UNIVERSIDADE FEDERAL DE SÃO JOÃO DEL-REI

ALEXANDRE SILVA DE ALMEIDA

Protocolos e Aplicações para Internet das Coisas Musicais

São João del-Rei

2020

ALEXANDRE SILVA DE ALMEIDA

Protocolos e Aplicações para Internet das Coisas Musicais

Dissertação submetida ao Programa de Pós-graduação em Ciência da Computação da Universidade Federal de São João Del-Rei, como parte dos requisitos necessários para a obtenção do título de Mestre em Ciência da Computação.

Trabalho aprovado. São João del-Rei, 14 de dezembro de 2020:

Prof. Dr. Rafael Sachetto Oliveira
Orientador

Prof. Dr. Flávio Luiz Schiavoni
Coorientador

Prof. Dr. Elder José Reioli Cirilo
Titular Interno

Profa. Dra. Teresinha Moreira de Magalhães
Titular Exteno

São João del-Rei
2020

Ficha catalográfica elaborada pela Divisão de Biblioteca (DIBIB)
e Núcleo de Tecnologia da Informação (NTINF) da UFSJ,
com os dados fornecidos pelo(a) autor(a)

A798p

Almeida, Alexandre Silva de.
Protocolos e Aplicações para Internet das Coisas Musicais / Alexandre Silva de Almeida ; orientador Rafael Sachetto Oliveira; coorientador Flávio Luiz Schiavoni. -- São João del-Rei, 2020.
77 p.

Dissertação (Mestrado - Ciência da Computação) -- Universidade Federal de São João del-Rei, 2020.

1. Internet das Coisas. 2. Redes de Computadores. 3. Sistemas de Computação Distribuídos Heterogêneos. I. Oliveira, Rafael Sachetto, orient. II. Schiavoni, Flávio Luiz, co-orient. III. Título.

Agradecimentos

A Deus, pela dádiva da vida e por ter me iluminado durante toda essa jornada, colocando pessoas especiais ao meu lado, que me apoiaram muito e certamente sem elas não teria conseguido.

Imensamente ao meu orientador Rafael Sachetto e coorientador Flávio Schiavoni, pela paciência, empenho e dedicação que tiveram, além do excelente profissionalismo e amizade que firmamos ao decorrer deste trabalho ao qual saio enriquecido de conhecimento.

Aos todos os professores, colegas do Mestrado e de Laboratório que compartilharam conhecimento, amizade e força.

Aos meus amigos que tiveram sempre me acompanhando e incentivando, em especial ao Felipe Valente e Marcus Vinícius pela enorme força e apoio.

A meus pais que são tudo para mim e familiares que estiveram sempre comigo.

A Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo incentivo na concessão de bolsas de estudos no primeiro ano do mestrado.

Por fim, o meu profundo agradecimento as pessoas que contribuíram para a realização deste trabalho, motivando-me intelectual e emocionalmente.

Resumo

A arte evidencia o entendimento social e cultural da sociedade, juntamente com a música, produz os costumes e a identidade das pessoas. No Brasil a música representa uma posição substancial na cultura e na sociedade, produzindo competências sociais, artísticas e culturais, destaca-se por nossa população ouvir mais músicas que muitos outros países. Em decorrência do isolamento social causado pelo coronavírus, o grupo artístico musical que produz e consome música foi muito afetado. Com o auxílio dos recursos tecnológicos, este projeto de pesquisa busca, soluções que possam minimizar o distanciamento social utilizando as concepções e desafios da Internet das Coisas Musicais, desenvolvendo um modelo com a criação de uma aplicação e protocolo para compartilhamento de música de forma síncrona e em tempo real, baseando-se em cenários que demonstram tal necessidade. A aplicação utiliza conteúdos do *YouTube* permitindo interação com os métodos musicais entre os usuários no compartilhamento de músicas e *Playlist*, apresentando aos usuários uma *interface* simples, prática e funcional com todos os elementos básicos de um reprodutor musical, possibilitando centenas e milhares de conexões simultâneas.

Palavras-chaves: Internet das Coisas, Internet das Coisas Musicais, protocolos, redes e aplicações musicais.

Abstract

Art demonstrates the social and cultural understanding of society, together with music, produces people's customs and identity. In Brazil, music represents a substantial position in culture and society, producing social, artistic, and cultural skills, it stands out for our population listening to more music than many other countries. As a result of the social isolation emitted by the coronavirus, the artistic group that produces and consumes the music was the most affected. With the contribution of technological means and resources, this research project seeks solutions that minimize social distance using the conceptions and challenges of the Internet of Musical Things, developing a model with the creation of an application and protocol for sharing music synchronously and in real-time, based on situations that demonstrate such a need. The application uses YouTube content allowing the interaction with musical content between users when sharing music and Playlist, providing users with a simple, practical, and functional interface with all the basic elements of a music player, allowing several simultaneously.

Key-words: Internet of Things, Internet of Musical Things, protocols, networks and musical applications.

Lista de ilustrações

Lista de abreviaturas e siglas

API	Application Programming Interface (Interface de Programação de Aplicativos)
CDMA	Code Division Multiple Access (Acesso Múltiplo por Divisão de Código)
CPU	Central Processing Unit
CSS	Cascading Style Sheets
D2D	Device-to-Device
GNU	General Public License
GPS	Global Positioning Satellite (Sistema de Posicionamento Global)
HBO	Home Box Office
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IEEE	Institute of Electrical and Electronics Engineers
IoMusT	Internet of Musical Things (Internet das Coisas Musicais)
IoT	Internet of Things (Internet da Coisas)
IP	Internet Protocol
ITU	International Telecommunication Union
JSON	JavaScript Object Notation
LAN	Local Área Network
LoRaWAN	Long Range Wide Area Network
LPWAN	Low-Power Wide-Area Network
M2M	Machine-to-Machine
MIDI	Musical Instrument Digital Interface
MIMO	Multiple-Input and Multiple-Output

MQTT	Message Queue Telemetry Transport
NMP	Networked Music Performances
NTP	Network Time Protocol
OMS	Organização Mundial da Saúde
P2P	Peer-to-Peer
PHP	Hypertext Preprocessor
RAM	Random Access Memory
RTCP	Real-time Transport Control Protocol
RTP	Real-time Transport Protocol
SSL	Secure Sockets Layer
TCP	Transmission Control Protocol
WPAN	Wireless Personal Area Network

Sumário

1 Introdução

A arte está presente no mundo com uma função primordial no desenvolvimento e formação das pessoas, produzindo competências sociais, artísticas e culturais. A arte por meio da música representa um papel fundamental e cultural de influência na sociedade e nos indivíduos, relacionando a música com os sons, inspirando o modo como os ouvintes se conectam e lidam em um ambiente.

No Brasil a música tem uma importante relevância na vida dos brasileiros, onde conforme pesquisa divulgada na (FOLHA UOL, 2018) 83% dos brasileiros são apaixonados por música, superando em 25% os Estados Unidos, Canadá e Reino Unido. Mostra-se evidente que a música está presente no dia a dia dos brasileiros em diferentes ambientes, onde 66% gostam de ouvir em momentos de descanso, 58% ao realizar tarefas domésticas, seguido de 53% que escutam em trânsito, 51% em reunião de amigos e 48% no caminho para trabalho.

No dia 11 de março de 2020 a OMS (Organização Mundial da Saúde) declarou o vírus da Covid-19 uma pandemia, recomendando isolamento entre as pessoas e enfatizando como a melhor forma de conter a disseminação do vírus. Em consequência do isolamento, o público passou a consumir mais arte, cultura, filmes, livros entre outros. Porém, o setor artístico musical foi um dos mais afetados de acordo com levantamento realizado com mais de 500 produtoras do Brasil (OUTRAS PALAVRAS, 2020; G1, 2020), indicando o anulamento de 8.141 eventos, com prejuízos estimados acima de R$ 480 milhões em 21 estados brasileiros, impactando um público previsto em oito milhões de pessoas.

Sendo assim, é fundamental que a ciência juntamente com a tecnologia estude e desenvolva novos recursos e ferramentas para aproximar o público à música, desenvolvendo aplicações e protocolos por meio da Internet das Coisas Musicais.

Qual o caminho, utilizando as tecnologias em cooperação com a música para alcançar meios de ampliar a interatividade das pessoas, minimizando o distanciamento social?

1.1 Objetivos

1.1.1 Objetivo geral

Diante da situação mundial atual, imposta pela pandemia e das recomendações da OMS, o objetivo deste trabalho consiste na elaboração de um modelo e implementação de aplicação e protocolo musical, visando permitir que grupos de pessoas compartilhem música e interajam remotamente, proporcionando a sensação de maior aproximação.

1.1.2 Objetivo específico

Para que os usuários finais utilizem os recursos da Internet das Coisas Musicais de maneira fácil e interativa, é preciso compreender e enfrentar vários desafios tecnológicos, artísticos e pedagógicos.

Isso inclui infraestrutura e protocolos de comunicação em tempo real e de forma sincronizada, com aplicações baseadas em processos e *interfaces* de músicas que propiciam a interação entre os usuários, oferecendo recursos para produção ou consumo de músicas.

A Internet das Coisas Musicais deve prover ambientes em que as pessoas possam acessar através de seus dispositivos, musicais com praticidade e simplicidade.

1.2 Metodologia

Este trabalho dissertará sobre a Internet da Coisas (ou *Internet of Things, IoT)*, mostrando os setores essenciais do domínio da IoT apresentando sua comunicação em rede com um breve resumo de seus principais protocolos.

Logo após, realizar uma pesquisa dos conceitos, abordando um levantamento da Internet das Coisas Musicais, elencando as principais dificuldades encontradas em promover soluções da Internet das Coisas Musicas, apresentando os requisitos de seus dispositivos apontando como é o Ecossistema da Internet das Coisas Musicais.

Para enriquecer o trabalho, serão apresentados alguns cenários demonstrando necessidades das pessoas interagirem com a música em grupo de forma compartilhada e em tempo real. Por intermédio dos cenários, mostraremos uma implementação prática com um modelo de aplicação e um protocolo em que os usuários, a partir dos seus dispositivos, consigam conectar de forma simples e prática as músicas, captando e realizando ações de forma interativa, com outras pessoas, promovendo encontros virtuais, onde todos os participantes possam estar sincronizados e ouvindo a mesma música.

A aplicação e o protocolo contará com a comunicação em tempo real, garantido a sincronicidade dos dados entre os usuários, provendo uma *interface* simples e objetiva, viabilizando uma conexão segura entre os usuários e o servidor. Ao final deste trabalho, esperamos mostrar que a solução apresentada, com a aplicação e o protocolo são totalmente implementáveis e adaptáveis a outros cenários.

2 Referencial Teórico

Neste capítulo serão expostos os conceitos da Internet das Coisas, descrevendo suas principais aplicações e implementações, assim como, protocolos e padrões utilizados. Em seguida, discutiremos sobre Internet das Cosias Musicais contextualizando o tema apontando suas características e desafios.

2.1 Internet das Coisas (IoT)

O termo Internet das Coisas ou do inglês *IoT - Internet of Things* foi popularizado em 1999 (GALEGALE, 2016), mas, ainda nos anos 80 o tema já era abordado pela união Europeia (SINGER, 2012). Pela descrição da IEEE, a Internet das Coisas é definida como uma rede de itens, incorporada com sensores que podem ser conectados à Internet (IEEE, 2019). No entanto, esta definição pode ser mais profunda quando abordado a Internet das Coisas, que a definida pela IEEE, uma vez que abrange uma numerosa quantidade de objetos e dispositivos conectados entre si ou pela Internet, através de uma identificação única para cada objeto que pode operar de maneira inteligente, sem a interferência das pessoas, fazendo com que troquem e processem informações e dados de qualquer lugar em qualquer hora (ROSA, 2019).

Para uma melhor percepção, a Internet das Coisas pode ser vista como um segmento a nossa Internet mundial, conectando os mais diversos dispositivos (coisas) heterogêneos. Estudos consideram o termo “coisas” como conjuntos de tecnologias e objetos em sistemas embarcados que possibilitam a comunicação, em sua maioria, através das redes de comunicação ou da Internet, viabilizando a interação das coisas entre si (DIREITOS DA COMUNICAçãO, 2019). Na Figura 1 percebemos um pouco do universo da Internet das Coisas, onde notoriamente observamos os mais variados objetos heterogênicos conectados, possibilitando, interagirem uns com os outros, exemplificando o universo da Internet das Coisas.

Figura 1 – Internet das coisas (OPICE BLUM, 2017)

Recentemente, com os crescentes avanços tecnológicos nas redes de sensores sem fio, sistemas embarcados, microeletrônica, comunicação móvel e computação ubíqua, a expansão da Internet das Coisas vem se tornado mais presente. Sua expansão acarretou muitas adversidades na comunicação entre objetos heterogêneos, protocolos e limitações de *software* e *hardware* (PIRES, 2015; WANG, 2015).

Nos últimos anos a Internet das Coisas, pelo potencial apresentado, torna-se cada vez mais objeto de pesquisas, tanto no meio acadêmico quanto no mercado industrial, sendo este termo amplamente buscado no Google, como mostra a Figura 2, ganhando enorme popularidade entre as consultas, possivelmente pela sua disseminação e grande crescimento em cima de sua expectativa (SANTOS, 2016). Esta crescente, faz com que grandes e pequenas organizações busquem tecnologias e conectividade na Internet das Coisas, ainda que os investimentos sejam altos, em pouco tempo seu retorno é alcançado conforme informação divulgada na (FORBES, 2019).

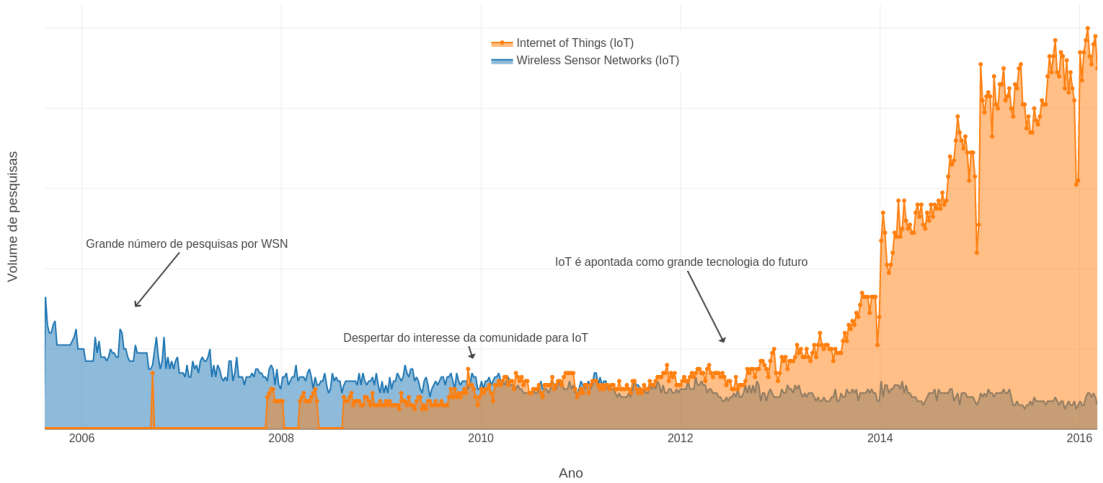

Figura 2 – Volume de pesquisas no Google sobre Internet das Coisas (SANTOS, 2016)

2.1.1 As coisas da Internet das Coisas

Muito provavelmente, é possível pensar na Internet das Coisas a partir de diferentes dispositivos que podem funcionar conectados por rede.

- **Telefones inteligentes**: os telefones inteligentes incorporam inúmeras funções como: máquinas fotográficas, recursos de localização através de GPS, acesso à Internet, comunicação e interações em redes sociais, *e-mail*, mensagens e conversas instantâneas com possibilidade de chamadas de voz e vídeo, praticidade em compras *online*, leitura de livros, artigos e notícias, diversidade de atrativos como jogos e musicas, notificações de lembretes e calendários, viabiliza assistir vídeos e filmes, utilização em meios corporativos e empresarias através dos aplicativos com as mais variadas soluções, substituindo diversas funções que antes só eram realizadas por computadores (SALES, 2018).

- **Cidades Inteligentes (*Smart City*)**: as cidades inteligentes englobam diversas aplicações e dispositivos conectados que coletam, processam e enviam dados e informações. Essa interação possibilita, por exemplo, o gerenciamento e monitoramento do trânsito de veículos, controle da iluminação pública, resposta e planos de contingência a situações climáticas como enchentes, alagamentos e incêndios, e até antecipação de possíveis catástrofes derivadas de monitoramento, como rompimento de barragens.

- **Carros Inteligentes**: uma realidade envolvendo as cidades inteligentes são os carros conectados e autônomos, que principalmente, com a implementação do 5G as indústrias automotivas e de transporte estão revolucionando a forma com os carros

estão sendo construídos, deixando de ser peças sem comunicação com o mundo ao seu redor e passando a se conectar com o entorno (INTEL, 2019).

- **Industria Inteligente e Industria 4.0**: a Internet das Coisas desempenhara um papel fundamental nos processos industriais. Hoje com o uso da aprendizagem de máquina, provendo capacidade de conhecimento de rotinas, processos, controles, monitoramento e automatização, a IoT mudará de forma significativa a maneira como os equipamentos industriais e dispositivos inteligentes se portarão na indústria, possibilitando tomadas de medidas importantes, como no controle de fluxo, pressão, temperatura, gás, água e até aprendizagem para decisões inesperadas e de forma emergencial dispondo uma maior sustentabilidade, eficiência e confiabilidade (ROSA, 2019).

- **Casas Inteligentes**: As casas inteligentes se baseiam em tecnologias e dispositivos inteligentes com a finalidade de propiciar aos moradores, um maior conforto e qualidade de vida. Isto é alcançado através da interação de dispositivos eletrônicos, que viabilizam a tomadas de decisões baseadas nas rotinas diárias, de alimentação, limpeza, saúde entre outros (ROSA, 2019).

- **Instrumentos Musicais inteligentes**: os *Smart Instruments* são instrumentos musicais que podem enriquecer a Internet das Coisas Musicais, acrescentando o aprimoramento de sensores e atuadores, incorporando inteligência computacional com mecanismos de processamento e síntese de som e oferece conexão ponto a ponto sem fio. Um de seus recursos fundamentais é sua interoperabilidade possibilita troca de informações musicais diversificada entre os objetos musicais, tecnologias vestíveis, *smarts phones*, equipamentos de realidade aumentada, realidade virtual, áudio e iluminação em larga escala para salas de concerto(TURCHET, 2017).

2.2 Internet da Coisas e as redes

A Internet das Coisas é combinada por conjunto de diversas tecnologias complementares, que com sua integração, proporcionam a possibilidade dos objetos comunicarem-se uns com os outros e com as pessoas. Esta conexão se dá por redes de computadores, que são compostas de diversos componentes conectados entre si, através de um meio físico. Esses componentes se comunicam seguindo padrões de comunicações conhecidos como protocolos. Os protocolos possibilitam que as mais variadas categorias de dispositivos troquem informações mutuamente (TANENBAUM, 2003) e (KUROSE; ROSS, 2006).

Em seu início, as redes de computadores eram utilizadas principalmente para conectar computadores. Com o avanço da tecnologia e a popularização de dispositivos com suporte a conexão de rede, as redes podem ser compostas por celulares, TVs, *notebooks*,

consoles de jogos, câmeras de vigilância, alarmes, telefonia, entre outros. Conforme (FORBES, 2019), a estimativa é que até o final de 2020 teremos 26 bilhões de dispositivos conectados no mundo.

