SIMP2P: Uma estratégia P2P de distribuição de texturas em Mundos Virtuais 3D

Marcelo Santos, Carlos Kamienski

Universidade Federal do ABC (UFABC)
Caixa Postal 15.064 – 91.501-970 – Santo André – SP – Brazil

{marcelo.batista,cak}@ufabc.edu.br

Resumo. *Mundos Virtuais Sociais permitem a criação de conteúdo dinâmico com personalização de imagens e alto grau de imersão com ambientes compartilhados em tempo real. No entanto, problemas no aumento do número de usuários conectados simultaneamente e a complexidade do ambiente compartilhado acarretam uma exigência maior de recursos de hardware e rede, gerando facilmente um gargalo no servidor. Propomos uma abordagem de compartilhamento P2P, utilizando o servidor de código livre OpenSim, implementando e avaliando uma solução que distribuí texturas de objetos, reduzindo o tráfego enviado pelo servidor e o tempo médio de transferência.*

Abstract. *Social Virtual Worlds allow users to dynamically create content such as customized images and offer a high degree of immersion with shared real time environments. However, some problems adversely affect their performance such as a limitation in the number of simultaneous users and the complexity of the shared environment, which impose a higher burden on network traffic and hardware resources of servers, thus making the server a bottleneck of the system. Here we propose and analyze the performance of a P2P architecture for virtual worlds using the OpenSim open source server for sharing textures among clients. Compared with the client/server approach, SimP2P significantly reduces the traffic sent by the server as well as the average download time for textures for the clients.*

1. Introdução

Nos últimos anos, a melhoria das tecnologias de rede e o aumento da capacidade de processamento dos computadores impulsionaram fortemente a utilização de ambientes virtuais. Estes ambientes proporcionam alto grau de imersão e interatividade que satisfazem condições gerais para uma experiência de comunicação, socialização e aprendizado, tornando os Mundos Virtuais um ambiente com enorme diversidade de aplicabilidade de acordo com a necessidade e imaginação do usuário. Essas características posicionam os mundos virtuais como um serviço de enorme potencial de crescimento, ao ponto de talvez em alguns anos, possuir um avatar (representação digital do usuário no mundo virtual) seja tão comum quanto possuir um endereço de e-mail ou conversar por um programa de mensagem instantânea.

Como ambientes virtuais estão se tornando cada vez mais populares, conseqüentemente a participação de usuários aumentou substancialmente, atingindo centenas de milhares de pessoas. Esses sistemas são tipicamente desenvolvidos usando a arquitetura Cliente/Servidor (C/S) que apresenta várias desvantagens técnicas e comerciais, especificamente na área de confiabilidade, custos de hardware, largura de

banda da rede, alto custo de processamento e manutenção. No entanto, esse modelo prevalece atualmente devido à facilidade de garantir a consistência do mundo, isolar a carga de processamento e simplificar a troca e atualizações de mensagens entre clientes [Lu et al. 2009]. Mesmo que tradicionalmente os sistemas para ambientes virtuais comerciais sejam baseados na arquitetura cliente/servidor, como o Second Life [Second Life 2010], essa abordagem freqüentemente não é considerada ideal para suportar uma carga de trabalho crescente. O OpenSimulator (ou OpenSim) [OpenSimulator 2010] é um servidor de mundos virtuais de código aberto que pode ser usado para construir um mundo virtual no estilo do Second Life. Consequentemente é também baseado na arquitetura C/S.

Jogos populares 3D do tipo MMOG (*Massively Multiplayer Online Game*), possuem boa escalabilidade quando comparado ao Second Life (SL) e OpenSim. Isso se deve a capacidade de criação dinâmica de conteúdo pelos usuários no SL e OpenSim, onde o cliente não possui todos os dados sobre os personagens e objetos do ambiente virtual em sua máquina, mas requisita-os a um servidor centralizado, de modo que os próprios usuários podem construir regiões do mundo da maneira como lhes aprouver em tempo real em um ambiente compartilhado. O custo dessa abordagem é a sobrecarga adicional nos recursos de hardware do servidor e de tráfego na rede entre cliente e servidor.

Esse trabalho apresenta uma proposta e avaliação de uma solução P2P para o servidor de mundos virtuais OpenSim onde as texturas dos objetos podem ser trocadas entre os clientes, o que diminui a sobrecarga no servidor e no seu enlace de saída com a Internet. Texturas são as imagens que constituem a característica externa dos objetos e que geram a maior parte do tráfego entre servidor e clientes [Liang et al. 2008]. A solução adotada é baseada numa abordagem P2P onde o servidor atua como um semeador de texturas que na maioria das vezes são trocadas entre os clientes. O estudo se deve à necessidade empírica encontrada de aprimorar o desempenho e a escalabilidade do servidor OpenSim. Os resultados mostram que utilizando uma rede P2P para distribuição de texturas há uma redução na utilização dos recursos de rede do servidor que chegam a 40%. O tempo médio de transferência das texturas no cliente comparado com a arquitetura C/S obteve uma melhora significativa de aproximadamente 30 segundos.

O restante do artigo é organizado da maneira descrita a seguir. Na seção 2 são citados os trabalhos relacionados. Na seção 3 há uma breve discussão sobre Mundos Virtuais. Na seção 4 são esclarecidos tópicos sobre o funcionamento do OpenSim e sua arquitetura. A seção 5 apresenta a estratégia de distribuição SimP2P. A seção 6 discute a metodologia. Na seção 7 são apresentados os resultados. Na seção 8 discutimos os resultados. Apresentamos as conclusões na seção 9. Por fim são citadas as referências.

2. Trabalhos Relacionados

Um dos trabalhos com grande importância para o desenvolvimento do projeto do SimP2P foi a análise feita do impacto das texturas no ambiente do SL [Liang et al. 2008], onde análises mostram que 61% a 88% do tráfego total no cliente foi gerado apenas por solicitações de texturas ao servidor. Regiões como Isis possui mais de 3 mil texturas que totalizam aproximadamente 335 MB de dados. Este trabalho mostrou o potencial da exploração do compartilhamento de texturas por meio de uma rede P2P.

