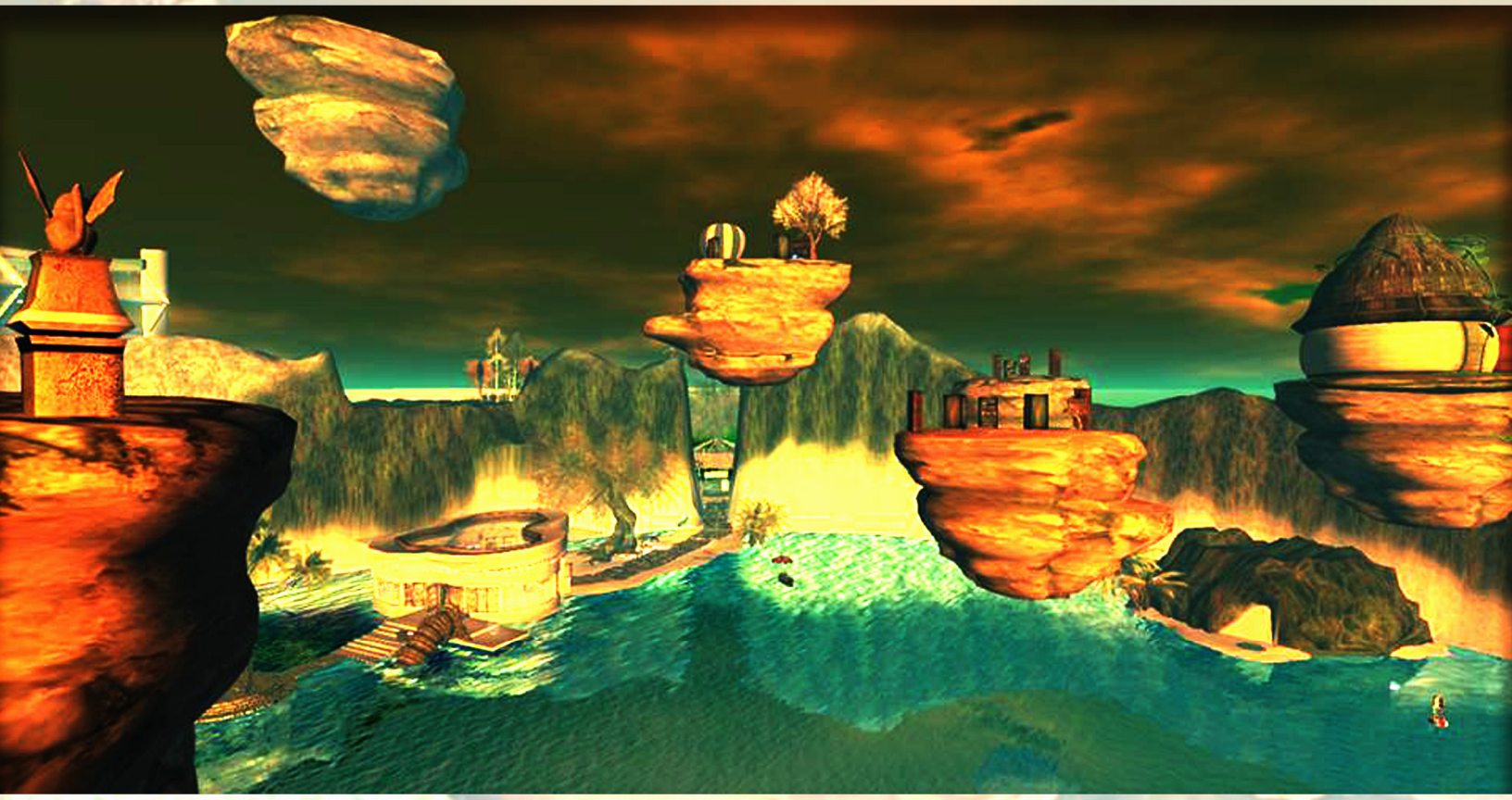
Journal of
Virtual Worlds Research
jvwresearch.org ISSN: 1941-8477

Assembled 2015

February 2015
Volume 8 No. 1



From: Alexandria - A Virtual Repository of Knowledge

Volume 8, Number 1
Assembled 2015
February 2015

Journal of Virtual Worlds Research
jvwresearch.org ISSN: 1941-8477

Volume 8, Number 1
Assembled 2015
February, 2015

Alexandria - A Virtual Repository of Knowledge

Joseph M. Graham
Independent Researcher

Eva Comaroski
Wizardry and Steamworks Consulting Group

Abstract

Virtual worlds are development platforms, similar to sandbox games where users are able to interact with each other with no predefined goal. Due to immersive visual feedback, these platforms have been successfully used in the past to create lesson plans and materials used for undergraduate and graduate education as well as recreation. To complement that effort, we have built a virtual library that would conveniently offer reference materials to support learning activities. We present Alexandria, a Linden-based virtual library that uses synchronization techniques based on the OpenSim Scripting Language and server-side file-synchronization software in order to allow the stashing and dissemination of educational and recreational material within Virtual Worlds.

1. Introduction

Stashing material in the real world is expensive due to the general high costs of space and the afferent maintenance that is required in order to preserve the material. Every library suffers from books being damaged or lost eventually. With the advances of technology, we have seen major benefits from stashing content digitally. Foremost, there are many cost-effective techniques (Lee, Slattery, Lu, Tang, & McCrary, 2002) that allow the sound preservation of material, ranging from the ability to back-up the content to the fact that the overall lifetime of digital medium outstretches by far the lifespan of paperback material (Ranstam, 2008).