Diferente da comunicação de dispositivos de redes tradicionais, a construção da Internet das Coisas pode ser composta primeiramente pela **identificação** dos objetos presentes na rede, de forma que possibilite a comunicação com a Internet, outros dispositivos e protocolos com identificação única, como visto na Figura 3 (SANTOS, 2016). As informações dos objetos precisam ser capturadas e compreendidas para realizar um processamento inicial e em seguida enviar as informações para um tratamento mais específico. Os **Sensores** são utilizados para captar informações sobre o ambiente e os **Atuadores** são responsáveis por compreender o ambiente e tomar alguma decisão com os dados repassados pelos **Sensores**. A **comunicação** está intimamente relacionada em como os objetos irão se conectar e trocar informações entre si. A fase de **computação** fica responsável por realizar o processamento e execução de algoritmos nos objetos. As coisas da Internet das Coisas podem prover diversos tipos de **serviços** como mapeamento do ambiente e detecção de alguma irregularidade ou anomalia no ambiente por meio de sensoriamento. A **semântica** é responsável pelo tratamento dos dados coletados de forma que as informações sejam úteis para a utilização dos recursos (SANTOS, 2016).

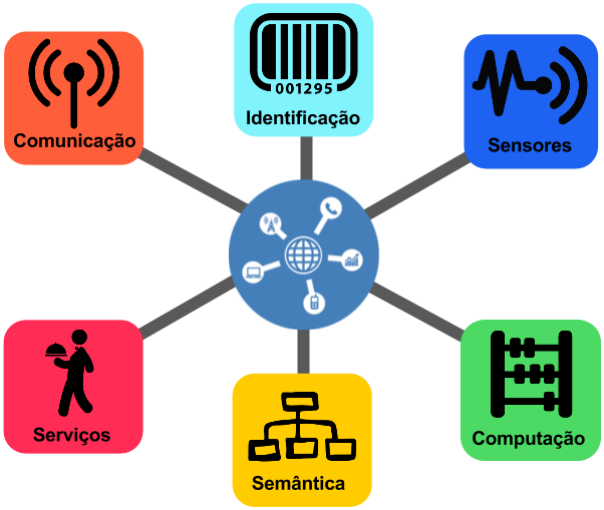

Figura 3 – Modelo para construção Internet das Coisas

Para a comunicação entre dispositivos, os seguintes modelos podem ser utilizados:

- **Modelo Cliente-Servidor**: um dos modelos mais utilizado nas redes tradicionais atualmente é o modelo cliente/servidor onde, através da troca de mensagens, o servidor provê algum serviço para os clientes (TANENBAUM; WETHERALL, 2011) .

- **Modelo Peer-to-Peer (P2P)**: as redes *peer-to-peer* destacam-se por serem descentralizadas, dispensando a utilização de um equipamento servidor, de modo que a comunicação ocorre de forma direta entre os dispositivos. Como na Internet das Coisas em sua maioria a comunicação deverá ser direta com os objetos ou dispositivos, este modelo de rede se torna essencial em sua aplicação.

- **Modelo Publisher/Subcriber**: no modelo cliente/servidor o cliente sempre inicia a comunicação com uma requisição ao servidor, no modelo *Publisher/Subscriber*, similar ao modelo *peer to peer*, ambos podem se manifestar a qualquer momento o interesse em se conectar. Este é o modelo utilizado pelo protocolo da IoT.

Além destes modelos de comunicação, os seguintes protocolos e tecnologias de rede estão fortemente ligados com a conexão de dispositivos IoT.

2.2.1 Wi-Fi

O padrão Wi-Fi é definido pela IEEE 802.11 (WI-FI ALLIANCE, 2019) e sua transmissão é realizada por radiofrequência propagada através do ar, onde seu alcance varia conforme sua frequência e obstáculos como paredes, árvores, outros dispositivos com mesma frequência ou canal no caminho do sinal. Desde sua criação em 1999 pela Wi-Fi Alliance a rede Wi-Fi já passou por diversas versões com variações na frequência, taxas de transmissão e alcance (STEFANUTO, 2016).

As redes Wi-Fi destacam-se, por compor uma imensa quantidade de equipamentos com sua tecnologia incorporada, tais como, televisões, computadores e similares, telefones celulares, vídeo games, impressores, geladeiras, caixas de som, equipamentos industriais e domésticos entre outros. Por sua ampla disponibilidade acaba sendo uma ótima solução para comunicação da Internet das Coisas e Internet das Coisas Musicais, ainda assim, requer um melhor comportamento em *hardware* pela numerosa quantidade de protocolos e recursos necessários, inclusive requer um alto consumo de energia, impossibilitando dispositivos limitados.

2.2.2 Bluetooth

O *Bluetooth* é padronizado pela IEEE 802.15.1 com objetivo de conectar redes pessoais sem fio de curtas distâncias (WPAN) se caracterizando pelo seu baixo custo e baixo consumo de energia. A transmissão dos dados em uma rede *Bluetooth* é realizada por radiofrequência, semelhante ao Wi-Fi, operando na faixa de frequências entre 2.4 GHz e 2.485 GHz.

Os dispositivos *Bluetooth* são conectados entre si através do emparelhamento. Quando temos dois ou mais dispositivos conectados, esse conjunto é chamando de *pi-*

conet, sendo que o dispositivo que iniciou a conexão é denominado mestre (*master*) e os demais escravos (*slaves*). Em uma rede *piconet* é possível acomodar até 8 dispositivos, porém, utilizando um recurso conhecido como sobreposição esta quantidade de dispositivos pode ser expandida de forma que um membro de uma *piconet*, seja ele mestre ou escravo, se conecta em outra *piconet* podendo assim transmitir dados entre as redes. Este recurso é chamado de *scatternet*. A Figura 4 ilustra o modelo da criação das *piconet* e das *scatternet* (STEFANUTO, 2016).

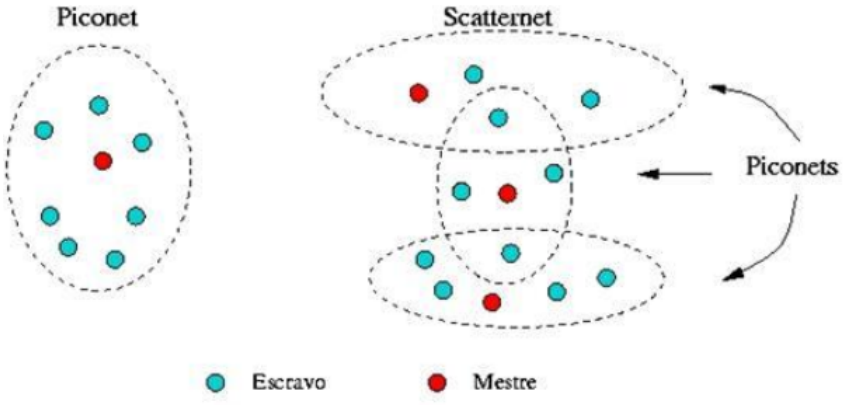

Figura 4 – Scatternet formada por outras piconets. Adaptado de (STEFANUTO, 2016)

Assim como o Wi-fi, o *Bluetooth* passou por diversas modificações e melhorias desde sua concepção. Na versão 1.2 a velocidade de transmissão era limitada a 1 Mbit/s. Com a evolução do protocolo e do *hardware*, houve um aumento significativo nesta velocidade, chegando em 50Mbit/s na versão 5.0.

2.2.3 ZigBee

O padrão ZigBee se destaca fortemente na Internet das Coisas, principalmente pela sua baixa vazão, permitindo uma considerável redução no consumo de energia. A ZigBee Alliance (ZIGBEE, 2019) é encarregada por administrar o protocolo garantindo sua interoperabilidade entre os mais diversificados dispositivos. Sua capacidade de transmissão pode chegar 250 kbps, suportando operar nas seguintes frequências: 2.4GHz, 868 MHz e 915MHz (SANTOS, 2016).

2.2.4 SigFox

Uma das preocupações da Internet das Coisas está relacionada com conexões sem fio e ao consumo de energia, com isso, a SigFox uma empresa francesa, desenvolveu uma solução de comunicação sem fio, baseada em *low-power wide-area network* (LPWAN), fazendo com que o *SigFox* destaque-se por operar com baixo consumo de energia, principalmente quando comparado as redes de celular tradicionais.

2.2.5 LoRaWAN

O LoRaWAN *(Long Range Wide Area Network)* (LORA ALLIANCE, 2019) traz em seus princípios parâmetros fundamentais voltados para Internet das Coisas, viabilizando criação de redes em escalas geograficamente distribuídas com o baixo consumo de energia, baixos custos, promovendo seu desenvolvimento externo, sendo móvel, flexível e lida com as questões de localidade, além de propiciar segurança em comunicações *full-duplex* (LORA ALLIANCE, 2019; LINK LABS, 2019; ROSA, 2019; SANTOS, 2016).

Essa tecnologia utiliza técnicas de espalhamento espectral, sendo um protocolo com uma alta área de cobertura, conseguindo garantir o baixo consumo possibilitando que dispositivos mais limitados consigam manter sua bateria por mais de ano. Opera nas frequências: 109 MHz, 433 MHz, 866 MHz e 915 MHz, conseguindo obter uma taxa de transmissão de até 50 kbps, com um alcance de até 5 km em áreas urbanas e 45 km em áreas rurais, chegando a uma capacidade de até 1 milhão de dispositivos conectados. O protocolo LoRaWAN proporciona adaptação ao protocolo IPv6, suas redes são associadas a diversos *gateways* e não apenas a um único, primordial aos requisitos da Internet das Coisas, como em controle inteligente de iluminação, lançados em altas altitudes para monitoramento de fortes chuvas, principalmente granizo, rastreamento de animais e crianças, medidores de energia e até monitoração de abalos sísmicos (LORA ALLIANCE, 2019; LINK LABS, 2019; ROSA, 2019; SANTOS, 2016).

2.2.6 Redes de dados em telefonia celular

A tecnologia da rede 5G surgiu em 2015 na Conferência Mundial de Radiocomunicações, com melhorias da tecnologia anterior (4G) e a finalidade de fornecer uma Internet com alta velocidade para dispositivos móveis, variando entre de 1 a 10 Gbps. Consequentemente, requer melhorias das tecnologias das redes existentes para que sejam capazes de proporcionar a conexão e desempenho que o 5G promete. A rede 5G prover em suas especificações, requisitos de conexões com dispositivos da Internet das Coisas em latência, eficiência e capacidade de vários dispositivos simultâneos (ROSA, 2019).

Com a utilização das redes 5G, que caracteriza uma conexão até 20 vezes mais rápida que as redes atuais, com sua abrangência em alcance e expansão, sendo uma alternativa para Internet das Coisas, fazendo com que a Internet das Coisas chegue aos lugares mais remotos e com maior disponibilidade dos serviços.

Para obtenção das altas taxas de transmissão, pode se utilizar células com pequenas áreas de cobertura, com o propósito de aumento da capacidade de tráfego e nas taxas de dados que serão trafegadas pelos usuários, evitando os congestionamentos nas áreas urbanas. Para melhoria da intensidade do sinal transmitido, pode-se usar a agregação das tecnologias M-MIMO e *Beamforming* o que proporciona uma maior taxa de transmissão

com um melhor desempenho, com relação às tecnologias anteriores. De acordo com *International Telecommunication Union (ITU)* a comunicação aceitável do 5G deve ser de 1 ms (IUT-R, 2019) enquanto os sistemas de celulares possuem latência de 10 ms, em alguns casos em comunicação D2D *(device-to-device)* da Internet das Coisas, a latência acaba sendo crucial (ROSA, 2019). O ITU estabelece que mais de um milhão de dispositivos devem estar conectados por quilômetros quadrados. Portanto, as redes 5G se tornam torna uma solução viável para atender a expansão e demandas da Internet das Coisas (ROSA, 2019).

2.2.7 Protocolo MQTT e Machine-to-Machine (M2M)

Um dos principais padrões para as comunicações da Internet das Coisas é o protocolo MQTT (Transporte de Telemetria da Fila de Mensagens). Este protocolo foi desenvolvido para aplicações em que objetos se comunicam e trocam informações com outros objetos em um modelo conhecido como Machine-to-Machine. Além de utilizar o sistema de troca de mensagens *publish-subscribe* (publicação e assinatura), este protocolo provê um mecanismo que viabiliza a garantia de entrega, respondendo perfeitamente a dispositivos móveis com baixo consumo de energia (MELO, 2018).

O MQTT é um protocolo assíncrono na troca de mensagens que visa ser leve e ajustável, facilitando os desenvolvedores de aplicações e serviços para Internet das Coisas. O MQTT suporta implementação em dispositivos e equipamentos com limitação de latência e largura de banda. Por ser assíncrono o dispositivo não precisa aguardar a resposta de uma mensagem enviada, podendo assim continuar executando outras atividades ou solicitações, e assim que a mensagem for respondida à mesma pode ser analisada e processada (SKALEX, 2017; SANTOS, 2016).

Usualmente o protocolo MQTT estabelece dois tipos de funcionalidades em rede, um intermediador de mensagens e um número de clientes. Neste modelo, o servidor é chamado de *Broker* e é responsável por receber as mensagens que foram enviadas pelos clientes e encaminhá-las para seus respectivos clientes de destino. Para as aplicações da Internet das coisas, um cliente pode ser qualquer dispositivo ou equipamento que requisita algum serviço, ou interação com o servidor *Broker* (SKALEX, 2017). A Figura 5 descreve um modelo de MQTT com publicação e assinatura (SKALEX, 2017; PROJETO IOT, 2018). Neste modelo, temos um sensor pulicando informações de temperatura para 2 dispositivos inscritos no tópico.

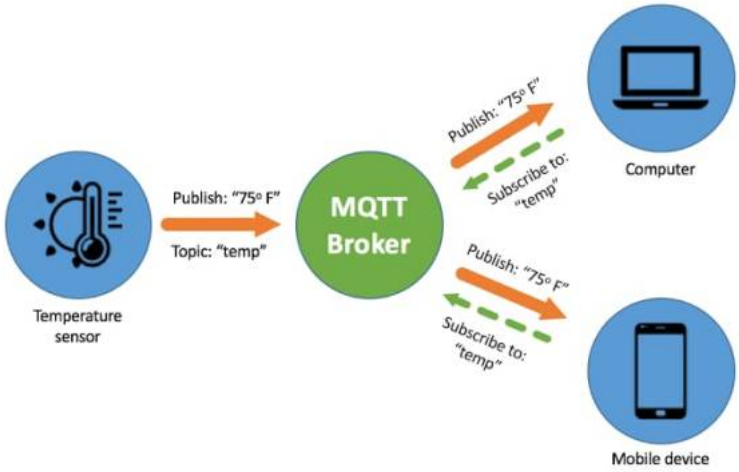

Figura 5 – Modelo Publisher/Subcriber com MQTT

As mensagens enviadas via protocolo MQTT são consideradas "leves" por possuírem um cabeçalho simples e otimizado. Além de ser um protocolo flexível, sua organização em tópicos permite que somente os clientes inscritos interajam com as mensagens enviadas (SKALEX, 2017).

2.3 Internet das Coisas Musicais (IoMusT)

A Internet das Coisas Musicais ainda é um assunto desafiador e pouco discutido, contudo, com o avanço da Internet das Coisas, este cenário chama a atenção e promovido, diversas discussões. Não há um conceito preciso de Internet das Coisas Musicais, podemos, apresentar como sua definição sendo "*um dispositivo de computação capaz de detectar, adquirir, processar ou atuar e trocar dados que atendam a um propósito musical*" ou como "*conjunto de interfaces, protocolos e representações de informações relacionadas à música que permitem serviços e aplicativos que atendem a um propósito musical com base nas interações entre seres humanos e Coisas musicais ou entre as próprias coisas musicais, no domínio físico e/ou digital*" (TURCHET, 2018).

De forma geral, a Internet das Coisas Musicais pode ser vista como uma extensão da Internet das Coisas, seguindo muitos conceitos no contexto musical. Neste modelo, diversos componentes conectados podem interagir entre si, pessoas com coisas musicais, objetos musicais com outros objetos musicais, público com objetos, artistas e público em ambientes locais e/ou remoto.

Outro emprego para Internet das Coisas Musicais seria sua utilização em gravação remota em estúdios de gravações, utilizando ferramentas de acompanhamento e integração com outros estúdios permitindo gravações de forma compartilhada, com instrumentos e acessórios necessários para produção de conteúdo musical distribuído geogra-

ficamente (LACEY, 2015). A Figura 6 demonstra um exemplo de uma ferramenta (OHM STUDIO, 2013).

Figura 6 – Exemplo de ferramenta gravação distribuída (LACEY, 2015)

Além disso, poderiam ser utilizadas misturas ao vivo de mixagens de áudio através da Internet das Coisas Musicais, possibilitando a transmissão ao vivo de um *show* em lugares diferentes em tempo real (LACEY, 2015).

2.3.1 Os dispositivos da IoMusT

Ao passo que a Internet das Coisas comporta os mais variados objetos conectados, a Internet das Coisas Musicais se específica no conjunto de objetos e dispositivos físicos musicais. Um objeto musical engloba dispositivos inteligentes, como pulseiras, relógios, dispositivos vestíveis, dispositivos com capacidade de controlar, capturar ou rastrear dados, ou informação, como gestos, movimentos, informações fisiológicas, além de *feedbacks* com propósitos musicais.

Entre os dispositivos possíveis para este contexto, estão equipamentos musicais embutidos com sensores aumentados capazes de reprodução de sons com proximidade das mãos e movimentos, como a guitarra *Sensus Smart Guitar*, apresentada na Figura 7 (ELK AUDIO, 2019; GUITAR MODERNE, 2016; HAZZARD, 2014).

Figura 7 – Sense Smart Guitar (GUITAR MODERNE, 2016)

2.3.2 O Ecosistema da IoMusT

A infraestrutura da Internet das Coisas Musicais deve ser capaz de comportar um ecossistema de dispositivos heterogêneos que interligue os mais variados dispositivos musicais, assim como, interação dos músicos e com o público em tempo de real, com baixa latência, alta confiabilidade e excelente sincronismo, seja no ambiente local, entre ambientes diferentes ou remoto (TURCHET, 2017; FILHO, 1998), como exemplificado na Figura 8.