Vários trabalhos tentam definir uma arquitetura P2P eficiente para mundos virtuais, mas as arquiteturas propostas são complexas, tentando prover escalabilidade desde a troca de informação pela rede quanto processamento distribuído [Lu et al. 2004][Hampel et al. 2006][Douglas et al. 2005][Lu et al. 2009][Fan et al. 2007]. As arquiteturas propostas exigem praticamente uma nova implementação de todo o software existente de mundos virtuais como o SL e OpenSim, tanto do lado cliente quanto servidor. Todas as propostas utilizam resultados baseados apenas em simulações devido à complexidade de aplicação. Há mudanças na forma de divulgação de eventos do mundo, atualização de posicionamento, forma de controle de área de interesse e persistência do mundo virtual, mas nenhum trabalho deu atenção especificamente à distribuição de texturas na rede por meio de uma rede P2P em malha.

3. Mundos Virtuais

Mundos Virtuais são ambientes eletrônicos que emulam espaços físicos complexos através de representações de ambientes simulados, contendo objetos e interações com seus avatares (personificação do usuário no mundo virtual) [Mennecke et al. 2008]. Os mundos virtuais sempre foram utilizados nas áreas de jogos e simuladores, mas atualmente as pessoas têm vislumbrado outras possibilidades, como socialização, educação e, obviamente, entretenimento.

Particularmente, para este trabalho, são de grande interesse os mundos virtuais sociais, também chamados de MMOW (*Massively Multiplayer Online World*) que são uma ramificação dos ambientes conhecidos como MMOG, mas que permitem grande flexibilidade ao usuário para exercer diversas atividades independentes de níveis ou estágios presentes em jogos [Kamienski et al. 2008]. Esses mundos virtuais apresentam diversos problemas técnicos, como a escalabilidade do número de avatares em uma mesma região, exigência de recursos para processamentos gráficos, reatividade das informações, segurança, tolerância a erros e persistência dos dados em um ambiente que está em constante mudança, dificultando sua distribuição por meio de uma tecnologia P2P.

O Second Life (SL) é atualmente o mais popular MMOW, obtendo números impressionantes como um pico de 88.200 avatares conectados simultaneamente em diferentes regiões [Blog SecondLife 2010]. Trabalhos recentes mostram que um cliente que se conecta ao Second Life utiliza em momentos de pico 350 Kbps de largura de banda em uma região popular quando o avatar está andando [Antonello 2008][Kinicki et al. 2008]. Além dos fatores técnicos, limitações financeiras para uso do SL impedem que haja uma maior popularização e utilização de toda sua capacidade devido a cobranças de taxas sobre certos serviços oferecidos, desde a taxação para disponibilização de um terreno para construção de alguns objetos até mesmo envio de imagens.

A popularização do uso de mundos virtuais envolve a facilidade de acesso às tecnologias do cliente e do servidor, assim como permitir o acesso gratuito baseado em plataformas abertas [Kamienski et al. 2008]. Foi desenvolvido um servidor alternativo de mundos virtuais de código aberto chamado de OpenSimulator, conhecido como OpenSim [OpenSimulator 2010], que permite a criação de um ambiente similar ao Second Life. A sua compatibilidade é estendida ao software cliente do SL (chamado de visualizador), pois o OpenSim surgiu a partir da engenharia reversa do Second Life. Para o desenvolvimento da estratégia proposta utilizamos o OpenSim como servidor de

Mundo Virtual pois o lado servidor do SL não é disponibilizado para se fazer modificações, testes ou medições.

4. OpenSim

O OpenSimulator (OpenSim) é um conjunto de servidores distribuídos que podem ter o seu código fonte modificado sem nenhuma restrição. A partir da instalação do servidor pode-se acessar o mundo virtual por softwares clientes como o Hippo [Hippo 2010] ou até mesmo o cliente do Second Life. O projeto do OpenSim está em desenvolvimento constante e a versão atual (0.7.2) apresenta grande estabilidade para a maioria de recursos típicos de mundos virtuais como o SL. Embora o projeto do OpenSim tenha sido iniciado em 2007, ele já possui características que o tornam uma opção adequada para várias aplicações de mundos virtuais, como: a) o software é gratuito, não sendo necessário pagar por nenhum módulo ou funcionalidade; b) capacidade de criação de conteúdo em tempo-real; c) suporte as linguagens de programação OSSL, LSL, C#, JavaScript e Visual Basic .NET; d) flexibilidade na distribuição de recursos em várias máquinas servidoras.

Alguns conceitos são importantes para entender a estrutura de um mundo virtual no OpenSim e no Second Life. **Região** é uma parte do mundo virtual que por padrão tem o tamanho de 256x256 metros. **Software Cliente** é utilizado para se conectar no mundo virtual, como o próprio cliente do Second Life. **Grid** é o conjunto de várias regiões interligadas de tamanho de 256x256 metros, podendo essas regiões ser vizinhas ou não entre si. **Primitiva** é a menor representação única de um objeto. **Textura** é a imagem na face de uma primitiva. **UUID** é um identificador único de 128bits que é usado para identificar avatares, objetos, texturas e regiões.

O OpenSim é baseado na arquitetura cliente/servidor, onde o servidor mantém todas as informações sobre o mundo virtual e envia as atualizações para os clientes que transformam os dados em uma realidade tridimensional em tempo-real. O OpenSim pode funcionar de dois modos: *Standalone* ou *Grid* (Figura 1). O modo *standalone* é limitado a um número pequeno de usuários, pois um único processo é responsável por toda a simulação. O modo Grid tem o potencial de ser mais eficiente conforme o número de usuários cresce, pois é possível distribuir serviços em máquinas diferentes.

