Trigate
Sistema Web para Palestras no Ambiente Virtual *OpenSim*

Renan Pinho Assi[1], **Rafael Luiz Cancian**[1]

[1]Instituto de Informatica e Estatística – Universidade Federal de Santa Catarina (UFSC)
Santa Catarina – SC – Brazil

Resumo. *Neste trabalho será explicado em detalhes todas as técnicas envolvidas para a construção de um sistema para web com funcionalidade de cadastro de usuários, gerenciamento de acessos, streaming de vídeo e sistema de chat. Além disso, será abordado a integração da mesma com o OpenSim, um ambiente de realidade virtual 3D.*

Abstract. *This thesis will explain in detail all the techniques involved to build an web application with user registration, access management and video streaming capabilities. Furthermore, the integration of this web application with OpenSim, a 3D virtual reality environment, will be addressed.*

1. Introdução

Ao longo dos anos as formas de ensino vêm se modernizando e a modalidade de educação a distância (EaD) vem ganhando cada vez mais espaço no Brasil e no mundo. As restrições para conter o avanço da Covid-19 também fomentaram o aumento não somente da EaD como também de regimes de trabalho em casa (*Home Office*) e reuniões e palestras na *web* [Quintino 2021][Mello 2021], o que, por sua vez, influenciou no crescimento do uso de ferramentas de comunicação *On-line* como o Zoom, Skype e outras mais. Também é preciso destacar a procura crescente de mundos virtuais para a adaptação ao contexto atual. Muitas entidades tem aderido a esses simuladores com o objetivo de trazer uma maior imersão entre as suas atividades com os seus participantes ou promover simulações de experimentos em ambientes controlados. Isso tem gerado uma nova onda de utilização massiva desses *softwares* que foram muito populares no começo da década de 2000 mas que tiveram um queda significativa no uso ao longo dos anos. O Second Life é um bom exemplo desse fenômeno. Durante a pandemia, registrou um aumento considerável na sua base de usuários, bem como um retorno significativo de usuários antigos [Kariuki 2020].

Apesar do cenário dos últimos anos ter trazido possibilidades de inovação, ainda existe uma lacuna entre as ferramentas disponíveis pois cada uma tem seu próprio nicho específico de atuação. Este trabalho se propõem a apresentar o desenvolvimento de uma plataforma que une os pontos principais das ferramentas usadas para comunicação *On-line* e que ao mesmo tempo esteja implementada dentro de um mundo virtual, possibilitando desta forma que qualquer usuário possa fazer sua apresentação, aula ou palestra sem precisar estar vinculado no mesmo.

2. Fundamentação Teórica

2.1. Ensino a Distância

A Educação a Distância (EaD) pode ser definida como ”o processo de ensino-aprendizagem, mediado por tecnologias, onde professores e alunos estão separados espacial e/ou temporalmente” (MORAN, 1994). Existem registros de mais de 200 anos da prática de EaD.

O marco inicial da EaD se deu no ano de 1728, onde anúncios de estudos por correspondência foram feitos em um jornal de Boston (ALVES, 2011). No Brasil, Segundo Mugnol em artigo publicado em 2009, "as primeiras iniciativas em educação a distância no Brasil se deram por meio de cursos por correspondência" [Mugnol 2009] e essas iniciativas começaram antes de 1900 no Rio de Janeiro [Rodrigues 2012]. Com os adventos da tecnologia se consolidando cadava vez mais no Brasil e no mundo, a EaD começou a ser aplicada em outros formatos. Atualmente, o Brasil que tem cerca de 134 milhões de internautas, segundo dados publicados pela Agencia Brasil em 2020, vem sendo muito receptivo ao formato de EaD via *internet*. Segundo dados do INEP apresentados pelo portal de notícias G1 [Tenente 2012], em 10 anos, o ensino a distância no Brasil cresceu 378,9% entre 2009 a 2019. Além disso, por conta da COVID-19, houve um aumento na procura por ensino na modalidade a distância. Segundo dados da *Udemy*, durante o período de Fevereiro a Abril de 2020, o número de inscrições em cursos na sua plataforma cresceu em 425% em escala global e 95% no Brasil [Udemy 2021].

Para atender a demanda crescente por EaD via internet os cursos ofertados precisam se apoiar em ferramentas que propiciem facilidades para os mais diversos objetivos como encontros entre educadores e educandos, disponibilização de conteúdos, simulação de experimentos, entre outros. Atualmente, grande parte dos cursos utilizam algum tipo de ambiente virtual de aprendizagem para gestão das disciplinas dos cursos e dos seus usuários, além disso, estão sendo usados plataformas de *streaming* de vídeo para aulas síncronas ou assíncronas. Por ultimo, deve-se destacar a utilização de ambientes virtuais de tridimensionais como uma nova classe de ferramentas que vem sendo exploradas nos cursos de EaD.

