UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
CURSO DE GRADUAÇÃO EM ENGENHARIA ELÉTRICA

LEONARDO SCHOLL STERNBERG

DESENVOLVIMENTO DE PAINEL ANUNCIADOR DE INFORMAÇÕES COM ATUALIZAÇÃO EM TEMPO REAL PARA ESTAÇÕES METROFERROVIÁRIAS

Porto Alegre
2021

LEONARDO SCHOLL STERNBERG

DESENVOLVIMENTO DE PAINEL ANUNCIADOR DE INFORMAÇÕES COM ATUALIZAÇÃO EM TEMPO REAL PARA ESTAÇÕES METROFERROVIÁRIAS

Este Projeto de Diplomação foi analisado e julgado adequado para a obtenção do grau de bacharel em Engenharia Elétrica e aprovado em sua forma final pelo Orientador e pela Banca Examinadora.

_____

Prof. Dr. Raphael Martins Brum – UFRGS

Orientador – UFRGS

_____

Prof. Dr. Marcelo Soares Lubaszewski – UFRGS

Responsável pelos Projetos de Diplomação da Engenharia Elétrica - UFRGS

Aprovado em: ____ de _____ de _____

Banca Examinadora:

_____

Prof. Dr. Luiz Tiarajú dos Reis Loureiro – UFRGS

_____

Prof. Dr. Tiago Oliveira Weber – UFRGS

_____

Eng. David Samuel Levenfus – Trensurb

AGRADECIMENTOS

Agradeço aos meus pais, Lennart e Claudia, que sempre me apoiaram nas minhas decisões, me ajudaram a crescer e servem de exemplo para eu ser cada vez melhor.

A minha irmã, Alessandra, que sempre esteve disposta a ajudar independente das situações ou dos nossos humores, e me demonstrou o incentivo de estudar e sempre tentar crescer na vida.

Aos meus primos e prima próximos Scholl, que para desconhecidos eu nem saberia explicar como a nossa família funciona, com pessoas tão únicas, complementares e incríveis cada um de sua forma. Conseguiram me proporcionar os melhores momentos sempre com situações inesperadas.

A todos os meus amigos e amigas, em especial ao Arateus Meneses e André Pocos, que me mostraram o mundo com olhos mais humanos e ensinaram a valorizar a rica variedade de opiniões.

Aos meus colegas de trabalho da Trensurb, em especial ao David Levenfus, Ricardo Riboldi, Jossel Gomes e Rodrigo Alves, que me fizeram crescer não somente como um profissional, mas também como pessoa em toda a minha caminhada na empresa, vocês são pessoas fantásticas.

E a todos os colegas de faculdade que conheci, que me mostraram que o mundo não é uma competição, e que só através da cooperação podemos mais facilmente passar pelas dificuldades e crescer como pessoas.

RESUMO

Neste trabalho, apresentamos um sistema de painéis anunciadores de informações para as estações da Trensurb, em que estes painéis se comunicam com um servidor central localizado no Prédio Administrativo através de uma arquitetura cliente-servidor. Durante o seu desenvolvimento, levantou-se os requisitos de obter-se uma apresentação no formato de lista de reprodução, com atualização das informações em tempo real, a criação de uma interface gráfica com o propósito de atualizar a lista de reprodução, a sincronização horária de todos os clientes das estações, de um servidor local de *backup* e de um serviço de acesso remoto. Para a criação dos painéis foi instalado um Raspberry Pi por monitor, e que se comunicam com demais dispositivos utilizando o protocolo de comunicação HTTP. Também, foi desenvolvido uma aplicação administrativa do sistema para atualizar e monitorar o servidor e os painéis. Para o sistema computacional, foi utilizado a linguagem de programação Python em todos os módulos do sistema, JavaScript nos clientes das estações dentro das diversas páginas *web* seguindo o modelo de HTML, e um banco de dados MySQL no servidor. Cinco serviços foram desenvolvidos para atuação dentro dos módulos funcionais do sistema, e juntos resultam no *software* da aplicação. Os testes e as implementações atuais mostram que o sistema criado funciona e atende a todos os requisitos. Este trabalho foi implementado em versão protótipo no saguão do Prédio Administrativo e na bilheteria da Estação Aeroporto. Pretende-se expandir a implementação para as demais estações e agregar novas funcionalidades, como atrelar de forma autônoma aos alertas operacionais, agrupar ao sistema já existente de sonorização e informar o tempo de chegada dos trens.

Palavras-chave: Painéis Informativos, Informação a Passageiros em Transportes, Trensurb, Trem Urbano.

ABSTRACT

In this work, we present an announcing panels information system for Trensurb's stations, in which these panels communicate with a central server located in the Administrative Building through a client-server architecture. During its development, requirements were raised to obtain a presentation in a playlist format, with updates of information in real time, the creation of a graphical interface with the purpose of updating the playlist, the time synchronization of all station's clients, a local backup server and a remote access service. For the panel's creation, a Raspberry Pi was installed per monitor, which communicates with other devices using the HTTP communication protocol. Also, an administrative system application was also developed to update and monitor the server and panels. For the computational system, the Python programming language was used in all the modules of the system, JavaScript in the station's clients within the various web pages following the HTML model, and a MySQL database in the server. Five services were developed to work within the functional modules of the system, and together they result in the application software. Current tests and implementations show that the system created works and meets all requirements. This work was implemented in a prototype version in the lobby of the Administrative Building and at the ticket office of the Airport Station. It is intended to expand the implementation to other stations and add new features, such as autonomously coupling operational alerts, adding to the existing sound system and informing the arrival time of the trains.

Keywords: Information Panels, Transport Passenger Information, Trensurb, Urban Train.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API – Interface de Programação de Aplicativos (*Application Programming Interface*)

CCO – Centro de Controle Operacional

CSS – Folhas de Estilo em Cascata (*Cascading Style Sheets*)

GECIN – Gerência de Comunicação Integrada

GUI – Interface Gráfica do Usuário (*Graphical User Interface*)

HDMI – Interface Multimídia de Alta Definição (*High-Definition Multimedia Interface*)

HTML – Linguagem de Marcação de Hipertexto (*HyperText Markup Language*)

HTTP – Protocolo de Transferência de Hipertexto (*Hypertext Transfer Protocol*)

ICMP – Protocolo de Mensagens de Controle da Internet (*Internet Control Message Protocol*)

IP – Protocolo de Internet (*Internet Protocol*)

NTP – Protocolo de Tempo para Redes (*Network Time Protocol*)

OS – Sistema Operacional (*Operational System*)

OSI – Interconexão de Sistemas Abertos (*Open Systems Interconnection*)

SEITEC – Setor de Projetos de Sistemas e Inovação Tecnológica

SEMOB – Setor de Mobilidade Urbana

TCP – Protocolo de Controle de Transmissão (*Transmission Control Protocol*)

UDP – Protocolo de Datagrama do Usuário (*User Datagram Protocol*)

UFRGS – Universidade Federal do Rio Grande do Sul

URL - Localizador Uniforme de Recursos (*Uniform Resource Locator*)

USB – Barramento Serial Universal (*Universal Serial Bus*)

VLAN – Rede Local Virtual (*Virtual Local Area Network*)

VNC – Computação em Rede Virtual (*Virtual Network Computing*)

SUMÁRIO

1. INTRODUÇÃO

A Empresa de Trens Urbanos de Porto Alegre S.A. foi criada em abril de 1980, através do Decreto nº 84.640, para implantar e operar uma linha de trens urbanos no Eixo Norte da Região Metropolitana de Porto Alegre, atendendo diretamente às populações dos municípios de Porto Alegre, Canoas, Esteio, Sapucaia do Sul, São Leopoldo e Novo Hamburgo. Ela possui mais de 40 trens em sua frota, atendendo a milhares de pessoas diariamente. A empresa mostrou a sua necessidade pela oferta à população dos municípios mencionados como uma alternativa de transporte com baixo custo e com maior rapidez, segurança, conforto e capaz de absorver uma demanda inicialmente prevista na casa dos 300 mil passageiros por dia. A Trensurb, ao longo da sua história, consolidou-se como uma empresa de transporte de passageiros, indutora de desenvolvimento social e econômico. Para cumprir sua missão, a empresa está organizada em cinco grandes áreas: Operação, Manutenção, Administração, Expansão e Comercial (TRENSURB [s.d.]).

Um serviço disponibilizado por diversas empresas metroferroviárias ou outras companhias de transportes são de painéis informativos, que tem a função de apresentar aos passageiros informações sobre operações, a instituição, segurança, saúde ou quaisquer outras pertinentes. A Trensurb também possui interesse nesse serviço, e este projeto apresenta uma alternativa a ser implementada voltada ao contexto da Trensurb.

Desta forma, neste trabalho é apresentado uma alternativa de modelo de sistema de painéis anunciadores de informação voltado para as estações da Trensurb. Estes painéis têm o objetivo de proporcionar uma melhor qualidade de serviço para os passageiros das estações, uma melhor acessibilidade nas estações, por motivos de segurança pública, para informar passageiros sobre questões operacionais, e para melhor informar usuários sobre medidas públicas como o de saúde em relação ao COVID-19. Estes painéis dentro da Trensurb têm a possibilidade de ser implementados em bilheterias, plataformas, acessos externos ou em ambientes internos. Devido a essa tecnologia não ser localizada gratuitamente na Internet ou como software livre, está proposto neste projeto uma alternativa de desenvolvimento destes painéis utilizando as ferramentas de *hardware* e *software* já disponíveis ou de fácil acesso na empresa. Buscou-se desenvolver um sistema robusto e flexível com

relação aos tipos de informações que podem ser apresentadas, funcionalidades e com potencialidade de expansão.

Como pré-requisitos do trabalho buscou-se: apresentar as informações no formato de lista de reprodução; a implantação de um computador central na sede da Trensurb que armazena as informações através de arquivos de um banco de dados; a criação de uma interface gráfica que facilite a atualização das informações apresentadas; o desenvolvimento de serviços de sincronização horária para os painéis; o tratamento para caso de falha de comunicação para um sistema de servidor local de *backup*; e o desenvolvimento de um serviço de acesso remoto entre os dispositivos. Também a fim de demonstrar a efetividade da solução buscou-se implementar a solução na versão protótipo em dois locais.