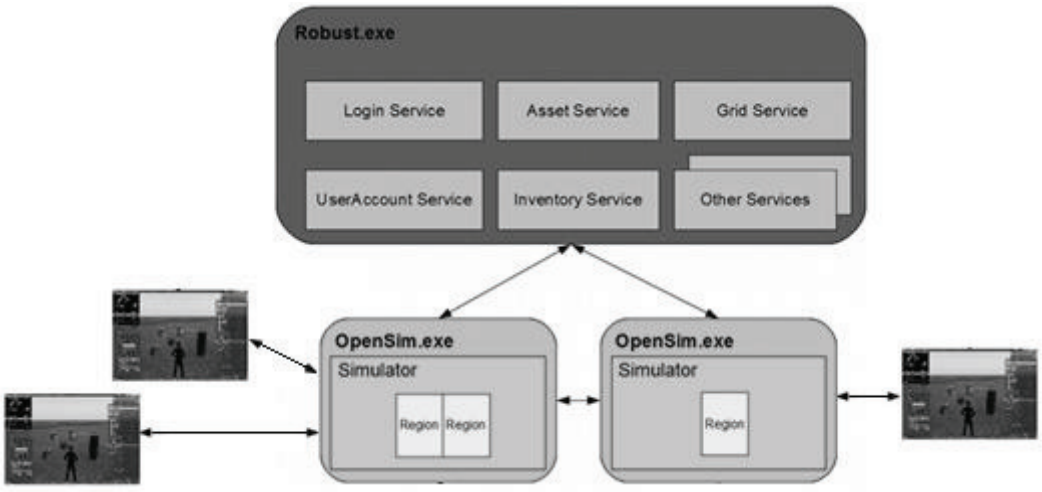


Figura 1 – Arquitetura OpenSim 0.7.2

O servidor **ROBUST** (*Redesigned OpenSim Basic Universal Server Technology*) [Robust 2010] substituiu os servidores Asset, Inventory e Grid das versões anteriores do OpenSim. O ROBUST é encarregado da autenticação das regiões e sua interligação, consultas no banco de dados para sons, texturas, imagens, scripts e objetos do inventário do avatar. O servidor **OpenSim**, também conhecido como servidor de região, é responsável por processar as interações físicas do mundo e avatares, executar scripts e controlar os objetos do mundo, exceto durante a fase de *login*, o cliente interage exclusivamente com o servidor OpenSim que pode ainda solicitar serviços ao servidor ROBUST (Figura 1).

5. A Arquitetura SimP2P

Dado o grande volume de tráfego gerado pela solicitação de texturas ao servidor do Mundo Virtual, definimos como SimP2P a estratégia que distribuiu texturas entre *peers*, diminuindo significativamente a quantidade de tráfego requisitada ao servidor e melhorando o tempo de resposta entre os clientes. O desafio fundamental na adaptação de um MMOW convencional para uma arquitetura P2P é cumprir as funcionalidades dos servidores centralizados em uma forma distribuída. Isto implica enfrentar duas questões essenciais [Lu et al. 2009]: 1) Dificuldades de gerenciamento de áreas de interesse de comunicação; 2) Persistência e consistência do mundo virtual.

A proposta da arquitetura SimP2P é baseada principalmente em duas alterações na arquitetura C/S do OpenSim. A primeira modificação feita para que a estratégia P2P seja eficiente, foi modificar a forma como o cliente requisita texturas, através da implementação de uma área de interesse (AI). Baseado no funcionamento do SL onde o servidor envia apenas informações de texturas, objetos e avatares num raio de 64 metros de uma circunferência [Liang et al. 2008], foi desenvolvido um cliente que se conecta ao servidor OpenSim requisitando apenas texturas na sua AI de forma semelhante ao SL. O padrão do servidor OpenSim envia informações de tudo que está dentro da região, ficando optativo ao cliente se deve ou não solicitar detalhes dos objetos, podendo solicitar texturas que talvez o usuário não necessite visualizar.

Visando tratar a questão da escalabilidade do ponto de vista de rede, foi utilizado um algoritmo baseado numa arquitetura de conexões em malha, redundante a falhas que se adapta bem a ambientes dinâmicos como o de mundos virtuais.

5.1. Arquitetura

Na arquitetura SimP2P (Figura 2) o servidor OpenSim se comporta como o rastreador (*tracker*) que fornece as informações sobre os *peers* engajados em um enxame (*swarming*), componentes semelhantes aos utilizados no modelo BitTorrent [BitTorrent 2010]. Inicialmente o *peer* solicita ao servidor de região a **lista de peers** disponíveis (e solicita uma nova lista periodicamente), mantendo assim uma lista atualizada de todos os avatares conectados na região que são obrigatoriamente ao mesmo tempo os *peers*. A partir da lista de *peers* o **gerenciador de parceiros** escolhe aleatoriamente, com uma distribuição uniforme, cinco *peers* para a abertura de uma conexão para solicitação de eventuais texturas que estejam na sua área de interesse. A troca de parceiros ocorre somente em caso de desconexão de algum deles. Após contatar o **distribuidor de texturas** de um dado *peer*, é recebida a porta disponível para que o módulo cliente (implementado como uma *thread*) estabeleça uma conexão para a transferência de texturas com o módulo servidor. O **distribuidor de texturas** ouve numa porta padrão

requisições feitas por algum **gerenciador de parceiros** de um *peer*, abrindo uma *thread* em uma porta disponível para que seja enviada uma ou mais texturas (Figura 3).

O *peer* mantém uma **lista de texturas recebidas** para compartilhamento com outros *peers*, além de manter uma **lista de texturas pendentes** e uma lista de todas as **texturas da região** informada assim que ele conecta-se. As texturas serão solicitadas quando necessário, ou seja, o *peer* sabe onde está a posição do objeto no mundo e suas texturas, mas ele não a solicita até estar na sua área de interesse. Após estabelecer uma conexão com outro *peer*, a **lista de texturas pendentes** é percorrida e enviada a todos aos parceiros, o que responder primeiro é que envia a textura pedida, várias texturas são transferidas em paralelo de acordo com a quantidade de conexões estabelecidas.