2.2. Ferramentas para Ensino a Distância

Neste trabalho buscou-se incluir características de 3 tipos de classes de ferramentas que são utilizados na EaD. Esta seção irá apresentar brevemente estas classes.

2.2.1. Ambientes Virtuais de Aprendizagem

Um Ambiente Virtual de Aprendizagem (AVA) pode ser descrito como uma plataforma de gerenciamento de materiais e pessoas, tendo como papel principal centralizar toda a estrutura de um curso. Dentro das mais variadas opções disponíveis, é de consenso que as principais características são o controle de acesso dos usuários, as funcionalidades para gerenciamento de materiais do curso e o acompanhamento administrativo do progresso dos alunos (Milligan, 1999) [Milligan 1999]. Todavia, cada implementação de AVA pode ter características especificas.

2.2.2. Plataformas de *Streaming* de Vídeo

Streaming de vídeo é a técnica de enviar pequenos pacotes de vídeo para o usuário final em tempo real pela *web*. Ao mesmo tempo que o usuário assiste um vídeo, ele também está o carregando. Essa técnica evita que o vídeo tenha que ser carregado completamente para ser visualizado (Hartsell, T., & Yuen, S., 2006). Com base nisso é possível definir que plataformas de *streaming* de vídeos são sites, aplicações ou *softwares* que

se utilizam dessa técnica para transmitir vídeos, sejam eles em tempo real (como *web* conferências) ou gravações. Apesar de não serem voltadas para um propósito específico, atualmente tem-se utilizado cada vez mais essas plataformas para educação e trabalho. Segundo a revista ISTOÉ, com as medidas de restrição impostas pela COVID-19, a demanda por plataformas de *streaming* de vídeo aumentaram. A plataforma *Skype*, por exemplo, obteve um crescimento de 70% de usuários em apenas 1 mês [Redação 2020]. O portal *TechRepublic* publicou dados de um monitoramento feito entre 17 de fevereiro a 6 de abril. Nesta pesquisa, das 6 plataformas monitoradas, o menor crescimento foi de 166% da plataforma *Skype*, ao passo que o maior crescimento foi da plataforma *Zoom*, com 552% [Bayern 2020].

2.2.3. Mundos Virtuais Tridimensionais

Um mundo virtual tridimensional (MVT) pode ser definido como um espaço modelado em 3 dimensões se utilizando de computação gráfica. Além disso, o mundo virtual tem como prerrogativa a dinamicidade, se adaptando/modificando as ações dos usuários (Schlemmera; Backesb, 2008). Em adição a essas características, muitos dos MVT disponibilizam linguagens de programação que são interpretadas dentro dos próprios ambientes, o que facilita a expansão e dinamicidade do mundo pelos usuários. As aplicações dos MVT são imensas, mas o interesse de aplicação desses *softwares* no EaD vem crescendo cada vez mais, visto que podem funcionar como ambientes sociais, lúdicos e até mesmo como simuladores.

3. O projeto UFSC3D

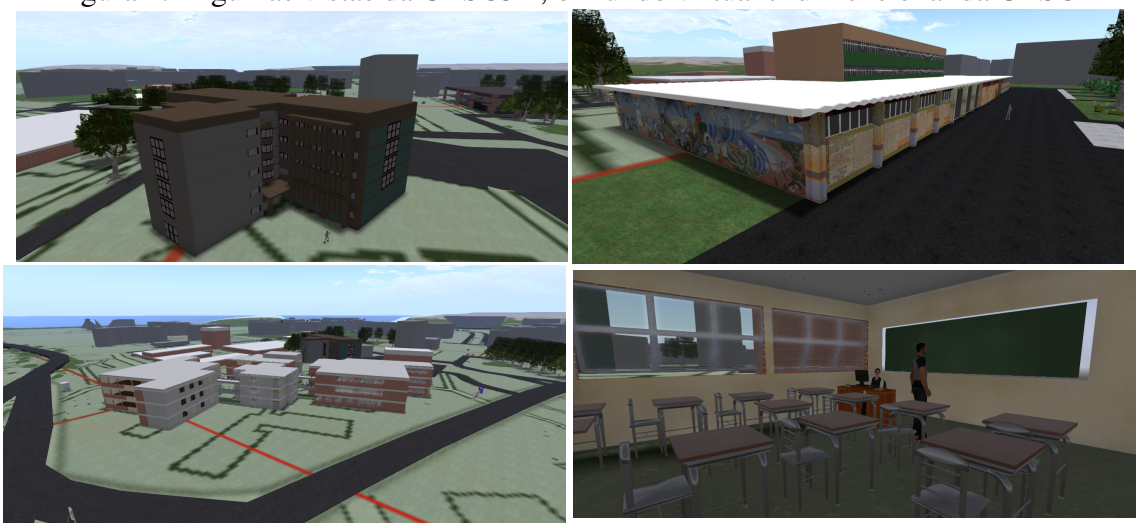
O projeto de extensão "Construção ambiente virtual 3D em multiverso para ensino, pesquisa e extensão", simplificadamente intitulado "UFSC3D" visa criar pelo menos um mundo virtual 3D que represente a UFSC e que permita que possam ser oferecidos pelo menos os serviços de divulgação da instituição, a realização de aulas, cursos e palestras, incluindo mecanismos de avaliação da aprendizagem, e de experimentação remota. Os mundos virtuais são ferramentas poderosas de simulação e possuem aplicação em diversas áreas. Embora seu uso tenha diminuído consideravelmente nos últimos anos, devido o advento das redes sociais, sua aplicabilidade e potencialidade não diminuíram. Muitas universidades e empresas privadas utilizam mundos virtuais como plataforma de ensino, divulgação, comunicação e comércio eletrônico.