Classical studies (Ladd & Mize, 1983) have shown that learning, whilst being a personal form of development, also may convey the "social experience" that grants students the ability to interact with their peers and be able to build connections that will further benefit their development. To that end, the digital medium provides an excellent technical way to preserve, distribute and annex material; however, digital medium falls short when it comes to providing social interactions between and among students.

In order to address these shortcomings, we have grafted the two worlds together - the factual dissemination of learning material along with the social experience of a real world library where students are able to meet, interact and even contribute to the library. The result is a virtual world library that contains learning material for students, as well as providing a game-like multi-player experience that allows students to meet virtually in order to disseminate data together. Taking one step further and benefitting from the development of media delivery, we have additionally added recreational material to supplement the library.

Alexandria is built using the OpenSim (Overte, 2011) software that is a virtual world platform similar to Linden Lab's Second Life (SL). The OpenSim platform allows the creation of a game-like sandbox, to which multiple clients can connect via the Internet; they can then interact with each other or with the virtual world. Contrasting with other virtual worlds, building platforms, such as Unity3D (Wang et al., 2010), OpenSim allows the building of content directly within the world without requiring additional tools (Messinger, Stroulia, & Lyons, 2008). This is performed by uploading assets such as textures, animations and sounds and then applying the assets to in-world objects or to clothes. Internally, OpenSim pre-generates a few defaults, such as the sky, or the default ground - which can then be changed, as well as fully implementing the basic protocols for multiple clients to connect and interact. While building Alexandria, these defaults have allowed us to focus on the library, rather than re-implementing stereotypical game-related mechanisms such as developing a network layer or designing elementary graphical elements that would be necessary for the virtual world.

2. Overview

The motivation behind creating a virtual library such as Alexandria is explained in the Motivation section. In the "Technical Background" the underlying technical mechanisms for data synchronization and in-world display of literature are explained. An overview of the library is provided with a current outlook of the library. Included is a table that sums up literature gathered so far. The article concludes with a discussion that explains why the authors believe that virtual world libraries could potentially be a future asset that could be comparable to real-world libraries.

3. Motivation

An old proverb advises to "not judge a book by its cover;" historically, that proverb relies on the lack of associative or contextual information that book covers hold in relation to their contents (Medin & Schaffer, 1978). Even in modern, real-world, large libraries spanning hundreds of thousands of books, when people reach a bookshelf, they are presented with a wide palette of book-choices arranged by author. Although they may have been following classifications such as those already provided by the Dewey system (Dewey, 1965) and still widely-used today, it is still uncertain for an individual which book they should pick. In practice, without being able to read the whole book, the selection process is performed by associating the author names, or by looking over the cover, perhaps an illustration or a short biography of the author on the back - all of these selection criteria being quite tertiary to the literature held between the covers. In the end, due to the lack of associative information, libraries tend to be consulted only when an individual knows precisely what they are looking for. For example, the Dewey decimal classification (DDC) is an exhaustive classification (Wiegand, 1998) that is apt to classify any type of literature and works very well for finding something specific.

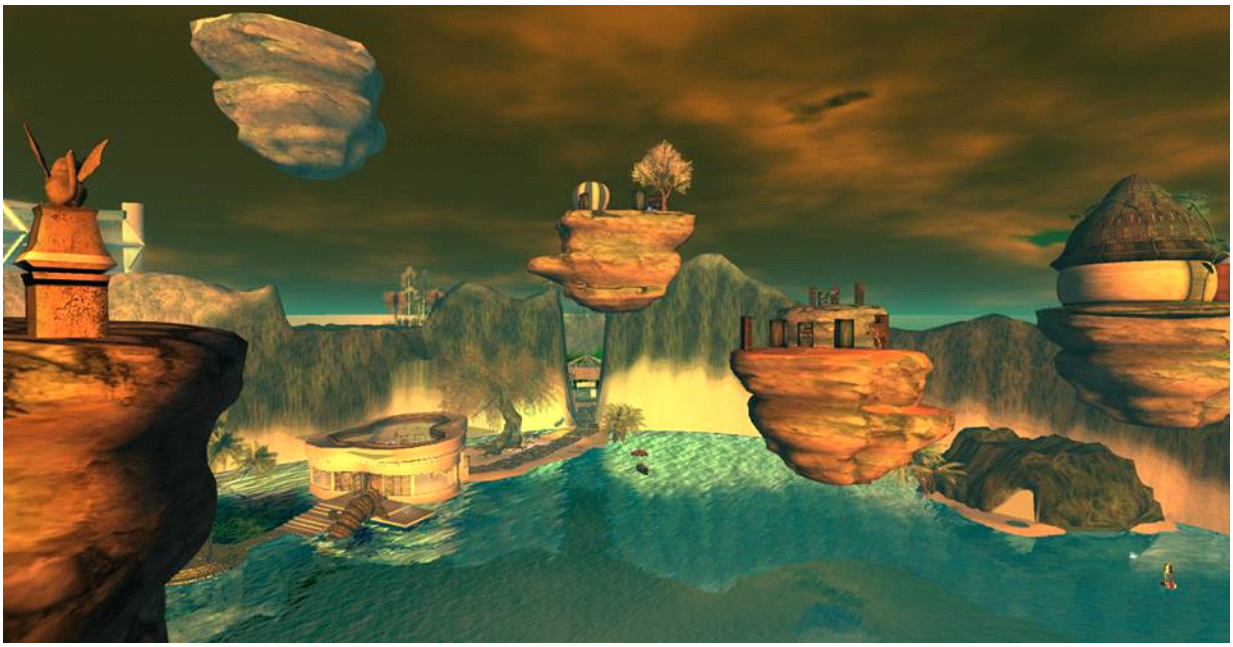


Figure 1. An overview of Alexandria structured as disparate floating islands, each themed by the type of literature that the shelves on the islands contain.