Este trabalho está organizado na seguinte forma de apresentação dos capítulos: no capítulo 2, os conceitos fundamentais usados no trabalho voltado ao transporte metroferroviário e conceitos de computação como protocolos e arquiteturas. No capítulo 3, a definição do problema voltado ao contexto da Trensurb. No capítulo 4, a especificação, definindo os serviços e os módulos do sistema. No capítulo 5, a implementação e resultados obtidos de cada módulo, com a demonstração do *software* e *hardware*. No capítulo 6, a conclusão e trabalhos futuros, em que é apresentado o ponto de desenvolvimento em que o projeto chegou e algumas sugestões de rumos de expansão do projeto. Por último, no capítulo 7, as referências usadas.

2. CONCEITOS FUNDAMENTAIS

Este capítulo traz seções sobre o transporte coletivo e conceitos fundamentais para o leitor que não está familiarizado com os temas.

2.1 Transporte Metroferroviário

O transporte coletivo é um sistema cuja característica principal é o transporte conjunto dos passageiros num único meio de transporte. Normalmente funciona em horários programados e rotas estabelecidas com um custo de embarque por passageiro. Sua principal função é conseguir transportar de forma rápida, eficiente, segura, com conforto e com qualidade pessoas de um local para o outro a fins de trabalho, estudo, viagem ou outros motivos. Ele pode ser um indutor de desenvolvimento social e econômico ao conectar pessoas de diferentes regiões. Ele é considerado pela Constituição Federal um serviço essencial, ou seja, a sua interrupção pode colocar em perigo a sobrevivência, a saúde e/ou segurança da população.

A Trensurb, a Empresa de Trens Urbanos de Porto Alegre S. A., atende os requisitos de transporte coletivo ao conectar as populações dos municípios de Porto Alegre, Canoas, Esteio, Sapucaia do Sul, São Leopoldo e Novo Hamburgo. Ela introduziu e motivou mudanças nos hábitos da população, alterando consideravelmente a realidades dos municípios. Isto porque sua implementação envolveu, não só a instalação do sistema metroviário, mas também a readequação da malha viária, o saneamento, a iluminação pública, a segurança de pedestres e outras obras complementares. Prevista inicialmente para absorver aproximadamente 300 mil passageiros por dia, consolidando-se como uma empresa essencial. Na Tabela 1 é apresentado a distribuição de passageiros transportados anualmente nos anos entre 2016 até 2019.

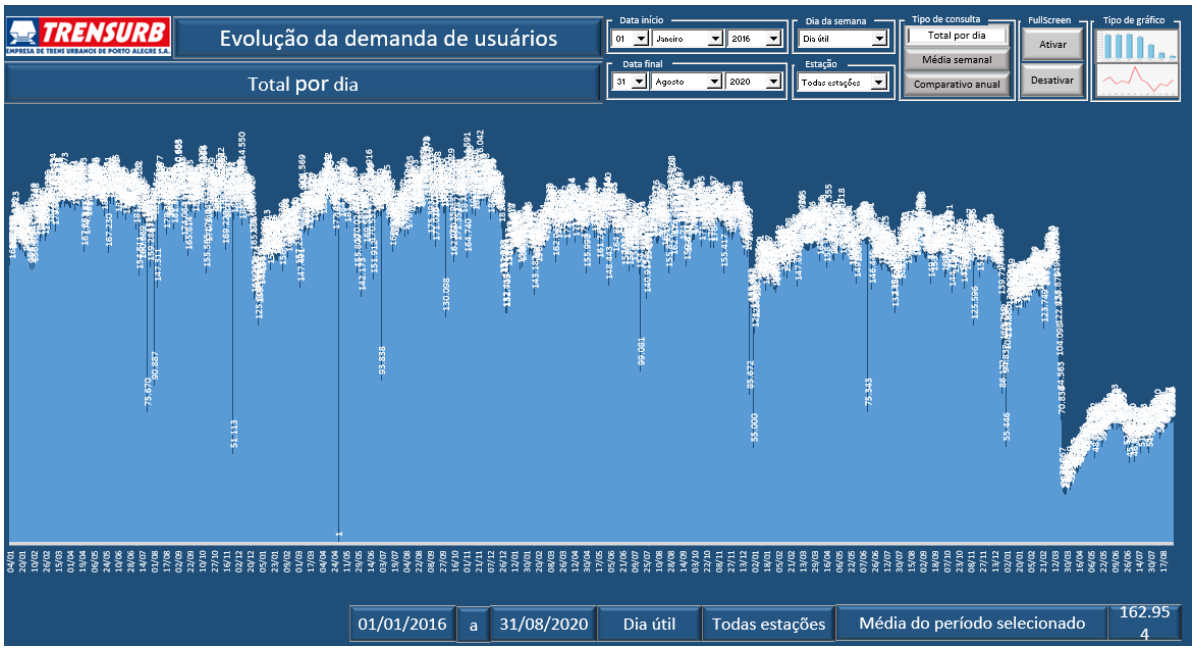
Tabela 1: Número de passageiros transportados mensalmente e anualmente no período entre 2016 e 2019 na Trensurb.

Mês	2016	2017	2018	2019
Janeiro	4.175.321	4.167.490	4.252.457	3.796.198
Fevereiro	4.029.833	3.699.351	3.579.070	3.710.771
Março	5.005.696	4.865.569	4.461.508	4.003.756
Abril	4.701.158	4.234.900	4.409.283	4.194.208
Maio	4.891.144	4.869.926	4.508.118	4.295.275
Junho	4.953.798	4.537.112	4.304.204	3.854.756
Julho	4.579.092	4.709.197	4.238.087	4.100.004
Agosto	5.020.373	5.151.402	4.801.092	4.371.332
Setembro	4.725.671	4.683.879	4.161.966	3.965.391
Outubro	4.827.798	4.762.087	4.668.737	4.189.108
Novembro	4.596.351	4.813.788	4.338.590	3.903.770
Dezembro	4.669.370	4.562.766	4.017.901	3.670.795
Total	**56.175.605**	**55.057.467**	**51.741.013**	**48.055.364**

Fonte: Setor de Mobilidade Urbana (SEMOB) – Trensurb.

Na Figura 1 é demonstrado um gráfico representativo do número médio de passageiros no período de 2016 até 2020, com média aproximada de 150 mil passageiros diariamente no período anterior ao COVID. Também é possível visualizar no gráfico a queda do número de passageiros pelo reflexo do período de pandemia.

Figura 1: Média do número de passageiros diariamente no período entre 2016 e 2020 na Trensurb.



Fonte: SEMOB – Trensurb.

2.2 Sistemas de Apresentação de Informações a Usuários do Transporte Coletivo

Os sistemas de apresentação de informações a usuários de transportes coletivos são ferramentas que têm por objetivo informar passageiros sobre situações operacionais, de segurança, o horário atual, do horário de passagem do transporte, aspectos institucionais, questões educacionais, podendo ser publicitária, assim como diversas outras informações pertinentes. Ele demonstra ser uma ferramenta essencial a fim de garantir uma melhor qualidade para os passageiros.

Existem diversas maneiras e meios utilizados para a apresentação destas informações, podendo ser em forma de Painéis de LED, totens digitais, monitores, telas ou outras formas. Analisando implementações de sistemas de apresentação de informações a passageiros em Porto Alegre, encontramos exemplos como a TV Minuto, circuitos embarcados encontrados nas telas dentro de ônibus e trem, em que inicialmente operava com atualização das informações via pen-drive e atualmente atualiza suas informações com sistema de transmissão e recepção por antenas. Sistemas de apresentação também se encontram nas telas do Aeroporto Internacional Salgado Filho, com monitores mostrando informações de horário de voos.

Em relação ao transporte metroferroviário é encontrada uma variedade de sistemas diferentes sendo implantados pelo mundo, com adaptações próprias para as necessidades locais. Este trabalho se inspirou em diversos sistemas já implementados, especialmente dos sistemas de metrô do Reino Unido e trens da França. Um exemplo de uma aplicação funcionando no trem de passageiros da França encontrado na Figura 2.

Figura 2: Apresentação de informações nos trens da França.

Fonte: (SNCFWORLD, [s.d.])

Em relação a fabricação de monitores, a maior empresa fabricante de sinalização e iluminação em transporte de ônibus, trem, avião militar e comercial pelo mundo é a Luminator. Ela possui uma parceira localizada no Brasil chamada Mobitec que já opera no metrô de São Paulo. Na Figura 3 os locais de operação da empresa Luminator encontrado em (LUMINATOR, [s.d.]).

Figura 3: Locais de operação da empresa Luminator.

Fonte: (MOBITEC, [s.d.])

O software utilizado na apresentação destas informações não é de domínio público, não encontrado no formato *open-source*. Uma alternativa e auxílio no desenvolvimento deste projeto, foram pesquisadas implementações e Interfaces de Programação de Aplicativos (API) similares feitos por pessoas físicas, em que um exemplo encontrado foi o tutorial do letreiro desenvolvido utilizando o Hardware Raspberry Pi e o banco de dados dos trens do Reino Unido, desenvolvido por Crocker-White Chris (2020). Na Figura 4 encontra-se a implementação do tutorial.

Figura 4: Implementação do tutorial do sistema caseiro de sinalização dos metrôs do Reino Unido.

Fonte: (BALENA, [s.d.])

