The Open Metaverse Currency (OMC)
A Micropayment Framework
for Open 3D Virtual Worlds

Frank Kappe and Michael Steurer

Institute for Information Systems and Computer Media
Graz University of Technology, Austria
{frank.kappe, michael.steurer}@iicm.tugraz.at

Abstract: Virtual worlds have become very popular recently, in particular 3D virtual worlds with a built-in virtual currency that enables providers and users to monetize their creations. Virtual Worlds based on the relatively new open-source and free-to-install "OpenSimulator" software currently lack such a virtual currency framework, in particular one that can be used across the boundaries of such virtual worlds (also known as grids).
In this paper we present the design of such a micropayment system for this class of virtual worlds. The functionality expected by the users of such a 3D environment is somewhat different from what is usually expected from Web payment systems. A particular challenge is the fact that parts of the software that our system is to be integrated with cannot be trusted, as they are run by unknown parties, and can easily be modified (as it is open-source software). Transaction values in such a virtual environment are usually very small, so the cost of a transaction is also of concern.
The system has already been implemented and is used by a number of such virtual worlds. We believe that we will already be able to include some first evaluation results in the final paper.

Keywords: 3D Virtual Worlds, Micropayment, Virtual Currency, OpenSimulator, Hypergrid, Open Metaverse.

1 Introduction

Virtual worlds are a special class of social software, where users are represented by avatars in a virtual environment. The avatars interact with each other in real time using chat or voice chat. In contrast to most other social networks, the users behind the avatars typically remain anonymous. Unlike Massively Multiplayer Online Games (MMOGs), which use similar technology, virtual worlds are not a game, with predefined goals, winners, and losers, but are more open-ended, with a focus on user interaction and creativity.

Virtual worlds have become very popular and are still growing [1]. A recent study by virtual world research firm Kzero [2] estimates over 800 million users in over 300

different virtual worlds. However, these virtual worlds are very different in nature. The majority targets children and teenagers as customers, but there are others where the average user is over 30 years old. Many of the virtual worlds are only two-dimensional in nature, running in a web browser.

Yesha Sivan defines as "Real Virtual Worlds" [3] virtual worlds which share the "3D3C" properties, *i.e.* three-dimensional, with an emphasis on Community, Creation, and Commerce. An example of such a "3D3C Real Virtual World" is Second Life [4].

Real virtual worlds allow users to create virtual goods (*e.g.* clothing, buildings, furniture, vehicles, weapons, animations, scripted objects), and sell them to other users who lack the skills or time to create them for themselves [5]. The global market for virtual goods is estimated to be approximately EUR 1,5 billion a year [6] with Second Life, Tencent, IMVU, Gaia Online, and Habbo Hotel among the leading players. In the virtual economy of Second Life alone, about 2 million US$ worth of virtual goods and services are traded between users on an average day [7]. According to statistical data published by Linden Lab [8], the company that runs Second Life, about 69.000 users made a profit in Second Life in February 2010. In 2009, there existed a few Second Life entrepreneurs, whose profits exceed 1 million US$ per year [9].

However, there is a significant obstacle for further growth: The current virtual worlds are walled gardens, completely isolated from each other. The situation can be compared to the pre-Web Internet, with companies like Compuserve, America Online, Prodigy, The Source, and others competing for users and publishers. The incompatibility required publishers to take bets on where to invest their money, which limited the growth of this industry. Finally, the Web came along, with open standards and open-source implementations, and removed this barrier to entry for both publishers and users.

2 Introducing the Open Metaverse

The term "Metaverse" has been coined in 1992 by Neal Stephenson in his science fiction novel "Snowcrash" [10], where he described an immersive 3D virtual world. In fact, this book has strongly influenced the design of virtual worlds such as Active Worlds and Second Life, which can therefore be regarded as an implementation of this vision. We will use the term "Open Metaverse" to describe virtual worlds like Second Life, but implemented using open-source software such as OpenSimulator (also known as OpenSim), Sun's Project Wonderland, and OpenCroquet. In addition to these "de facto" standards, a few "official" standardization initiatives try to standardize communication protocols and file formats of virtual worlds [11].