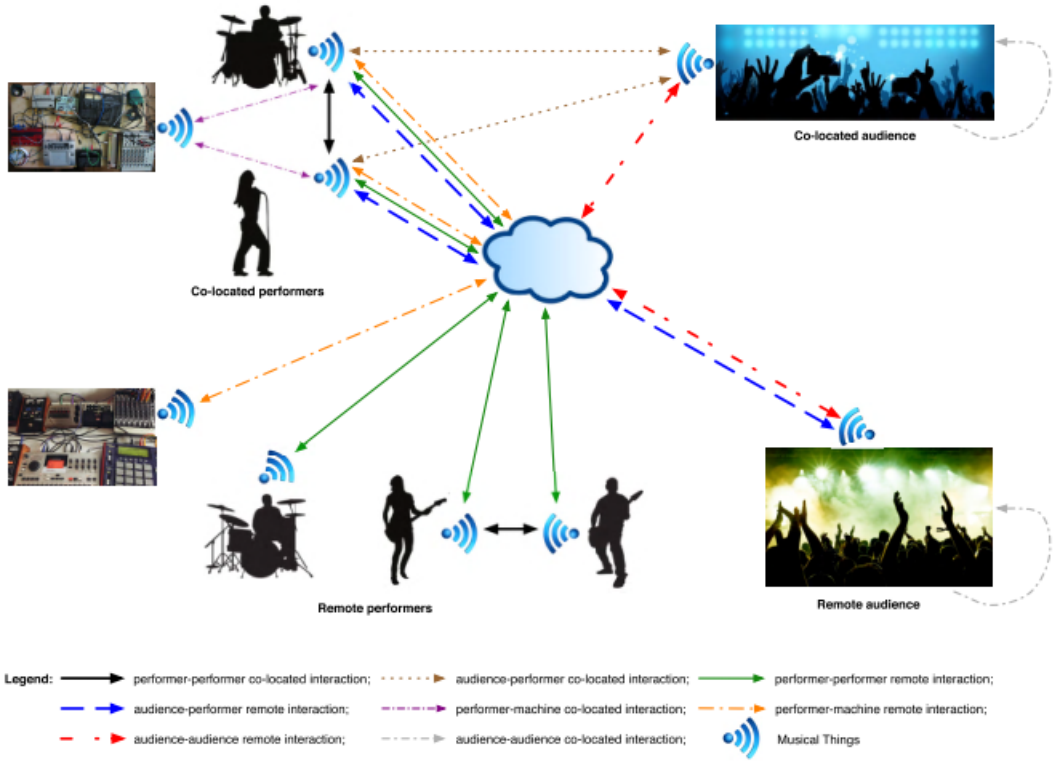

Figura 8 – Ecossistema IoMusT (FILHO, 1998)

Como exemplo, a participação do público com interações visuais e sonoras, por meio da manipulação de objetos virtuais ou através de *smarts phones* pode criar um ambiente de realidade interativa permitindo a exibição do conteúdo virtual no palco. Citando um caso análogo de realidade virtual em 360 graus, mostrado na Figura 9, temos o público posicionado no centro de um palco em um festival de orquestra da Quinta Sinfonia de *Beethoven* (O'BANNON, 2015; O'BANNON, 2018).

Figura 9 – Realidade virtual na orquestra (BRADY DYER, 2016)

Para exemplificar melhor o âmbito da Internet das Coisas Musicais, podemos considerar para sua integração em três grupos tecnológicos:

- **Coisas Musicais**: na abordagem da Internet das Coisas Musicais, crescem novas tendências relacionadas a dispositivos musicais inteligentes com sensores e atuadores capazes de conectar-se a Internet e que poderão impactar a relação entre espectadores e artistas/músicos, alcançando os desafios com o sincronismo e a interoperabilidade. As coisas musicais podem ser usadas para reprodução de conteúdo musical ou aprendizagem de experiências musicais e controle das coisas que podem estar conectadas a uma rede local, ou remota. Coisas musicais podem ser instrumentos ou dispositivos inteligentes, ou qualquer outro dispositivo em rede, utilizado para controlar, gerar, rastrear ou processar conteúdo musical.

- **Conectividade**: através da infraestrutura de redes locais e remotas que os objetos e dispositivos das coisas musicais mantém sua comunicação, seja em rede sem fio ou na Internet, através dos recursos de *hardware* e *software*, bem como as tecnologias, os padrões e protocolos de comunicação. A comunicação é crucial na Internet das Coisas Musicais, tendo como requisitos importantes o sincronismo, conectividade em tempo real, comunicações com baixa latência, alta confiabilidade, segurança e qualidade, são exigências primordiais, principalmente em apresentações com interações ao vivo, com uso de realidade virtual e realidade aumentada, entre outros (TURCHET, 2018).

- **Aplicativos e serviços**: os aplicativos auxiliam a interatividade entre os objetos e humanos, provendo diversos recursos e serviços para os usuários e músicos. Os aplicativos podem ser desenvolvidos em APIs para Web com propósitos musicais que permitam a interação dos usuários com os músicos ou com os objetos (TURCHET, 2018).

Quando aplicada a Internet das Coisas Musicais a comunicação e interação entre músicos, instrumentos e público ativo e passivo terão uma direção totalmente ampliada, possibilitando cenários de alcance em grandes escalas de públicos e músicos. Como, por exemplo, participações em ambientes *online* com *interfaces* virtuais, possibilitando centenas ou milhares de participantes com sensações próximas, como se estivessem participando ao vivo ou no palco utilizando recursos da realidade aumentada e realidade virtual, partilhando de objetos e instrumentos musicais inteligentes, como iluminação, sugestões de conteúdo ou ritmo musical, que com a coleta e análise das informações, novas experiências podem ser criadas ou melhoradas (TURCHET, 2018).

2.3.3 As possibilidades da IoMusT

Algumas possibilidades de interações podem ser proporcionadas pela Internet das Coisas Musicais, por meio dos diversos cenários participativos co-localizados em uma sala, evento ou *show*, ou ainda em cenários abertos reunindo diversos participantes em um ambiente totalmente virtual. Ainda quando combinado a outras tecnologias como Realidade Virtual e Realidade Aumentada, o dinamismo de criação, experiência e aprendizagem musical se expande, permitindo ao público uma participação mais próxima, com a sensação de estar ao vivo no evento ou participando diretamente com os músicos na apresentação, ou concerto. Por meio dos recursos oferecido pela realidade aumentada, as performances interativas e participativas em consertos, por exemplo, podem ser aprimorados com experiências entre a participação da plateia e dos músicos em ambientes virtuais envolvendo dispositivos interativos.

A Internet das Coisas Musicas estende os benefícios, recursos e aperfeiçoamentos entre artistas e músicos, equipamentos e dispositivos, interação e o comportamento do público, e propõe melhorias e experiências entre os fornecedores de equipamentos musicais. Como exemplos, utilização de *smartphones* para controlar o som de instrumentos musicais inteligentes, ou ainda, formas de que através de relógios ou pulseiras inteligentes, músicos possam interagir. O público, os músicos e os dispositivos podem interagir durante um *show*, de forma que os músicos tenham um *feedback* da plateia através de capturas fisiológicas, obtendo respostas emocionais da plateia, de modo a controlar o timbre dos instrumentos ou comportamento dos equipamentos do palco, como iluminação, telões, entre outros. Da mesma forma que os sons dos instrumentos inteligentes podem ser enviados através

de vibrações em forma de ritmos para as pulseiras, vestimentos inteligentes ou relógios inteligentes do público (TURCHET, 2017; GREATEST HIT STATION, 2008).

A informações capturadas e processadas pela Internet das Coisas Musicais podem ser utilizadas pelos músicos para aprendizagens e experiências de gostos do público e clientes no fornecimento de serviços específicos para melhoria de equipamentos, dispositivos, produtos e conteúdo musical (TURCHET, 2017).

As diversas informações capturas e compreendidas pela Internet das Coisas Musicais podem ser usadas para aprendizagem dos músicos, entendimento e gosto dos participantes e clientes no fornecimento de serviços específicos, melhoria de equipamentos, dispositivos e produtos, entre outros (TURCHET, 2017).

2.3.4 Desafios da Internet das Coisas Musicais

Com o surgimento e crescimento da Internet surgiram muitas oportunidades envolvendo integração de músicas com redes, explorando domínios das apresentações musicais em redes (*Networked Music Performances* - NMPs), com criação de tecnologias de hardware e software como *codecs*, estruturas, protocolos de baixa latência para as apresentações musicais em redes. No entanto, alguns desafios ainda estão presentes para tornar esta solução amplamente disponível.

2.3.4.1 Protocolos de comunicação

A infraestrutura tecnológica precisa de alguns avanços para conseguir prover comunicação entre os dispositivos e conectividade necessária para as coisas musicais. Isto engloba sensores capazes de capturar áudio e informações específicas de um emissor e um receptor que compreenda a leitura dessas informações conseguindo processá-las e respondê-las, permitindo a interação e sincronismo das coisas e dos envolvidos. Para isso, precisamos de uma rede com componentes e dispositivos que reaja com eficiência, baixa latência, confiabilidade e segurança.

Geralmente, a Internet das Coisas Musicais propõe criações e adaptações das camadas e protocolos da pilha TCP/IP, podemos utilizar como exemplo, para garantir as baixas latências na camada física, as mensagens serem essencialmente curtas podendo ocasionar restrições nas taxas de dados. Na camada de rede, os protocolos de roteamentos precisarão serem otimizados para atrasos baixos e não para altas taxas de transferências de dados, além de métodos de decisões de roteamento distribuído, capazes de minimizar o atraso. Novas técnicas de controle e gerenciamento baseados em otimização deverão ser capazes de configurar e alocar os recursos adequados do plano de dados em diferentes domínios para criar serviços de ponta a ponta com baixa latência (TURCHET, 2017).

Nas abordagens de análise de dados devem ser implementados técnicas e ferra-

mentas capazes de armazenamento e processamento de grandes quantidades de dados relacionados a música, como *Big datas*, *web* semântica, ontologias, de modo que, quando necessário e solicitada as informações, estejam disponíveis.

2.3.4.2 Quebras de paradigma

Algumas quebras de paradigmas devem ser exploradas, como:

- Meios com que os integrantes da plateia tenham interatividade diretamente com o artista/músico no palco;
- Recursos de métodos de composição de músicas que proporcionam autonomia aos integrantes, de forma individual ou em conjunto, com diversos indivíduos, provendo sensações mútuas, de modo que estivessem interagindo ao vivo, sem comprometer a qualidade e performance, visando a diversidade de cada indivíduo em sua percepção de sensibilidade, preferência e aptidão;
- Mecanismos que possibilitam, criação e ensaios de músicas e *shows* de forma genérica, utilizando a tecnologia para integrar cada expressão participativa individual e em grupos;
- Questões relacionadas à segurança, privacidade das informações e desempenho (TURCHET, 2017).

2.3.4.3 Desempenho

O desempenho musical em redes é crucial para expansão de aplicativos da Internet que remodelará a forma de interação musical propiciando através das redes, interações entre músicos remotos. Para isso, é indispensável assegurar o desempenho de forma que alguns requisitos necessários para execução de áudios em redes sejam estabelecidos, como atraso de rede, qualidade de áudio, sincronização satisfatória entre os músicos, além da coleta e análise de experiências interativas.

Assegurar captura e transmissão de fluxos de áudio pela Internet, ainda é um desafio, pois a sincronização e o desempenho da rede podem ser comprometidas pelos atrasos de pacotes e atrasos de processamento com os músicos estando em diferentes locais geograficamente distribuídos.

2.3.4.4 Baixa latência

Mesmo com a Internet móvel 5G oferecendo altas taxas de transmissão de dados, a latência até então acaba sendo objeto de muitos estudos, devido as rigorosas exigências de latência da Internet das Coisas Musicais, principalmente se tratando de transmissões ao vivo com interconexões de diversos instrumentos musicais.

Por outro lado, outros mercados estão estimulando, assim como, na Internet das Coisas Musicais, a melhoria das taxas de transmissão em rede, como exemplo, classes emergentes de serviços, como jogos digitais, telepresença, realidade virtual, realidade aumentada e controle de missão crítica. Uma alternativa seria utilização de estudos envolvendo "*Tactile Internet*" que operam com uso de latências bastante baixas com atrasos de pouquíssimos milissegundos (não excedendo 5ms), abrangendo recursos para tecnologias com frequências sem fio na faixa de 10 a 300 GHz com taxas de dados em giga bits por segundo em distâncias curtas, sendo crucial agregação dessa tecnologia nos equipamentos da Internet das Coisas Musicais (TURCHET, 2017).

A Internet das coisas musicais considera diversas situações que exigem baixa latência e atrasos como captura e processamento de sons, toques e notas de equipamentos musicais, transmissão das vozes dos músicos, *feedback* visual de movimentos e gestos de outros integrantes em tempo real. Sendo assim, é necessário alcançar requisitos rigorosos de latências na casa de algumas dezenas de milissegundos, sendo estimados o limite de atrasos entre 20 e 30 ms, podendo variar de acordo com o cenário e distância (ROTTONDI, 2016).

2.3.4.5 Sincronização

Como na Internet das Coisas Musicais a comunicação em sua maioria ocorrerá em topologias remotas em locais distribuídos ou por meio da Internet, a sincronização acaba sendo mais um requisito fundamental a ser analisado para garantir seu desempenho satisfatório no fluxo de dados emitidos principalmente pelos objetos e dispositivos que não compartilham o mesmo relógio. Algumas técnicas podem ser adotadas para alinhamento na transmissão dos fluxos de modo ao envio de dados de áudio e/ou vídeo e controle sem comprometer a qualidade, além do controle de perda de pacotes de rede ou para resolver o problema de tremulação de rede, evitando um pacote atingirem seu destino após o tempo de reprodução programada (ROTTONDI, 2016).

Para a Internet das Coisas Musicais, podem ser utilizado o controle com pacotes de áudios e/ou vídeos coesos com a data e hora, com os protocolos RTP / RTCP e NTP (LAZZARO; WAWRZYNEK, 2001; XYLOMENOS, 2013). Há algumas alternativas para sincronização dos relógios de forma que pacotes de controle com as informações do relógio principal podem anexados aos dados de *streaming* de áudio e enviado a receptor para que os relógios sejam sincronizados e enquanto a sessão estiver estabelecida o NTP envia atualizações periódicas. Contudo, o protocolo NTP para sincronismos de relógios possui atrasos consistentes em comunicações remotas, podendo comprometer a comunicação exigida nas músicas em tempo real. Já em comunicações locais através de redes LANs, o atraso de 0,2 ms não compromete a performance da Internet das Coisas Musicais. Algumas possibilidades já exploradas, quando relacionado as redes de longas distâncias, seriam so-

luções sustentadas pelo GPS (*Global Positioning Satellite*, ou Sistema de Posicionamento Global) e CDMA (*Code Division Multiple Access*, ou Acesso Múltiplo por Divisão de Código), que alcançam precisões aceitáveis, sendo na ordem de microssegundos.

A latência, no entanto, depende do tipo de dado a ser compartilhado. Em redes clientes/servidor que utilizam dados MIDI, por exemplo, um nó mestre envia o relógio a todos os clientes, onde os valores são baseados na frequência de *clock* do *hardware* do nó mestre (ROTTONDI, 2016).

2.3.4.6 Integração e facilidade de participação

Garantir a comunicação e interoperabilidade dos mais diversos dispositivos heterogênicos musicais também é um desafio, tendo em vista que não há compatibilidade total entre instrumentos eletrônicos, não eletrônicos, gestuais, áudios e vozes. Alguns protocolos proprietários embutidos em alguns equipamentos possibilitam essa comunicação, outros já usam a *Interface* Digital de Instrumentos Musicais (MIDI - *Musical Instrument Digital Interface*) para transporte de mensagens em instrumentos eletrônicos (ROTTONDI, 2016).

2.3.4.7 Redes em tempo real

Um recurso para Internet das coisas musicais seria a utilização das redes de computadores em tempo real para comunicação entre músicos e bandas em locais distribuídos remotamente para executar e reproduzir músicas como se estivessem no mesmo ambiente. Para exemplificar, suponha um músico que queria realizar um *show* ao vivo, mas sem necessidade de transportar toda sua banda e os instrumentos musicais. Uma solução seria cada instrumento possuir sensores capazes de enviar os sons e dados pela rede, possibilitando a banda e o músico estarem remotos, com som real como se estivessem no mesmo local realizando o *show* ao vivo como exposto na figura 10 (LACEY, 2015).

Figura 10 – Exemplo de Show em rede distribuído (LACEY, 2015)

3 Cenários de comunicação

Neste capítulo iremos descrever, através de cenários práticos um modelo de uma solução para conteúdo musical interativo para Internet das Coisas Musicas.

Empregaremos em nossa proposta o conceito de cliente-servidor (descrito na seção 2.2) de forma que todas as interações dos usuários são enviadas para servidor que realizará o trabalho de atualizar em tempo real, todos os usuários conectados, mantendo-os sincronizados.

3.1 Cenário 1 - música interativa

Neste cenário apresentaremos a interatividade entre os usuários e uma aplicação musical. Iremos considerar como exemplo, um grupo de amigos em suas residências que, em um momento de laser, reúnem-se de forma remota para descontrair.

Por meio da solução proposta, todos os usuários conseguirão ouvir as mesmas músicas e terão acesso a uma *Playlist* sincronizada e compartilhada em tempo real, utilizando recursos da plataforma do *YouTube*, concedendo a todos os mecanismos de interação, como pausar, reproduzir, avançar, voltar as músicas e alterar a barra de reprodução da música em execução. De acordo com a Figura 11, este cenário utiliza um servidor central, que atuará como o responsável por efetivar os comandos dos usuários, assim teremos vários usuários conectados no servidor, realizando ações a todo momento. Vale destacar, que a conexão é bidirecional e contínua, e o servidor pode estar em uma rede remota, como na figura, ou em uma rede local.

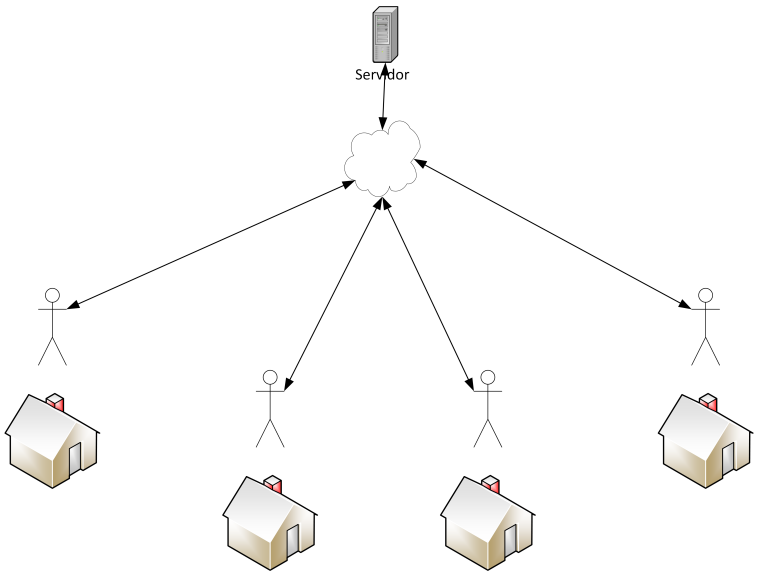

Figura 11 – Modelo do Cenário 1 com a interação do servidor e clientes

Inicialmente, os usuários devem estabelecer a conexão com o servidor, através de troca de mensagens, como apresentado na Figura 12. Nesta Figura temos um usuário "x1" solicitando a conexão com o servidor da aplicação, e o servidor aceitando a conexão.

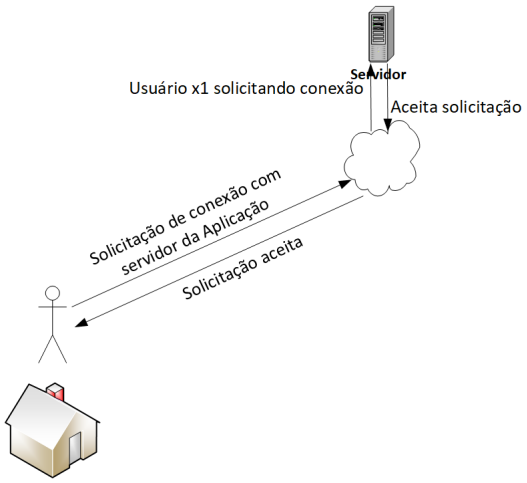

Figura 12 – Cenário 1, Primeiro passo: conexão com servidor

Após o estabelecer a conexão, os usuários podem interagir com as músicas da *Playlist*, usando os recursos de pausar, reproduzir, voltar e avançar. Todas essas ações serão enviadas para servidor, que realizará o processamento das requisições e enviará uma mensagem para todos os usuários conectados. Esta mensagem contém a ação executada, para que todos os clientes sejam atualizados. Na Figura 13 a), é exibido o usuário "x1", efetuando uma ação por meio do recurso de próxima música. O servidor recebe a instrução, e replica a alteração para todos os usuários conectados como demostrado na Figura 13 b). Desta forma, qualquer usuário conectado pode efetuar alterações nas músicas que estão

sendo reproduzidas ou na *Playlist*, que o servidor se encarregará de notificar a todos a alteração realizada.