Browsing a library with thousands of books becomes extremely difficult if there is no previous knowledge of the material sought after. Alexandria tries to address such shortcomings by providing much more than book covers, searching system or timeline-based triage. Alexandria is built by creating floating islands (Figure 1), each decorated and themed by the literature that is placed on those islands. It is thus convenient to "just browse" the repository by following the decorations and artwork. For example, an individual looking for wartime journalism will find a floating island decorated with sculptures of soldiers, a burning vehicle and perhaps a house on fire that contains the bookshelves with the corresponding literature.

Alexandria does not address data archival and preservation as much as it addresses the delivery, display and triage of that data. Technically, as will be explained further, the technology of Alexandria

uses a database as a backend (namely, MySQL but SQLite is also possible) so it inherits the qualities of that database. However, the improvement over other similar database-based libraries is the ability to include artistic hints and pointers in order to deliver thematic literature repositories at a glance.

4. Technical Background

SecondLife (SecondLife, 2003), a product of Linden Labs (Atkinson, 2008), provides a programming language, called the Linden Scripting Language (LSL). While LSL provides most of the necessities of object manipulation, the language has no constructs for storing large chunks of data within the virtual world. In short, using LSL one is able to read data from a document format called Notecards (an in-game text storage solution by Linden Lab, and OpenSim), but one cannot write to Notecards. For building libraries such as Alexandria, this read-only limitation makes the bulk-upload of textual material prohibitive - short of having to manually copy and paste the contents of every document.

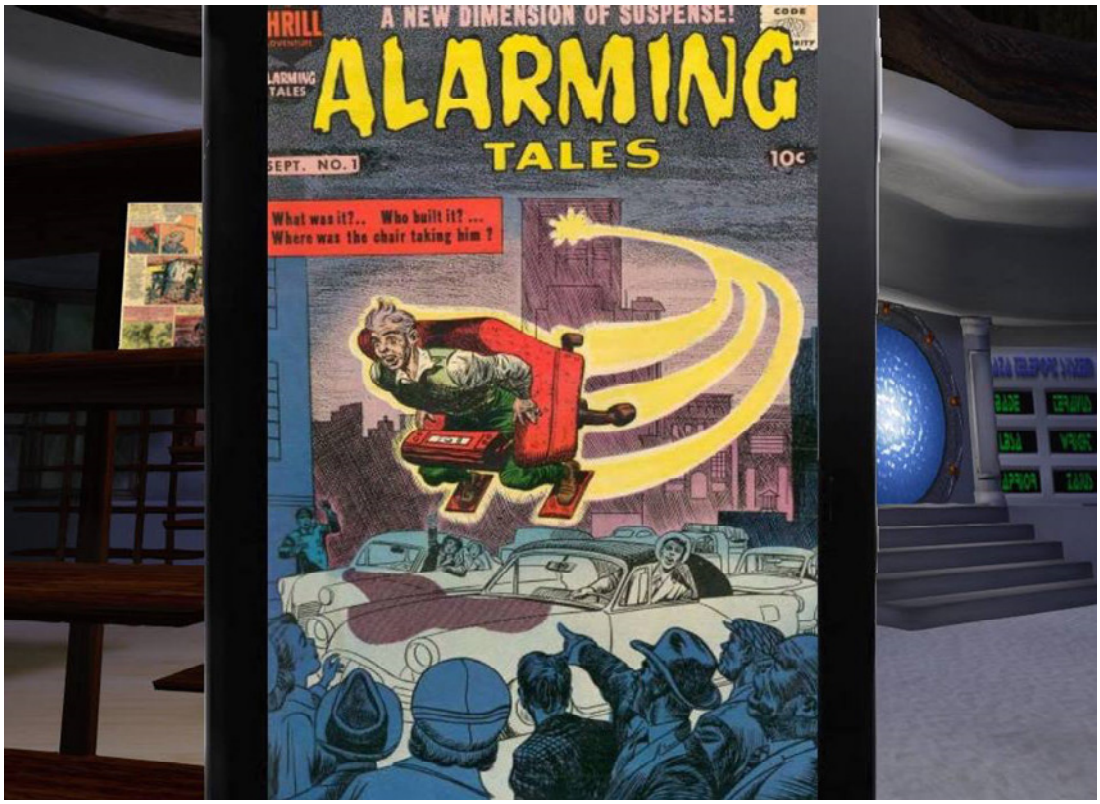


Figure 2. A tablet-like book rendered in Alexandria on the screen with the ability to browse chapters as well as turn pages whilst still being able to interact in-world with other people.

On the other hand, OpenSIm allows C# (C Sharp, a Microsoft Programming Language) scripting within the virtual world and additionally provides extensions to LSL, called the OpenSim Scripting Language (OSSL) that is able to generate Notecards. Programming in LSL and using the OSSL extensions we have created a technical framework for Alexandria that allows the import of textual material within the virtual world.

Nevertheless, the Notecard format created by Linden Lab is limited and only able to render plain Unicode text as well as integrate landmarks or pictures. Another limitation is that scripts cannot read Notecards that span more than 255 bytes. This is insufficient in the case where technical documents have to be stored on Alexandria. The uploads are thus split into two main categories: documents that do not include figures or that do not rely on extensive typography (such as mathematical renderings) are uploaded using the Notecard format, otherwise technical documents are converted to images first and then uploaded directly into Alexandria as textures and stored on tablet-like devices within the virtual world that students can read (Figure 2).