The vision of the open metaverse is to do for the 3D Internet what Apache and Mozilla did for the 2D Web. In fact, some limited interoperability (teleporting) between Second Life and OpenSim has already been demonstrated, and we already see some OpenSim-based virtual worlds appear [12].

However, the situation with 3D worlds is more complicated than in the 2D Web: Even if all virtual worlds would use the same protocol, viewer, and 3D object

representation, it still would not be possible to *e.g.* buy a virtual good in one world, and use or sell it in another. Today's virtual worlds are like islands, each with their own currency, user profiles, permissions system, and asset repositories.

However, this may soon change. The Hypergrid [13] is an extension to the OpenSim protocol which enables teleporting between OpenSim-based virtual worlds (which are called grids). You can even take virtual assets (*e.g.* clothing) with you (more precisely, your avatar) as you roam the OpenSim Hypergrid.

A key component for making the Open Metaverse a "real virtual world" according to Sivan's definition is the "commerce" aspect. This in turn requires a payment system, which is not yet part of the OpenSim framework.

3 Requirements

Designing and implementing a payment system for the OpenSim environment poses some interesting requirements (and challenges), both technical and non-technical. We can divide these issues roughly into the three groups functionality, trust, and cost.

3.1 Functionality

The functional requirements of a payment system for 3D virtual worlds are in some aspects quite different from the "traditional" payment systems found in the 2D Web or even the real world. In the OpenSim case, the expectation of the users is to find the same level of functionality available in the virtual world "Second Life", which is arguably the most successful implementation of a micropayment system, with 50 million US$ worth of transactions per month [14]. Also, as we will describe in more detail in the "System Architecture" section, the client software used is compatible with the Second-life client (which is open source). Therefore it makes sense to model the micropayment system for OpenSim after what is available in Second Life.

Unlike earlier implementations [15], we did not want to use in-world objects representing virtual wallets or head-up displays, because of the negative impact on simulator performance and the necessity to allow script permissions to everybody. Specifically, our micropayment system has to support the following use cases:

1. User A pays user B, while both users are online in the same simulator and see each other. In the user interface, this can be done by user A right-clicking on the avatar of user B, and selecting "pay" from a pie-chart menu.
2. Like use case 1, but user B is not online (or out of sight). Still, user A can bring up B's profile and select "pay" in the profile window.
3. User A likes to pay an object in the virtual word owned by user B. This can be achieved in the user interface by right-clicking on the object, and then selecting "pay" similar to use case 1. The owner of the object (B) will receive the money, but in addition an event (the money-event) is raised in the script running in the object, so that the object can react to it.

4. User A wants to buy an object in the virtual word owned by user B. Again, this is done by user A right-clicking the object and then selecting "buy" in the pie menu (see Figure 1). User B has to set the object's properties to "sellable" and set a price beforehand. User B will get the money; in addition the object (or a copy of it) will be delivered to the inventory of user A.