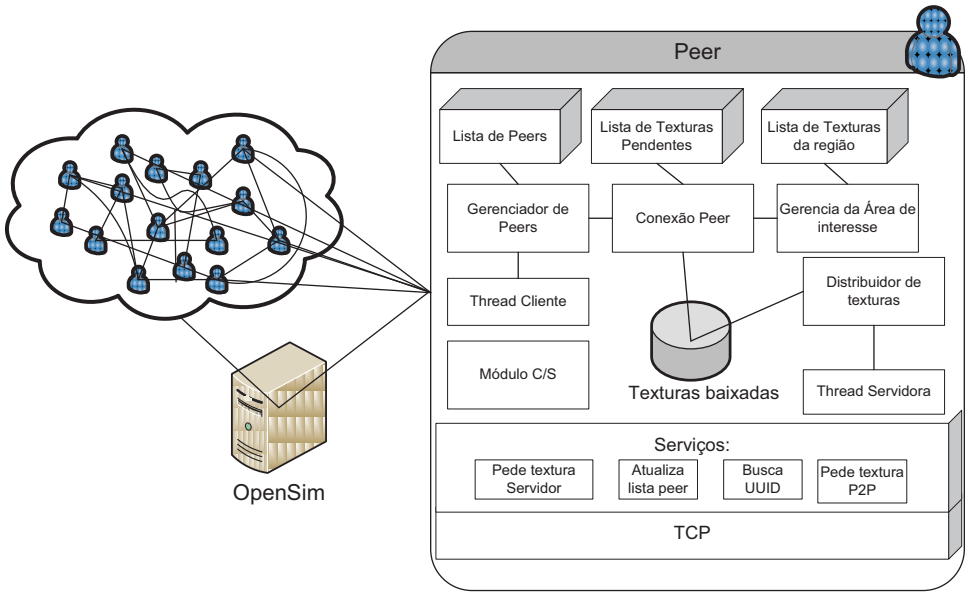


Figura 2 – Arquitetura SimP2P

As conexões entre os parceiros não são obrigatoriamente de duplo sentido, ou seja, um *peer* "A" pode apenas requisitar textura para o *peer* "B", e o *peer* "B" requisita texturas para outro peer "C" e não para "A". A Figura 3 apresenta a seqüência para o estabelecimento das conexões entre os *peers*.

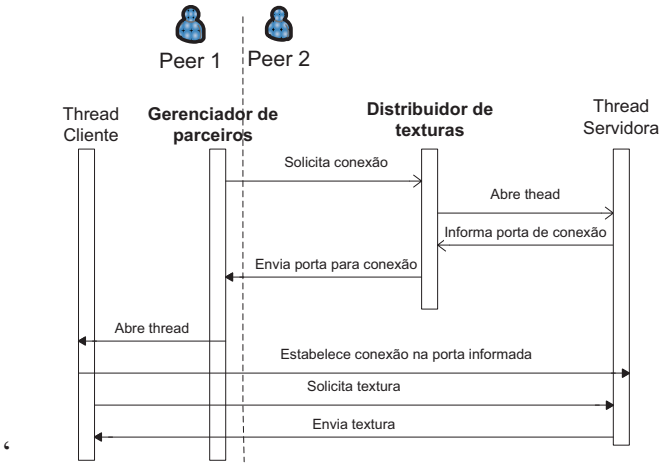


Figura 3- Sequência de estabelecimento da conexão entre *peers*

Para manter a persistência do mundo virtual, as informações de objetos são enviadas para o servidor da forma tradicional, mas as solicitações são feitas primeiro para rede P2P, mantendo a consistência do mundo para o cliente que caso solicite uma textura para 3 dos 5 peers e esses não a tenham, então a textura é solicitada pelo **módulo C/S** diretamente ao servidor OpenSim. Assim pode-se garantir que as texturas serão obtidas do servidor no início quando não estão disponíveis em nenhum *peer* e também diminui a dependência da rede P2P. Em outras palavras, a arquitetura SimP2P se comporta como C/S na ausência de peers que possam trocar texturas.

Durante uma conexão, os clientes mantêm as texturas em memória e não requisitam novamente. Quando um objeto é modificado um novo ID é criado para cada textura nova, ou seja, o sistema está preparado para modificações dinâmicas e não ocorre inconsistência de *cache*. Modificações em uma região não ocorrem com muita frequência e por isso esse aspecto não foi abordado no artigo.

6. Metodologia

A arquitetura tradicional C/S do OpenSim é comparada com a estratégia proposta de distribuição de texturas por meio da rede P2P. Para manter a comparação mais próxima da realidade, ou seja, de como um cliente humano realmente se comporta exigindo recursos do servidor, foi preciso caracterizar padrões de comportamento de um avatar e utilizar um ambiente de rede heterogêneo que são descritos nas seções seguintes.

6.1. Coleta de dados no Second Life

É importante sabermos quanto tempo uma avatar permanece conectado, pois isto interfere diretamente na estratégia P2P avaliada. As texturas dos objetos são solicitadas uma única vez durante uma conexão. Sendo assim, o tráfego de saída do servidor e de entrada do cliente é maior no período de solicitação das texturas, pois o restante do tráfego é de controle de movimentação e informações de consistência do Mundo Virtual. Portanto foi desenvolvido um software cliente para simular ações humanas (*bot*) no ambiente do Mundo Virtual através da coleção .Net de bibliotecas escritas em C# para interação em ambientes virtuais que é compatível tanto com o Second Life quanto OpenSim, conhecida como libOpenMetaverse [LibOpenMetaverse 2010].

Foram realizadas coletas no SL por ser um ambiente popular no qual o OpenSim foi baseado. A coleta foi realizada durante aproximadamente um mês em quatro regiões que proporcionam diferentes formas de entretimento para o usuário (Tabela 1). O *bot* coletou informações do tempo de permanência de todos os avatares conectados simultaneamente com o *bot*. O tempo de duração da coleta em cada região foi distinto devido ao mecanismo de detecção e desconexão de *bots*, onde a região bane o avatar quando ele é considerado um *bot*, sendo este controle mais rigoroso na região *Sexy Island* e *Moose Beach*.

Tabela 1 – Descrição das regiões coletadas no SL

Região	Duração média da conexão	Descrição	Duração da coleta
Dance Island	25.12 minutos	Região com uma pista de dança, onde os avatares tocam objetos que os fazem dançar.	180 horas
Brasil	23.34 minutos	Região que os avatares participam de um jogo RPG onde cada um pertence a um clã e representa um personagem que pode duelar com outros.	240 horas

Sexy Island	15.73 minutos	Região com conteúdo adulto. Avatares se reúnem param simular práticas num mundo semelhante a uma praia.	85 horas
Moose Beach	11.48 minutos	Região que retrata um ambiente semelhante a uma praia para avatares interagirem.	93 horas

6.2. Modelo de tempo de permanência

Os dados coletados no SL, descritos na seção anterior, revelaram que o ambiente da região influencia no tempo médio de permanência do avatar e que 65% ou mais, de cerca de 30.000 avatares detectados, passaram menos de 20 minutos conectados na região (Figura 4).