Um mundo virtual que represente a UFSC poderia ser usado para aulas virtuais à distância (em tempo real ou não), oferecimento de cursos, realização de feiras virtuais (como a SEPEX por exemplo), passeios virtuais em diferentes cenários (como os diferentes campi e as Fortalezas da Ilha por exemplo), possivelmente guiados por NPC (personagens virtuais interativos), comércio de serviços e produtos e várias outras opções, incluindo pesquisa em algumas áreas de conhecimento.

Este trabalho está vinculado ao projeto UFSC3D e fornece uma plataforma *on-line* para comunicação em tempo real de palestrantes com usuários do OpenSim, denominada Trigate. Seu objetivo é viabilizar a interação de usuários leigos em relação aos mundos virtuais tridimensionais, como a UFSC3D, pois uma conexão direta no multiverso não seria necessária. A plataforma Trigate precisa ter serviços para comunicação bidirecional

por texto e um *streaming* de vídeo capaz de suportar uma sala de usuários, além de ser acessível de forma externa ao mundo virtual, fazendo com que o usuário da plataforma consiga se comunicar com os usuários do multiverso. Algumas visões da versão desse mundo virtual podem ser vistas na figura 1.

Figura 1. Algumas vistas da UFSC3D, o mundo virtual tridimensional da UFSC



4. Solução Proposta

4.1. Arquitetura do Sistema

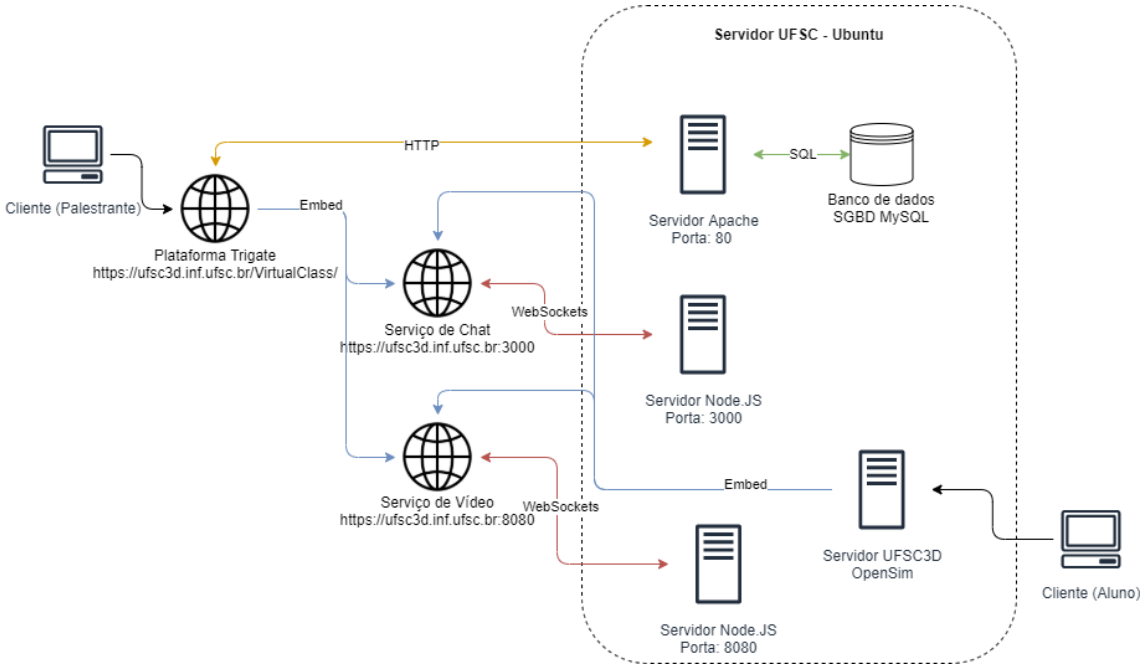
O sistema *Trigate* é composto por diferentes componentes e portanto alguns aspectos da estrutura precisam ser destacados. Os componentes do sistema estão embasados em um servidor Ubuntu versão 18.04 da UFSC, ao todo são 2 microsserviços (*chat* e *streaming* de vídeo), uma aplicação *web* (portal Trigate) e um banco de dados.