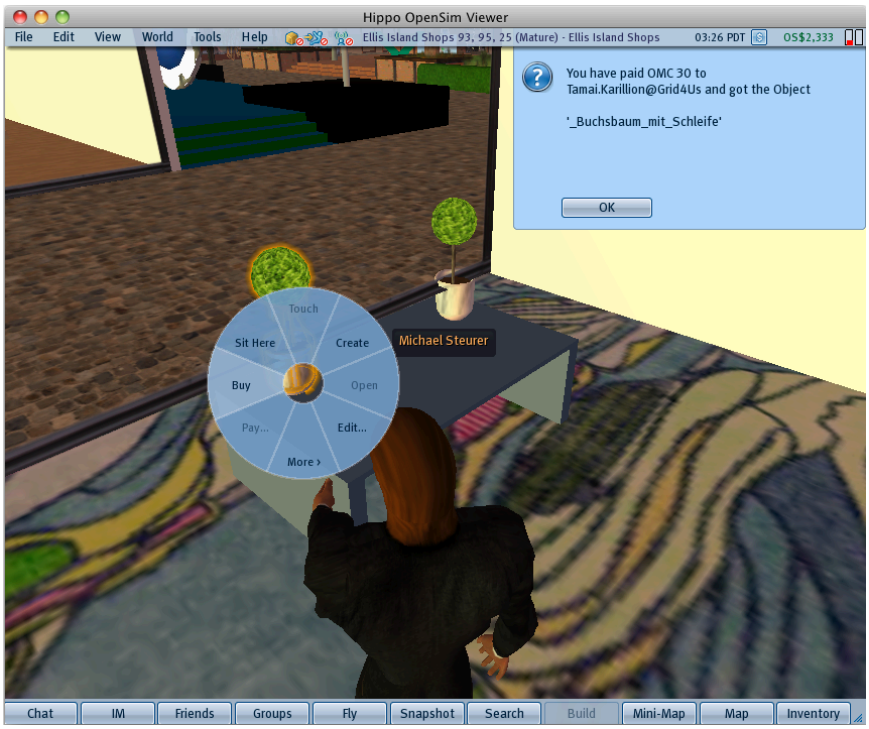

Figure 1: "Object Buy" dialog with the notification on the payment made.

5. User A wants to buy land currently owned by user B. User A initiates the process by selecting "buy land" form the land properties page. User B has set the land properties to "sellable" and set a price beforehand. Once the money has been paid, the OpenSim database will be updated to reflect the transfer of ownership.
6. An object owned by user B pays (virtual) money to user A. Note that user B will typically not be on-line when this happens (nor does user A have to be online). Therefore, the owner (B) will have to specifically give debit permissions to the object beforehand, to access the user's account and send money on the user's behalf without further confirmation.

These are the six cases necessary for a smooth in-world user experience similar to that of Second Life. Additional use cases were included to support buying objects and the like from more traditional user interfaces like a Web shop, but these are less interesting and beyond the scope of this paper.

3.2 Security and Trust

Obviously, security is a key concern of any payment system. In a single-provider virtual world (like Second Life), the provider will usually not only run the virtual world as such, but also be providing the micropayment system, as well as means to buy the virtual currency (an exchange), and take the role of the central bank responsible for keeping the value of the currency stable. Therefore, the user will have to trust only the provider of the virtual world.

In contrast, in the OpenSim environment, the people running the individual servers (so-called simulators) are many, and users should not need to trust them. Therefore, it makes sense to separate the functionality of the payment system provider from that of the provider of the virtual world, so that the payment authorization process effectively bypasses the unsecure OpenSim servers. Users will of course still have to trust the payment system provider, but not necessarily the provider of the specific simulator they are currently in, as the (virtual) money never leaves the realm of the payment provider.

For the user experience, this will mean that – unlike in Second Life – for the use cases 1 – 5 the payment will have to be confirmed at the external website of the payment system provider, where users will have to have an account to participate in the virtual commerce. This is somewhat more inconvenient than a complete in-world experience, but is the price to be paid for security. An alternative would be to embed the payment functionality in a modified viewer, and enforce its use, which would probably be seen as even more inconvenient.

Use case six is a special security concern. The owner of the object paying (virtual) money on the owner's behalf will have to not only confirm the debit permissions in the virtual world, but in addition on the payment providers website. This is necessary to restrict any provider of a simulator to fake these permissions. Still, the script running inside the object paying out can be read and modified by the administrator of the simulator, which is why the users using this functionality must trust the simulator provider, or preferably run the simulator themselves.

3.3 Transaction Costs – Virtual Currency

The value of a user-to-user transaction in a virtual world is typically very small, *i.e.* less than a Dollar or Euro. It is therefore important to keep the transaction costs for user-to-user transactions close to zero. This requirement rules out the classic forms of payments like credit cards, debit cards, or PayPal.