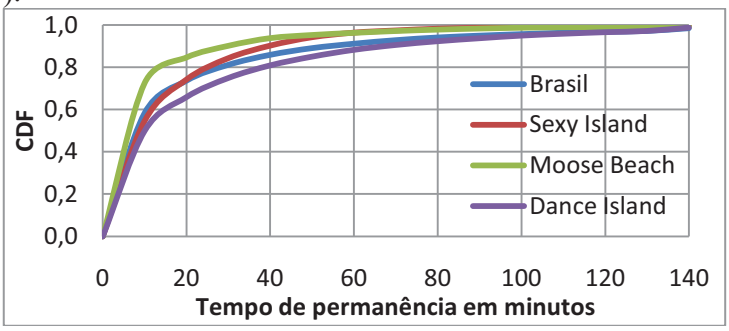


Figura 4 – Distribuição acumulada de probabilidade do tempo de permanência por avatar em regiões distintas do Second Life.

Baseado nos resultados obtidos com o teste Kolmogorov-Smirnov, o tempo de permanência dos avatares foi modelado com a distribuição exponencial.

6.3. Padrão de movimentação

Pela falta de um modelo de movimentação existente, foi utilizado uma cadeia de Markov (Figura 5) de dois estados para que o avatar alterne entre ações de parar e andar, semelhante a um cliente humano, com 50% de chance de manter o mesmo estado ou alternar. Estudos mostram que o avatar passa mais tempo parado do que andando [Varvello et al. 2008], por isso o tempo de permanência no estado andando é gerado por uma distribuição uniforme que varia entre 20 e 45 segundos enquanto o estado parado varia entre 60 e 120 segundos.

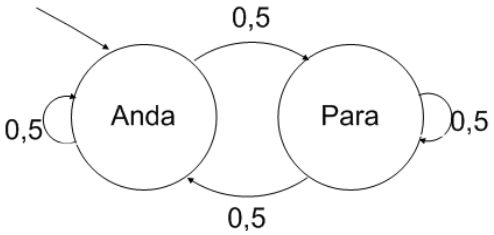


Figura 5 – Cadeia de dois estados que modela movimento de um avatar

6.4. Clientes e a rede PlanetLab

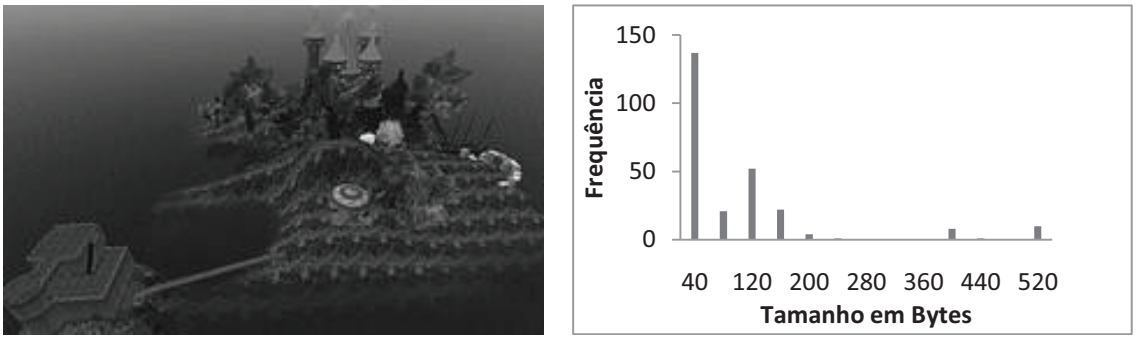
Os experimentos foram realizados no laboratório de redes virtual PlanetLab (PL), desenvolvido para o estudo de novas tecnologias e protocolos de rede. Foram utilizadas 100 máquinas localizadas em diferentes partes do mundo para representar o lado cliente conectado ao servidor de mundos virtuais, cada máquina representou um avatar. Como as máquinas no PL possuem recursos compartilhados o servidor OpenSim ficou localizado na Universidade Federal do ABC numa máquina dedicada fora da rede

PlanetLab. Assim podemos avaliar o comportamento do servidor de forma precisa em uma máquina robusta que oferece tanto uma boa largura de banda disponível com ótima configuração de hardware.

Cada máquina armazena dados de pedidos de texturas no banco de dados MySQL e captura o tráfego de rede com o tcpdump. Durante o experimento as máquinas utilizadas do PlanetLab foram monitoradas e não foram detectados problemas em relação a disponibilização recursos de hardware e rede, pois foram escolhidas máquinas com grande quantidade de memória, disco e capacidade de processamento livre.[1] O servidor utilizado possui sistema operacional Linux - openSUSE 11.2-64bits com 8GB de memória RAM e um processador Intel Xeon Quad Core X3430 2.40GHz e um link compartilhado de 1Gbps.

6.5. Cenário

Utilizando a máquina servidora descrita anteriormente, configuramos e utilizamos a versão 0.7.2 do OpenSim com a região Fairie Castle [Mundos virtuais para OpenSim 2010] (Figura 6). O número de primitivas contidos na região é de 2700, no entanto o número de texturas é de 255, pois é possível ter objetos diferentes com texturas iguais. Se somarmos o tamanho de todas as texturas obtemos 19 MB, que é um valor pequeno comparados a regiões populares no Second Life que possuem regiões que chegam a ter mais de 1500 texturas que contabilizam mais 200 MB de dados [Liang et al. 2008]. É importante ressaltar que os clientes solicitam apenas uma única vez uma mesma textura durante cada conexão.



a) Visão da Região Fairie Castle b) Histograma do tamanho das texturas

Figura 6 – Região Fairie Castle

Pouco mais de 50% das texturas desta região são menores que 40 bytes e possuem tamanho máximo de 512 bytes.