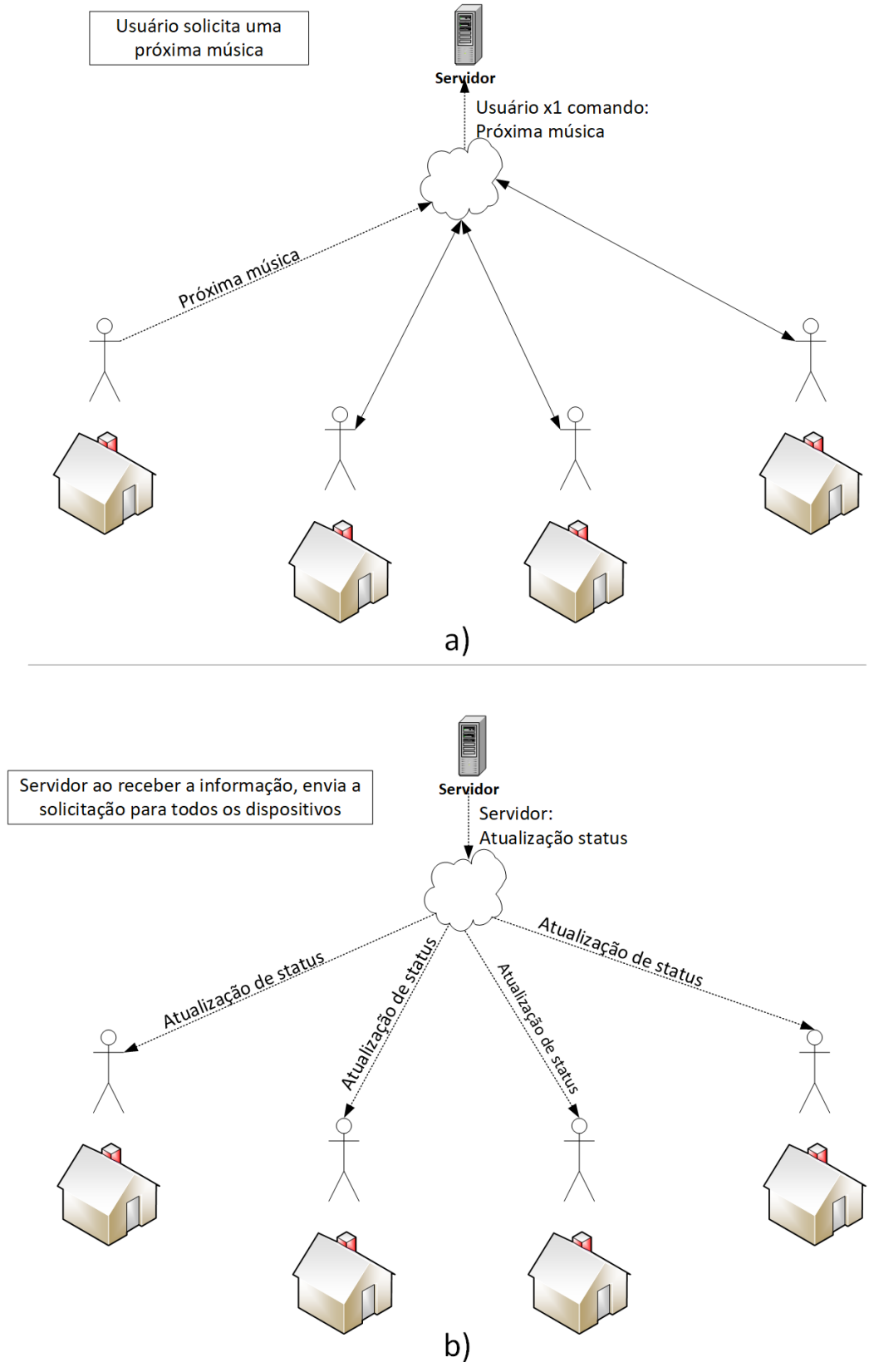

Figura 13 – Cenário 1: Realização de ações

3.2 Cenário 2 - música interativa com um usuário controlador

Este cenário é semelhando ao anterior, com a diferença da presença de um controlador das interações músicas que serão alteradas para todos os dispositivos conectados. Em muitas situações, o papel do moderador acaba sendo fundamental, para evitar desordem ou mesmo conteúdos impróprios.

Através de seus dispositivos conectados ao servidor, que receberá todas as requisições dos usuários, a princípio ele não as replica diretamente, como no cenário anterior, enviando-as para um moderador (controlador) que tem a função de controlar as ações nas músicas que serão executadas e reproduzidas nos dispositivos. Como exibido na Figura 14, o controlador também é um usuário conectado ao servidor, com a função de receber todas as requisições enviadas ao servidor e definir qual ação será executada, exemplo similar de um DJ em um evento.

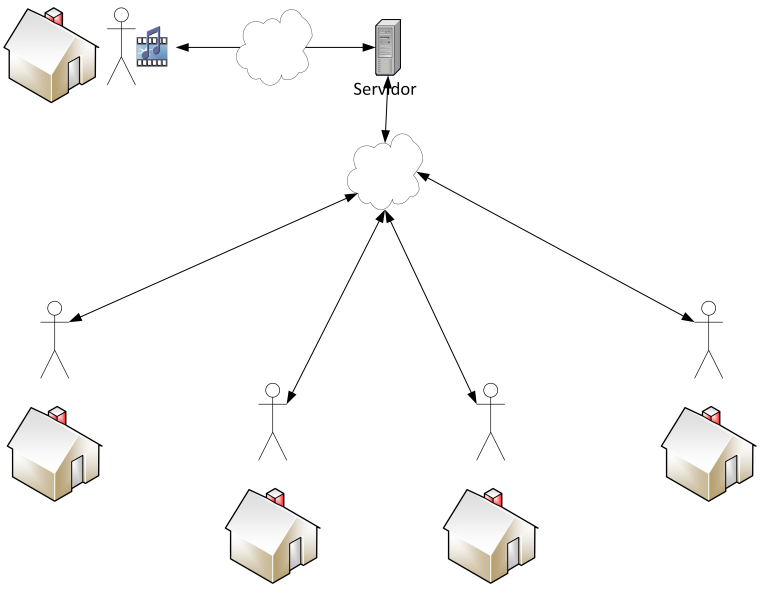

Figura 14 – Modelo Cenário 2: Música interativa com um controlador

Neste cenário, os usuários podem enviar suas instruções de pausar, reproduzir, avançar e voltar as músicas para servidor, que ao receber, envia para o controlador, conforme exibido na Figura 15. O controlador então, define as instruções que serão executadas, e envia novamente ao servidor que encaminha a todos os usuários conectados.

Na Figura 15 a), o usuário “x1"executa a ação de próxima música, que é recebida pelo servidor. Ao receber, o servidor envia ao usuário controlador que então decide o que fazer. A Figura 15 b) exemplifica uma ação em que o usuário controlador aceitou, enviando a instrução de próxima música para o servidor, que enviou a ação para todos os usuários conectados.

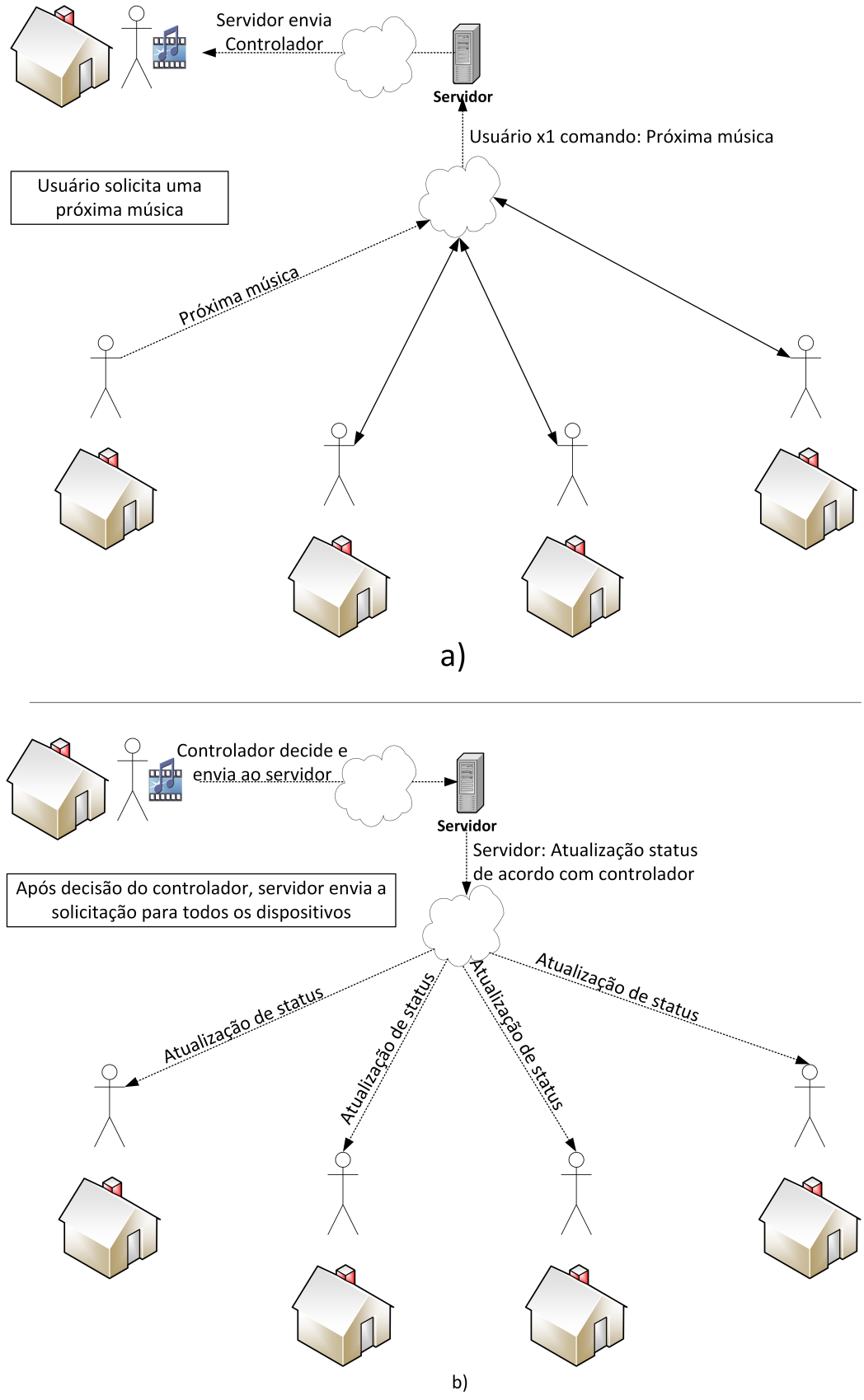

Figura 15 – Cenário 2: Atualização de requisição com controlador

3.3 Cenário 3 - Espacialização do som pelos dispositivos de usuários

Este cenário pode ser muito útil no contexto de Internet das Coisas Musicais. Nele, tempos a possibilidade de, além da interação, a utilização dos dispositivos dos usuários como os smartphones, para reproduzir os sons no ambiente.

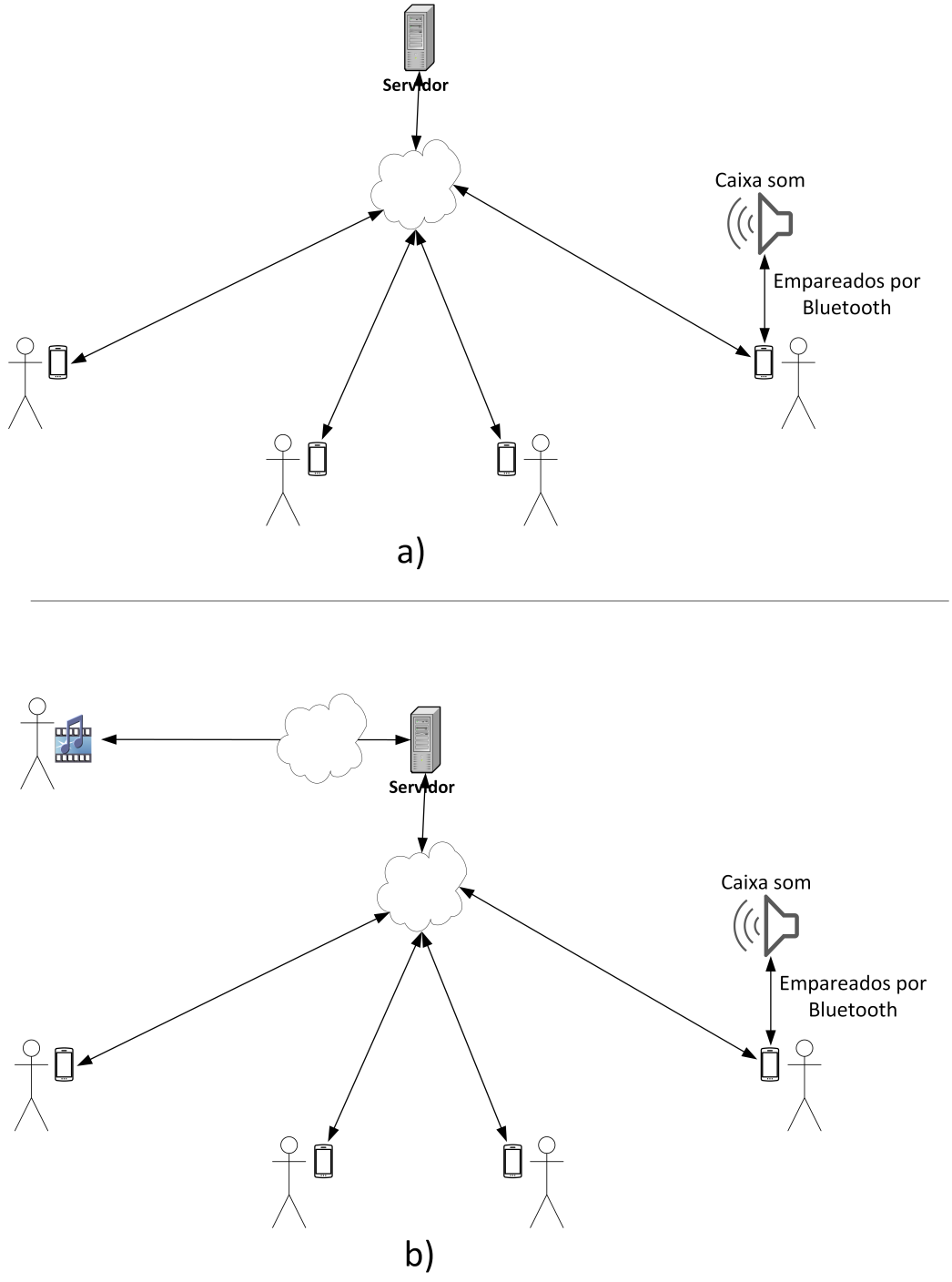

Figura 16 – Cenário 3: Espacialização do som pelos dispositivos de usuários

Neste cenário os usuários através de seus dispositivos conectados ao servidor central, Figura 16 a) poderão enviar e receber ações para as músicas, podendo utilizar um usuário moderador, conforme mostrado na Figura 16 b) que definirá quais ações serão aplicadas nos dispositivos conectados, que podem atuar como caixas de som em vários ambientes distintos, ampliando o espaço de alcance dos sons.

Após conectarem ao servidor, é permitido aos usuários o envio de músicas e ritmos por meio de seus dispositivos, ficando a critério do usuário moderador (quando utilizado) definir as músicas que serão reproduzidas. Conforme mostrado na Figura 17 a), o usuário "x3" envia uma música para ser reproduzida. A música é envida ao servidor que, ao receber, encaminha a solicitação para moderador, que recebe a informação e envia ao servidor, que propaga para todos os usuários conectados, como exposto na Figura 17 b) em que os usuários podem utilizar seus dispositivos, ou caixas de sons auxiliares para reproduzir a música no ambiente.

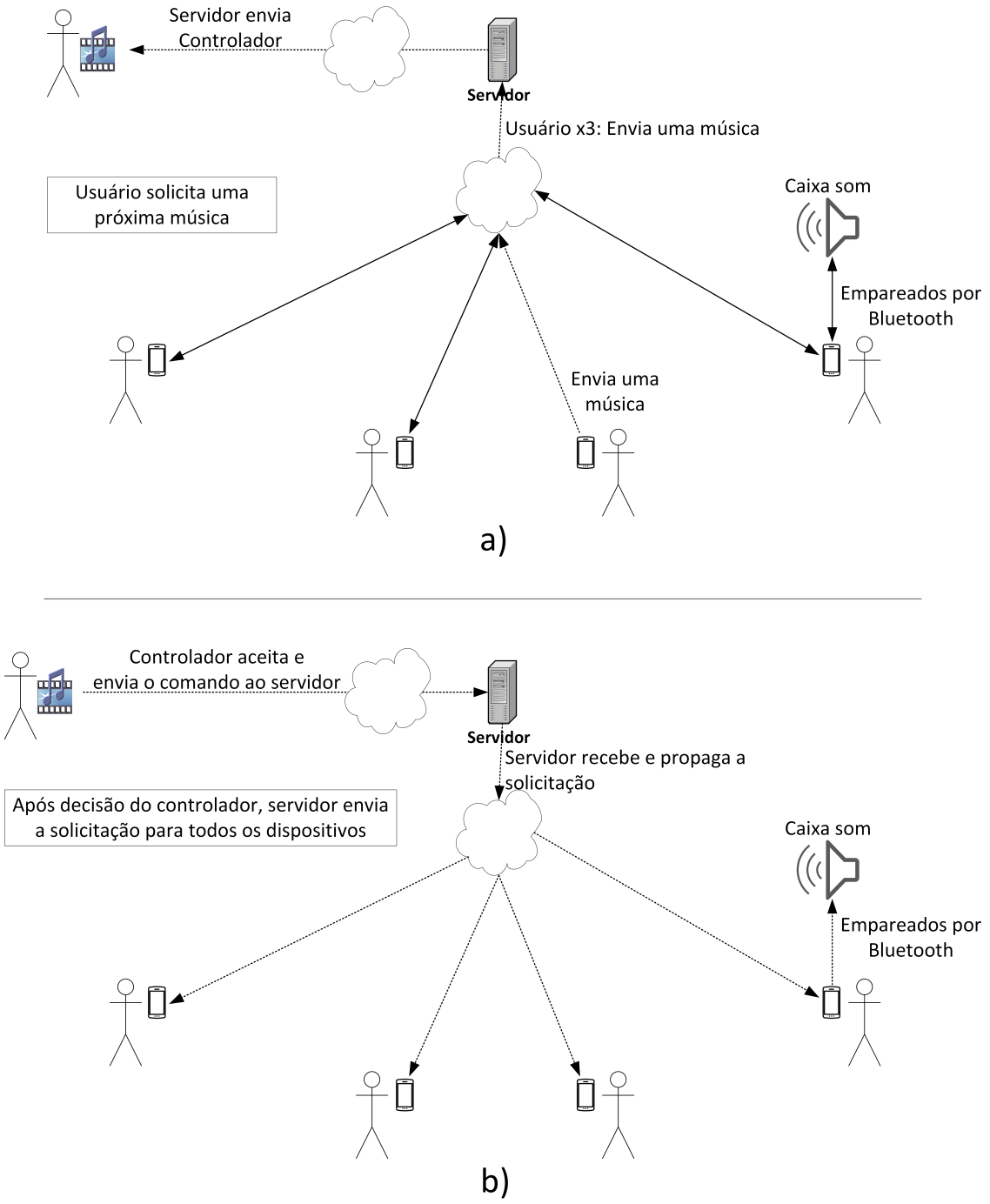

Figura 17 – Cenário 3 - Troca de mensagens

3.4 Cenário 4 - Dispositivo do usuário para espacialização em veículos automotivos

Este cenário é similar ao cenário anterior, viabilizando aos dispositivos, conectarem a outros dispositivos, por exemplo, por meio do *Bluetooth*, ou outras tecnologias, como os recursos multimídias e conexões auxiliares dos veículos automotivos.