A aplicação *web* do portal está hospedada na porta 80, no subdiretório *Virtual-Class*. Essa aplicação conta com o *Front-End* do portal e da API Rest que contém métodos para a comunicação com o banco e outras funcionalidades. Os serviços de *chat* e vídeo são embutidos no portal por meio de *iframes*, o que confere uma maior independência entre todos os agentes do sistema.

Os microsserviços de *chat* e *vídeo* são hospedados por servidores *Node.js* nas portas 3000 e 8080 respectivamente. Esses serviços utilizam a biblioteca *socket.io* para viabilizar a comunicação em tempo real usando o protocolo *full-duplex WebSockets*.

Por fim, o Banco de dados, que foi construído com o SGBD *MySQL*, tem como função armazenar todos os dados pertinentes a plataforma Trigate, sendo mais especificamente os dados de cadastro dos usuários palestrantes e as salas que são cadastradas manualmente.

Figura 2. Diagrama dos Componentes do Sistema.



Fonte: elaborado pelo autor, 2021.

4.2. Tecnologias e Infraestrutura

Para a construção da plataforma do sistema e de seus microsserviços foram necessários a utilização de várias tecnologias. Para uma maior integrabilidade, se optou por manter as mesmas linguagens de programação entre os microsserviços e a plataforma. Além disso, foram escolhidas linguagens consolidadas no mercado com o auxilio de poucas bibliotecas, visando assim, diminuir a quantidade de refatoração do código ao longo do tempo em decorrência de possíveis descontinuações das mesmas.

Para o *Front-End* da plataforma e dos microsserviços foram utilizados HTML e CSS para a estruturação visual e JavaScript como linguagem de programação para a criação de funcionalidades dinâmicas. A biblioteca Jquery, bem como os *frameworks* Bootstrap e ELC - Easy Constructor também forma utilizadas no desenvolvimento *Front-End*. Essas tecnologias foram escolhidas em razão da sua popularidade e estabilidade ao longo dos anos.

Para o desenvolvimento *Back-End*, foi utilizado PHP e JavaScript (Node.JS) como linguagens de programação. As bibliotecas Socket.io e PHP-JWT foram escolhidas para facilitar certas funcionalidades. A biblioteca Socket.io tem como objetivo facilitar a comunicação via protocolo WebSocket, enquanto a PHP-JWT facilita a manipulação de *tokens* JWT.

Para as aplicações de *chat* e *streaming* de vídeo, foi escolhidos o servidor Node.JS, pois o mesmo suporta a comunicação via *WebSocket* pela implementação da biblioteca Socket.io. Para o portal da plataforma, foi utilizado o servidor Apache pela popularidade do mesmo. O SGBD utilizado é o MySQL e sua escolha se deu pela ampla utilização do mesmo, o que confere muita credibilidade e fácil acesso a conteúdos explanativos.

4.3. API REST

A *API* usada pelo sistema foi construída em *PHP* utilizando um paradigma misto de programação. Para se fazer a virtualização das tabelas do banco de dados foi utilizado o paradigma de orientação a objetos. O paradigma procedural foi utilizado para construir cada método da API. Todas as consultas feitas ao banco de dados utilizam a interface *PDO* (*PHP Data Objects*). Essa interface oferece vantagens como a preparação das consultas *SQL*, o que faz uma pré-verificação se a consulta funciona da maneira esperada, executando a consulta a qual ela foi designada evitando, deste modo, ataques do tipo *SQL injection*. Outra vantagem é que o tratamento de erros nas requisições é facilitado, pois são oferecidos métodos para receber informações sobre o resultado de uma requisição.

Para a *API* do sistema, foi elaborado um padrão de *endpoints* para retratar com clareza a ação alvo. A estrutura do padrão é relativamente simples:

<Url base>/ + api/ + <escopo da ação> + /<ação>

Onde:

- *Url base*: *Url* onde o sistema reside.
- Escopo da ação: Referente ao grupo que será atingido pela ação. Um exemplo seria o acesso a informações de usuários, logo o escopo da ação seria "*users*".
- Ação: Efetivamente a ação que será feita. No mesmo exemplo supracitado, a ação seria "*get_user_info*".

Apesar de preferencialmente começar com a palavra "*get*" no começo de todas as ações que farão requisições do tipo *GET*, não existe uma padronização para nomear as ações com relação ao tipo da requisição.