6.6. Plano de trabalho

O objetivo do experimento é analisar o ganho de um estratégia P2P aplicada à distribuição de texturas em um ambiente virtual com o OpenSim, analisando o impacto tanto no cliente quanto servidor variando a taxa de permanência do cliente. O cliente alterna sempre entre dois estados de On e Off, com tempos determinados pela distribuição exponencial. Mantemos o tempo de On e Off proporcionais para que a quantidade de avatares conectados ao mesmo tempo fosse sempre a mesma, em média,

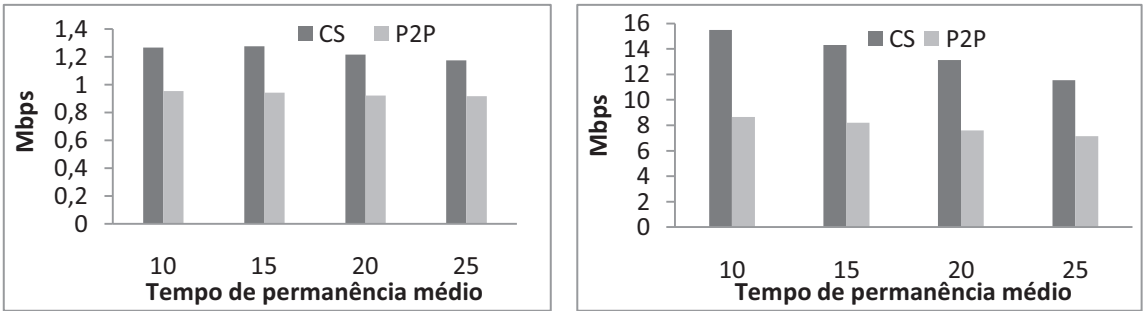
[1] Os experimentos foram refeitos várias vezes até que um conjunto de servidores estáveis e com recursos de hardware adequados fossem encontrados.

em cada experimento. Foram realizados experimentos com 6 horas de duração contínua onde foram desprezados os primeiros 30 minutos para que se tivesse uma coleta sem um período transiente de adaptação da rede. Consideramos o período de 10 minutos como uma replicação, pois não foi observada variação excessiva nesse intervalo no cliente. Assim considerando apenas as 5,5 horas temos 33 replicações para cada caso. Foram calculados os intervalos de confiança, mas as barras de erros não são mostradas nos gráficos devido a pouca variabilidade encontrada. Os tempos médios de permanência foram variados entre 10, 15, 20 e 25 minutos, acompanhados pelo respectivo tempo de Off 2, 3, 4 e 5. Foi executado tanto a abordagem C/S quanto P2P com uma área de interesse de 64 metros de raio e as mesmas 100 máquina configuradas no PlanetLab. Quando um mesmo usuário alterna entre estados de ON e OFF não é utilizado o cache da conexão anterior.

7. Resultados Obtidos

7.1. Visão do Servidor

A Figura 7 mostra resultados para a vazão média de entrada e saída no servidor. Pode-se observar que a abordagem P2P provoca uma redução substancial no tráfego de texturas enviadas do servidor para o cliente, que em média se situa na faixa de 40%. O tráfego de entrada no servidor também diminui, mas não de maneira tão acentuada como na saída de dados, pois os clientes precisam solicitar muitas texturas ao servidor e apenas enviar informações de controle de movimentação.



a) Vazão de entrada b) Vazão de saída

Figura 7 - Vazão média no lado do servidor

Outra observação importante é a diminuição do tráfego de saída gerado por texturas à medida que o tempo de permanência (On) dos avatares aumenta. Isso se deve ao fato de que quando passam muito tempo em uma região os avatares frequentemente obtem todas as texturas que necessitam e então praticamente param de solicitar essas informações, pois já solicitaram boa parte das texturas do ambiente. Como as texturas não costumam ser alteradas com muita frequência, um maior tempo de permanência dos avatares gera menor carga na rede e no servidor. O tráfego de entrada não é afetado significativamente pelo tempo de permanência, pois consiste apenas do envio de informações de atualização de posicionamento e sobrecarga de pedidos de solicitações de texturas por parte do cliente.

7.2. Visão do Cliente

Analisando o cliente que se conecta ao servidor, ou seja, o *bot* desenvolvido, que simula o comportamento de um avatar, observa-se novamente ganhos significativos. A

economia de recursos da rede no lado do servidor é compensada por um aumento do tráfego de entrada e saída dos clientes, porque eles trocam texturas entre si. No entanto, o aumento de tráfego é bem dividido entre os clientes pelo algoritmo P2P e os aumentos não são significativos tanto na entrada quanto na saída, podendo ser suportados por clientes conectados através de enlaces ADSL por exemplo.

Resultados da observação da vazão média de entrada e saída para vários tempos de permanências são mostrados na Figura 8. É possível observar que ocorre um aumento leve do tráfego de entrada nos clientes (de não mais que 20%), mas que não ultrapassa 220 Kbps. É extremamente importante observar que embora haja uma grande redução de pedidos ao servidor, não ocorre sobrecarga na taxa de *upload* nos peers. O tráfego de saída aumenta substancialmente, chegando a quadruplicar, mas a média do tráfego de saída não ultrapassa a taxa de 50 Kbps. Observa-se também que o aumento do tempo de permanência gera diminuição do tráfego do cliente, como já foi mostrado para o servidor. Há uma redução visível do tráfego de entrada para tempo de permanência de 10 e 15 minutos, mas com tempos maiores a redução é menos intensa.

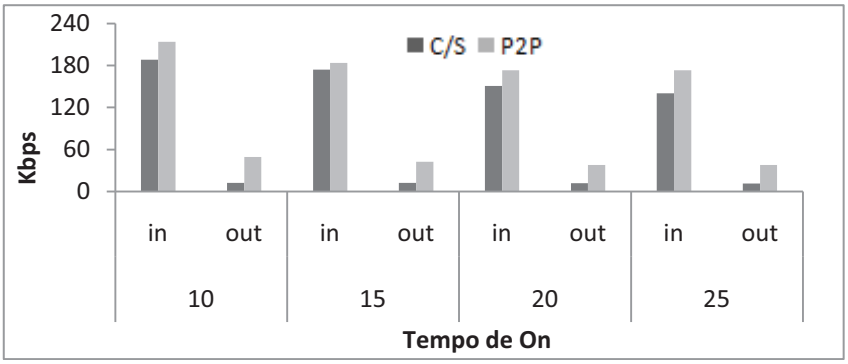
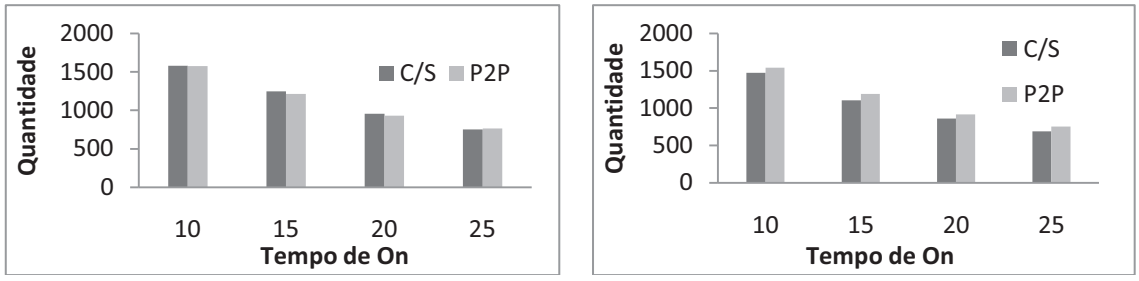