4.1 Automatic Document Synchronization

Alexandria is built using "themed" floating islands, where every island is decorated to the specifics of the type of literature that can be found on that island. For example, the American poets floating island is roughly a fusion between the Gothic styles of Edgar Allan Poe, coupled with the rural style in the industrial era. Similarly, the ancient Greek literature island is decorated using statues and cultural-specific characteristics pertaining to the Greek islands.

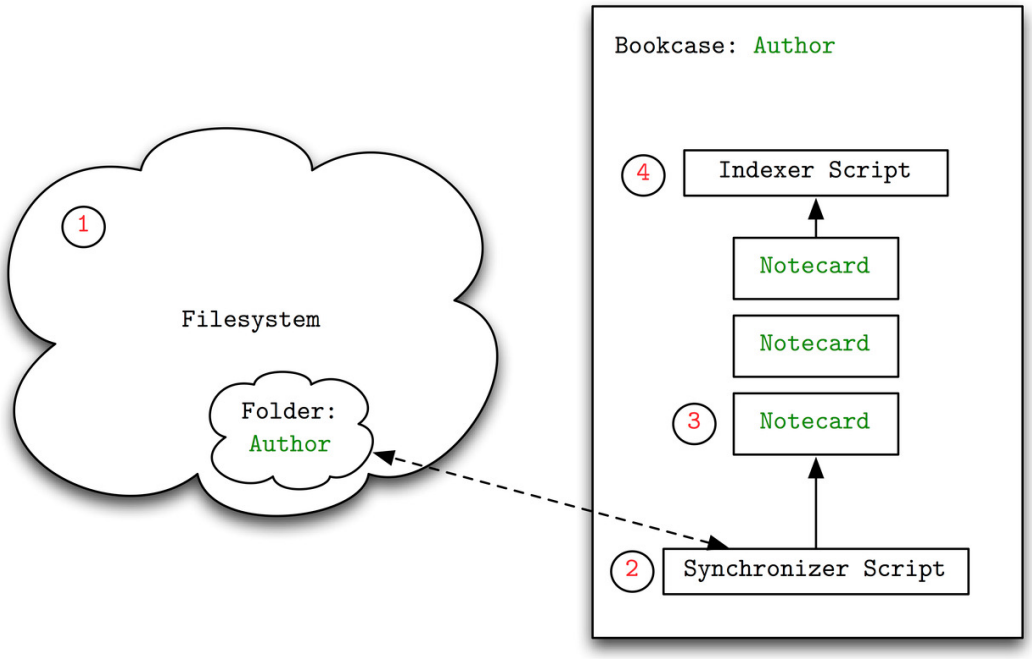


Figure 3. The filesystem contains several folders named after the author (1). These folders get processed from an in-world synchronization script (2), which reads the documents in the folder and generates Notecards within the bookcase (3). Additionally, and indexer script (4) reads the Notecards whenever an individual searches for a keyword.

On these themed-islands, bookcases are placed that are given the name of the author whose work they will contain. The underlying filesystem contains a folder that carries the same name of that

bookcase. A synchronizer script, illustrated in Figure 3, scans the folder on the filesystem for documents.

Whenever a document is added or deleted from the named folder on the filesystem, the synchronizer script regenerates the bookcase by creating or deleting Notecards. Adding a new author is a task that consists of creating a folder on the filesystem named after the author and then creating a bookcase on Alexandria, giving the bookcase the name of the author and adding the synchronizer script to the bookcase.

The "filesystem" in Figure 3 is to be thought of as a generic filesystem that could have an underlying network or cloud-based storage (Wu, Ping, Ge, Wang, & Fu, 2010). The synchronization script does not care about the lower-layers of Input-Output (IO), but is designed to be aware of folders and text-files. This gives great opportunities for collaboration; for example, we have been using Dropbox (Drago et al., 2012) cloud storage as a mounted share on the server that Alexandria runs on in order to allow several contributors to upload material into Alexandria. Once a bookshelf is created, named and the synchronization scripts are set-up, users can start to add documents that the synchronization script reads and automatically adds the documents to the corresponding bookshelf. Given limited resources and personnel, we have found that this method of building a library by crowdsourcing (Huberman, Romero, & Wu, 2009; Brabham, 2008) was by far more feasible than hiring a dedicated staff. Other filesystems based on high-level IO abstraction, for example filesystems in userspace (FUSE, 2009) or distributed filesystems such as CODA (Kistler & Satyanarayanan, 1992) and even distributed systems (Lauvset, 2001), are also possible options for storage backends.

4.2 Searching and Indexing

The indexer script illustrated in Figure 3 communicates with an in-world object called the "Oracle" in order to locate the position of bookcases in Alexandria whenever an individual (a human being, called an "agent" in Linden Lab terms) searches for a given keyword.

When an agent searches for keywords using the "Oracle", the query is relayed over a communication channel to all the bookcases on Alexandria. The indexer script returns a positive result if those keywords appear in the name, description or within the documents contained in the bookcase. This is actually performed by a weighted search, for example, so that if an agent requests the name "Poe" the indexer script within the Edgar Allan Poe bookcase will be more likely to return a result in time than if the keyword was found within the documents.