The solution that most virtual worlds have adopted is to use a virtual currency, with all user-to-user transactions taking place in the virtual currency within the realm of the provider of the virtual world, usually without any transaction fee. Fees are only charged when the virtual currency is bought or sold in exchange for real-world currency. This typically happens in larger amounts than the individual user-to user transactions and consequently this exchange fee is less of an issue.

In addition to the reduction in transaction cost, using a virtual currency offers the users the benefit that the goods can be offered worldwide, but without the hassle of

constantly adjusting the prices in real-world currencies as a result of moving exchange rates. For the provider, using a virtual currency can reduce the liability of the provider in case things go wrong, *e.g.* Linden Lab states in the Terms of Service of Second Life:

"Regardless of terminology used, Linden Dollars represent a limited license right governed solely under the terms of this Agreement, and are not redeemable for any sum of money or monetary value from Linden Lab at any time. You agree that Linden Lab has the absolute right to manage, regulate, control, modify and/or eliminate such Currency as it sees fit in its sole discretion, in any general or specific case, and that Linden Lab will have no liability to you based on its exercise of such right." [16]

For these reasons, we have chosen to implement our micropayment system by introducing a new virtual currency, the Open Metaverse Currency (OMC), with the Open Metaverse Cent (OM¢) as currency unit. To be fully functional, this requires that in addition to the micropayment system itself, there is a framework including an exchange that lets users buy and sell OMC for real-world currency, and a kind of central bank that controls the money supply. In this paper, however, we concentrate on the aspects of the micropayment system.

4 Architecture

In this section we show the design and the architecture of the entire payment mechanism but focus on a general view instead of detailed implementation issues.

4.1 Modified Event Handling

In Section 3.1 we have already described use cases of user interactions that require a transfer of money. All these mechanisms are already implemented in the client software, further referred to as OpenSim Viewer or in the server software, further referred to as Simulator. The Simulator has a modular design and the modules detect events sent by the OpenSim Viewer with event listeners and act on the received parameters with event handlers. Existing server implementations are not aware of money and, for instance, immediately deliver objects with the event handler in case of a "Buy Object" event initiated by the client and detected by the event listener.

To add money capabilities to the Simulator we have to split the event listener from the event handler and add the entire money transfer process in between. The methods in the event handler will only be executed if the actual payment succeeded. Figure 2 depicts this extension with an additional step in between the event handler and the event listener.

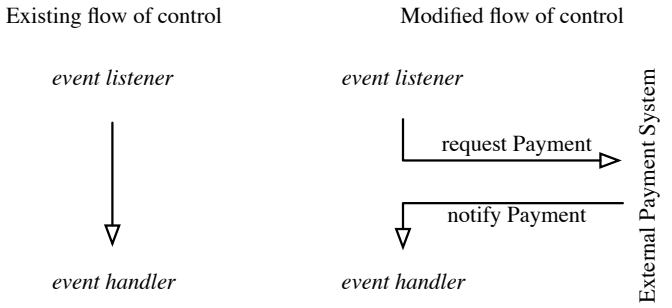

Figure 2: Split the event listener from the event handler in the simulator to do the payment

The structure of a grid comprises numerous different simulators responsible for all listen-events, event handlers, and everything in between. The source code of the simulators is publicly available and allows to run the service with potentially maliciously modified sources. This implies that every simulator owner could abuse a user's money account. To get rid of this problem the module that implements the event listener and event handler for money transfer does not contain any *control logic* but redirects all money related events to an external and trusted web service referred to as "Money Gateway". It puts all simulators under one umbrella and connects Simulators with the payment provider.

If the simulator detects a money-related event it sends a request to this centralized instance for further processing and waits for a notification to continue with the event handler. The money gateway employs the payment provider for the actual money transfer and therefore every user needs an account there. Users provide the payment sensitive information, *e.g.* login credentials, only to the payment provider and therefore bypass the untrusted simulator and the money gateway.