Assim, podemos utilizar para organização e realização de carreatas ou protestos como, por exemplo, manifestações que ocorreram solicitando fim do isolamento e reabertura do comércio, conforme informações da (VALOR GLOBO, 2020; VEJA ABRIL, 2020). Também pode ser empregado nas tradicionais carreatas políticas em períodos eleitorais. Este cenário possibilita ainda pensar a Internet das Coisas Musicais aliada aos conceitos de Cidades Inteligentes e Carros Inteligentes.

Neste cenário, os usuários se conectam ao servidor, conforme cenário anterior e os sons seriam reproduzidos nos dispositivos, com todos os veículos e aparelhos reproduzindo a mesma música, podendo utilizar, caso necessário uma intervenção de um moderador, que além de músicas, poderá enviar áudio de voz (informações e notícias) sincronizadas e em tempo real com maior organização e alcance da propagação do som, conforme Figura 18.

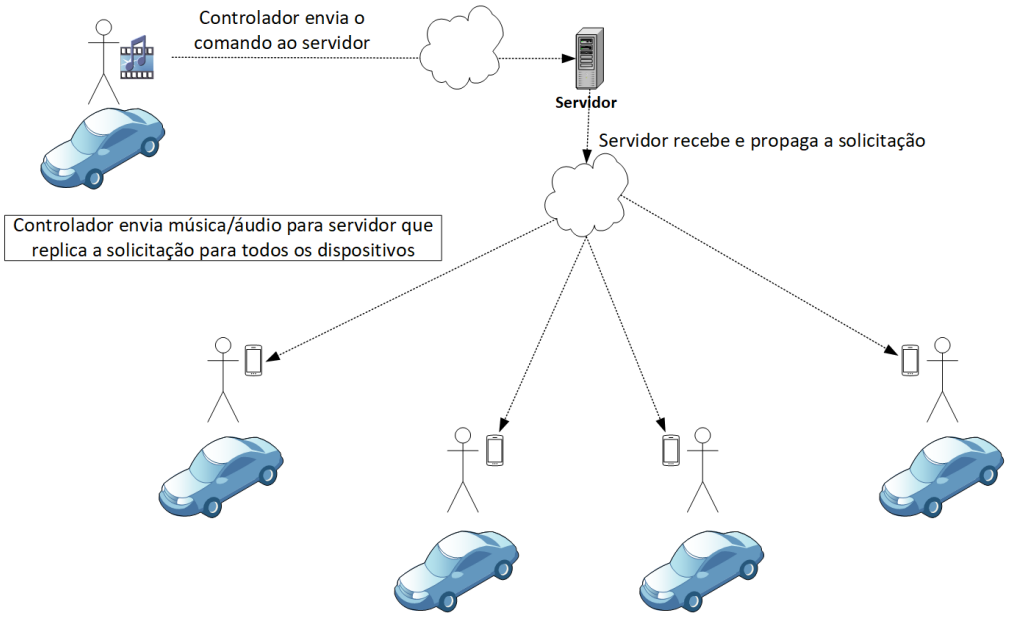

Figura 18 – Cenário 4 - com veículos

Além dessas duas situações, os recursos destes modelos podem ser utilizados nos tradicionais eventos de sons automotivos, de modo que os usuários podem interagir enviando ou alterando *status* das músicas, como nos cenários 1 (3.1) e 2 (3.2).

Podemos adaptar esse cenário para eventos presenciais ao vivo na modalidade *drive-in* no qual os usuários assistem os *shows* ao vivo no local, ficando no veículo.

4 Implementação da aplicação interativa e Resultados

A aplicação proposta foi implementada a partir dos cenários apresentados na seção 3. Com ela, apresentamos um protótipo funcional focado na execução distribuída de música em ambientes locais e/ou remotos, proporcionando que os usuários, por meio do navegador *Web*, interajam entre si, reproduzindo as mesmas músicas/vídeos e compartilhando a *playlist*, por intermédio do conceito da Internet das Coisas Musicais.

Devido a restrições de tempo, iremos implementar um protótipo que disponibiliza a aplicação descrita no cenário 3.1, demonstrando a aplicabilidade e validando a implementação, servindo de modelo para implementação dos demais cenários. Consequentemente, serão implementados um servidor e um cliente *Web*.

O cliente será acessado pelo navegador *Web*, utilizando a aplicação onde os usuários interagem uns com os outros, com uma *interface* amigável e simples.

O servidor será responsável por gerenciar e controlar todas as requisições dos usuários, de modo que, após algum usuário realizar alguma ação, está ação será enviada através de mensagens para servidor, que enviará mensagens com as ações para todos os usuários conectados.

4.1 Protocolo e Mensagens da aplicação

Os protocolos definem um conjunto de padrões e parâmetros que certificam a comunicação e transmissão de dados entre dispositivos, através de métodos que organizam o tráfego e conciliam procedimentos necessários para estabelecer a comunicação entre as partes, controlando o formato de como os dados serão enviados e recebidos, o meio por onde serão transmitidos e procedimentos de identificação e endereçamento (ISHIHARA, 2006).

Após o estabelecer a conexão entre cliente e servidor, várias mensagens de controle podem ser trocadas entre eles. Para exemplificarmos nosso protocolo de comunicação, mostraremos a troca de mensagens envolvidas na funcionalidade "*Play*". A Figura 19 demonstra a comunicação entre dois usuários: Cliente "x1" e "x2" com servidor.

O cliente "x1" executa a funcionalidade do protocolo do evento de *Play*, enviando uma mensagem "JSON" encapsulada ao servidor, conteúdo um action: 'play'. Como na Figura 19, o servidor recebe a mensagem e a desencapsula, realizando uma verificação condicional "*if*", identificando que a mensagem é reconhecida do tipo '*Play*'. Encapsula

uma nova mensagem contendo um estado (*state*) de notificação (notify_state('play')), passando o parâmetro '*play*', enviando-a para todos os clientes conectados. Como nesse exemplo, temos apenas o cliente "x2" conectado, o mesmo recebe a mensagem, com a função websocket.onmessage(JSON(message)), que tem a mensagem encapsulada. Após receber, o cliente desencapsula a mensagem e realiza a ação contida na mesma.

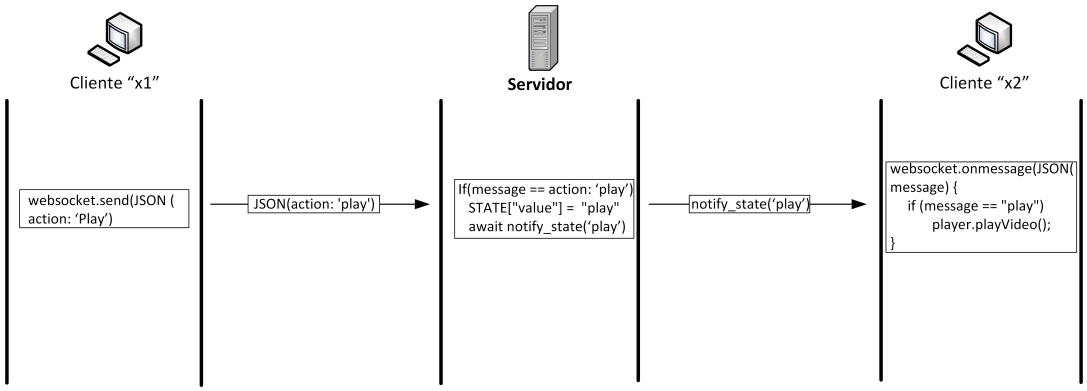

Figura 19 – Protocolo da funcionalidade Play

Na Figura 20 é demonstrado a troca de mensagens entre 3 (três) clientes e o servidor. Na primeira interação, o cliente "x1" executa a funcionalidade "*Play*", assim como no exemplos anterior. Ao receber a mensagem, o servidor desencapsula, realiza a verificação, em seguida, encapsula uma nova mensagem com uma notificação com o parâmetro '*play*' e envia para todos os clientes conectados. Após receberem a mensagem, os clientes podem executar a ação descrita na mesma. Seguindo esse modelo, na segunda interação, o cliente "x2" executa a funcionalidade "*Pause*", que é enviada ao servidor e entregue aos clientes "x1" e "x3". Por fim, na terceira interação o cliente "x3" executa a funcionalidade "*Next*", é enviada ao servidor, que ao receber, envia para os clientes "x1" e "x2".

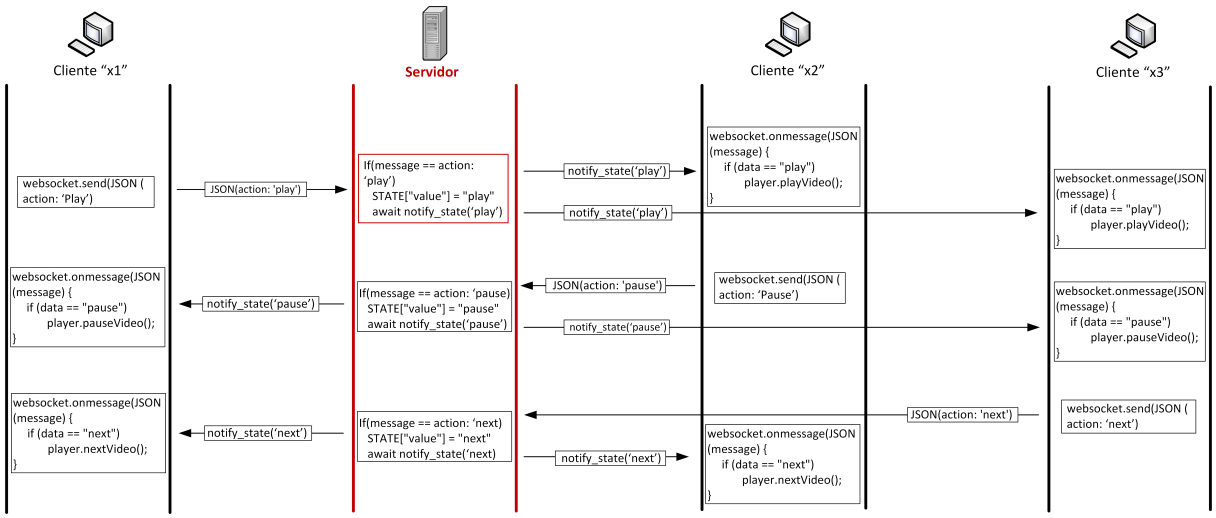

Figura 20 – Protocolo da funcionalidade Play, Pause e Next com 3 clientes conectados

As outras funcionalidades da aplicação seguem protocolos muito similares aos exemplificados acima e podem ser lidas com mais detalhes a partir do código fonte disponibilizado em: https://github.com/silvaale2/app_musica_interativa e aplicação em: https://alexandre.ufsj.edu.br/

4.2 Implementação do Cliente

A princípio, realizaremos a comunicação do cliente com o servidor, utilizando a linguagem *JavaScript* e a tecnologia de *WebSockets*. Uma aplicação *JavaScript* pode se conectar a um servidor através da seguinte função:

```
websocket = new WebSocket("ws://alexandre.ufsj.edu.br:6789/");
```

Depois disso, com o servidor configurado, o cliente e servidor já podem estabelecer conexão e troca de dados.

Escolhemos utilizar *WebSockets*, pois diferente do protocolo HTTP em que usualmente as conexões são fechadas após a recepção de uma resposta, com *WebSockets* a comunicação é feita por conexões contínuas, com isso, a conexão só é finalizada quando uma das partes solicita (COMPUTACAO; FIRMINO, 2011; GONÇALVES, 2014).

Tradicionalmente os *WebSockets* utilizam os recursos do HTTP para estabelecer a conexão entre o cliente e servidor. Após estabelecida a conexão, o protocolo mantém a conexão bidirecional, retirando o cabeçalho HTTP, provendo uma conectividade contínua entre cliente e servidor.

Como o objetivo é prover interatividade com a música por intermédio da plataforma do *YouTube*, devemos permitir a conexão de vários usuários simultâneos. Para isso,

foi criado no cliente a função para utilizar a API do *YouTube*, por meio da linguagem *JavaScript*.

A API do *YouTube* com *JavaScript*, oferece recursos de exibir e controlar vídeos do *YouTube*, vale ressaltar, que esse é um modelo, podendo ser usado como base para outras plataformas de música e/ou vídeo que possua API ou recursos semelhantes a API do *Youtube*. Conforme disponível em (ZINE, 2015), a plataforma do *YouTube* dispõe de uma API de *player* com *JavaScript*, com isso, algumas funções da API foram implementadas na aplicação.

Para possibilitarmos a execução da API na aplicação, devemos incluir a seguinte *tag*:

```
<script src="https://www.youtube.com/iframe_api"></script>
```

Com o suporte da API na página dos usuários, podemos utilizar seus recursos, portanto, iremos implementar suas funções, para que sejam carregadas nos navegadores, permitindo interação com os outros usuários sincronicamente.

As funções da API do *YouTube* utilizadas na aplicação foram:

- `onYouTubeIframeAPIReady()` esta função foi utilizada por oferecer recursos de carregar os vídeos na aplicação e criação da *playlist*
- `initialize()`, esta função atua nos controles de carregamento dos vídeos, além de atualizar a apresentação dos tempos decorridos na barra de progresso.

Além disso, para o controle dos vídeos pelos usuários foram utilizadas as funções:

- `formatTime(time)`
- `player.playVideo()`
- `player.pauseVideo()`
- `player.nextVideo()`
- `player.previousVideo()`

Com essas funções, os usuários conseguem voltar, avançar, pausar, parar, iniciar, visualizar os tempos e efetuar ações na barra de progresso.

Precisamos enviar as ações do cliente para o servidor, de forma que, ele receba as informações e as reenvie para todos os clientes conectados. Neste exemplo estamos enviando a ação de iniciar uma reprodução de vídeo (*play*), quando o botão de *play* é acionado (*onclick*), é executado a função:

```
play.onclick = função (ação) {
    websocket.send(JSON.stringify({ action: 'play' }));
}
```

Essa função envia uma mensagem "JSON" simples (`{"action":"play"}`) para o servidor, que por meio de suas funções fica aguardando essa informação e ao recebê-la, envia a ação para todos os usuários conectados, de acordo com apresentado na seção 4.1.

Por fim, todos os usuários conectados devem receber a ação (mensagem), podendo executar algum evento ou instrução com a informação recebida, em nosso caso, essa ação deve executar a função de reproduzir uma música/vídeo por meio da API do *Youtube* conforme exibido na Figura 21, utilizando a função:

```
websocket.recebe_msg = function (action) {
    action = JSON.parse(action);
    if (action == "play") {
       playVideo();
    }
    else
       imprime_tela("Ação inválida: ", action);
}
```

recebe as mensagens do servidor *WebSocket* (websocket.recebe_msg), recendo a "ação" (mensagem "JSON"), realizando uma verificação do que fazer com a mensagem, no caso, executou a função de reproduzir vídeo "playVideo()".

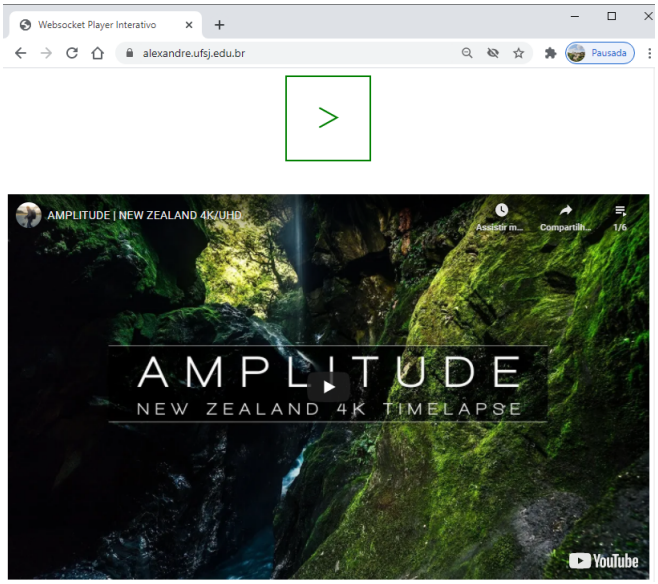

Figura 21 – Interface da Aplicação no cliente

Agora, com o exemplo do iniciar (*play*), usaremos os mesmos parâmetros para criar condições para as demais funcionalidades (*pause, stop, next, previous* e barra de progresso).

Após incrementado as funcionalidades, percebemos a necessidade de incrementar a funcionalidade que permita adicionarem vídeos na *Playlist*. Para implementação foi necessário a criação de um vetor e um identificador, para identificar o vídeo corrente e a quantidade de elementos da *Playlist*. Com a identificação do vídeo corrente, garantimos que novos vídeos sejam adicionados ao final da *Playlist*, além de assegurar a reprodução do vídeo corrente e não retornar ao início da *Playlist*.

Vale destacar que, como estamos utilizando a API do *YouTube* (ZINE, 2015), para que os usuários adicionem vídeos, será necessário enviar apenas o ID dos vídeos, que pode ser obtido de qualquer vídeo do *YouTube*, através do seu *link*, copiando seu final, depois de "?v=", por exemplo, do vídeo: https://www.youtube.com/watch?v=6LK5CZPDMfk, o ID que é: "6LK5CZPDMfk"portanto, esse ID que tem que ser adicionado a Playlist, como pode ser visto na interface do cliente exibido na Figura 22. Nesta mesma Figura, mostramos também uma área para exibir os vídeos que estão na *Playlist*, qual o vídeo corrente e o total de vídeos da *Playlist*.