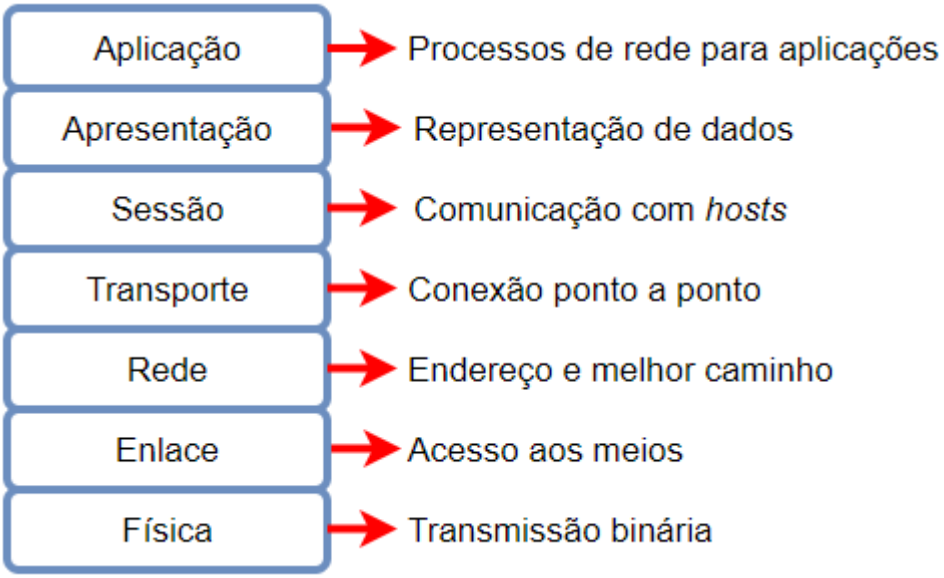
Desta forma, é possível visualizar diversos exemplos de implementações em pequena e grande escala pelo mundo. Isto nos fornece uma visão geral do projeto.

2.3 Rede de Computadores e Modelo de Comunicação

De acordo com Serpanos e Wolf (2014), a rede de computadores ou rede de dados é um conjunto de dois ou mais dispositivos eletrônicos de computação interligados por um sistema de comunicação digital guiados por um conjunto de regras ou protocolos de rede a fim de compartilhar informações, serviços, recursos físicos ou lógicos. As conexões podem ser estabelecidas usando mídia com ou sem fio. Os exemplos mais comuns de rede de computadores são a Internet, Intranet de uma instituição, rede doméstica e pública. Para garantir a comunicação entre os dispositivos, modelos de comunicação precisam ser estabelecidos para garantir a comunicação.

O modelo Interconexão de Sistemas Abertos (OSI), de acordo com Tanenbaum, Wetherall (2011), é um modelo conceitual que caracteriza e torna padrão a comunicação de função de um sistema de telecomunicação ou computação sem levar em conta sua estrutura interna e tecnologias subjacentes. Seu objetivo é a interoperabilidade de diversos sistemas de comunicação com protocolos padrão, garantindo a comunicação *end-to-end* entre tipos de sistemas. O modelo particiona o fluxo de dados em um sistema de sete camadas de abstração, desde a implementação física de transmissão de bits em um meio de comunicação até a representação de dados de mais alto nível. Essas camadas (chamadas também de funções) possibilitam conexão, troca de informações entre máquinas e definir diretivas genéricas para a elaboração de redes de computadores independente da tecnologia utilizada, seja ela curta, média ou longa distância. A representação das suas camadas está disposta na Figura 5.

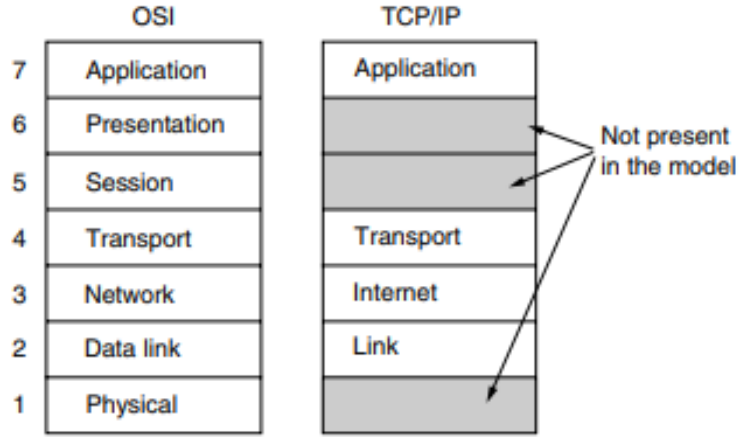
Figura 5: Camadas abstratas do modelo OSI.



Fonte: O autor.

O modelo OSI é capaz de fornecer os fundamentos sobre os protocolos de comunicação, mas é bastante teórico sendo geralmente utilizado só para referências de modelos lógicos. Na prática outros modelos de comunicação são implementados, sendo o Protocolo de Controle de Transmissão/Protocolo de Internet (TCP/IP) o mais utilizado no mundo atualmente e que será utilizado neste projeto, encontrado em dispositivos de computadores, celulares, assim como outros dispositivos gerais de comunicação. A seguir na Figura 6 e na Tabela 2 a apresentação das camadas do modelo TCP/IP e a comparação com o OSI.

Figura 6: Comparação das camadas abstratas do modelo OSI com modelo TCP/IP.



Fonte: (TANENBAUM; WETHERALL, 2011)

Tabela 2: Comparação das camadas abstratas do modelo OSI com modelo TCP/IP.

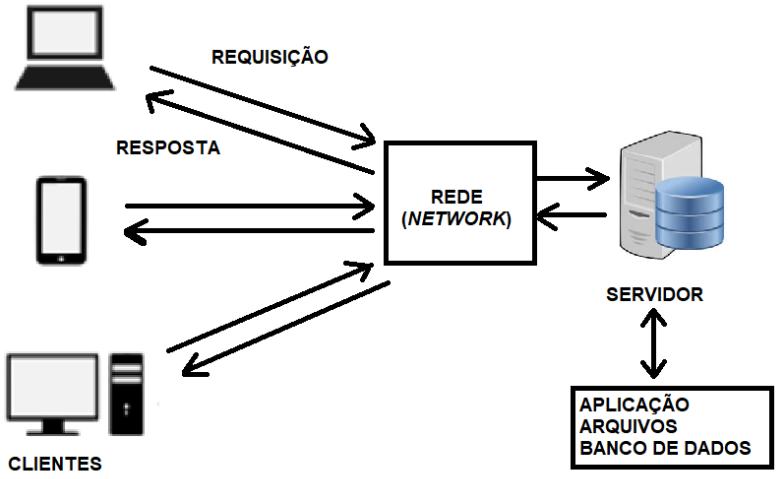
OSI	TCP/IP
Desenvolvido por grupos de trabalho da ISO (*International Standard Organization*).	Desenvolvido por pesquisadores da DARPA (*Defense Advanced Research Projects Agency*).
Desenvolvido previamente aos protocolos que seriam implementados sobre o modelo.	Desenvolvido após os protocolos que seriam implementados sobre o modelo.
Clara distinção entre interface, serviços e protocolos.	Pouca distinção entre interface, serviços e protocolos.
Total independência entre a implementação de diferentes camadas.	Há interdependência entre camadas, o que dificulta a substituição dos protocolos.
Utiliza a camada de rede para definir padrões e protocolos de roteamento.	Utiliza a camada de Internet (IP) para definir padrões e protocolos de roteamento.
Camada de transporte é apenas orientada à conexão.	Camada de transporte pode ser orientada à conexão ou sem conexão.
Camada de enlace de dados e a física são camadas separadas.	Camada de enlace e camada física combinadas em uma única camada.
As camadas de sessão e apresentação fazem parte do modelo OSI.	Não há camada de sessão e apresentação no modelo TCP/IP.
Tornou-se referencial teórico para a compreensão da Teoria de Redes, não mais sendo utilizado na prática.	Tornou-se o padrão *de facto* da Internet e das redes de computadores em geral.
Desenvolvido por grupos de trabalho da ISO (*International Standard Organization*).	Desenvolvido por pesquisadores da DARPA (*Defense Advanced Research Projects Agency*).

Fonte: (TANENBAUM; WETHERALL, 2011) (RUSSELL, 2013).

2.4 Arquitetura Cliente-Servidor

A arquitetura cliente-servidor é uma estrutura de aplicativo distribuída que particiona tarefas ou cargas de trabalho entre os provedores de um recurso ou serviço, chamados servidores, e solicitantes de serviços, chamados clientes. Frequentemente, clientes e servidores se comunicam em uma rede de computadores em hardware separado, mas tanto o cliente quanto o servidor podem residir no mesmo sistema. Um *host* de servidor executa um ou mais programas de servidor, que compartilham seus recursos com os clientes. Um cliente geralmente não compartilha nenhum de seus recursos, mas solicita conteúdo ou serviço de um servidor. Os clientes, portanto, iniciam sessões de comunicação com os servidores, que aguardam as solicitações de entrada. Exemplos de aplicativos de computador que usam o modelo cliente-servidor são e-mail, impressão em rede e a sites gerais encontrados na Internet. Na Figura 7 é representado a arquitetura cliente-servidor.

Figura 7: Arquitetura cliente-servidor.



Fonte: O autor.

Assim como as demais arquiteturas, esta oferece vantagens e desvantagens em relação a outras. Seus pontos positivos gerais são:

- ter seus recursos centralizados, facilitando sua manutenção, reparo e atualização das informações ou do *software*;
- o armazenamento da maioria dos dados nos servidores, que geralmente possuem controles de segurança muito maiores que os clientes;

- tem seu funcionamento com vários tipos de clientes diferentes com níveis de capacidades variado.

Seus pontos negativos gerais são:

- Os clientes podem requisitar serviços, mas não podem oferecê-los para outros clientes. Desta forma, se o sistema não for bem projetado, pode ocorrer a sobrecarga do servidor, pois quanto mais clientes mais informações irão demandar da rede;
- A maioria das informações estão agrupadas em um único nó no servidor, em que se ocorrer uma perda de conexão deste, todo o sistema ficará fora do ar. Para contornar parcialmente este problema, diversos tratamentos podem ser feitos dependendo da topologia em questão.