É importante ressaltar que o acesso à plataforma funciona com *tokens*. Cada usuário cadastrado, no ato de acessar o sistema, recebe um *token* temporário. Esse *token* será utilizado para determinar se o usuário tem acesso a ação que se pretende performar. Nesta *API*, alguns métodos não necessitam de *token* para serem utilizados (são métodos globais que podem ser utilizados por todos os tipos de usuários e até mesmo não usuários do sistema), porém, outros métodos exigem um *token* válido. Um *token* é válido se seu tempo de expiração não foi atingido e se suas informações internas são válidas (caso haja uma alteração nas informações internas, o *token* não será decifrado com sucesso). Além disso, outras informações internas do *token* podem validar ou invalidar sua utilização, como por exemplo o nível de permissão do usuário.

4.4. Aplicação de *Chat - Trigate Pigeon*

Pigeon é a aplicação de *chat* em tempo real desenvolvida ao longo do projeto. Essa aplicação se utiliza da biblioteca *socket.io* para trocar mensagens via *websockets* com outros usuário da aplicação. Um serviço simples que exige apenas um tratamento nas mensagens para retirar caracteres especiais e evitar ataques do tipo *XSS injection*. Esse tratamento, para simplificar ainda mais a mecânica da aplicação, é feita no lado do cliente apenas, muito embora, para uma versão final o aconselhado seria migrar o tratamento para o lado do servidor, evitando assim que o usuário possa driblar a mecânica de segurança implementada. Além do mecanismo de filtragem de mensagens, foi preciso implementar um ouvinte no lado do servidor que recebe uma mensagem e envia ao demais usuários da

aplicação. Um ouvinte no lado do cliente, que recebe uma mensagem e a coloca na área de texto da aplicação, também se faz necessário. Para otimizar a aplicação, as mensagens pertencentes a um usuário são colocadas diretamente na sua área de texto local, sem esperar a resposta do servidor.

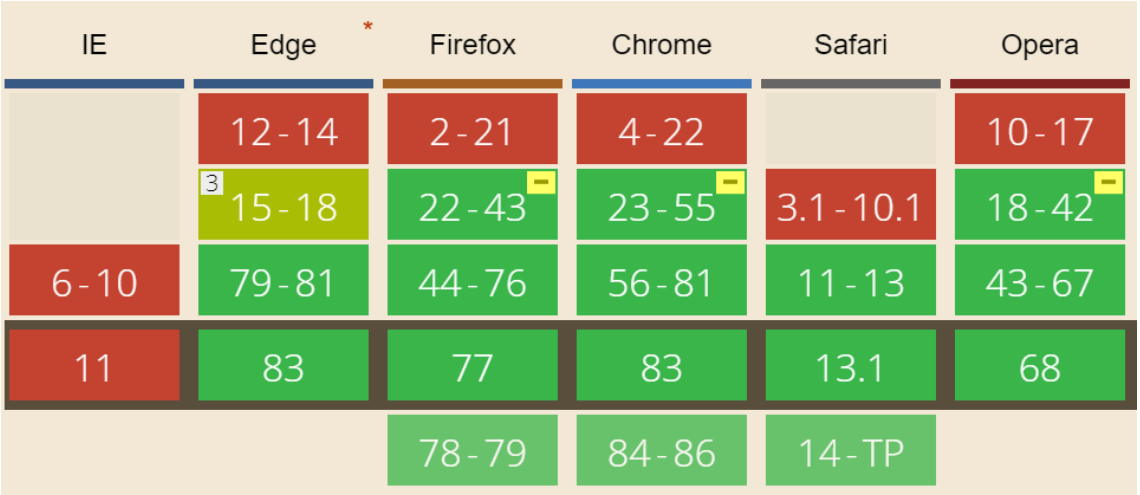
4.5. Aplicação de Video *Broadcasting*

Trigate Video Broadcasting ou TVB é uma aplicação de vídeo chamada em tempo real na modalidade *broadcast*, o que significa que a mídia compartilha e o áudio flui apenas de um usuário para os demais usuários. A TVB utiliza *WebRTC* e *RTCMultiConnection*, um *framework* que permite comunicação em vídeo e/ou áudio em tempo real. O projeto é apoiado por grandes empresas como o *Apple*, *Google* e *Microsoft*. Esse *framework* implementou nos navegadores várias funcionalidades para o propósito de comunicação em tempo real e é preciso destacar duas funcionalidades:

- ***getUserMedia***: Funcionalidade para pedir acesso a câmera ou tela do dispositivo e/ou microfone. Além de pedir acesso, também recebe as *streams* das mídias caso seja permitido.
- ***RTCPeerConnection***: Habilita a comunicação entre *Peers*. Também performa outras ações para viabilizar o envio das mídias entre os pontos conectados.

Na figura abaixo, é possível ver quais navegadores oferecem suporte ao *WebRTC*.

Figura 3. Tabela de compatibilidade do WebRTC.



Fonte: `https://bloggeek.me/webrtc-browser-support`, 2021.

O *framework open-source RTC Multi Connection* também foi utilizado. Este *framework* oferece facilidades para conexões *peer-to-peer*, tornando o processo de conectar mais usuários em uma mesma sala mais fácil.