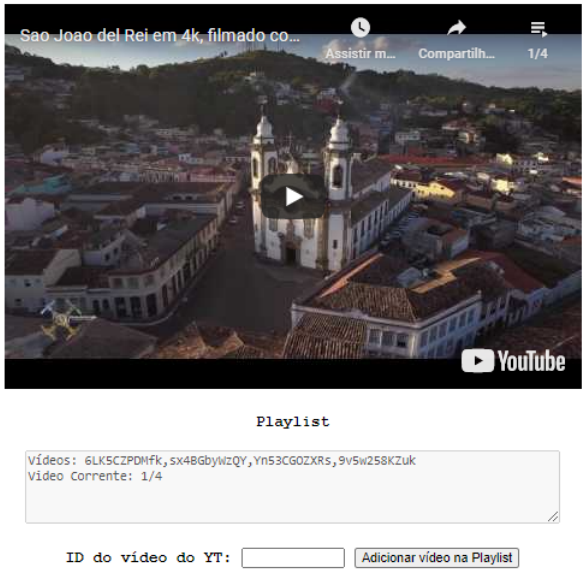

Figura 22 – Interface Playlist

Também implementamos na aplicação um *chat* para troca de mensagens de texto, possibilitando a comunicação entre os usuários, acrescentando maior interatividade com a aplicação. A implementação do *chat* segue os princípios anteriores, onde foi criado uma *string* com uma cadeia de caracteres, que é enviada para servidor. Na figura 23 mostramos a área em que os usuários visualizam as mensagens enviadas, abaixo um campo nome, para identificação e na frente um campo com a mensagem que o usuário deseja enviar, e

um botão para que a mensagem seja enviada.

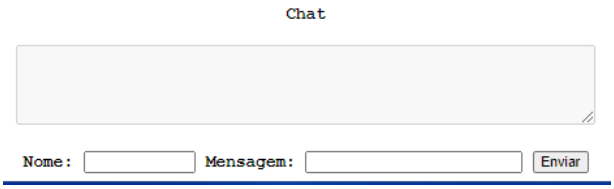

Figura 23 – Chat

4.3 Implementação do Servidor

Esta aplicação utiliza 2 servidores: um servidor de página *Apache* para fornecer os componentes estáticos (páginas HTML, CSS, imagens e *JavaScript*) e um servidor *WebSocket* que servirá como replicador de mensagens para os usuários. A opção por utilizar *WebSockets* se deu por esta tecnologia ser mais compatível com a Internet das Coisas Musicais e suas preocupações como a redução do tempo de resposta e a latência (COMPUTACAO; FIRMINO, 2011; GONÇALVES, 2014).

A implementação da aplicação do servidor *WebSocket* foi escrita na linguagem de programação *Python*. Inicialmente é necessário desenvolver a comunicação do servidor com os clientes, de forma que este fique esperando conexões por meio dos protocolos HTTP e HTTPS. Para isso, utilizaremos uma comunicação assíncrona, com uso das funções da linguagem Python *asyncio* e *await*. A primeira indica que uma função deve ser executada de forma assíncrona. A segunda significa que a conexão será paralisada naquele ponto aguardando novas mensagens futuras.

```
asyncio.get_event_loop().run_until_complete(servidor)
asyncio.get_event_loop().run_forever()
```

No exemplo acima, o parâmetro "servidor" é o *socket* criado para comunicação entre o cliente e o servidor *WebSocket*, no qual os módulos e bibliotecas do *WebSocket* já foram instalados e importados, "*asyncio*" faz parte da biblioteca e é usado para programação concorrente, *threads* entre outros, sendo necessário definir o endereço e porta do servidor *WebSocket*:

```
servidor = websockets.serve("Endereço_servidor", porta)
```

Para que seja possível enviar comandos para os clientes, precisamos manter uma lista com todos os clientes conectados. Para isso utilizamos a seguinte função, que é executada após a conexão de um cliente:

```
async def registrar_usuario (websocket):
    USERS.add(websocket)
```

Quando o usuários sair ou sua conexão falhar a função:

```
async def remover_usuario(websocket):
    USERS.remove(websocket)
```

tem a responsabilidade de remover os usuários do servidor.

Agora que já temos recursos de comunicação com o servidor *WebSocket* e funções para adicionar e remover usuários, precisamos de mecanismos para que o servidor *WebSocket* envie os eventos para todos os usuários conectados sempre que um novo usuário conectar ao servidor. Para isso, podemos utilizar meios de notificações com envio de mensagens, notificando todos os usuários em tempo real quando um novo usuário conectar ou desconectar, com as funções:

```
async def usuarios ():
    return json.dumps({"tipo": "usuarios", "contador": len(USERS)})
```

temos o controle da quantidade de usuários conetados, sendo assim, é possível saber sempre que um novo usuário conecta ou desconecta do servidor.

Para que todos os clientes sejam notificados deste evento, a função:

```
async def notificar_usuarios():
    if USERS:
        message = state_event()
        await asyncio.wait([user.send(usuários()) for user in USERS])
```

se encarrega de enviar notificações de usuários para os clientes conectados. Essa função obtém o novo estado e grava na variável "*message*". Após isso, envia a mensagem para todos os usuário.

Portanto, quando um novo usuário conectar ao servidor e a função que adiciona usuário é executada, ao seu final, inserimos a chamada da função notificar_usuarios():

```
async def registrar_usuarios (websocket):
    USERS.add(websocket)
    notificar_usuarios()
```

Isso faz com que o evento de um novo usuário registrado, seja enviado para todos os outros usuários conectados no servidor.

Em sequência, precisamos implementar as funções que recebem e registrem os eventos que serão notificados para todos os clientes. Consequentemente, definiremos quais serão os eventos e ações que devem ser notificados. Como estamos trabalhando com reprodução de música/vídeo, criaremos ações para os botões de um reprodutor, no caso, os implementados no cliente (*play, pause, stop, next, previous* e barra de reprodução).

Para receber e notificar as ações para todos os usuários, podemos adaptar os conceitos da função de notificar usuários, alterando para notificar ações dos usuários:

```
async def acoes():
    return json.dumps({"tipo": "acoes", **ACAO})

async def notificar_acoes():
    if USERS:
        asyncio.wait([user.send(acoes()) for user in USERS])
```

Assim como na notificação dos usuários, precisamos criar a função das ações e executar dentro dela a função de notificar_acoes() para que todos os usuários recebam a informação. Neste exemplo, a ação de *play* será recebida pelo servidor e enviada para todos os usuários conectados.

```
async def register(websocket):
    websocket.send(notificar_ações())
    async for message in websocket:
        acao = json.loads(message())
        if acao["action"] == "play":
            ACAO["value"]  = "play"
            await notificar_ações()
        else:
            imprime_tela("Comando inválido: ", acao)
```

Ao receber a mensagem em "json.loads(message())", é verificado o tipo da mensagem, alterando seu estado no servidor em "ACAO["value"] = "play", por fim, é enviada a mensagem de notificação com o estado para todos os usuários "await notificar_acoes()".

Para segurança da aplicação, realizamos a instalação do protocolo SSL no servidor, realizando as importações dos módulos e das chaves, em seguida foi efetuado as configurações necessárias no servidor, assim como, as configurações do *Apache*. Para segurança da aplicação no HTML (cliente) alteramos linha:

```
websocket = new WebSocket("ws://alexandre.ufsj.edu.br:6789/");
```

para:

```
websocket = new WebSocket("wss://alexandre.ufsj.edu.br:6789/");
```

e no servidor *WebSocket* alteração da linha:

```
start_server = websockets.serve(counter, "alexandre.ufsj.edu.br", 6789)
```

para:

```
start_server = websockets.serve(counter, "alexandre.ufsj.edu.br", 6789,
        ssl=ssl_context)
```

Para concluímos ativação do SSL e carregamento das chaves de segurança, foram incluídas as linhas:

```
ssl_context = ssl.SSLContext(ssl.PROTOCOL_TLS_SERVER)
path_cert = "/var/www/html/certificado.pem"
path_key = "/var/www/html/chave.key"
ssl_context.load_cert_chain(path_cert, keyfile=path_key)
```

4.4 Resultados

A aplicação foi instalada em um servidor virtual, com o sistema operacional Linux Debian GNU 10, com CPU Intel Xeon Gold 5220 - 2.20 GHz com 4 Cores, 16 G de memória RAM e 100 G de armazenamento em disco. Foram instalados os módulos do *Websocket, Python3, SSL, Apache2, PHP* nas versões mais atuais para funcionamento da aplicação.

A *interface* da aplicação com todas as funcionalidades estão apresentadas na Figura 24 que mostra as ações que os usuários podem realizar na aplicação.

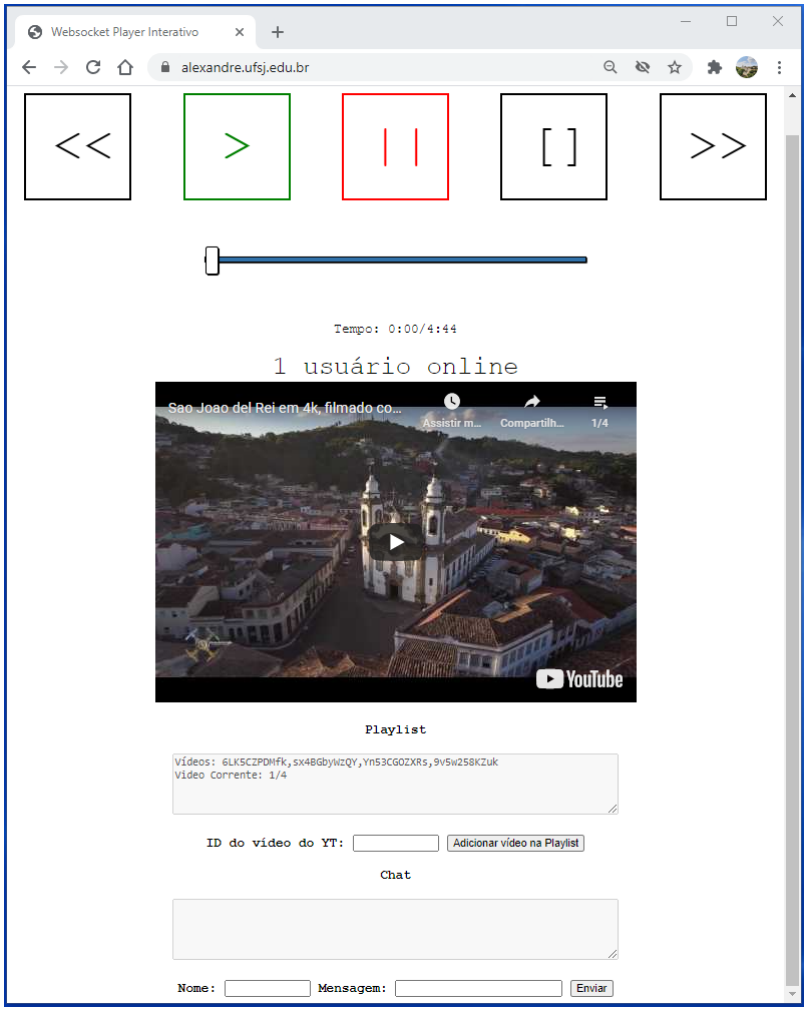

Figura 24 – Interface da Aplicação

https://alexandre.ufsj.edu.br/

As funcionalidades da aplicação foram testadas durante o desenvolvimento do projeto, através de testes locais e remotos. Os testes foram realizados utilizando-se de computadores, *smartphones* e *tablets*, com provedores de Internet banda larga e dados móveis de celulares. Foram executados testes de rede com diversos usuários e em vários navegadores diferentes verificando o tempo de resposta, tanto no cliente quanto no servidor.

Os resultados foram satisfatórios em todas as fases, demonstrando atender todas as expectativas, inclusive com uso dos dados móveis os tempos de respostas foram satisfatórios.

A Figura 25 mostra a *interface* de um teste com a interação de quatro usuários em navegadores diferentes (*Chrome, Firefox, Edge e Internet Explorer*), utilizando a funcionalidade do *chat* que permite a todos os usuários conectados, enviarem e receberem mensagens de texto. Os testes mostraram que tanto a *interface* quanto as funcionalidades se comportaram de maneira muito similar nos 4 navegadores testados.

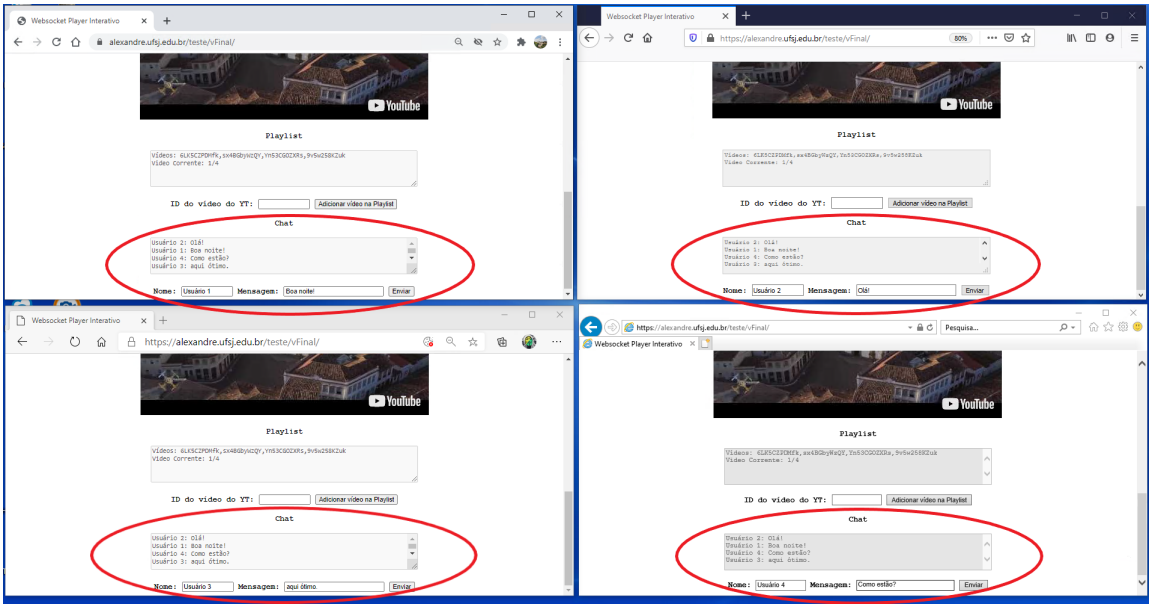

Figura 25 – Funcionalidades em navegadores diferentes

Outra avaliação realizada foi o desempenho do servidor, conforme observado na Figura 26 a) foram testadas 56 conexões simultâneas, em seguida 70 conexões simultâneas como na Figura 26 b), para concluir os testes, chegamos a 102 conexões simultâneas como visto na Figura 26 c). Os testes foram realizados por diversos usuários, em diferentes localidades, entre diferentes dispositivos, utilizando vários navegadores e provedores de Internet, incluindo dados móveis de celulares.

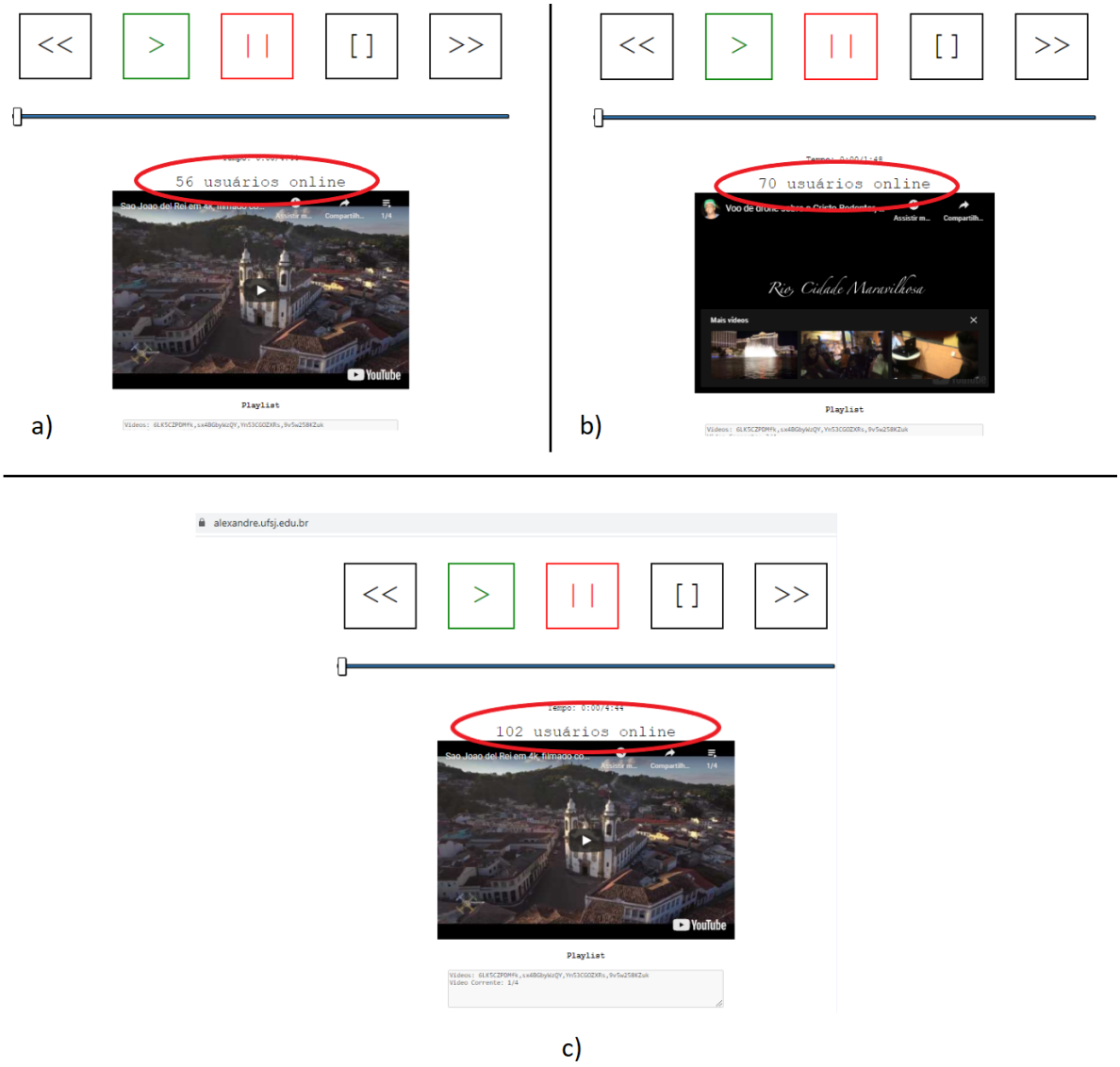

Figura 26 – Teste conexões simultâneas servidor

As conexões foram capturadas pelo servidor por meio do comando “netstat -anp” com um filtro para capturar somente as conexões na porta da aplicação “| grep 6789”, de acordo com os dados exibidos na Figura 27.

```
root@trabalho:/home/alexandre# netstat -anp | grep 6789
tcp        0      0 192.168.2.244:6789      0.0.0.0:*               OUÇA       16212/python3
tcp        0      0 192.168.2.244:6789      131.108.200.226:52783   ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      177.37.127.30:51250     ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      168.121.90.126:4836     ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      179.108.200.163:4523    ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      168.121.90.126:4845     ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      131.108.200.226:51393   ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      177.37.127.30:50580     ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      191.215.138.199:60610   ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      131.108.200.226:51865   ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      177.37.127.30:49842     ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      190.108.121.254:41188   ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      179.108.200.163:3075    ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      177.37.127.30:48936     ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      131.108.200.226:52288   ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      179.108.200.163:3299    ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      192.168.0.121:52427     ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      131.108.200.226:51743   ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      168.121.90.126:4851     ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      192.168.0.121:52420     ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      177.55.205.121:30436    ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      191.215.138.199:33575   ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      177.37.127.30:51258     ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      131.108.200.226:51740   ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      131.108.200.226:51866   ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      190.108.121.254:41216   ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      131.108.200.226:51058   ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      179.108.200.163:4973    ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      131.108.200.226:52645   ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      177.37.127.30:50144     ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      131.108.200.226:51946   ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      179.108.200.163:4926    ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      177.55.205.121:30120    ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      179.108.200.163:4804    ESTABELECIDA 16212/python3
tcp        0      0 192.168.2.244:6789      168.121.90.126:4843     ESTABELECIDA 16212/python3
```

Figura 27 – Captura das conexões simultâneas servidor

Realizamos também o monitoramento de desempenho de performance do servidor virtual por meio da ferramenta gráfica do *Nutanix Prism* (NUTANIX, 2020), solução de máquinas virtuais utilizada no *Datacenter* onde o servidor está hospedado. Na Figura 28 apresentamos o consumo de recursos do servidor antes dos testes com as conexões simultâneas.