2.5 *Back-end* e *Front-end* de um Sistema

Em engenharia de *software*, os termos *front-end* e *back-end* referem-se à separação de interesses entre a camada de apresentação (*front-end*) e a camada de acesso a dados (*back-end*) de um pedaço de *software*, ou a infraestrutura física ou *hardware*. No modelo cliente-servidor, o cliente geralmente é considerado o *front-end* e o servidor geralmente é o *back-end*. Na arquitetura de *software*, pode haver muitas camadas entre o *hardware* e o usuário final. O *front* é uma abstração, simplificando o componente subjacente ao fornecer uma interface amigável, enquanto a parte do *back* geralmente lida com armazenamento de dados e lógica. A distinção destes dois termos é importante, pois o tratamento é feito de maneira bastante distinta em cada uma destas camadas, dada a sua natureza e suas diferentes necessidades.

3. DEFINIÇÃO DO PROBLEMA

A Trensurb, ao longo da sua história, consolidou-se como uma empresa de transporte de passageiros, indutora de desenvolvimento social e econômico, com o seu maior objetivo de, além de diversos outros socioculturais, garantir o transporte seguro, rápido e de qualidade a todos os seus passageiros da região de operação de seus municípios. O Setor de Projetos de Sistemas e Inovação Tecnológica (SEITEC) busca desenvolver e implementar inovações que auxiliem a empresa como um todo, desde a Operação, Manutenção, Administração, Expansão e Comercial. Uma das necessidades que surgiram na empresa foi o desenvolvimento de uma ferramenta capaz de melhor informar e se comunicar com usuários nas estações, tanto em períodos de normalidade quanto de anormalidades. Notou-se que uma das maiores razões de desconforto dos passageiros é a possível desinformação que acontece em momento de atraso ou problemas durante a operação, recaindo à SEITEC o desenvolvimento de tal ferramenta. Também, atualmente no período de pandemia do COVID-19, ferramentas informativas para a população demonstram-se valiosas para fins de saúde pública.

Neste projeto foi desenvolvida uma alternativa passível de ser implementada com o objetivo de garantir a apresentação destas informações. Vislumbrou-se a implementação de Painéis Anunciadores de Informação em todas as 22 estações, a serem implementados nas Bilheterias, Plataformas, Acesso Externo, Aeromóvel, possivelmente para uso interno da empresa, assim como outros locais pertinentes. As informações dispostas nestas telas seriam de cunho informacional, educacional, operacional, de possíveis falhas, relativas à segurança, horário atual e estimativa de chegada de trens, ou outras informações.

Nas estações e trens da Trensurb já existem algumas telas informativas, apresentando propagandas e com uma pequena porcentagem para o espaço de informações da Trensurb, que são locais alugados e de propriedade da TV Minuto. Pelo fato do pequeno espaço de tempo para informações de interesse da Trensurb, e com a demora para a atualização destas (com tempo médio para atualização de 3 dias), estas telas não seriam capazes de cumprir o propósito requerido. O desenvolvimento deste projeto não possui o objetivo de competir espaço com a TV Minuto, pois está voltada a cunho não comercial e sem apresentação de propagandas.

Na Figura 8 e 9 fotos da TV Minuto e do protótipo do Painel Anunciador de Informação localizados na Estação Aeroporto.

Figura 8: Foto da TV Minuto na Estação Aeroporto.

Fonte: O autor

Figura 9: Foto do Painel Anunciador de Informação na Estação Aeroporto.

Fonte: O autor

Desta forma, este projeto possui o objetivo de criar:

- Painéis que apresentam informações no formato de lista de reprodução com atualização em tempo real. Em caso de situações de alerta ou anormalidades na operação, informações prioritárias devem ser dadas aos passageiros das estações, sobrescrevendo os termos que seriam apresentados até a normalidade da operação;
- Implementação de um computador central (que por consequência segue a arquitetura cliente-servidor), a ser localizado na sede da Trensurb, que tem a função de atualizar as informações apresentadas. Tanto o Centro de Controle Operacional (CCO) quanto a Gerência de Comunicação Integrada (GECIN) devem ter acesso a esse servidor;
- Criação de uma interface gráfica que facilite a atualização das informações, arquivos e do banco de dados referente a esta aplicação;
- Desenvolvimento dos serviços de sincronização horária para os painéis;
- Elaboração de um servidor de *backup* nas estações em caso de falha de comunicação entre as telas e o computador central;
- Desenvolvimento de serviço de acesso remoto entre os dispositivos.

Um dos fundamentos usados no desenvolvimento deste projeto foi de fornecer uma alternativa eficiente, de baixo custo, robusta, flexível e expansível. Desta forma, deve-se ter o cuidado para não se obter soluções com processos “engessados”, a fim de garantir a sua eventual expansão. Também existiu a prioridade de utilização de soluções com ferramentas já existentes ou de fácil acesso na empresa.

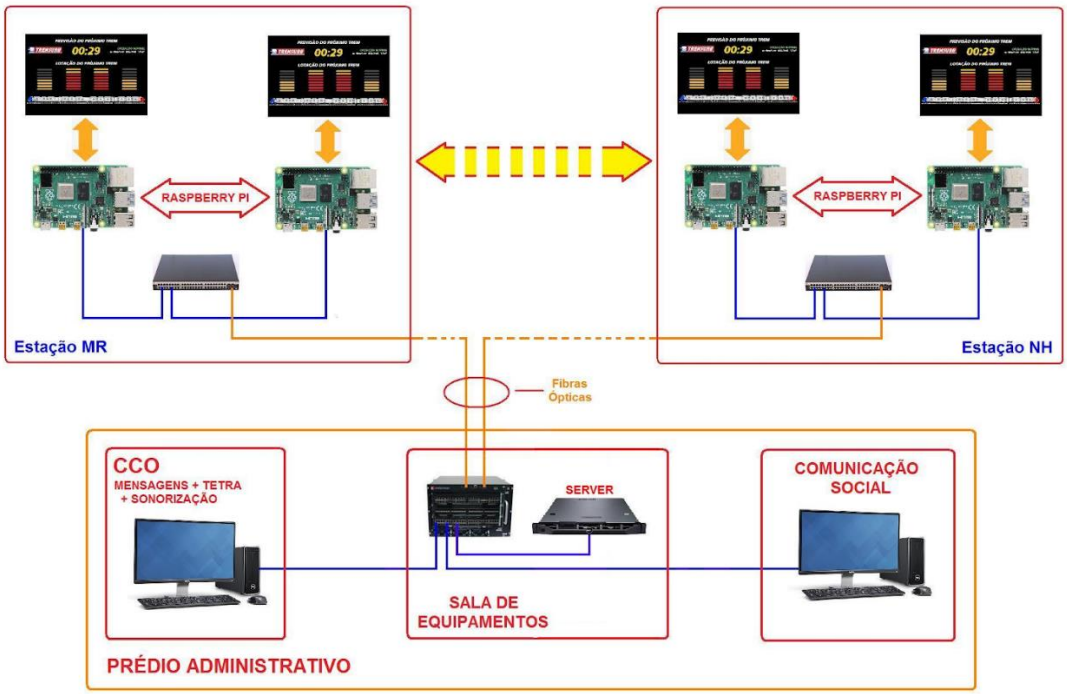
4. ESPECIFICAÇÃO

Neste capítulo são explicadas as especificações do trabalho, através de uma visão geral, análise dos serviços e os módulos em que estes serviços estão sendo executados.

4.1 Visão Geral do Sistema Proposto

O sistema proposto é baseado em uma arquitetura cliente-servidor, conforme visto na seção 2.4. Cada um dos painéis, que são os clientes, realiza requisições para o computador central da aplicação localizada na sede da Trensurb, que tem a função de armazenar as informações, receber as requisições, computar a lógica e executar respostas para os clientes. A escolha desta arquitetura deve-se ao fato de que estamos tratando com um sistema dinâmico, em que as informações a serem apresentadas podem ser modificadas constantemente, de maneira remota, em tempo real e com flexibilidade das informações a serem mostradas. Por este motivo foi adotado o protocolo de comunicação Protocolo de Transferência de Hipertexto (HTTP) para o serviço de apresentação de informações, o mesmo utilizado em sites gerais de Internet. A visão geral do sistema é apresentada na Figura 10, em que é visível na sala de equipamentos do Prédio Administrativo da Trensurb o servidor da operação conectado à rede interna da Trensurb, também conectado à rede interna está um computador do CCO e da Comunicação Social com a função de administrar o sistema, e, conectado à rede interna da Trensurb através de fibra óptica, a rede local da estação metroferroviária, que associa um ou mais dispositivos computacionais que apresentam as informações para as telas das estações metroferroviárias.

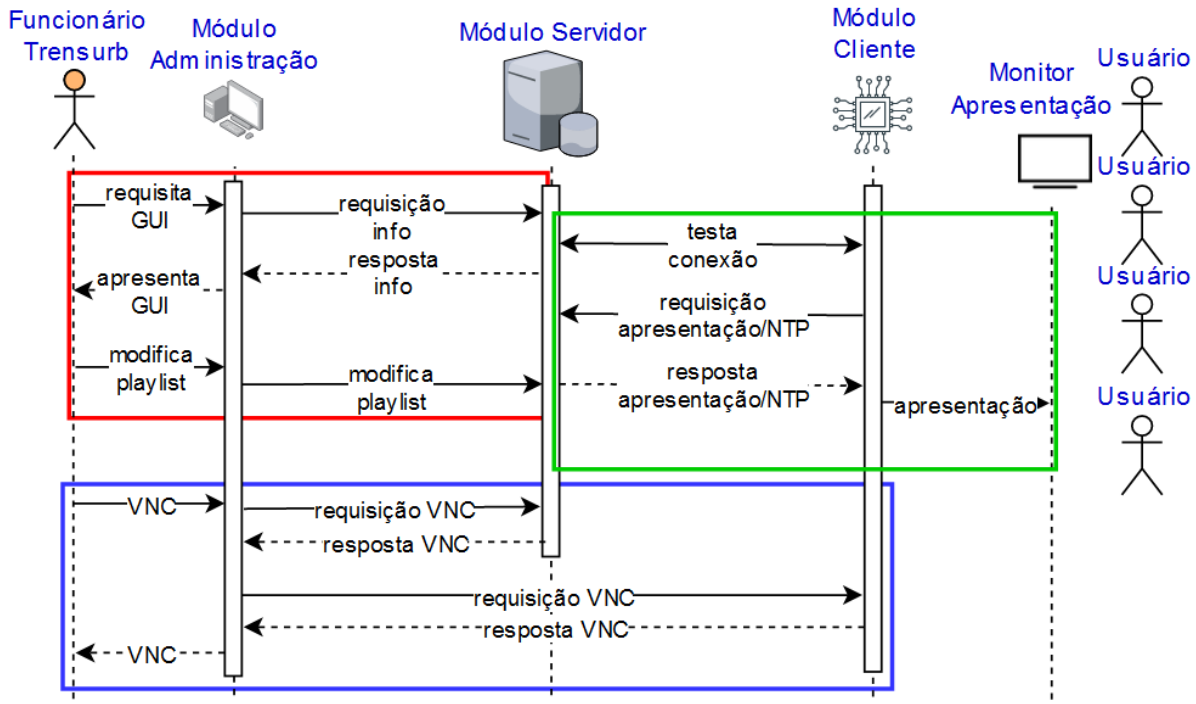
Figura 10: Visão geral do sistema.