The weighting is performed by racing the indexer scripts and by setting a timeout so that first the name of the bookcase is checked, then the description and if there is sufficient time left over before the query expires, matches within the document are returned as well. This ensures that only the most relevant bookcases are selected for a given user-supplied query. The concept of racing the bookcases using timeouts is inspired from race-conditions (Carr, Mayo, & Shene, 2001) in programs but made useful for the purpose of delivering relevant data.

When an indexer script within a bookcase triggers a positive result for a given search-query, it sends the name and global coordinates of the bookcase to the "Oracle" (Figure 4). The procedure of placing responses on shared communication channels can be seen in UNIX (a family of operating systems) terms as making use of data pipes. After the timeout, the "Oracle" organizes the responses from the bookcases and offers the agent a pop-up menu with the names of the bookcases. The agent can

then select an item from the menu and sit on the "Oracle" in order to quickly travel to the bookcase location.

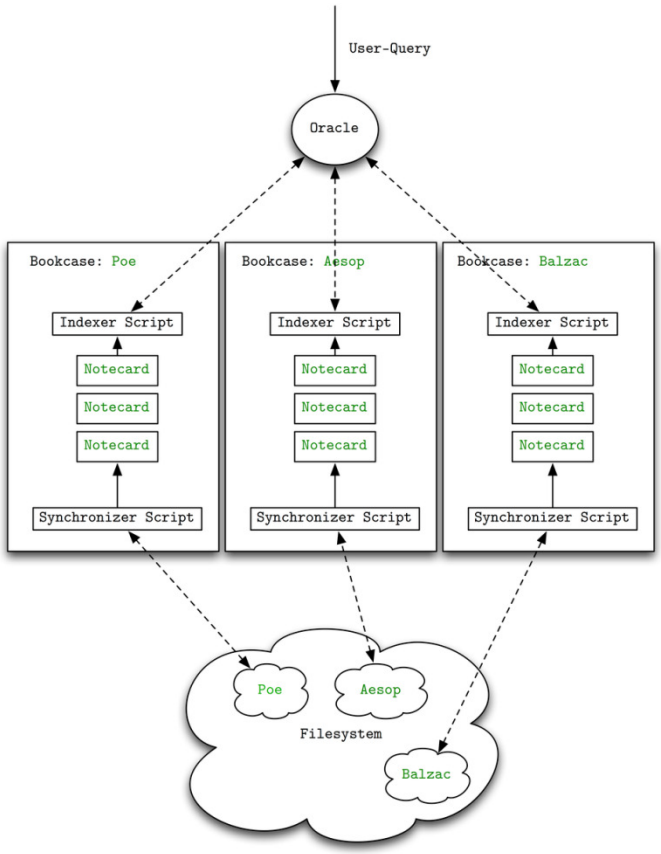


Figure 4. Whilst the Notecards are generated using the synchronizer script, the "Oracle" provides the means to query the bookcase whenever an agent searches for a keyword.

Additionally, all the bookcases contain a third script called "giver" that hands out the Notecards whenever an agent clicks (or "touches", in Linden terms) a bookcase.

4.3 Programming Notes

The indexer, oracle and giver script are programmed in LSL while the synchronization script is written using C# and OSSL. The usage of different programming languages is intentional because OpenSim does not have C# or OSSL enabled by default. This is used to Alexandria's advantage because Alexandria can be forked into a development grid, where content is created and uploaded, and separately one can maintain a publicly accessible grid that only hands out and indexes the literature. Moving content between the development and the production grid is performed by OpenSim's region-wide export capabilities called OARs - essentially an OAR file is just a compressed tape-archive (IEEE, 1992) of assets.

4.4 Catastrophic Failure

The simple design that Alexandria uses for content addition secures the library with multiple levels of protection in the event of a catastrophic failure. First, the OAR system allows the creation of

incremental backups, as well as running a mirrored-image of Alexandria on the display grid - thereby offering redundancy. Second, all the literature is stored in the cloud, so that Alexandria does not have a single point of failure (Armbrust et al., 2010). In the event that contributors lose all their data, then that data will still be available on the cloud and on the machines of the contributors.

OpenSim can use either MySQL or SQLite as a storage backend. Within that database, all the assets are stored, along with the Notecards containing the literature. In the event that the database becomes corrupted, the documents are still present on the cloud and, in the event where a reconstruction is needed, the synchronization script will automatically regenerate the bookshelves without requiring human intervention. Using Dropbox for this purpose is perhaps not the best solution because if a participant chooses to delete files, then they will be deleted from all other machines. In Alexandria's case, all participants are considered trustworthy for the task of populating the library with literature.

5. Overview of The Library

Alexandria shows itself to be a tool capable of expanding and becoming a useful item for schools around the globe. The ability to perform backups and restore items back into the grid when destroyed or removed by accident is invaluable in making sure that there is no catastrophic failure of the system and that any loss by a power outage is minimal. Given the virtual setting, the interaction between students and teachers may take place in a more comfortable environment that allows for an easy oversight of students using the system. Social interaction is promoted as well, allowing groups of students to collaborate and learn in a safe and fun environment for them. Since Alexandria provides contextual clues to books, specific genres can be quickly located and used for research, reports and discussions. Furthermore, the virtual world provides a user inventory that allows them to take the books and visit other grids where lessons may be planned. For example, a physics book can be taken from Alexandria and brought to a physics simulation, so that equations and principles are at hand. This eliminates the stress induced by having to search through a backpack for a specific book if forgotten in a dorm or classroom.