Figura 8 – Tráfego de entrada e saída no cliente

A Figura 9 mostra a quantidade média de texturas solicitadas e recebidas por clientes comparando as versões C/S e P2P para diferentes tempos de permanência. É possível observar que a abordagem C/S, em alguns casos, solicita um número maior de texturas (que não representa 2% de aumento). Isso ocorre devido à dificuldade de replicar um mesmo experimento em um cenário real distribuído na Internet com máquinas compartilhas por vários usuários (Planetlab). Já no caso das texturas recebidas a abordagem P2P possui um aumento significativo em relação a abordagem C/S pois o tempo para recebimento de uma textura por meio da rede P2P em média é menor que 1 segundo, enquanto que utilizando a arquitetura tradicional do OpenSim esse tempo chega perto de 40 segundos (Figura 11). É mais provável que uma textura solicitada seja recebida pela abordagem SimP2P do que C/S antes que usuário entre em estado de OFF, resultando em um aumento de texturas recebidas na abordagem P2P (Figura 9b).



a) Texturas solicitadas b) Texturas recebidas

Figura 9 – Quantidade de texturas recebidas e solicitadas (média por cliente)

Analisando apenas quando utilizamos a estratégia SimP2P, vemos que a Figura 10 mostra que em geral cerca de 80% das texturas são trocadas entre os peers clientes e o restante ao servidor, o que demonstra a eficiência da solução desenvolvida. O tempo de permanência não afeta significativamente esses dados, mesmo que a cada nova conexão de um cliente ele se comporte como um novo avatar não havendo cache das texturas solicitadas anteriormente.

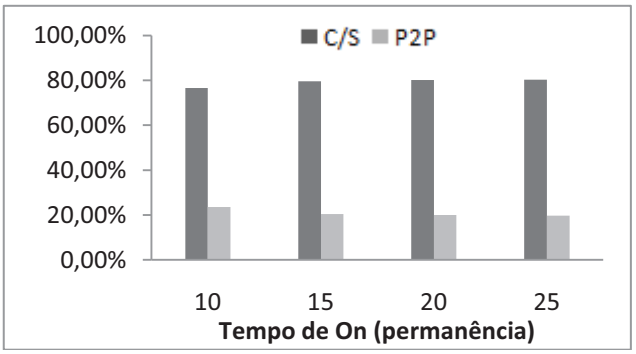
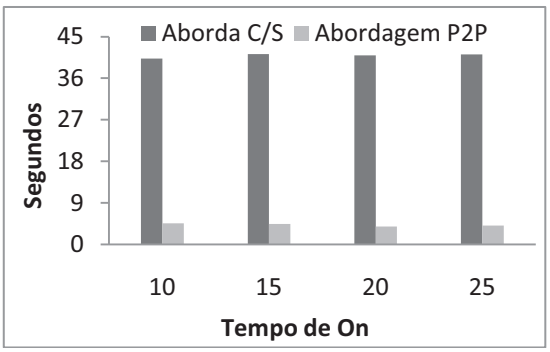
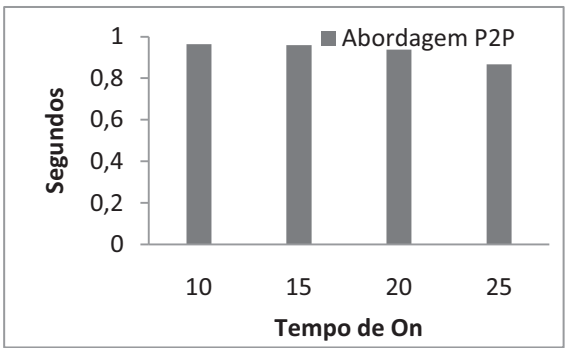


Figura 10 – Porcentagem de texturas obtidas pelo servidor ou pela rede P2P

O tempo de resposta ao usuário sofre uma diminuição de enorme vulto com a abordagem P2P, medido pelo tempo médio para transferências das texturas (Figura 11). A Figura 11a mostra a enorme redução do tempo médio de transferência de texturas entre o servidor e os clientes na abordagem P2P comparada com a abordagem C/S. É importante lembrar que na abordagem P2P o servidor também tem um papel ativo na transferência de texturas. Enquanto que na abordagem C/S o tempo de transferência atinge valores altos como 40 segundos, na abordagem P2P o tempo fica em torno de 5 segundos. Embora possa parecer um tempo excessivo, nesses ambientes os usuários estão acostumados com transferências parciais de texturas. Algumas vezes até as texturas dos avatares são impactadas e os usuários os vêem como uma figura totalmente branca ou apenas a forma de uma fumaça enquanto é feito a transferência.



a) Transferência entre servidor e clientes



b) Transferência entre os peers

Figura 11 – Tempo médio de transferência de texturas solicitadas

Por outro lado, somente 20% das texturas são obtidas do servidor, sendo as 80% restantes obtidas dos *peers* com tempo médio de 1 segundo (Figura 11b). Esse resultado mostra que além de reduzir a demanda no servidor e aumentar pouco o tráfego entre os clientes, a Qualidade de Experiência (QoE) é significativamente melhor com o uso da solução P2P. Esses resultados também indicam que para um cenário com número maior de texturas, o resultado terá uma grande probabilidade de ser ainda melhor.