A TVB funciona com um sistema de salas, ao conectar no microsserviço, o usuário é colocado na sala a qual o parâmetro de *query* (parâmetro colocado após o *url*) *room-id* indica. Atualmente, a aplicação foi configurada para funcionar em um modo automático de entrar/criar sala. Caso não exista ninguém na sala, a aplicação presume que este usuário é um palestrante e então pede acesso a tela do computador e ao microfone. Se já existe algum usuário na sala, a aplicação recebe a mídia que está sendo compartilhada (som e áudio) e então aplica na tela do novo usuário conectado. O ponto principal dessa aplicação é a utilização de servidores de estabelecimento de conectividade interativa (*ICE Servers*). Seu papel é descobrir o *IP* público da máquina que está acessando a aplicação *web* e o tipo de *NAT* que esta máquina está. É preciso observar que esse passo viabiliza a conexão entre *peers* e após isso, as mídias serão compartilhadas diretamente entre os *peers*.

4.6. Integração na UFSC3D

As aplicações do sistema *Trigate* foram projetadas para serem utilizadas independente de qualquer outro *software* que não o navegador e por esse motivo elas podem, inclusive, serem utilizadas separadas do sistema Trigate. Essa característica confere uma facilidade em termos de integração com outros sistemas. O requisito mínimo para rodar as aplicações é ter um *browser* que suporte operações assíncronas e *websockets* e *WebGL*. As aplicações foram integradas em um objeto 3D, em forma de televisor, utilizando a funcionalidade *Shared Media* do *Open Simulator*.

Figura 4. Exemplo de microsserviço aplicado no OpenSim usando a função *Shared Media*.

Fonte: elaborado pelo autor, 2021.

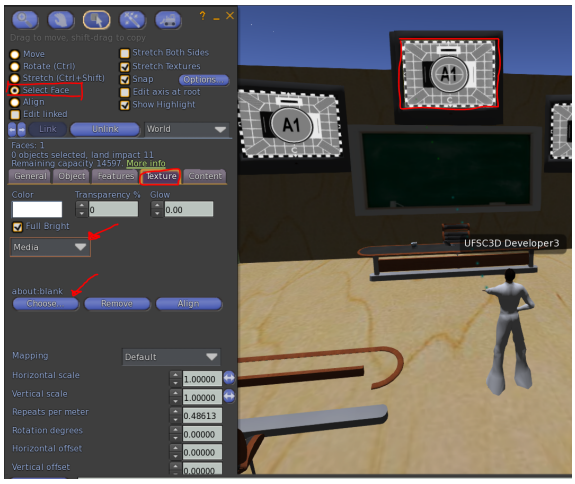
Essa funcionalidade permite a visualização de um conteúdo na *web* dentro do mundo virtual. A aplicação se dá na textura de um objeto. Para isso, primeiramente é preciso selecionar o objeto e escolher a opção ”*Edit*”:

Figura 5. Exemplo de aplicação de textura usando *Shared Media* - Sem título 1.

Fonte: elaborado pelo autor, 2021.

Um menu com informações do objeto aparecerá, selecionaremos a seção de de escolher faces e a opção ”*Select Face*”. Isso possibilita a escolha da face do objeto que iremos aplicar o microsserviço como textura. Com a face selecionada (clicando nela), selecionaremos a seção ”*Texture*”. O tipo de textura a ser usada é ”*Media*”. Por fim, apertaremos no botão ”*Choose*”:

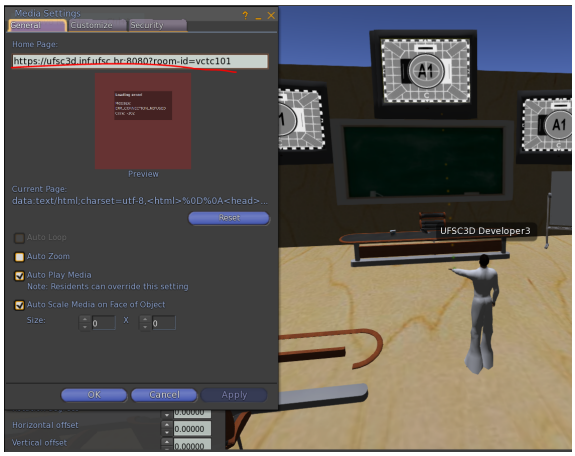
Figura 6. Exemplo de aplicação de textura usando *Shared Media* - Passo 2.



Fonte: elaborado pelo autor, 2021.

Finalmente, podemos aplicar um *url* para ser a textura da face selecionada. Após inserir o *url*, devemos aperta no botão ”*apply*”:

Figura 7. Exemplo de aplicação de textura usando *Shared Media* - Passo 3.



Fonte: elaborado pelo autor, 2021.

Após a aplicação, o usuário pode interagir com o site aplicado na textura do objeto como se estivesse usando a página diretamente no navegador.