Fonte: Ricardo André Riboldi – técnico eletrônico do SEITEC - Trensurb.

Desta forma, o sistema geral é composto por uma camada de *back-end* e duas de *front-end*. Isto se deve ao fato de que temos duas *personas* ou usuários do sistema, uma voltada ao funcionário da Trensurb e outra voltada aos passageiros das estações. A camada de *back-end* é composta pelo controlador e servidor local de *backup* nas estações, pela rede interna da Trensurb, pelo servidor geral, seus arquivos e o banco de dados. A primeira camada de *front-end* é composta pelos painéis apresentadores de informações nas estações, e a segunda camada de *front-end* é composta pela interface gráfica que o funcionário da Trensurb utiliza. Outra maneira de podermos analisar o sistema e seus serviços é através do diagrama de sequência, disposto na Figura 11.

Figura 11: Diagrama de Sequência do sistema.



Fonte: O autor.

O diagrama de sequência facilita a visualização das ações que cada uma das partes funcionais do sistema tem a função de tomar, assim como os serviços necessários. No diagrama a ordem superior até a inferior indica a ordem da leitura do gráfico, mas na prática muitos destes processos podem ocorrer de forma paralela. As partes funcionais do sistema assim como seus serviços está explicado em maior detalhe nas seções 4.3, 4.4 e 4.5

4.2 Definição dos Serviços do Sistema

Feita a definição do problema, podemos ter agora uma visão geral do projeto. Ela pode ser vista como a composição de cinco serviços, sendo estes:

- O serviço de apresentação de informação aos painéis anunciadores de informação, com o desenvolvimento de uma interface gráfica para a atualização das informações, e de painéis para apresentar as informações seguindo a arquitetura cliente-servidor;
- O serviço de teste de conexão, realizado com um PING periódico do servidor para os clientes, que são as telas das estações;
- O serviço de servidor local de *backup*, entrando em vigor em caso de perda de conexão entre o cliente e o servidor, não entrando em funcionamento em caso de normalidade de conexão;
- O serviço de sincronização horária das telas, feita com um servidor Protocolo de Tempo para Redes (NTP);
- O serviço de acesso remoto dos painéis, usado para visualização, atualização de arquivos e de *software* de maneira remota.

Com a visão dos serviços necessários para a composição do projeto, foram analisados esses dentro das camadas do modelo TCP/IP, com as escolhas e uma descrição breve de seus motivos disposto na Tabela 3. Esses cinco serviços serão executados ao longo dos módulos do sistema, explicados dentro das seções 4.3, 4.4 e 4.5.

Tabela 3: Escolhas de projeto dos serviços seguindo o modelo TCP/IP.

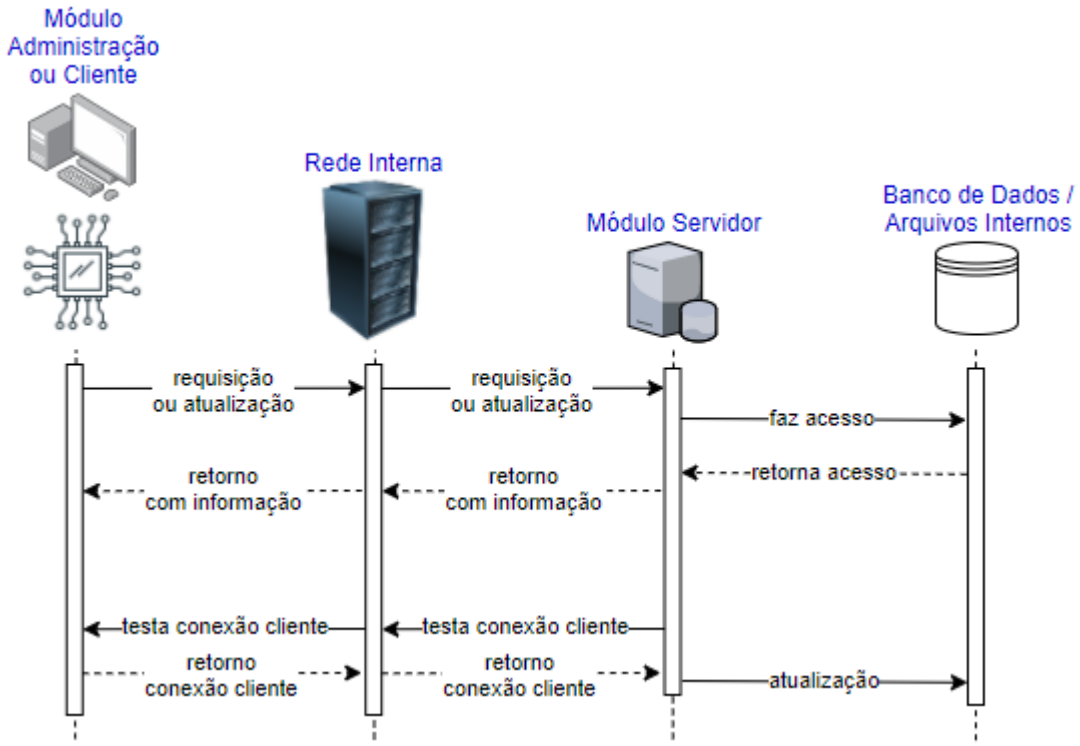
Camadas	Serviço de Apresentação de Informações	Serviço de Teste de Conexão	Serviço de Servidor Local de *Backup*	Serviço Sincronizador de Horário	Serviço de Visualização e Atualização Remota
Aplicação	HTTP (padrão da biblioteca escolhida Flask do Python)	PING (escolha de implementação para o serviço pois demanda pouco da rede)	HTTP (padrão da biblioteca escolhida Flask do Python)	NTP (padrão para sistemas de sincronização de horário)	VNC (padrão para sistemas de acesso remoto)
Transporte	TCP (padrão da biblioteca escolhida Flask do Python, garante confiabilidade do transporte da informação)	Protocolo de Mensagens de Controle da Internet (ICMP) do tipo 8 e 0 (padrão para serviços de PING)	TCP (padrão da biblioteca escolhida Flask do Python, garante confiabilidade do transporte da informação)	UDP (padrão do serviço de sincronização de horário, garante rapidez do transporte da informação)	TCP (padrão do serviço de acesso remoto)
Internet	IPv4 (padrão da biblioteca escolhida Flask do Python)	IPv4 (padrão para o serviço escolhido de PING)	IPv4 (padrão da biblioteca escolhida Flask do Python)	IPv4 (padrão do programa escolhido Meinberg)	IPv4 (padrão do serviço escolhido de acesso remoto)
Host / Rede	Ethernet (arquitetura disponibilizada na Trensurb)	Ethernet (arquitetura disponibilizada na Trensurb)	Serviço Local (servidor e cliente no mesmo *hardware*)	Ethernet (arquitetura disponibilizada na Trensurb)	Ethernet (arquitetura disponibilizada na Trensurb)

Fonte: O Autor.

4.3 O Módulo Servidor

O servidor é uma das três partes funcionais necessárias para o funcionamento do sistema. De acordo com a seção 2.4, ele é responsável pelo armazenamento e atualização das informações, recebimento das solicitações dos clientes e resposta dos serviços requisitados. O servidor está localizado na camada de *back-end* do sistema, funcionando de maneira automatizada em que modificações nele não devem ser feitas diretamente, mas pelo módulo de administração. Os serviços diversos do servidor necessitam estar programados de forma bastante robusta, pois pretende-se conectar muitos clientes ao sistema, com uma estimativa inicial superior a 100. O diagrama de sequência detalhado do servidor está disposto na Figura 12.

Figura 12: Diagrama de Sequência do módulo servidor.



Fonte: O autor.

Os serviços que o servidor é capaz de fornecer estão descritos com o seu passo a passo nas subseções 4.3.1, 4.3.2 e 4.3.3.

4.3.1 Serviço de Apresentação de Informações

O serviço de apresentação de informações é responsável pela apresentação correta das informações nas telas dos clientes nas estações. Ele é composto pelo banco de dados localizado diretamente no servidor, pelo servidor, pelos controladores e monitores dos clientes nas estações, pela rede da Trensurb conectando todos e pelo módulo de administração. A seguir a descrição das funções do servidor para este serviço.

1. O servidor aguarda a requisição de um cliente usando o protocolo HTTP, mantendo uma de suas portas aberta para conexão.
2. Ao receber uma requisição de um cliente para saber qual o próximo termo a ser apresentado, o servidor acessa suas informações internas e o banco de dados para fazer tal decisão. Neste passo o servidor também atualiza no seu histórico que foi feita esta requisição, podendo ser acesso o registro em caso de algum erro.
3. O servidor acessa o vídeo (formatos .mp4 e .ogg), imagem (formato .png e .jpg) ou tela armazenada (página HTML) internamente a ser apresentada na estação e responde ao cliente o conteúdo. No caso de o conteúdo não ser encontrado ou com caso de erro (por exemplo corrompimento do vídeo), o servidor ao invés fornece uma tela alternativa, pulando este termo e não travando o serviço.