O tempo médio de transferência é praticamente 40 vezes menor na abordagem P2P (Figura 11), aumentando a qualidade de experiência vivenciada pelo cliente, pois o mundo a sua volta carregará mais rápido. Observa-se ainda que o tempo médio das texturas solicitadas ao servidor na abordagem P2P é inferior quando comparado com as texturas pedidas diretamente ao servidor na abordagem tradicional C/S, devido a exigência grandes de recursos da rede e gerenciamento do próprio simulador.

8. Discussão

A abordagem P2P representada pela arquitetura SimP2P mostrou-se mais eficiente em relação a abordagem C/S, em aspectos como taxa de transferência, quantidade de texturas recebidas e duração do tempo de transferência. Embora a idéia seja relativamente simples, a redução de tráfego no servidor foi expressiva. Mesmo havendo variação no tempo de permanência do avatar os ganhos se mantiveram em torno de 40%. É importante ressaltar que ambas as abordagem utilizaram o *bot* com a implementação de um algoritmo que gerencia a área de interesse do avatar, sendo a análise realizada extremamente justa entre C/S e P2P, pois se usou a mesma semente geradora de número aleatórios desde do tempo de permanência até os mesmos movimentos dos avatares.

Poucas mudanças no servidor são necessárias para que a rede SimP2P seja formada. Na implementação realizada foi preciso apenas fazer com que o servidor enviasse uma lista de peers para outros peers. No entanto, o anonimato permanece como uma questão em aberto. Os mundos virtuais são atraentes para várias pessoas pela ausência de leis, regras sociais cabíveis de punição ou até mesmo a distância da sua vida real, de modo que talvez essas pessoas não aprovariam ter seu avatar identificado por um endereço IP.

A análise do ponto de vista do cliente e servidor revelou que a sua vazão é influenciada pelo tempo de permanência do cliente. Havendo uma redução da média do volume de dados gerado quando há um aumento no tempo de permanência pelo fato de uma textura ser pedida uma única vez durante toda a conexão. Ou seja, o cliente solicita uma grande quantidade de texturas, mas passa muito tempo sem solicitá-las quando seu tempo de conexão é muito alto. Quando o cliente para de solicitar texturas o tráfego gerado é devido ao controle e atualizações de movimentação dos avatares, que é muito inferior ao tráfego gerado pela transferência de texturas.

9. Conclusão

Neste trabalho foi proposto, implementado e avaliado uma rede P2P integrada a mundos virtuais 3D distribuindo imagens que representam a textura de objetos, tendo tal abordagem um desempenho bastante superior a arquitetura C/S tradicional. Um dos pontos fortes está no cliente que não exige modificações excessivas na arquitetura atual. Com a rede proposta se ganha escalabilidade mantendo aspectos essenciais para o funcionamento de mundos virtuais como persistência dos dados e melhoria da qualidade de experiência do usuário devido a taxas mais altas de transferência de texturas, tendo grande impacto também na diminuição da carga de tráfego de rede no servidor.

Com base nos resultado vimos que a arquitetura SimP2P é uma plataforma estável e com grande potencial de evolução.

Em trabalhos futuros pretende-se analisar outras estratégias de distribuição de texturas por meio de uma rede P2P e realizar avaliações em regiões do Second Life.

Referências

Antonello, R. T. (2008). Análise e Modelagem de Tráfego do Mundo Virtual Second Life. Dissertação de Mestrado. Centro de Informática (CIn). Universidade Federal de Pernambuco (UFPE), Março de 2008.

Arquitetura ROBUST (2010)- http://opensimulator.org/wiki/ROBUST, acessado em Novembro de 2010.

BitTorrent (2010), http://www.bittorrent.com/, acessado em junho de 2010.

Blog SecondLife (2010) - https://blogs.secondlife.com/community/features/blog/2009/04/16/the-second-life-economy--first-quarter-2009-in-detail, acessado em novembro de 2010.

Douglas, S., Tanin, E., and Harwood, A. (2005), "Enabling Massively Multi- Player Online Gaming Applications on a P2P Architecture," in Proc. Of ICIA. IEEE.

Fan, L., Taylor, H., and Trinder, P. (2007), "Mediator: A Design Framework for P2P MMOGs," in Proc. of NetGames. ACM, 2007.

Fan, L., Trinder, P. and Taylor, H. (2009), "Design Issues for Peer-to-Peer Massively Multiplayer Online Games", 2nd InternationalWorkshop on Massively Multiuser Virtual Environments (MMVE), at IEEE Virtual Reality (VR '09).

Hampel, T., Bopp, T. and Hinn, R. (2006), "A P2P Architecture for Massive Multiplayer Online Games," ACM NetGames.

Hippo OpenSim Viewer (2010) - http://mjm-labs.com/viewer/, acessado em Novembro de 2010.

Kamienski, Carlos., A., Fernandes, Stênio F.L., Silva, Cledja K.R., (2008) "Mundos Virtuais: Histórico, Avaliação e Perspectivas". Mini-Curso do WebMedia 2008.

Kinicki, J., Claypool, M. (2008), "Traffic Analysis of Avatars in Second Life", ACM NOSSDAV 2008.

Liang, H., Motani, M. and Tsang, W. (2008), "Textures in Second Life: Measurement and Analysis," IEEE ICPADS workshop P2P-NVE, Dec. 2008.

LibOpenMeverse (2010) - http://www.openmetaverse.org/ , acessado em Dezembro de 2010.

Lu, H., Knutsson, B., Xu, W. and Hopkins, B. (2004), "P2P Support for Massively Multiplayer Games," in Proc. of INFOCOM. IEEE, pp. 7–11.

Mennecke, B., McNeill, D., Ganis, M., Roche, E., Townsend, A., Lester, J. (2008) Second Life and Other Virtual Worlds: A Roadmap for Research, Coommunications of the ACM, 2008.

Mundos para OpenSim (2010) - http://opensimworlds.com, acessado em Novembro de 2010.

OpenSimulator (2010), http://opensimulator.org, acessado em dezembro de 2010.

Second Life (2010), http://opensimulator.org, acessado em novembro de 2010.

Varvello, M., Picconi, F., Diot, C., and Biersack, E. (2008). Is There Life inSecond Life? In Proc. of CONEXT'08, Madrid, Spain, 2008