Figura 8. Texturas aplicadas no OpenSim.

Fonte: elaborado pelo autor, 2021.

5. Testes e Validação

Boa parte dos testes foram feitos simulando a utilização por parte de um usuário, porém, em alguns casos houve a necessidade de fazer testes mais elaborados. Neste seção serão discutidas algumas abordagens de testes utilizadas neste projeto.

5.1. Testes simulando usuário

Esse conjunto de testes teve como função validar funções chaves do sistema. Primeiramente, foi testado o acesso a plataforma utilizando dados incorretos para testar os mecanismos de bloqueio e dados corretos para testar o acesso efetivo. Também foram feitos em todas as funcionalidades menores da plataforma, como o fluxo de *logout* da plataforma, a retomada de páginas após um recarregamento do navegador e o acesso às salas virtuais. Esses testes foram aplicados tanto em um computador como em um celular, para testar a responsividade e *bugs* de *layout*. As ferramentas de *Chat* e de vídeo *broadcasting* também foram testadas nesse formato. Visualmente, foi avaliada a sua implementação dentro do portal. Quanto à ferramenta de *chat*, duas abas do navegador simularam dois diferentes usuários se comunicando via *chat* com o objetivo de avaliar a disposição das mensagens na tela, o tamanho das fontes e o tempo de resposta do servidor em relação a entrega das mensagens. O mesmo teste foi feito com o microsserviço de vídeo, mas com o objetivo de testar o funcionamento da funcionalidade *broadcast* para diferentes dispositivos. Por fim, ambos os serviços foram testados utilizando o usuário de testes da UFSC3D

Apesar de simples, a aplicação desses testes ajudaram na resolução de muitos *bugs* e também no aperfeiçoamento da usabilidade do sistema.

5.2. Testes de carga

Foram aplicados testes de carga para avaliar as condições dos componentes em relação ao servidor. Para API REST da plataforma, foi utilizado a ferramenta Postman, que permite a criação de cenários de testes com requisições HTTP. Para esse teste foram feitas 2000 requisições à API, tendo como resultado uma média de 13 ms por chamada.

Para testar a capacidade de carga do serviço de vídeo e *chat* foi elaborado um teste para simular condições de uso em situações reais. Foram utilizados 4 dispositivos

diferentes para este teste. 1 dispositivo tomou o papel do palestrante, sendo usado para fazer a captura da tela em tempo real e também para acompanhar a gravação dentro da UFSC3D e 3 dispositivos foram destinados a simular os participantes da palestra, onde foram abertos 15 abas dos serviços de *chat* e vídeo para assistir o vídeo e interagir com o *chat*.

Este teste teve duração de 30 minutos e durante este tempo foi capturado um vídeo em contagem regressiva para testar o *delay* da transmissão do vídeo. Algumas mudanças de telas e outros exercícios foram feitos com o mesmo propósito. Com o serviço de *chat*, foram enviadas mensagens de várias abas, escolhidos de forma aleatória, com o objetivo de também testar o *delay* entre o envio e recebimento das mensagens.

6. Resultados

Com o final dos testes, pode se observar que o sistema pode ser acessado de forma satisfatória tanto em computadores como em dispositivos móveis, isso pela propriedade de responsividade da plataforma, ou seja, a capacidade de se adequar aos mais diferentes formatos de telas. Apesar dos testes terem revelado que nos dispositivos móveis não é possível compartilhar a tela (por limitações do próprio navegador) e consequentemente apresentar uma palestra, foram feitos testes usando a câmera dos dispositivos e nessa modalidade é possível abrir uma *stream* de vídeo e com isso permitir que a palestra seja feita utilizando celulares e *tablets*.

Com os testes de carga foi possível verificar que o servidor consegue lidar com uma carga significativa de usuários. Os testes com a API mostraram que se vários usuários acessarem a ferramenta em um mesmo tempo, o servidor consegue dar boa vasão as requisições e manter o funcionamento do sistema com um tempo normal de resposta. Também foi mostrado que para cenários similares aos reais, onde se estima um número considerável de usuários, o sistema consegue manter as funções de *chat* e vídeo com tempos de *delay* irrisórios, mantendo assim um bom funcionamento no geral. Os testes também demostraram boa conectividade entre os participantes em relação ao *streaming* de áudio, que se mostrou muito satisfatório até quando testado em conexões mais lentas. Apesar de tudo, uma acurácia melhor seria possível se fossem aplicados testes em condições reais de funcionamento, ou seja, com uma quantidade similar ao de uma turma de um curso da UFSC.

Por fim, a independência das aplicações mostrou uma facilidade muito grande quanto a integração das mesmas com outras aplicações, permitindo que a ferramenta possa atendar não só o mundo virtual da UFSC3D, mas também outros mundos virtuais ao mesmo tempo.