4.3.2 Serviço de Teste de Conexão

O serviço de teste de conexão é responsável por informar se a conexão ainda está disponível entre o servidor e cliente. Existe esta necessidade, pois no serviço de apresentação de informações o servidor apenas aguarda solicitações do cliente, ficando incapaz de reconhecer em caso de desconexão. Sua implementação é feita através de dois métodos complementares realizados em paralelo. O primeiro segue através do protocolo PING, usado para testar a conexão com envio de um pequeno pacote de dados, evitando o sobrecarregamento da rede. O segundo método segue através de temporizadores iniciados sempre que um cliente envia uma requisição no serviço de apresentação de informações. O serviço de teste de conexão é composto

pelo servidor, pelos controladores dos clientes nas estações e pela rede da Trensurb conectando todos. A seguir a descrição das funções do servidor para este serviço.

Primeiro Método:

1. O servidor realiza periodicamente um teste de *ping* para cada um dos clientes, testando a conexão com estes.
2. Cada cliente quando receber seu *ping* devolverá um *pong* para o servidor. Desta forma, o servidor ao receber o *pong* do cliente garante que a conexão está disponível e informa o tempo que demorou entre o *ping* e o *pong*. No caso do não recebimento do *pong* considera-se que o cliente em questão está indisponível.
3. Dependendo da resposta do *pong* o servidor atualiza a informação de conexão do cliente para o banco de dados.

Segundo Método:

1. No funcionamento do serviço de apresentação de informações, sempre que o servidor receber uma solicitação de um cliente, será disparado um temporizador dentro do servidor com o tempo do termo a ser apresentado para aquele cliente acrescido de uma folga.
2. Quando o tempo do temporizador acabar, se aquele cliente não realizar outra requisição neste período considera-se que ocorreu algum travamento nele. Isto garante um complemento da informação de testes, mas agora voltado a travamentos no cliente mesmo quando conectado à rede.
3. Em caso de travamento, o servidor atualiza a informação de conexão do cliente para o banco de dados.

4.3.3 Serviço Sincronizador de Horário

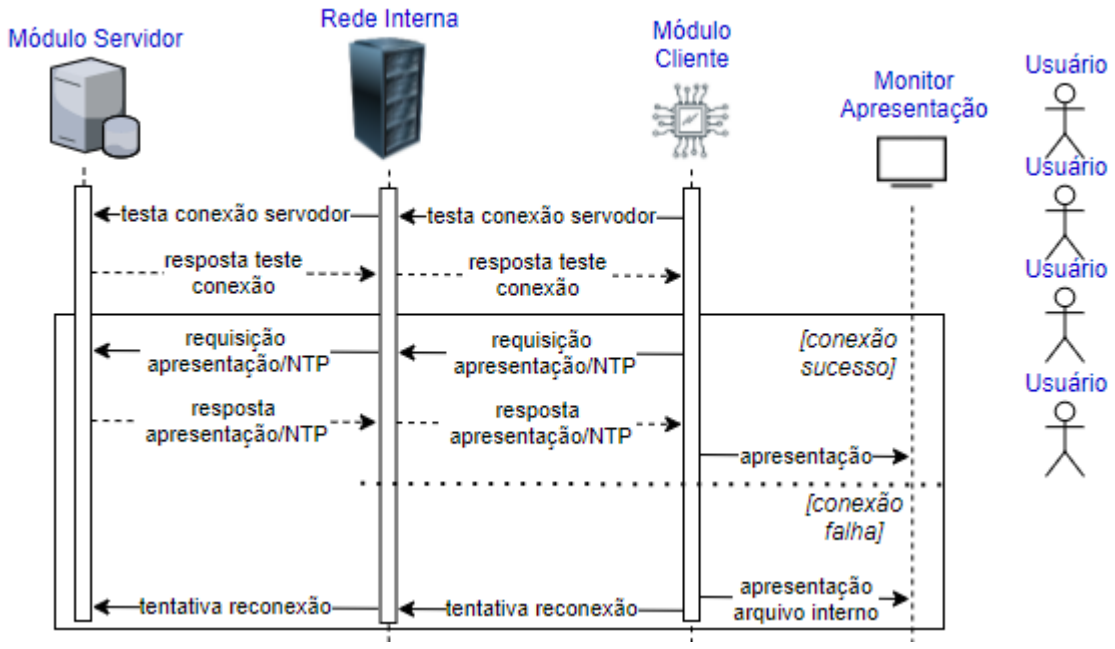
O serviço sincronizador de horário, como o próprio nome diz, tem a função de sincronizar os horários de todos os clientes com o horário do servidor, garantindo a apresentação na ordem correta das informações e de forma simultânea nas telas dos clientes nas estações. Ele é composto pelo servidor, pelos controladores dos clientes nas estações e pela rede da Trensurb conectando todos. A seguir a descrição das funções do servidor para este serviço.

1. O servidor aguarda a requisição de um cliente usando o protocolo NTP, mantendo uma de suas portas aberta para conexão.
2. Ao receber uma requisição de um cliente para sincronizar horário, o servidor acessa seu relógio interno e envia uma resposta com o seu horário.

4.4 Módulo Cliente da Estação

O cliente é outra das três partes funcionais necessárias para o funcionamento do sistema. De acordo com a seção 2.4, ele é responsável por enviar solicitação dos serviços ao servidor, e após o recebimento da resposta deve fazer a apresentação da informação recebida. No caso dos Painéis Anunciadores de Informação, cada cliente é composto por um controlador, disposto na camada de *back-end*, e um monitor apresentando informações aos usuários nas estações, disposto na camada de *front-end* do sistema. Todo o cliente funciona de maneira automatizada e sem necessidade de interferência. O diagrama de sequência detalhado do cliente está disposto na Figura 13.

Figura 13: Diagrama de Sequência do módulo cliente da estação.



Fonte: O autor.

Os serviços que o cliente é capaz de fornecer estão descritos com o seu passo a passo nos subcapítulos 4.4.1, 4.4.2 e 4.4.3.

4.4.1 Serviço de Apresentação de Informações

O serviço de apresentação de informações é responsável pela apresentação correta das informações nas telas dos clientes nas estações. Ele é composto pelo banco de dados localizado diretamente no servidor, pelo servidor, pelos controladores e monitores dos clientes nas estações, pela rede da Trensurb conectando todos e pelo módulo de administração. A seguir a descrição das funções do cliente para este serviço.

1. Ao inicializar o cliente ou acabar o termo sendo apresentado no momento, o controlador do cliente envia uma requisição com protocolo HTTP para a porta correta do servidor.
2. No caso do não recebimento da resposta, o controlador inicializa o serviço de servidor local de *backup*. Em caso de recebimento da resposta com o vídeo, imagem ou tela a ser apresentada e o tempo de apresentação desta, ele passa essa informação para o monitor. Devido ao uso do protocolo HTTP, o programa mais adequado é um navegador geral de Internet funcionando em tela cheia e modo *kiosk*[1].
3. Ao acabar o tempo de apresentação, o controlador realiza uma nova requisição HTTP para o servidor.

4.4.2 Serviço Sincronizador de Horário

O serviço sincronizador de horário, como o próprio nome diz, tem a função de sincronizar os horários de todos os clientes com o horário do servidor, garantindo a apresentação na ordem correta das informações e de forma simultânea nas telas dos clientes nas estações. Ele é composto pelo servidor, pelos controladores e monitores dos clientes nas estações e pela rede da Trensurb conectando todos. A seguir a descrição das funções do cliente para este serviço.

1. O cliente periodicamente realiza uma requisição NTP para atualização do horário para a porta correta no servidor.
2. Ao receber a resposta do horário atual do servidor, o controlador atualiza seu relógio interno.

[1] Modo que trava a tela de apresentação, utilizado em casos de aplicações de janela única.

4.4.3 Serviço de Servidor Local de *Backup*

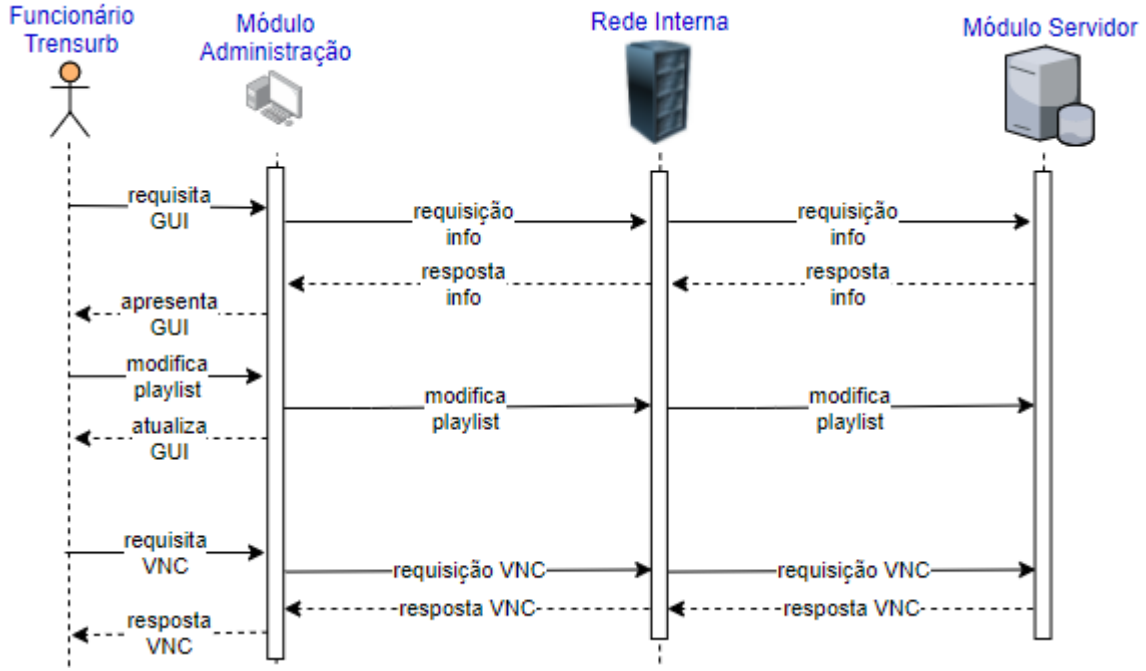
O serviço de servidor local de *backup* é responsável por garantir a continuidade de informações nas telas das estações mesmo que em caso de falhas de conexão entre o cliente e o servidor. O serviço normalmente não entra em funcionamento, somente usado em caso de adversidades. Os testes usados para o serviço entrar em funcionamento são realizados durante o serviço de apresentação de informações descrito em 4.4.1, entrando em funcionamento quando notar-se uma falha de comunicação. Se isto ocorrer, o controlador deve conseguir prover informações para a tela de maneira autônoma até o restabelecimento de conexão. O serviço é composto pelos controladores e monitores dos clientes nas estações. A seguir a descrição das funções para este serviço.