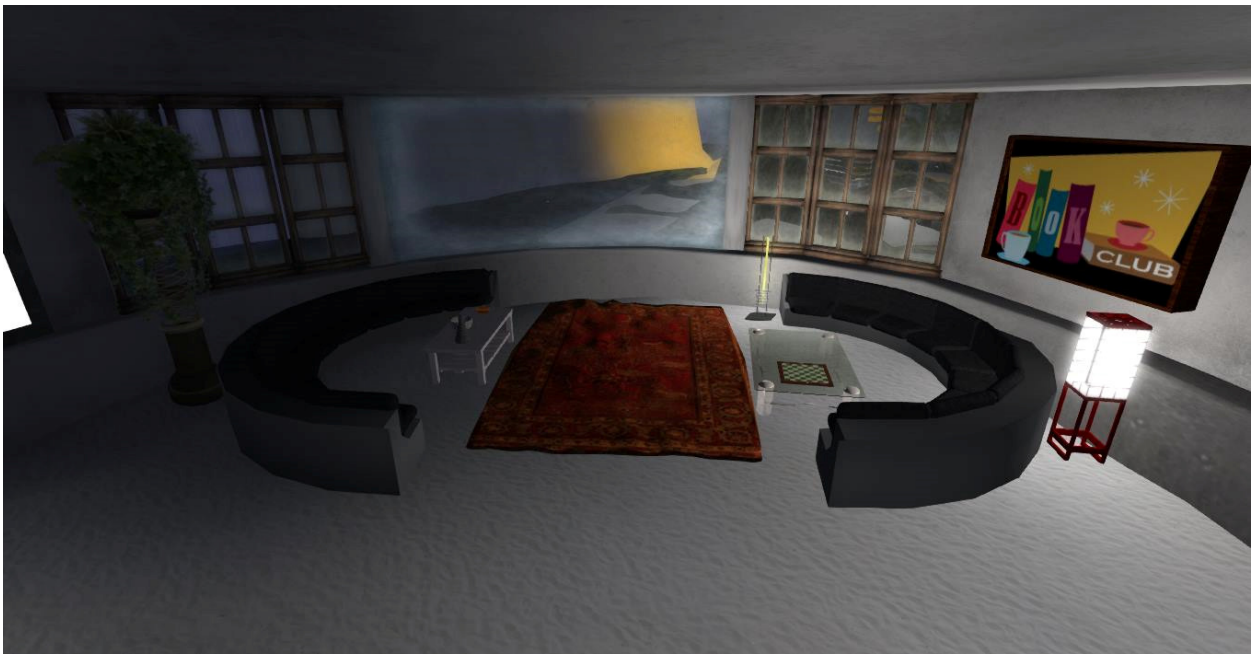


Figure 5. The "Book Club" landing point where students can meet and discuss as well as use the Oracle just outside to browse the available literature.

Alexandria's main landing point includes HyperGate access to two gate networks in OpenSim. Since this is also an area for new users to get acclimatized, there are couches, books, and help available for those new to virtual worlds (Figure 5). Studies have shown (Blackmon, 2012) that the social interaction between scholars, even just by chatting, can lead to a better understanding of the course material or the knowledge acquired from the literature that Alexandria contains.

Next to the landing area is a collection of popular books for browsing, so that they are easily accessible, especially for those that are new to virtual worlds. Also included is a teleport at each island along with the art center and landing area for easy teleport between islands and areas. The art center houses a collection of works ranging from classics including Michelangelo and Da Vinci, up to more modern artists such as Andy Warhol and Picasso. A large overview of the Library is visible from the windows in the art gallery, along with plenty of chairs to sit and look out over the main bay. The design attempts to combine visual art with literary art.