7. Conclusões

Neste trabalho, buscou-se implementar uma ferramenta para palestras com integração ao mundo virtual UFSC3D hospedado no *OpenSim*. Como objetivo, o sistema deveria entregar funcionalidades de *streaming* de vídeo e *chat* em tempo real e o palestrante deveria poder se comunicar, apenas utilizando a plataforma por meio desses serviços, com usuários do *OpenSim*. Além disso, a plataforma deveria contar com um sistema de autentificação para os palestrantes, permitindo que o usuário pudesse se cadastrar na plataforma, via confirmação de *e-mail*, para efetivamente usá-la.

Conforme relatado na seção de resultados, o objetivos que tangem a capacidade do sistema ser utilizado em tempo real em situações reais foram atingidos. Além disso, o sistema é utilizado de forma totalmente independente ao *OpenSim*, que era um dos objetivos pretendidos. Todavia, em função do tempo para a conclusão do trabalho e da migração do servidor externo ao da UFSC, que foi utilizado para desenvolver o projeto, para o servidor final da UFSC, não foi possível reestabelecer um servidor de *e-mail*, o que tornou o sistema de cadastro inativo.

Apesar do sistema estar em fase inicial, com algumas implementações ainda não amadurecidas totalmente, a capacidade de expansão, conforme foi abordado nesta seção, e com os resultados dos testes aplicados, conclui-se que a implementação de um sistema para palestras em mundos virtuais é viável.

8. Recomendações para Trabalhos Futuros

Para a expansão deste trabalho, recomenda-se a implementação de um provedor de *e-mail* no servidor da UFSC para que o sistema de cadastro possa ser restabelecido ou a integração do mesmo ao sistema da UFSC. Como novas funcionalidades, se recomenda a implementação de papéis diferenciados para usuários cadastrados na plataforma e a construção de uma plataforma administrativa para gerenciar o sistema. A inserção de informações na tela do palestrantes, como a quantidade de usuários presentes na palestra e por quais plataformas estão acessando, também pode ser uma possibilidade de expansão do sistema. Outra adição valiosa seria a construção de uma nova aplicação que transmitisse o áudio dos usuários no mundo virtual para a plataforma, pois proveria maior interatividade entre mundos, visto que o formato atual é de transmissão unilateral do palestrante para os ouvintes.

Ainda se tratando de novas funcionalidades, mas com um viés mais ambicioso, a integração do mundo virtual com óculos de realidade virtual se mostra muito atrativa, bem como a construção de um visualizador do mundo virtual próprio para *web* para ser utilizado dentro da plataforma. Por fim e não menos ambicioso, se sugere a criação de um mundo virtual 3D próprio para substituir ou ser usando em conjunto com o atual *OpenSim* da UFSC3D utilizado neste trabalho.

9. Referências

Alves, L. (2011). Second educação a distância: conceitos e história no brasil e no mundo. *Revista Brasileira de Aprendizagem Aberta e a Distância*, 10(7):84–92.

Bayern, M. (2020). Zoom grew by 574% in less than two months, but skype for business reigns supreme. Acessado: 26 de agosto de 2021.

BlogGeek.Me (2020). Webrtc browser support on desktop and mobile. Acessado: 26 de agosto de 2021.

Eliane Schlemmer, L. B. (2008). Metaversos: novos espaços para construção do conhecimento. *Revista Diálogo Educacional*, 8(24):519–532.

Kariuki, D. (2020). Pandemic spurs second life usage, book club, lower non-profit prices. Acessado: 26 de agosto de 2021.

Mello, D. (2021). Home office foi adotado por 46% das empresas durante a pandemia. Acessado: 26 de agosto de 2021.

Milligan, C. (1999). The role of virtual learning environments in the online delivery of staff development. Acessado: 26 de agosto de 2021.

Moran, J. (1994). O que é educação a distância. Acessado: 26 de agosto de 2021.

Mugnol, M. (2009). A educação a distância no brasil: Conceitos e fundamentos. *Diálogo Educacional*, 9(27):335–349.

Quintino, L. (2021). Ofertas de vagas em regime home office crescem mais de 300% em 2020. Acessado: 26 de agosto de 2021.

Redação (2020). Número de usuários do skype cresceu 70% durante a quarentena. Acessado: 26 de agosto de 2021.

Rodrigues, C. (2012). Cursos por correspondência, hoje em desuso, recebiam mais de mil cartas por dia. Acessado: 26 de agosto de 2021.

Taralynn Hartsell, S. C.-Y. Y. (2006). Video streaming in online learning. *AACE Review*, 14(1):31–43.

Tenente, L. (2012). Em 10 anos, aumenta quase 5 vezes número de alunos que entram em cursos à distância do ensino superior, diz inep. Acessado: 26 de agosto de 2021.

Udemy (2021). Online education steps up: What the world is learning (from home). Acessado: 26 de agosto de 2021.