1. O controlador na estação ao reconhecer que perdeu conexão com o servidor inicia o serviço de servidor local de *backup* e acessa a sua memória interna.
2. Uma lógica interna realiza a computação do próximo termo escolhido a ser apresentado dentro dos que ele já possui armazenado.
3. Periodicamente o controlador realiza novas tentativas de reconexão com o servidor. Em caso de sucesso de reconexão, o serviço de apresentação de informações será retomado, mas em caso de falha o serviço de servidor local de *backup* continua atuando.

4.5 O Módulo de Administração

O módulo de administração é a última das três partes funcionais necessárias para o funcionamento do sistema. Ele é responsável, como o próprio nome diz, por administrar, monitorar e visualizar o estado de funcionamento do sistema. Ele tecnicamente é um cliente do servidor, mas diferente dos controladores das estações ele consegue coletar e modificar informações diretamente no banco de dados e nos arquivos do servidor. Como no seu funcionamento ocorrem diversas interações humanas com os funcionários da Trensurb para visualizar o sistema, foi desenvolvido uma Interface Gráfica do Usuário (GUI) que facilita o uso do módulo. Desta forma, este módulo é composto por uma GUI na camada de *front-end*, que obtém suas informações do módulo servidor no *back-end* do sistema. O diagrama de sequência detalhado do cliente está disposto na Figura 14.

Figura 14: Diagrama de Sequência do módulo de administração.



Fonte: O autor.

Os serviços que o módulo de administração é capaz de fornecer estão descritos com o seu passo a passo nas subseções 4.5.1, 4.5.2.

4.5.1 Serviço de Apresentação de Informações

O serviço de apresentação de informações é responsável pela apresentação correta das informações nas telas dos clientes nas estações. Ele é composto pelo banco de dados localizado diretamente no servidor, pelo servidor, pelos controladores e monitores dos clientes nas estações, pela rede da Trensurb conectando todos e pelo módulo de administração. A seguir a descrição das funções do módulo de administração para este serviço.

1. O computador administrador realiza requisições periódicas das informações dispostas no banco de dados e dos arquivos internos no servidor, obtendo informações sobre o estado de conexão com cada um dos clientes, os nomes e descrições dos arquivos da lista de reprodução sendo apresentado no momento, além de outras informações.
2. O funcionário da Trensurb ao acessar o cliente administrador tem acesso a uma GUI que apresenta todas as informações pertinentes ao sistema de maneira clara, realiza mudanças nas listas de reprodução e insere novos vídeos, imagens ou telas no sistema.
3. As mudanças feitas pela GUI modificam os arquivos e o banco de dados do servidor.

Por questão de facilidade de modificação das informações em múltiplas telas, utilizou-se uma lista de reprodução geral. Primeiro, o funcionário da Trensurb escolhe o arquivo a alocar na lista de reprodução geral, escolhendo o arquivo, o tempo de reprodução e uma descrição do arquivo. Este arquivo após é copiado diretamente de maneira automatizada para o módulo servidor. Desta forma, quando o funcionário escolher modificar a lista de reprodução de um dos clientes, ele deve escolher qual termo da lista de reprodução geral deseja introduzir ou qual termo da lista de reprodução do local escolhida deseja ser retirado ou modificado. Isto garante rapidez para realizar mudanças de programação.

4.5.2 Serviço de Visualização e Atualização Remota

O serviço de visualização e atualização remota possui a função conseguir ter acesso ao computador ou controlador de forma remota, com acesso gráfico a sua tela ou aos seus arquivos. Para isto foi utilizado o protocolo Computação em Rede Virtual (VNC), largamente utilizado para este tipo de serviço. Este serviço facilita a visualização de eventuais erros nas telas das estações, execução de testes, depuração, atualização do sistema ou de outros programas. Este serviço é bastante comum e encontra-se uma variedade de programas já desenvolvidos que executam essa função. Ele é composto pelo servidor, pelos controladores dos clientes nas estações, pela rede da Trensurb conectando todos e pelo módulo de administração. A seguir a descrição das funções do módulo de administração para este serviço.

1. O computador do módulo de administração envia uma requisição de conexão remota através do IP e senha correspondente ao computador do local desejado.
2. Se a requisição for respondida com sucesso o computador do módulo de administração recebe acesso completo ao computador do local, com a visualização das informações sendo apresentadas no momento, dos arquivos internos, e consegue executar modificações diretas neste computador.

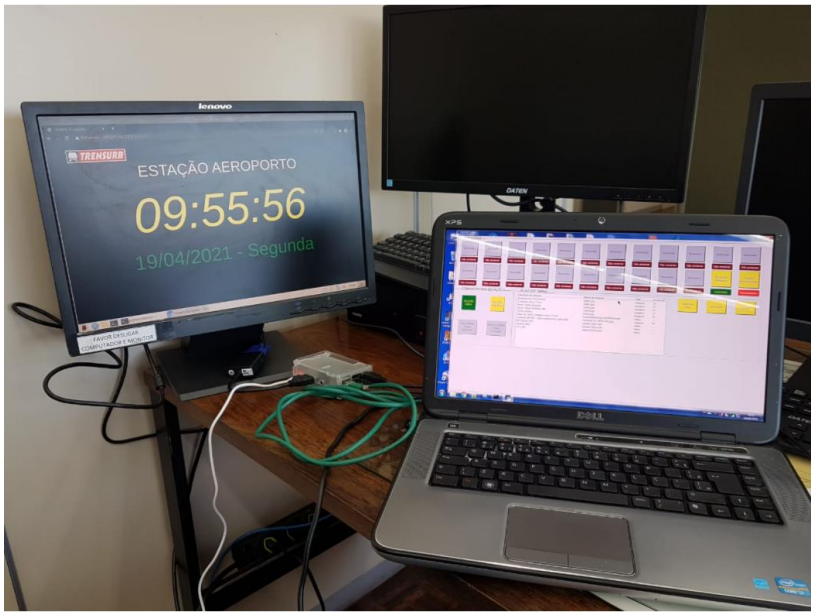
5. IMPLEMENTAÇÃO E RESULTADOS

Neste projeto, estas escolhas foram baseadas no enfoque de obter um sistema eficaz e de baixo custo, utilizando ferramentas já dispostas, de fácil acessibilidade para a Trensurb ou de baixo custo. Também foi avaliado fatores como robustez, expansibilidade e flexibilidade. Após algumas iterações do programa, conseguimos obter o alinhamento de como o programa deve ser implementado e chegando a sua versão atual. Neste projeto, existiram contribuições dos membros da SEITEC, e de maior ênfase do David Samuel Levenfus e Ricardo André Riboldi, em que auxiliaram em questões de alinhamentos entre setores, métodos de implementação e ajudas gerais. Desta forma, o projeto pode ser desenvolvido pelo autor executada exatamente da forma desejada pela empresa.

Todos os módulos do sistema estão conectados através de um bastidor de Rede Local Virtual (VLAN), que garante conectividade entre dispositivos e performance. Esta VLAN interliga a Sala de Equipamentos da sede da Trensurb à Sala de Equipamentos de todas as Estações através de fibra óptica. Como os dispositivos conectados estão isolados da *internet*, a VLAN garante um nível de segurança para invasões externas à Trensurb, que necessitam invadir a rede como um todo da empresa inicialmente. Desta forma, o desenvolvimento de protocolos de segurança não foi a prioridade inicial no projeto.

O projeto até o momento somente foi implementado no saguão de entrada do Prédio Administrativo e na Bilheteria do Aeroporto, mas existem pretensões de expansão dos seus locais e funcionalidades. Também, parte do serviço de apresentação de informações ainda não foi desenvolvido, como a referente a interrupções que o CCO deseja fazer nas telas das estações em caso de alerta. Na Figura 15 está apresentada a bancada de testes das implementações, em que o notebook está simulando o Módulo Servidor e o Módulo de Administração, e conectado a ele por um cabo de rede está um Raspberry Pi, simulando o Módulo Cliente da Estação apresentando as informações em um monitor.

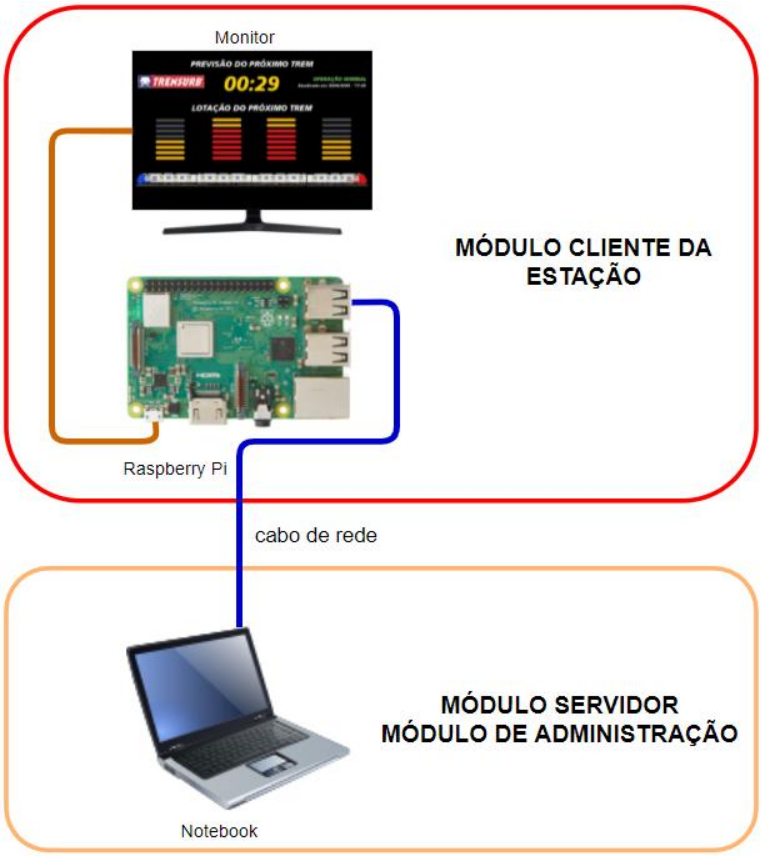
Figura 15: Bancada de testes das implementações.



