

In the first stage of the research, the simulator was tested in a local environment and with a local remote connection. In a second stage, the simulator will be experimented remotely and as an extension of the “3DUPB” campus, which will be exploited as a testbed for conducting experimentations and case studies with a larger user base (MMO context).

In the first case it was difficult to simulate a multi-user context. Therefore tests were performed using multiple user accounts created with an OS utility pCamBot [35] which can be found in the bin directory of the OS distribution.

A collaborative scenario might be: a) the master teacher (i.e. the one teaching the class) logins; b) the master teacher introduces the acronym of the current activity and course, year of study, semester in the COURSE_PANEL. The WHITEBOARD is initialized accordingly, displaying information from the COURSE_PANEL; c) the course begins, open discussions may take place on a communication channel; d) students address questions using a copy of the NOTECARD_QA; e) teacher reads them and replies on a communication channel; f) the teacher initializes a simple quiz on the quiz_PANEL to assess the attention and information retention during the class; f) students respond to the notecards and the score is saved in the database and correlated with the LMS; g) the teacher indicates an acronym for a course assignement (taken from the LMS), indicating tasks and due date. The results of the project (eg. captures, links to Power Point or files) may be loaded next time on the EXHIBIT_PANELS, indicating the acronym of the assignement.The scores of the assignments are employed by 3DCampSim's gamification engine for calculation of students' gamification score [10]

## 8. Discussion and findings

There are several mechanisms to create new OS functionalities, which are insufficiently exploited. The scripting functions are powerful and versatile, covering almost any desired functionality, including communication with the databases and web services, and could also be easily debugged. Instead they required a learning curve to understand the specificity of interactions in a 3D virtual world (e.g. events) and to program the atomic functionalities. The add-ons could isolate the variable functionalities which are to be later modified without interrupting the 3D simulator, but they proved to be more difficult to debug. The core distribution of OS contains several source code of region modules, as a a starting point and technical support.

The RM are useful for region extension, integration with external specialized libraries, or for dynamic content creation (rezzing new objects). The load of the RM required a restart of the server.

The management of the content, user groups and rights were performed using the Inventory, and it proved to be a challenging task in a distributed and multi-user environment.

## 9. Conclusions and further work

In this paper we evaluated different levels of development and extensibility of customized OS virtual worlds: LSL and OSSL scripting functions; Region modules; MOD functions; in-world and remote communication capabilities.

Several learning objects were implemented for collaborative work, dynamic information display or tracking the usage of the environment. A detailed analysis and workflow was done to determine the functionalities that can be provided by scripting and the ones suitable for add-ons. Region modules were tested to assess the difficulty of implementation and practical usage of connecting the virtual world with external services/applications. In a future research, the RM will be used for dynamic modification of certain objects according to data collected during the usage of the environment.

**Acknowledgement**

The work of the first author is supported by the Sectoral Operational Programme Human Resources Development (SOP HRD), financed from the European Social Fund and the Romanian Government under the contract number POSDRU/159/1.5/S/137390/.

## R E F E R E N C E S

[1] *M. Savin-Baden*, A Practical Guide to Using SECOND LIFE in Higher Education, Ed. Mc. Graw Hill, 2010.
[2] *University of Cincinnati virtual campus,* http://ucsim.uc.edu/blog/services/virtual-environments-development/opensim, Accessed May 2015.
[3] *University of Arts Zurich New Genres Grid,* http://opensim-edu.org/blog/2012/01/zurich-university-of-the-arts/, Accessed May 2015.
[4] *Virtual University of Edinburgh grid,* http://www.hyperica.com/2012/01/openvue-2/, Accessed May 2015.
[5] *3DUPB Project web site*, http://3d.pub.ro, Accessed May 2015.
[6] *F. Pop, M. I. Andreica, C. M. Dobre, R. Rughinis, A. Moldoveanu, C.A. Boiangiu and M. Ionescu*, "eUPB: Towards an Integrated e-Service Platform in Large Scale Distributed Environments", Conference: Proceedings CSCS-18, The 18-th International Conference on Control Systems and Computer Science, Volume: 2.
[7] *A. Moldoveanu, A. Morar and V. Asavei,* "3DUPB - The Mixed Reality Campus: A glimpse at how mixed reality systems can shape the future", Revista Română de Interacțiune Om-Calculator 6 (1), 35-56, 2013.
[8] *A. Moldoveanu, A. Gradinaru, O. M. Ferche and L. Stefan,* "The 3D UPB Mixed Reality Campus Challenges of mixing the real and the virtual", in Proceedings of the 18th International Conference on System Theory, Control and Computing, Sinaia, 2014, pp. pp. 544-549.