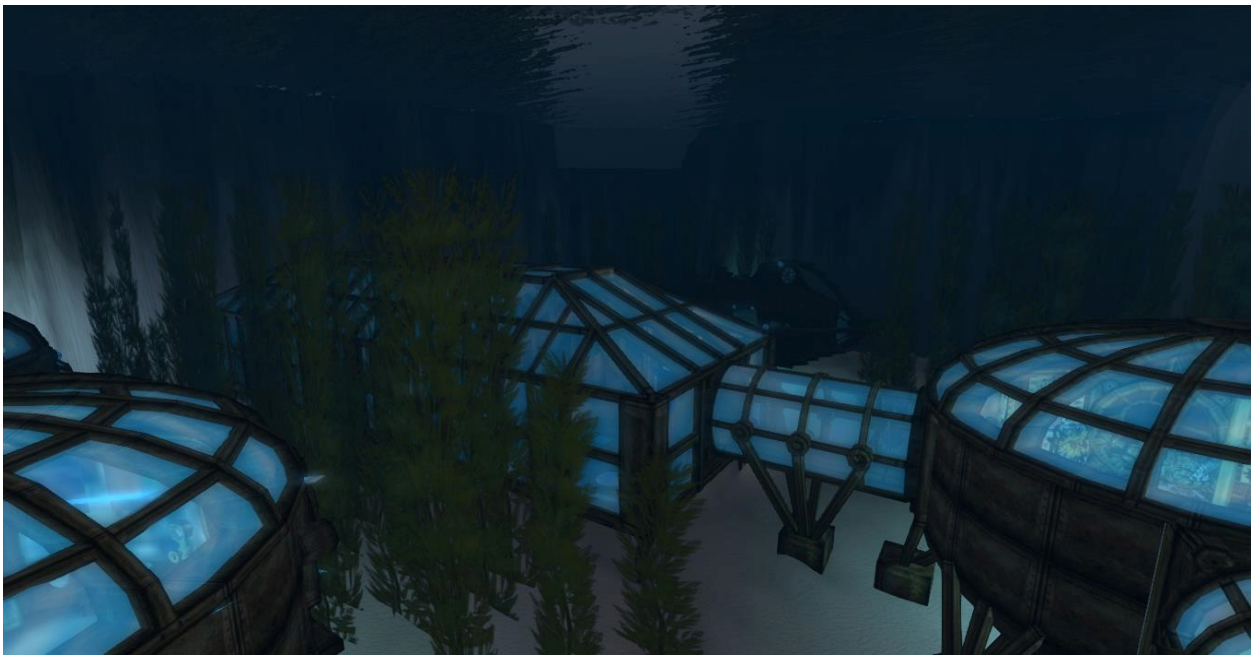


Figure 6. Alexandria's submerged library built under the water line and themed after Jules Verne's "Twenty Thousand Leagues Under the Sea" that contains most of Jules Verne's works as well as other Steampunk-themed science fiction literature.

Under the art center, on a dock, is the tunnel to the underwater Science Fiction section, themed after Jules Verne's Twenty Thousand Leagues Under the Sea (Figure 6). Included with the Science Fiction area is a collection of game cheat codes and tips, arranged alphabetically for easy access. The design of Alexandria includes a possible visual experience in addition to the literature that is available. A submerged Steampunk-themed repository can offer the visual feedback as well as a suitable environment for reading. Not only do the rich visuals help with the triage of the available literature (as mentioned previously) but also it helps by immersing the reader into a world similar to that depicted in the literary works (Warburton, 2009). Having a rich perceptual overview of the literature helps students retain knowledge by being able to trace the connections between what they have read and the images that they remember.

Figure 7. The floating island dedicated to American poets sorted automatically in various bookshelves containing poems on Notecards

A comparison between Alexandria and classic libraries is necessary to understand the benefits. Unlike a normal library, there is no need for large numbers of staff, no need for electricity for lighting and heat, and no need for re-shelving of books (Figure 7). As in a normal library, books can be added and shelved, but the shelving is automatic, and, unlike a real world library, there is no need to push a cart of books or to re-shelve them. Additional benefits help those users who are unable to travel to a real world library. With Internet access and a computer, they can access the library from anywhere and receive books to read.

6. Further Work

Further work on Alexandria includes many new ideas, including the conversion of content to a widely recognizable format such as "ePub" (Marinai, Marino, & Soda, 2011) by sending a requested book to an user via E-mail on a request popup of a shelf. Expanding the library from primarily English to many more languages so that novels, poems, and other items become more relevant to people of other nationalities is one future goal. Also planned are document uploading systems for teachers to easily add to the library documents such as notes or a lesson plan or even homework for students. Additional communications are planned with students to have them send, receive, print, and/or edit documents.

The design of the library allows for asynchronous additions and design expansion, so that it can be reconfigured for virtually any use. Additions such as classrooms, teacher areas, and planned areas for

social events would allow for virtual meetings and events, even if a student cannot be in the classroom physically.

One other possible avenue to explore is to use a distributed cloud-based storage system based on peer-to-peer networking such as BitTorrent Sync (Pounds, 2014), which would eliminate a common point of failure and offer data duplication amongst the contributors.

7. Conclusions

Alexandria contains collected literature from various sources with many literary works supplied by the Gutenberg project (Gutenberg, 1971). At the time of this writing the library spans multiple DDC classes that are displayed in either Notecard or tablet format. DDC sorting seems to be inadequate for sorting modern books by the very way it classifies them. Many derivatives of literary genres such as Science Fiction, Fantasy, Teen Paranormal, Mystery, Crime Investigation, Children's Books, Self-Help are not included in DDC.

Documents on Alexandria (DDC)	Number
French Satire and Humor	66
Literature	664
Poetry	4632
English Fiction	351
Ethics	269
Aristotelian Philosophy	1
Platonic Philosophy	1
Biographies	1
German Essays	90
Other Germanic Literatures	12
German Fiction	2
Fiction	243
Science and Religion	1
Genetics and Evolution	2
Drama	12
Total	6347

Figure 8. The number of documents present on Alexandria using the DCC classification at the time of writing. The poetry section appears to be the largest because the value refers to the number of poems rather than the number of authors.

Each of these categories has stronger subdivisions, for example as a subset of Science Fiction such as Cyberpunk, Steampunk, and Retro book types which are not amongst the DDC system. Follwong the DDC system however, Alexadria currently contains a large repertoire of literature as can be seen from the classification in Figure 8. Most of the categories could be further sub-divided, for example, to include comic books which in the DDC system have no precise division. However, using the rich visual environment and Alexandria's thematic design, searching a certain type of a subdivision can be accomplished by using the visual aids and turning to the corner dedicated to comic books.

Alexandria's benefits are numerous, and far outweigh any negatives. For example, the ability for anyone worldwide to access a library at any hour, have materials organized by their topics, as well as a way to talk to colleagues at any time, is something that very few real world libraries have. The additional help of automatic shelving and ability to instantly find any book (via the oracle platform) and instantly send a user to their requested material is also unparalleled in the real world, but easily possible in virtual space. Any catastrophic failure in Alexandria is virtually unheard of, with the ability to cloud store books and documents, and automatically re integrate them into the system, a feature that is not quite so easily available in the real world; if a fire hits, either the fire, or water damage from attempts to put out fire will occur, making losses inevitable. The additional use of themed book locations is also unparalleled in the real world. While children's book areas are often brightly designed to attract attention, adult book sections are often drab shelves with little individuality beyond every shelf being identical. As Alexandria expands, the ability to personalize and tailor the library to fit user's needs is easily accomplished, requiring little more than a few days' work, whereas rebuilding a real world library could take months, or years. As such, the Alexandria platform for libraries puts it in a league of its own and it will continue to expand and grow in years to come.