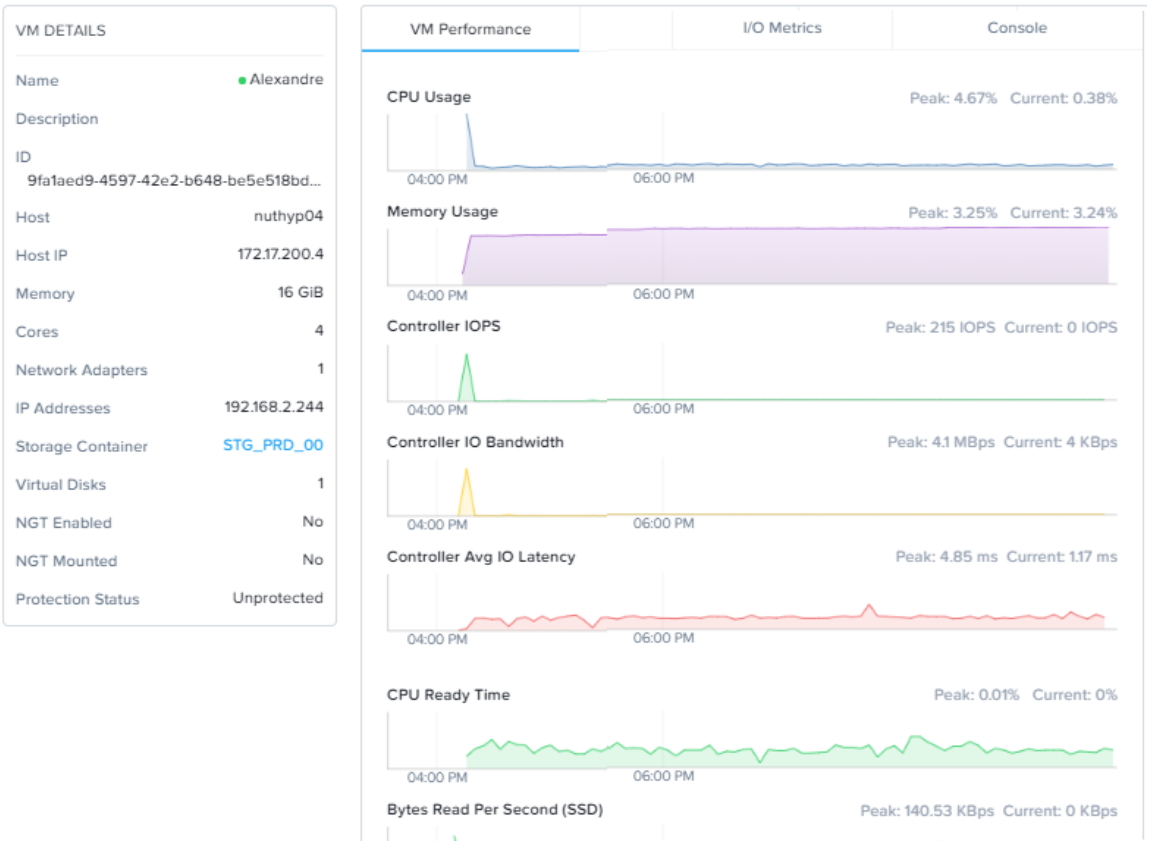

Figura 28 – Apresentação desempenho do servidor antes das conexões

Na Figura 29 observamos o consumo de recursos do servidor após as 102 conexões simultâneas.

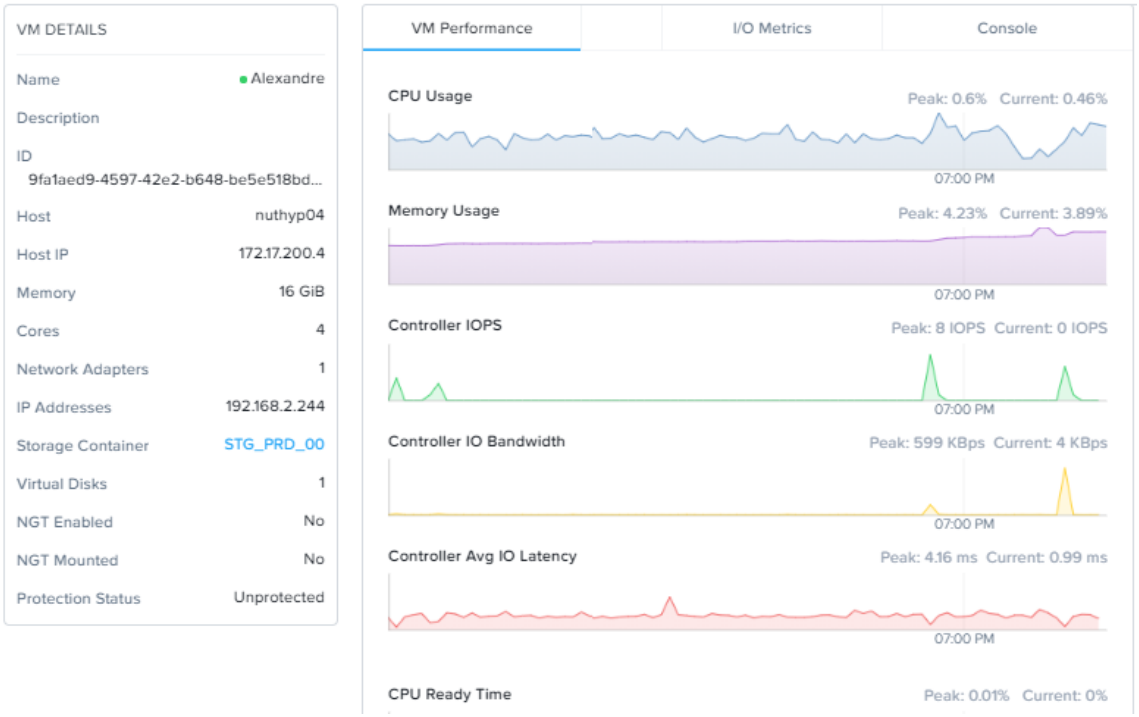

Figura 29 – Apresentação desempenho do servidor após as conexões

Após os testes de carga, verificamos que nosso servidor tem baixa exigência de processamento (*CPU usage*) e memória (*Memory usage*). Por exemplo, na Figura 27 vimos um pico de uso de 3,25% de memória com o servidor no estado de repouso (sem nenhum cliente conectado) e um pico de 4,23% com 102 conexões simultâneas. Sendo assim o tempos de respostas das ações dos usuários mantiveram-se aceitáveis, não sobrecarregando nem criando gargalo no servidor, atendendo as expectativas.

5 Considerações Finais e trabalhos futuros

5.1 Considerações Finais

Nosso trabalho de pesquisa apresentou um levantamento das necessidades da Internet das Coisas Musicais, demonstrando, como um estudo de caso, um modelo de aplicação interativa com os usuários e a Internet das Coisas Musicais.

A pesquisa partiu da hipótese de uma aplicação e o protocolo de comunicação em tempo real, gerado das necessidades dos cenários apresentados, garantido a sincronicidade dos dados e entre os usuários, por uma *interface*, viabilizando conexões seguras entre os usuários e servidor. Ao final do desenvolvimento, foi apresentado um modelo de aplicação com um protocolo demonstrando que objetivos foram atendidos.

Neste contexto, ressaltou a relevância da Internet das Coisas Musicais, com sua grande contribuição para sociedade, por meio da arte e da cultura, elencando sua grande dificuldade de expansão por ser pouco explorada.

Foi apresentando uma aplicação modelo, utilizando apenas o primeiro cenário, sendo necessário adaptá-la e implementá-la aos demais cenários.

Com os testes realizados, com pouco mais de 100 conexões simultâneas os resultados foram satisfatórios, cabendo realizar os testes com maior público, com centenas e milhares de conexões simultâneas, para ter uma melhor análise de desempenho da aplicação.

Outro fator desafiador está relacionado a latência, sincronismo e segurança, visto que muitas das soluções da Internet das Coisas Musicais requerem baixa latência, sincronismos satisfatórios com respostas rápidas e segurança entre as partes.

5.1.1 Ferramentas Similares

O *Spotify* lançou em 11 de maio deste uma ferramenta em sua plataforma que permite aos usuários a criação de seções grupais de música. A plataforma possibilita até 100 participantes e função ainda está em fase de testes (beta) e requer assinatura *premium* (não está disponível na versão gratuita). Lembrando que o *YouTube*, utilizado em nosso trabalho de pesquisa, é gratuito. Para mais informações e o passo-a-passo de como utilizar o recurso em (TECHTUDO, 2020).

A extensão *Teleparty* (antiga *Netflix Party*) para navegadores *Chrome e Edge*, oferece recursos de interatividade entre as pessoas, de forma sincronizada, possibilitando assistirem os conteúdos das Plataformas da *Netflix, Disney, Hulu e HBO*, disponibilizando

bate-papo em grupo interativo e sincronizado, contando com uma rede com mais de 10 milhões de pessoas (MICROSOFT EDGE, 2020; TELEPARTY, 2020).

5.1.2 Trabalhos Futuros

Embora este trabalho de pesquisa tenha alcançado suas expectativas em apresentar um modelo de aplicação funcional para Internet das Coisas Musicais com compartilhamento de solução musical utilizando o primeiro cenário, fica o desafio da adaptação para implementação dos demais cenários, principalmente empregando um controlador, que receba todas as ações dos usuários e decida qual conteúdo será executado. Também é necessária a realização de testes de carga, com uma maior quantidade de usuários conectados. Para o cenário 4, um estudo mais aprofundado de rede móveis se faz necessário, já que os veículos estarão em movimento e provavelmente utilizando conexões por dados móveis.

Outro projeto interessante, seria adaptar a solução com o protocolo da MQTT, que durante apresentado na pesquisa demonstrou ser um protocolo aceitável permitindo ampliar a abrangência da solução.

Para aplicações em eventos ao vivo, é necessário maior esforço em estudos e desenvolvimento de protocolo e aplicação com redução da latência e com respostas mais rápidas para requisições, de acordo com as demandas exigidas por tal aplicação.

Referências

OPICE BLUM. *Regulamentação da Internet das Coisas pautada no fomento aos negócios e não em barrá-los*. 2017. Disponível em: <https://opiceblum.com.br/nao-categorizado-/regulamentacao-da-internet-das-coisas-pautada-no-fomento-aos-negocios-e-nao-em-barra-los>. Acesso em: 27 de maio de 2020 às 11:13h. Citado 2 vezes nas páginas 6 e 14.

SANTOS, B. P.; SILVA, L. A.; CELES, C.; BORGES, J. B.; NETO, B. S. P.; VIEIRA, M. A. M.; VIEIRA, L. F. M.; GOUSSEVSKAIA, O. N.; LOUREIRO, A. Internet das coisas: da teoria à prática. *Minicursos SBRC-Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuıdos*, p. 31, 2016. Citado 7 vezes nas páginas 6, 14, 15, 17, 19, 20 e 21.

STEFANUTO, I.; SANTOS, J. A. M. dos; TORRES, C. T. Evolução das redes sem fio: Comparativo entre wi-fi e bluetooth. *Caderno de Estudos Tecnológicos*, v. 4, n. 1, 2016. Citado 3 vezes nas páginas 6, 18 e 19.

LACEY, L. *Como a Internet das coisas poderia impactar Composição Musical, Produção*. 2015. Disponível em: <https://ask.audio/articles/how-the-internet-of-things-could-impact-music-composition-production-performance/p>. Acesso em: 19 de novembro de 2019 às 22:24h. Citado 3 vezes nas páginas 6, 23 e 30.

GUITAR MODERNE. *Sensus Smart Guitar*. 2016. Disponível em: <https://www-.guitarmoderne.com/gear-2/sensus-smart-guita>. Acesso em: 24 de novembro de 2019 às 21:56h. Citado 3 vezes nas páginas 6, 23 e 24.

FILHO, P. M. Direitos autorais na internet. *Ciência da Informação*, SciELO Brasil, v. 27, n. 2, 1998. Citado 2 vezes nas páginas 6 e 24.

BRADY DYER. *360 grau Video Orchestra Wellington "ODES TO JOY"VR Beethoven Symphony No 9*. 2016. Disponível em: <https://www.youtube.com-/watch?v=xQLB3g0Kqm>. Acesso em: 24 de novembro de 2019 às 23:10h. Citado 2 vezes nas páginas 6 e 25.

FOLHA UOL. *Brasileiro é mais apaixonado por música do que americanos e ingleses, diz pesquisa*. 2018. Disponível em: <https://f5.folha.uol.com.br/musica/2018/11/brasileiro-e-mais-apaixonado-por-musica-do-que-americanos-e-ingleses-diz-pesquisa.shtm>. Acesso em: 12 de junho de 2020 às 20:36h. Citado na página 11.

OUTRAS PALAVRAS. *Como a pandemia atinge o mundo da Cultura*. 2020. Disponível em: <https://outraspalavras.net/outrasmidias/como-a-pandemia-atinge-o-mundo-da-cultura>. Acesso em: 11 de novembro de 2020 às 21:08h. Citado na página 11.

G1. *Coronavírus já causou prejuízo de mais de R$ 480 milhões no mercado musical do Brasil, mostra pesquisa*. 2020. Disponível em: <https://g1.globo.com/pop-arte-/musica/noticia/2020/04/04/coronavirus-ja-gerou-prejuizo-de-mais-de-r-480-milhoes-no-mercado-musical-do-brasil-mostra-pesquisa.ghtm>. Acesso em: 01 de novembro de 2020 às 20:35h. Citado na página 11.

GALEGALE, G. P.; SIQUEIRA, É.; SILVA, C. B. H.; SOUZA, C. A. d. Internet das coisas aplicada a negócios-um estudo bibliométrico. *JISTEM-Journal of Information Systems and Technology Management*, SciELO Brasil, v. 13, n. 3, p. 423–438, 2016. Citado na página 13.

SINGER, T. Tudo conectado: conceitos e representações da internet das coisas. *Simpósio em tecnologias digitais e sociabilidade*, v. 2, p. 1–15, 2012. Citado na página 13.

IEEE. *IEEE Internet of Things*. 2019. Disponível em: <https://iot.ieee.org/definition-.htm>. Acesso em: 27 de outubro de 2019 às 14:15h. Citado na página 13.

ROSA, L. d. S. P.; BARCELOS, R. G.; PRADO, Y. P.; REAL, Y. Aplicações do 5g em internet das coisas (iot). 2019. Citado 4 vezes nas páginas 13, 16, 20 e 21.

DIREITOS DA COMUNICAçãO. *Publicado Decreto do Plano Nacional da Internet das Coisas*. 2019. Disponível em: <https://direitodacomunicacao.com/internet/publicado-decreto-do-plano-nacional-da-internet-das-coisas>. Acesso em: 12 de dezembro de 2019 às 14:10h. Citado na página 13.

PIRES, P. F.; DELICATO, F.; BATISTA, T.; BARROS, T.; CAVALCANTE, E.; PITANGA, M. Plataformas para a internet das coisas. *Minicursos SBRC-Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, 2015. Citado na página 14.

WANG, F.; HU, L.; ZHOU, J.; ZHAO, K. A survey from the perspective of evolutionary process in the internet of things. *International Journal of Distributed Sensor Networks*, SAGE Publications Sage UK: London, England, v. 11, n. 3, p. 462752, 2015. Citado na página 14.

FORBES. *5 Internet Of Things Trends Everyone Should Know About*. 2019. Disponível em: <https://www.forbes.com/sites/bernardmarr/2019/02/04/5-internet-of-things-trends-everyone-should-know-about/65c696444b1>. Acesso em: 02 abril 2019 às 18:18h. Citado 2 vezes nas páginas 14 e 17.

SALES, H. F. S.; SILVA, F. M. d. S. M.; LIMA, B. d. J. L. Adaptação da escala de uso compulsivo de internet para avaliar dependência de smartphone. *Avances en Psicología Latinoamericana*, v. 36, n. 1, p. 155–166, 2018. Citado na página 15.

INTEL. *5G Connected Vehicles and Autonomous Driving: 5G Is Key to Fully Realizing Connected and Autonomous Vehicles*. 2019. Disponível em: <https://www.intel.com.br-/content/www/br/pt/communications/5g-connected-vehicle.htm>. Acesso em: 05 de novembro de 2019 às 9:35h. Citado na página 16.

TURCHET, L.; FISCHIONE, C.; BARTHET, M. Towards the internet of musical things. *smc2017*, smc2017, p. 1–8, 2017. Citado 5 vezes nas páginas 16, 24, 27, 28 e 29.

TANENBAUM, A. *Redes de computadores*. [S.l.]: CAMPUS - RJ, 2003. ISBN 9788535211856. Citado na página 16.

KUROSE, J. F.; ROSS, K. W. *Redes de Computadores e a Internet: uma abordagem top-down*. São Paulo: Person Addison Wesley, 2006. (3). Citado na página 16.

TANENBAUM, A.; WETHERALL, D. *Redes de computadores*. PRENTICE HALL BRASIL, 2011. ISBN 9788576059240. Disponível em: <https://books.google.com.br-/books?id=fnfwkQEACAA>. Citado na página 17.

WI-FI ALLIANCE. *Wi-Fi Alliance*. 2019. Disponível em: <https://www.wi-fi.or>. Acesso em: 24 de outubro de 2019 às 17:31h. Citado na página 18.

ZIGBEE. *The Zigbee Alliance*. 2019. Disponível em: <https://zigbee.org/zigbee-for-developers/about-us>. Acesso em: 30 de outubro de 2019 às 18:14h. Citado na página 19.

LORA ALLIANCE. *What is the LoRaWAN® Specificatio?* 2019. Disponível em: <https://lora-alliance.org/about-lorawa>. Acesso em: 28 de outubro de 2019 às 17:21h. Citado na página 20.

LINK LABS. *SigFox Vs. LoRa: A Comparison Between Technologies & Business Models*. 2019. Disponível em: <https://www.link-labs.com/blog/sigfox-vs-lor>. Acesso em: 28 de outubro de 2019 às 16:54h. Citado na página 20.

IUT-R. *Minimum requirements related to technical performance for IMT-2020 radio interface(s)*. 2019. Disponível em: <https://www.itu.int/md/R15-SG05-C0040/e>. Acesso em: 03 de novembro de 2019 às 15:10h. Citado na página 21.

MELO, R. C. d. S. Sistema de monitoramento de consumo de água utilizando o protocolo de comunicação mqtt. Universidade Federal de Uberlândia, 2018. Citado na página 21.