4.2 Bypassing Untrusted Software for Security

In the previous section we have stated that the actual payment process is hidden from the Simulator due to security concerns. To do so, a different and secure channel is needed because the client software suffers from the same problems as the simulator does – it cannot be trusted. To get rid of this constraint we establish the secure connection between the user and the payment provider with a common (and trusted) web browser. Secure HTTP connections (HTTPs) provide sufficient authentication mechanisms and cannot be wiretapped.

In case of a money related event the money gateway processes the simulator's request and redirects the user to the website of the payment provider in an external web browser. Most payment providers offer special programming interfaces to create payment requests that only need to be confirmed by a user. To do so, the user needs an account with the payment provider, provides credentials for the authentication, and finally confirms the payment created by the money gateway. On success the money is transferred and the payment provider informs the money gateway about the successful

payment. The money gateway redirects this notification to the simulator and prompts the interrupted event to be continued.

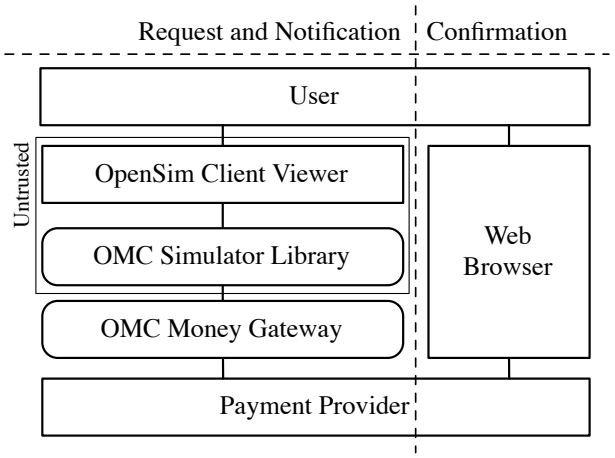

Figure 3: General architecture of a secure and trustworthy payment system in OpenSim based environments.

Figure 3 depicts a block diagram of the involved parties for a money transfer. As mentioned in Section 3.2 we cannot trust the OpenSim client viewer and simulator, and therefore need the bypass to directly communicate using a trusted web browser. Figure 4 shows an example of a message exchange for handling a money event.

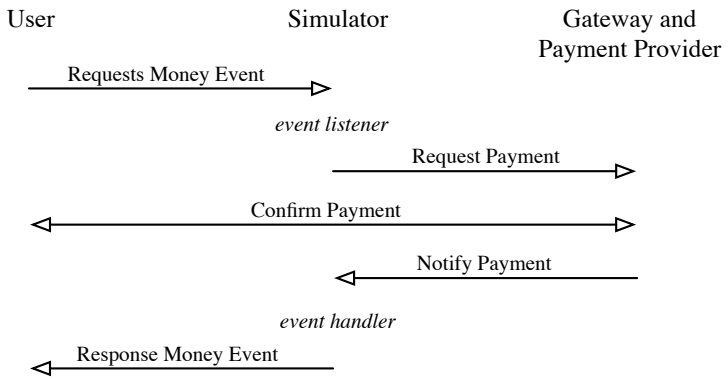

Figure 4: Requests, confirmation, and notifications for a successful and secure money transfer.

4.3 Design Limitations

We have already described the requirement of a secure channel for the payment confirmation due to the untrusted client viewer and simulator. The early stage of the entire simulator development raises another critical problem related to the security of the notification messages. All notifications are unencrypted XML-RPC and could origin from the money gateway, but also from anywhere else. Currently, the only available way to exchange information over an encrypted and authenticated channel is a secured HTTP request from the simulator to the money gateway.

Instead of directly transmitting the information of a successful payment to the Simulator, the money gateway just sends an identifier. With this identifier the simulator can fetch all necessary information about the confirmed payment from the money gateway via the encrypted channel. However, we consider the term "notification" as a synonym for the entire information transfer from the money gateway to the simulator.