References

Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., Zaharia, M. (2010, April). A view of cloud computing. *Communications of the ACM, 53* (4), 50–58. Retrieved from http://doi.acm.org/10.1145/1721654 .1721672

Atkinson, T. (2008). Inside Linden Lab. *TechTrends*: Linking Research & Practice to Improve Learning, 52(3), 16-18.

Blackmon, S. J. (2012). Outcomes of chat and discussion board use in online learning: A research synthesis. *Journal of Educators Online, 9*(2). Retrieved from http://www.thejeo.com/Archives/Volume9Number2/BlackmonPaper.pdf

Brabham, D. C. (2008). Crowdsourcing as a model for problem solving an introduction and cases. Convergence: *The International Journal of Research into New Media Technologies, 14*(1), 75-90.

Carr, S., Mayo, J., & Shene, C. K. (2001, October). Race conditions: A case study. J*ournal of Computer Science Collection, 17*(1), 90–105. Retrieved from http://dl.acm.org/citation.cfm?id=772488.772504

Dewey, M. (1965). *Dewey; Decimal classification and relative index*. New York; Lake Placid Club Education Foundation, 1965, 2v. (ix, 2153 p.)

Drago, I., Mellia, M., Munafo, M., Sperotto, A., Sadre, R., & Pras, A. (2012, November). Inside dropbox: understanding personal cloud storage services. In *Proceedings of the 2012 ACM conference on Internet measurement conference* (pp. 481-494). ACM.

FUSE. (2009). Filesystem in userspace. Retrieved February 15, 2015, from http://fuse.sourceforge.net/.

Gutenberg. (1971). Project Gutenberg [Digital Library]. Retrieved February 15, 2015, from http://www.gutenberg.org/wiki/Main Page

Huberman, B. A., Romero, D. M., & Wu, F. (2009). Crowdsourcing, attention and productivity. *Journal of Information Science, 35*(6), 758–765. Retrieved from http://dx.doi.org/10.1177/0165551509346786

IEEE. (1992). IEEE standards interpretations for IEEE standard portable operating system interface for computer environments (ieee std 1003.1-1988). IEEE Std 1003.1-1988/INT, 1992 Edition. doi: 10.1109/IEEESTD.1992 .106983

Kistler, J. J., & Satyanarayanan, M. (1992). Disconnected operation in the Coda file system. ACM *Transactions on Computer Systems (TOCS)*, *10*(1), 3– 25. Retrieved from http://doi.acm.org/10.1145/146941.146942

Ladd, G., & Mize, J. (1983). A cognitive-social learning model of social-skill training. *Psychological Review, 90*(2), 127-57.

Lauvset, K. A. J. (2001). Tos: Kernel support for distributed systems management. In *proceedings of the sixteenth ACM symposium on applied computing, Las Vegas*. ACM Press.

Lee, K. H., Slattery, O., Lu, R., Tang, X., & McCrary, V. (2002). The state of the art and practice in digital preservation. *Journal of Research of the National Institute of Standards and Technology, 107*(1), 93–106.

Marinai, S., Marino, E., & Soda, G. (2011, September). Conversion of PDF books in ePub format. In *Proceedings of International Conference on Document Analysis and Recognition (ICDAR), 2011* (pp. 478-482). IEEE.

Medin, D. L., & Schaffer, M. M. (1978). Context theory of classification learning. *Psychological Review, 85*(3), 207–238.

Messinger, P. R., Stroulia, E., & Lyons, K. (2008). A typology of virtual worlds: Historical overview and future directions. *Journal for Virtual Worlds Research, 1*(1). Retrieved from https://journals.tdl.org/jvwr/index.php/jvwr/article/view/291

Overte. (2011). Opensim. Retrieved February 15, 2015, from http://opensimulator.org/wiki/Main_Page

Pounds, E. (2014). Introducing BitTorrent Sync 1.4: An easier way to share large files. Retrieved February 15, 2015, from http://blog.bittorrent.com/2014/08/26/introducing-bittorrent-sync-1-4-an-easier-way-to-share-large-files/

Ranstam, J. (2008). Data handling, statistical computing, and archiving. *ActaRadiol, 49*(10), 1137-9. Retrieved February 15, 2015, from http://www.biomedsearch.com/nih/Data-handling-statistical-computing-archiving/18608011.html

SecondLife. (2003). Second life website. Retrieved February 15, 2015 from http://www.secondlife.com.

Wang, S., Mao, Z., Zeng, C., Gong, H., Li, S., & Chen, B. (2010). A new method of virtual reality based on Unity3d. In *Proceedings of 18th international conference on Geoinformatic,* (pp. 1–5).

Warburton, S. (2009). Second Life in higher education: Assessing the potential for and the barriers to deploying virtual worlds in learning and teaching. *British Journal of Educational Technology, 40*(3), 414-426.

Wiegand, W. A. (1998). The ”Amherst Method”: The origins of the Dewey decimal classification scheme. *Libraries & Culture*, 175–194.

Wu, J., Ping, L., Ge, X., Wang, Y., & Fu, J. (2010). Cloud storage as the infrastructure of cloud computing. In *Proceedings International Conference on of Intelligent Computing and Cognitive Informatics (icicci), 2010* (pp. 380–383).