SKALEX. *Blockchain and P2P Web: Is It 'Web 3.0'?* 2017. Disponível em: <https:/-/www.skalex.io/blockchain-p2p-web>. Acesso em: 08 de novembro de 2019 às 14:42h. Citado 2 vezes nas páginas 21 e 22.

PROJETO IOT. *Projeto IoT – Contagem de Fluxo de Pessoas - Programação e Explicação do CloudMQTT*. 2018. Disponível em: <https://el86a.wordpress.com/2018-/11/27/programacao-e-explicacao-do-cloudmqtt>. Acesso em: 09 de novembro de 2019 às 11:29h. Citado na página 21.

TURCHET, L.; FISCHIONE, C.; ESSL, G.; KELLER, D.; BARTHET, M. Internet of musical things: Vision and challenges. *IEEE Access*, v. 6, p. 61994–62017, 09 2018. Citado 3 vezes nas páginas 22, 25 e 26.

OHM STUDIO. *Together sounds better*. 2013. Disponível em: <https://www-.ohmstudio.com>. Acesso em: 20 de novembro de 2019 às 21:47h. Citado na página 23.

ELK AUDIO. *The Smart Guitar: Elk Powered prototype*. 2019. Disponível em: <https://elk.audio/sensus-smart-guitar>. Acesso em: 24 de novembro de 2019 às 21:47h. Citado na página 23.

HAZZARD, A.; BENFORD, S.; CHAMBERLAIN, A.; GREENHALGH, C.; KWON, H. Musical intersections across the digital and physical. 2014. Citado na página 23.

O'BANNON, R. *Why Virtual Reality and a Symphony Orchestra are a Natural Fit*. 2015. Disponível em: <https://www.bsomusic.org/stories/why-virtual-reality-and-a-symphony-orchestra-are-a-natural-fit>. Acesso em: 24 de novembro de 2019 às 22:51h. Citado na página 25.

O'BANNON, R. *Virtual Reality at the Orchestra*. 2018. Disponível em: <https://hackernoon.com/virtual-reality-at-the-orchestra-703ca7f3ce9>. Acesso em: 24 de novembro de 2019 às 23:04h. Citado na página 25.

GREATEST HIT STATION. *Learn These Tips About Downloading Music On The Internet*. 2008. Disponível em: <http://www.hits-station.org/index.php/2008/04/03-/learn-these-tips-about-downloading-music-on-the-internet>. Acesso em: 11 de novembro de 2019 às 11:03h. Citado na página 27.

ROTTONDI, C.; CHAFE, C.; ALLOCCHIO, C.; SARTI, A. An overview on networked music performance technologies. *IEEE Access*, IEEE, v. 4, p. 8823–8843, 2016. Citado 2 vezes nas páginas 29 e 30.

LAZZARO, J.; WAWRZYNEK, J. A case for network musical performance. In: ACM. *Proceedings of the 11th international workshop on Network and operating systems support for digital audio and video*. [S.l.], 2001. p. 157–166. Citado na página 29.

XYLOMENOS, G.; TSILOPOULOS, C.; THOMAS, Y.; POLYZOS, G. C. Reduced switching delay for networked music performance. In: *Packet Video Workshop (Poster Session)*. [S.l.: s.n.], 2013. Citado na página 29.

VALOR GLOBO. *Carreatas pelo país pedem fim do isolamento*. 2020. Disponível em: <https://valor.globo.com/politica/noticia/2020/04/19/carreatas-pelo-pas-pedem-fim-do-isolamento.ghtm>. Acesso em: 09 de outubro de 2020 às 18:48h. Citado na página 39.

VEJA ABRIL. *Brasília tem carreata pelo fim do isolamento social*. 2020. Disponível em: <https://veja.abril.com.br/politica/brasilia-tem-carreata-pelo-fim-do-isolamento-social>. Acesso em: 09 de outubro de 2020 às 18:52h. Citado na página 39.

ISHIHARA, G. L. Protocolo de comunicação para uma rede de sensores sem fio: teoria e implementação em soc. 2006. Citado na página 40.

COMPUTACAO, E. D.; FIRMINO, E. C. D. M. Websocket para aplicacoes web em erlang. 2011. Citado 2 vezes nas páginas 42 e 46.

GONÇALVES, G. S.; BASTOS, P. H. O.; OLIVEIRA, D. de. O uso de websockets no desenvolvimento de sistemas baseados em uma arquitetura front-end com api. In: SBC. *Anais da I Escola Regional de Sistemas de Informação do Rio de Janeiro*. [S.l.], 2014. p. 57–63. Citado 2 vezes nas páginas 42 e 46.

ZINE. *How to Control YouTube's Video Player with JavaScript*. 2015. Disponível em: <https://tutorialzine.com/2015/08/how-to-control-youtubes-video-player-with-javascrip>. Acesso em: 21 de setembro de 2020 às 18:27h. Citado 2 vezes nas páginas 43 e 45.

NUTANIX. *Elevate your Private Cloud with the Power of Hyperconvergence*. 2020. Disponível em: <https://www.nutanix.com/go/what-is-nutanix-hyperconverged-infrastructure?utm_source=google_adwordsutm_medium=paid_searchutm_campaign=Google_Search_NA-BZ-Brand-Alpha-Englishutm_term=nutanix\%2520prismutm_experience=gclid=CjwKCAiA_eb-BRB2EiwAGBnXXvQetr6BU8ax400OwAhzjnVLqPJagr3s5Ez5TM9-_LKFfyJu_56WQBoClukQAvD_Bw>. Acesso em: 16 de dezembro de 2020 às 19:26h. Citado na página 53.

TECHTUDO. *Spotify: como iniciar uma sessão grupal para ouvir música com amigos*. 2020. Disponível em: <https://www.techtudo.com.br/dicas-e-tutoriais/2020/05/spotify-como-iniciar-uma-sessao-grupal-para-ouvir-musica-com-amigos.ghtm>. Acesso em: 16 de dezembro de 2020 às 18:13h. Citado na página 56.

MICROSOFT EDGE. *Netflix Party is now Teleparty*. 2020. Disponível em: <https://microsoftedge.microsoft.com/addons/detail/netflix-party-is-now-tele-/igbncjcgfkfnfgbaieiimpfkobabmkce>. Acesso em: 16 de dezembro de 2020 às 18:39h. Citado na página 57.

TELEPARTY. *A new way to watch TV together*. 2020. Disponível em: <https://www-.netflixparty.com>. Acesso em: 16 de dezembro de 2020 às 18:29h. Citado na página 57.

Anexos

ANEXO A – Aplicação: Códigos Fonte

A.1 Código do Cliente (index.html)

```
<!DOCTYPE html>
<html>
<!-- HTML 4 -->
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<!-- HTML5 -->
<meta charset="utf-8" />

<head>
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery
  .min.js"></script>
  <title>Websocket Player Interativo</title>
  <link rel="stylesheet" type="text/css" href="stylesheet.css"
  media="screen" />
</head>

<body>
  <div class="buttons">
    <div class="voltar button"> << </div>
    <div class="play button"> > </div>
    <div class="pause button"> || </div>
    <div class="reiniciar button"> [] </div>
    <div class="avancar button"> >> </div>
  </div>

  <div>
<input type="range" id="progBar" value="0">
  </div>

  <div style="display: block;">
<div id="ytplayer"></div>
<br />
```

```
<p>Tempo: <span id="contador"></span></p>
  </div>

  <script>
var newtime;
$('#progBar').on('mouseup touchend', function (e) {
  newTime = player.getDuration() * (e.target.value / 100);
  var obj = { "action": "progresso", "numero": newTime };
  websocket.send(JSON.stringify(obj));
});
var output = document.getElementById("contador");
  </script>

  <div class="state">
    <span class="users">?</span> online
  </div>

  <div id="video-placeholder"></div>

  <div>
<b>Playlist</b>
  </div>

  <div>
<textarea disabled id="itensPlaylist" style="width: 550px; height:
70px;"></textarea>
  </div>

  <div>
<b>ID do vídeo do YT:</b> <input type=text id=p1 style="width: 100px;">
<button id=botaoMudarPlaylist onclick="mudaPlaylist();">Adicionar
vídeo na Playlist</button>
  </div>

  <div>
<b>Chat</b>
  </div>

  <div>
```

```
<textarea disabled id="campoMensagemReceber" style="width: 550px;
height: 70px;"></textarea>
  </div>

  <div>
<b>Nome:</b>
<input type=text id=nomeUsuario style="width: 100px;">
<b>Mensagem:</b>
<input type=text id=campoMensagemEnviar style="width: 200px;">
<button id=botaoEnviar onclick="enviarMensagem();">Enviar</button>
  </div>

  </div>

  <script src="https://www.youtube.com/iframe_api"></script>
  <script>
    var videoIDs = [
     '6LK5CZPDMfk',
     'sx4BGbyWzQY',
     'Yn53CGOZXRs',
     '9v5w258KZuk'
    ];

    document.addEventListener('keypress', function (e) {
      if (e.which == 13) {
        $('#botaoEnviar').click();
      }
    }, false);

    function mudaPlaylist() {
      var obj = { "action": "atualizarPlaylist", "p1":
      document.getElementById("p1").value };
        websocket.send(JSON.stringify(obj));
    }

    function enviarMensagem() {
      if (document.getElementById("nomeUsuario").value == "")
        alert("Escreva o nome do usuário.");
      else {
```

```
        var obj = { "action": "enviar", "texto":
        document.getElementById("nomeUsuario").value + ": " +
        document.getElementById("campoMensagemEnviar").value + "\n" };
        websocket.send(JSON.stringify(obj));
      }
    }

/////////API YOUTUBE//////////////////
  var player;
  var currentVideoId = 0;
  function onYouTubeIframeAPIReady() {
    player = new YT.Player('video-placeholder', {
      width: 600,
      height: 400,
      playerVars: {
        playlist: '6LK5CZPDMfk, 7sbbb7Dt3D8, KS2oecN-aVU, 4jqH1rLAQuM',
          'controls': 0,
          'modestbranding': 1,
          'rel': 0,
          'showinfo': 0
      },
      events: {
        'onReady': initialize
      }
    });
  }

  function formatTime(time) {
    time = Math.round(time);
    var minutes = Math.floor(time / 60),
      seconds = time - minutes * 60;
      seconds = seconds < 10 ? '0' + seconds : seconds;
      return minutes + ":" + seconds;
  }

  function updateProgressBar(event) {
    $('#progBar').val((player.getCurrentTime() / player.getDuration())
    * 100);
    output.innerHTML = formatTime(player.getCurrentTime()) + "/" +
```

```
    formatTime(player.getDuration());
  }

  function initialize(event) {
currentVideoId = 0;
x = currentVideoId + 1;
updateProgressBar();
time_update_interval = setInterval(function () {
updateProgressBar();
    }, 1000)
  }

//////////////WEBSOCKETS///////////////
  var play = document.querySelector('.play'),
pause = document.querySelector('.pause'),
users = document.querySelector('.users'),
avancar = document.querySelector('.avancar'),
voltar = document.querySelector('.voltar'),
reiniciar = document.querySelector('.reiniciar'),
enviar = document.querySelector('.enviar'),
websocket = new WebSocket("wss://alexandre.ufsj.edu.br:6789/");

  play.onclick = function (event) {
websocket.send(JSON.stringify({ action: 'play' }));
  }

  pause.onclick = function (event) {
websocket.send(JSON.stringify({ action: 'pause' }));
  }

  voltar.onclick = function (event) {
websocket.send(JSON.stringify({ action: 'voltar' }));
if (currentVideoId == 0) {
}
else {
  var obj = { "action": "atualizaContador", "valorCorrente":
  --currentVideoId };
  websocket.send(JSON.stringify(obj));
}
```

```
  }

  avancar.onclick = function (event) {
websocket.send(JSON.stringify({ action: 'avancar' }));
if (currentVideoId < videoIDs.length - 1) {
  var obj = { "action": "atualizaContador", "valorCorrente":
  ++currentVideoId };
  websocket.send(JSON.stringify(obj));
}
else {
}
  }

  reiniciar.onclick = function (event) {
websocket.send(JSON.stringify({ action: 'reiniciar' }));
  }

  websocket.onmessage = function (event) {
data = JSON.parse(event.data);
switch (data.type) {
  case 'state':
if (data.value == "play") {
  player.playVideo();
}
else if (data.value == "pause") {
  player.pauseVideo();
}
else if (data.value == "avancar") {
  // player.nextVideo();
}
else if (data.value == "voltar") {
  //   player.previousVideo();
}
else if (data.value == "reiniciar") {
  player.seekTo(0);
  player.pauseVideo();
}
else if (data.value == "progresso") {
  player.seekTo(data.numero);
```

```
}
else if (data.value == "atualizaContador") {
  currentVideoId = data.valorCorrente;
     player.loadVideoById(videoIDs[data.valorCorrente]);
  x = currentVideoId + 1;
  document.getElementById("itensPlaylist").value = "Vídeos: " +
  videoIDs + "\n" + "Video Corrente: " + x +"/" +
  videoIDs.length;
}
else if (data.value == "enviar") {
  document.getElementById("campoMensagemReceber").value =
  document.getElementById("campoMensagemReceber").value +
  data.texto;
}
else if (data.value == "atualizarPlaylist") {
  document.getElementById("p1").value = data.p1;
  videoIDs.push(document.getElementById("p1").value);
      player.loadPlaylist({
    playlist: videoIDs
  });
  player.loadVideoById(videoIDs[data.valorCorrente]);
  x = currentVideoId + 1;
  document.getElementById("itensPlaylist").value = "Vídeos: " +
  videoIDs + "\n" + "Video Corrente: " + x +"/" +  videoIDs.length;
}
break;
  case 'users':
users.textContent = (
  data.count.toString() + " usuário" +
  (data.count == 1 ? "" : "s"));
  break;
  default:
console.error(
  "unsupported event", data);
}
  };
  </script>
</body>
```

```
</html>
```

A.1.1 CSS: Estilo da página Web (stylesheet.css)

```
body {
font-family: "Courier New", sans-serif;
text-align: center;
}

div {
margin-top: 20px;
}

.buttons {
font-size: 4em;
display: flex;
justify-content: center;
}

.button,
.value {
line-height: 1;
padding: 2rem;
margin: 2rem;
border: medium solid;
min-height: 1em;
min-width: 1em;
}

.button {
cursor: pointer;
user-select: none;
}

.play {
color: green;
}
```

```
.pause {
color: red;
}

.value {
min-width: 2em;
}

.state {
font-size: 2em;
}

input[type=range] {
-webkit-appearance: none;
margin: 18px 0;
width: 50%;
}

input[type=range]:focus {
outline: none;
}

input[type=range]::-webkit-slider-runnable-track {
width: 100%;
height: 8.4px;
cursor: pointer;
box-shadow: 1px 1px 1px #000000, 0px 0px 1px #0d0d0d;
background: #3071a9;
border-radius: 1.3px;
border: 0.2px solid #010101;
}

input[type=range]::-webkit-slider-thumb {
box-shadow: 1px 1px 1px #000000, 0px 0px 1px #0d0d0d;
border: 1px solid #000000;
height: 36px;
width: 16px;
border-radius: 3px;
```

```
background: #ffffff;
cursor: pointer;
-webkit-appearance: none;
margin-top: -14px;
}

input[type=range]:focus::-webkit-slider-runnable-track {
background: #367ebd;
}

input[type=range]::-moz-range-track {
width: 100%;
height: 8.4px;
cursor: pointer;
box-shadow: 1px 1px 1px #000000, 0px 0px 1px #0d0d0d;
background: #3071a9;
border-radius: 1.3px;
border: 0.2px solid #010101;
}

input[type=range]::-moz-range-thumb {
box-shadow: 1px 1px 1px #000000, 0px 0px 1px #0d0d0d;
border: 1px solid #000000;
height: 36px;
width: 16px;
border-radius: 3px;
background: #ffffff;
cursor: pointer;
}

input[type=range]::-ms-track {
width: 100%;
height: 8.4px;
cursor: pointer;
background: transparent;
border-color: transparent;
border-width: 16px 0;
color: transparent;
}
```

```
input[type=range]::-ms-fill-lower {
background: #2a6495;
border: 0.2px solid #010101;
border-radius: 2.6px;
box-shadow: 1px 1px 1px #000000, 0px 0px 1px #0d0d0d;
}

input[type=range]::-ms-fill-upper {
background: #3071a9;
border: 0.2px solid #010101;
border-radius: 2.6px;
box-shadow: 1px 1px 1px #000000, 0px 0px 1px #0d0d0d;
}

input[type=range]::-ms-thumb {
box-shadow: 1px 1px 1px #000000, 0px 0px 1px #0d0d0d;
border: 1px solid #000000;
height: 36px;
width: 16px;
border-radius: 3px;
background: #ffffff;
cursor: pointer;
}

input[type=range]:focus::-ms-fill-lower {
background: #3071a9;
}

input[type=range]:focus::-ms-fill-upper {
background: #367ebd;
}
```

A.2 Código do Servidor (server.py)

```
#!/usr/bin/env python

import asyncio
```

```
import json
import logging
import websockets
import ssl

logging.basicConfig()

STATE = {"value": 0}

USERS = set()

def state_event():
  return json.dumps({"type": "state", **STATE})

def users_event():
  return json.dumps({"type": "users", "count": len(USERS)})

async def notify_state():
  if USERS:  #
    message = state_event()
    await asyncio.wait([user.send(message) for user in USERS])

async def notify_users():
  if USERS:  #
    message = users_event()
    await asyncio.wait([user.send(message) for user in USERS])

async def register(websocket):
  USERS.add(websocket)
  await notify_users()

async def unregister(websocket):
  USERS.remove(websocket)
  await notify_users()

async def counter(websocket, path):
  await register(websocket)
  try:
    await websocket.send(state_event())
```

```
    async for message in websocket:
      data = json.loads(message)
      if data["action"] == "play":
        STATE["value"] = "play"
        await notify_state()
      elif data["action"] == "pause":
        STATE["value"] = "pause"
        await notify_state()
      elif data["action"] == "voltar":
        STATE["value"] = "voltar"
        await notify_state()
      elif data["action"] == "avancar":
        STATE["value"] = "avancar"
        await notify_state()
      elif data["action"] == "reiniciar":
        STATE["value"] = "reiniciar"
        await notify_state()
      elif data["action"] == "progresso":
        STATE["value"] = "progresso"
        STATE["numero"] = data["numero"]
        await notify_state()
      elif data["action"] == "enviar":
        STATE["value"] = "enviar"
        STATE["texto"] = data["texto"]
        await notify_state()
      elif data["action"] == "atualizarPlaylist":
        STATE["value"] = "atualizarPlaylist"
        STATE["p1"] = data["p1"]
        await notify_state()
      elif data["action"] == "atualizaContador":
        STATE["value"] = "atualizaContador"
        STATE["valorCorrente"] = data["valorCorrente"]
        await notify_state()
      else:
        logging.error("unsupported event: {}", data)
  finally:
    await unregister(websocket)

# Create the ssl context (to use https)
```

```
ssl_context = ssl.SSLContext(ssl.PROTOCOL_TLS_SERVER)
path_cert = "/var/www/html/flavio/certificado.pem"
path_key = "/var/www/html/flavio/chave.key"
ssl_context.load_cert_chain(path_cert, keyfile=path_key)

start_server = websockets.serve(
  counter, "alexandre.ufsj.edu.br", 6789, ssl=ssl_context)

asyncio.get_event_loop().run_until_complete(start_server)
asyncio.get_event_loop().run_forever()
```