[9] *L. Stefan, F. Moldoveanu and A. Moldoveanu*, "Blended Learning in a Mixed Reality-based 3D Multi-User Virtual Environment", Proceedings of the 10th International Scientific Conference "eLearning and Software for Education" Bucharest, April 24 - 25, 2014, Ed. Universitatii Nationale de Aparare "Carol I", vol. I, pp. 105-112.

[10] *C. Gutl, V. Chang and S. Freudenthaler, "*How to Support More Flexible Learning Settings in Second Life", in Auer, M.E. and Schreurs, J. (ed), 13th International Conference on Interactive Computer Aided Learning, Sep 15 2010, pp. 129-141. Hasselt, Belgium: Kassel University Press.

[11] *L. Ştefan and F. Moldoveanu,* "Gamified 3D Virtual Learning Environment For Improved Students' Motivation And Learning Evaluation. A Case Study On 3DUPB Campus", in Proceeding of the 11th International Scientific Conference eLearning and software for Education, Bucharest, April 23-25, 2015, Ed. Universitatii Nationale de Aparare "Carol I", vol. I, pp. 94-101.

[12] *"3DCampSim" Google Plus blog*, http://tinyurl.com/p9ec5fu, Accessed May 2015.

[13] *J.M. Smart, J. Cascio and J. Paffendorf*, "Metaverse Roadmap Overview, 2007", in Accelerated Studies Foundation. Accessed May 2015.

[14] *Open Cobalt website,* http://www.opencobalt.org/, Accessed May 2015.

[15] *OpenSimulator website,* http://opensimulator.org, Accessed May 2015.

[16] *Open Wonderland website,* http://openwonderland.org/, Accessed May 2015.

[17] *Virtual World Framework website,* https://virtual.wf/demos.html, Accessed May 2015.

[18] *Leman F. Gül, Ning Gu, Anthony Williams*,"Virtual worlds as a constructivist learning platform: evaluations of 3D virtual worlds on design teaching and learning", 2008, Accessed July 2014 www.itcon.org

[19] *E. Shudayfat, F. Moldoveanu, A. Moldoveanu*, "A 3D Virtual Learning Environment for Teaching Chemistry in High School", Annals of DAAAM for 2012 & Proceedings of the 23rd International DAAAM Symposium, Volume 23, No.1, pp. 423-428.

[20] *Sloodle website*, https://www.sloodle.org/, Accessed May 2015.

[21] *K. Andreas, T. Thrasyvoulos, D. Stavros and P. Andreas,* "Collaborative Learning in OpenSim by Utilizing Sloodle", in Em: 6th Advanced International Conference on Telecommunications (AICT). pp. 9-15, 2010.

[22] *Diva Distribution* http://metaverseink.com/Downloads.html, Accessed May 2015.

[23] *Simonastick distribution,* http://simonastick.com/, Accessed May 2015.

[24] *OSGrid Distribution* http://www.osgrid.org/index.php/downloads, Accessed May 2015.

[25] http://opensimulator.org/wiki/RegionModules, Accessed May 2015.

[26] *MonoDevelop* http://www.monodevelop.com/, Accessed May 2015.

[27] *A. Henckel and C. V. Lopes, "*Stellarsim: A plug-in architecture for scientific visualizations in virtual worlds". In O. Akan et al. (Eds.), Facets of virtual environments (Vol. 33, p. 106-120). Springer Berlin Heidelberg, 2010. Accessed January 2015.

[28] http://opensimulator.org/wiki/OSSL_Script_Library/ModInvoke, Accessed May 2015

[29] http://opensimulator.org/wiki/REST, Accessed May 2015.

[30] http://opensimulator.org/wiki/OpenSim.Region.DataSnapshot, Accessed May 2015.

[31] http://opensimulator.org/wiki/Related_Software, Accessed May 2015.

[32] http://opensimulator.org/wiki/Webinterface, Accessed May 2015.

[33] http://opensimulator.org/wiki/Wifi, Accessed May 2015.

[34] *Horizon Report*, 2015 *Higher Education Edition,* http://www.nmc.org/nmc-horizon/, Accessed March 2015.

[35] http://opensimulator.org/wiki/PCampBot, Accessed May 2015.