The simulator needs to store all received parameters from the Viewer to be able to continue the execution on a notification of a successful payment. Unfortunately, the listen-event and the "Confirm Payment" message from Figure 4 are not necessarily temporally close. As described in Section 4.1 we have extracted the entire control logic from the simulator module to make the layer as thin as possible. We send all parameters received by the event listener to the money gateway but do not store it on the simulator. To continue with the processing on a successful payment all necessary parameters are sent with the notification message. The additional overhead of the notification message due to the sent parameters is acceptable because the stateless design makes the entire system more reliable and easier to monitor.

5 Conclusions

The complete micropayment framework has been implemented and put into actual use, with the Austrian company VirWoX [17] as external payment system provider and money exchange. Obviously, it is crucial for the acceptance of the new currency and the associated payment system that the provider enjoys the trust of the users. VirWoX has earned a reputation as the leading independent exchange for trading Linden dollars (the virtual currency used in Second Life) in exchange for real-life currency. In business since beginning of 2008, until the time of writing (March 2010) over 3 billion Linden dollars (about 8 million Euros) have been exchanged there. Therefore we believe that OpenSim users will trust the new currency.

At the time of writing, about 400 simulators in 5 different grids are connected to the system. We believe that in the final paper we will be able to provide some first information about its use.

6 References

1. Gartner Inc.: 80 Percent of Active Internet Users Will Have A "Second Life" in the Virtual World by the End of 2011. http://www.gartner.com/it/page.jsp?id=503861 (2007)
2. Mitham, N.: Virtual Goods: Good for Business? Journal of Virtual Worlds Research Vol 2, No 4 https://journals.tdl.org/jvwr/article/view/864/629 (2010)
3. Sivan, Y.: 3D3C Real Virtual Worlds Defined: The Immense Potential of Merging 3D, Community, Creation, and Commerce. (2008)
4. Rymaszewski, M., Au, W.J., Wallace, M., Winters, C., Ondrejka, C., Batstone-Cunningham, B., Rosedale, P.: Second Life: The Official Guide. SYBEX Inc. (2006)
5. Chesney, T., Noke, H.: Virtual World Commerce: An Exploratory Study. SSRN eLibrary http://ssrn.com/paper=1286036 (2008)
6. ENISA: Virtual Worlds - Real Money. European Network and Information Security Agency, http://www.enisa.europa.eu/media/press-releases/2008-prs/virtual-worlds-real-money (2008)
7. Linden Lab: 2009 End of Year Second Life Economy Wrap up, https://blogs.secondlife.com/community/features/blog/2010/01/19/2009-end-of-year-second-life-economy-wrap-up-including-q4-economy-in-detail
8. Linden Lab: Economic Statistics, http://secondlife.com/statistics/economy-data.php
9. Wagner, J.: Top Second Life Entrepreneur Cashing Out US$1.7 Million Yearly; Furnishing, Events Management Among Top Earners., http://nwn.blogs.com/nwn/2009/03/million.html
10. Stephenson, N.: Snow Crash. Bantam Books (1992)
11. Jakobs, K.: Real Standards for Virtual Worlds, Why and How? Journal of Virtual Worlds Reserach Vol 2, No 3 https://journals.tdl.org/jvwr/article/view/717/517 (2009)
12. Childers, B.: Run your own virtual reality with OpenSim. Linux Journal 2009, 6, (2009)
13. Fishwick, P.A.: A Introduction to OpenSimulator and virtual environment agent-based M&S applications. Proceedings of the 2009 Winter Simulation Conference,
14. LindenLabs: 1 Billion Hours, 1 Billion Dollars Served: Second Life Celebrates Major Milestones for Virtual Worlds. http://lindenlab.com/pressroom/releases/22_09_09 (2009)
15. Schrank, D., Gütl, C., Kappe, F.: Design and Implementation of a Payment System for Virtual Worlds. WASET 2010, Paris, France (to appear)
16. LindenLabs: Second Life: Terms of Service, http://secondlife.com/corporate/tos.php
17. VirWoX: The Virtual World Exchange, http://www.virwox.com