Fonte: O autor.

Na Figura 16 está apresentado um diagrama destas conexões.

Figura 16: Conexões da bancada de testes das implementações.



Fonte: O autor.

As soluções de software utilizadas foram:

- Para a camada de *back-end* foi escolhido realizar toda ela em Python. O Python é uma das linguagens de programação mais utilizadas no mundo, caracterizada por ser de alto nível, orientada a objetos, interpretada e de uso geral. Sua filosofia enfatiza uma fácil leitura e implementação. O principal motivo é, pois, esta linguagem possui bastante suporte na Internet devido ao uso generalizado, com diversos artigos e documentos para pesquisas, facilitando consideravelmente a implementação. Outro motivo é que já existem outros funcionários da Trensurb com domínio da linguagem em caso de necessidade de auxílio.
- Para a primeira camada de *front-end* dos painéis anunciadores foi escolhido utilizar o protocolo HTTP. O programa mais adequado que utiliza este protocolo e consegue apresentar informações variadas de forma flexível é um navegador geral de Internet. Para a visualização de qualquer informação da maneira desejada, as páginas são desenvolvidas utilizando códigos Linguagem de Marcação de Hipertexto (HTML), Folhas de Estilo em Cascata (CSS) e JavaScript armazenados no servidor. Este modelo destes três códigos é padrão de páginas *web*, em que o HTML tem a função de esqueleto da página, o CSS a estilização e a programação em JavaScript possibilitam a implementação de páginas dinâmicas. Desta forma é garantido um sistema expansível, em que se desejado uma nova funcionalidade basta desenvolver uma nova página *web* seguindo os mesmos padrões já encontrados em outras aplicações gerais. Foi decidido que a apresentação de vídeos deve ser apresentada no formato *streaming* na primeira fase do protótipo na própria página web. Quando um termo é finalizado, o JavaScript da página HTML automaticamente apresenta o próximo termo. Transferência de vídeo, que é outro formato viável e/ou complementar ao *streaming*, pode ser implementável futuramente, mas cuidados devem ser feitos para não ocasionar sobrecarga da rede ao transferir arquivos para muitos clientes simultaneamente.
- Para a segunda camada de *front-end* do Módulo de Administração, foi escolhido utilizar o Python novamente. O Python também possui auxílio no desenvolvimento de GUIs, compartilhando os mesmos benefícios do *back-end* com facilidade de leitura e implementação.

- Foi decidido utilizar um banco de dados alocado em paralelo a arquivos do servidor. Os bancos de dados relacionais, que é o caso deste projeto, são recomendados para dados altamente estruturados, que necessitem de integridade, consistência e segurança através de processos internos automatizados, e reduzem o custo computacional necessário para atualizar, armazenar e retirar os dados. Como as mesmas informações estão sendo compartilhadas e modificadas ao mesmo tempo por diferentes partes do sistema, garantir a integridade dos dados é um dos motivos principais da utilização desta ferramenta.

A conexão entre todas as partes funcionais do sistema através de uma VLAN, utilizada a da sonorização. Ela interconecta todas as partes funcionais do sistema, ou se em um mesmo local com cabo Ethernet (como todos os computadores deste sistema no Prédio Administrativo), ou se em locais diferentes através de fibra ótica (como conexão entre Prédio Administrativo – Estação Aeroporto). Na Figura 17 uma foto da conexão do bastidor.

Figura 17: Conexões da VLAN.

Fonte: O autor.

5.1 O Módulo Servidor

Para a implementação do servidor, descrito na seção 4.3, utilizamos um computador *desktop*, do tipo padrão de escritório com o sistema operacional Windows. Apesar de ele não ser um dispositivo ideal para atuar como servidor dedicado ou de menor custo, neste caso era o dispositivo que se encontrava disponível. Este computador atualmente encontra-se instalado na GECIN, mas o objetivo inicial era de colocá-lo na Sala de Equipamentos do Prédio Administrativo da sede da Trensurb, e existem pretensões futuras para isto. Este servidor está conectado através de um cabo Ethernet ao bastidor da VLAN da Sala de Equipamentos do Prédio Administrativo.

No servidor foram instalados diversos programas dependendo da necessidade do serviço. Para aplicações gerais o banco de dados escolhido foi o MySQL. O MySQL é uma base relacional *open-source* e estruturada SQL. Ele é um dos bancos de dados mais utilizados no mundo, com características de alta performance, seguro, modular e com um dos seus enfoques para aplicações *web*, que é o caso deste projeto. Ele também é bastante utilizado em aplicações Flask, que tem explicação na subseção 5.1.1. Neste banco de dados foram inseridas informações das estações com nome, termo da playlist sendo apresentado, resultado do teste de conexão, variáveis auxiliares e um registro da atividade de eventos. Na Tabela 4 e 5 as tabelas do banco de dados.

Tabela 4: Tabela “Playlists” do banco de dados.

id (PRIMARY KEY Integer)	Chave primária usado para identificação.
estacao (FOREIGN KEY String)	Locais sendo apresentados.
termo (Integer)	Termo da Playlist sendo apresentado no local no momento.
conexao (Integer)	Armazena Status de conexão com o local.
aux_conexao (Integer)	Variável auxiliar usado nos testes de conexão com os locais.

Tabela 5: Tabela “Playlist_especifica” do banco de dados.

estacao (PRIMARY KEY String)	Chave usada para identificar local de apresentação.
termo (Integer)	Termos da Playlist sendo apresentados no local.
nome (String)	Nome do arquivo da Playlist Específica
tipo (String)	Tipo de arquivo da Playlist Específica.
tempo (Integer)	Tempo de apresentação do arquivo.
descrição (String)	Descrição para identificação do arquivo pelo usuário. Não usado para lógica do sistema.

5.1.1 Serviço de Apresentação de Informações

Como descrito em 5.1, procurou-se por um *framework* em Python capaz de atuar como servidor HTTP. Para este serviço foi escolhido utilizar a biblioteca Flask. Outras bibliotecas poderiam ser usadas no seu lugar (tal como os testes iniciais foram feitos utilizando Django ao invés de Flask), mas o Flask demonstra seu valor por implementações simples, flexíveis, escalonáveis e modulares. Ele não tem camada de abstração de banco de dados, validação de formulário, ou quaisquer outros componentes onde bibliotecas pré-existentes de terceiros fornecem funções. No entanto, o Flask oferece suporte a extensões que podem adicionar recursos do aplicativo como se eles foram implementados no próprio Flask. Existem extensões para mapeadores relacionais de objetos, validação de formulário, tratamento de *upload*, várias tecnologias de autenticação aberta e várias ferramentas comuns relacionadas ao *framework*. Desta forma ele oferece ao programador um controle maior e visualização do que está sendo feito e não ferramentas “engessadas”, importante pois este é um projeto bastante específico. Também esta biblioteca oferece um sistema já desenvolvido com *threads*, garantindo paralelismo e rapidez quando processado em um computador. Exemplos de aplicações que usam essa biblioteca incluem Pinterest, Linkedin e Netflix.

Neste programa, ou aplicação como chamado comumente, em Flask é desenvolvido uma *blueprint* para cada estação, que são módulos da aplicação. Cada *blueprint* contém um código em Python que contém os IPs dos controladores daquela estação (e impedindo acesso em caso de IP externo), uma relação de Localizadores

Uniformes de Recursos (URLs) aceitas naquela estação e a relação direta entre arquivos e banco de dados para as páginas de URL. Desta forma se mais um cliente de estação deseja ser adicionado, basta incrementar o IP deste na *blueprint* já desenvolvida ou criar uma nova *blueprint*. O sistema Flask foi inicializado na porta padrão 5000.

A fim de facilitar a interligação entre o Flask e o Mysql e modificações no banco de dados, foi utilizada a biblioteca Flask-SQLAlchemy, que dá suporte ao SQLAlchemy na aplicação Flask. Ela tem o objetivo de simplificar a execução de tarefas simples como pesquisas, inserções e modificações no banco de dados através de ferramentas já desenvolvidas. Outra vantagem é que ela tem suporte diretamente na sua documentação com MySQL.

A seguir um resumo das bibliotecas externas do Python sendo utilizadas para este serviço:

- Flask: provê o servidor HTTP.
- Mysql-connector: conecta código em Python com banco de dados MySQL.
- Flask-SQLAlchemy: interliga Flask ao MySQL e facilita modificações no banco de dados.

5.1.2 Serviço Teste de Conexão

O serviço do teste de conexão foi implementado de duas formas complementares. A primeira utiliza o protocolo PING, um método simples com aplicações usuais de teste de conexão, não usado para troca de informações. Desta forma, usando-se deste método com pouca transferência de dados, evita-se um sobrecarregamento da rede. Este serviço implementado em Python opera com um timer periódico que envia as solicitações para cada um dos clientes e aguarda a resposta *pong* destes. Após é atualizado essa informação no banco de dados MySQL. A segunda forma opera também com *timers* periódicos iniciados quando um cliente pede uma requisição para o servidor Flask, em que se este cliente ao não realizar outra requisição depois que o timer acaba, considera-se que ocorreu algum travamento no cliente.

A seguir um resumo das bibliotecas externas do Python sendo utilizadas para este serviço:

- Moviepy: realiza a leitura do tempo de vídeos de maneira automatizada.

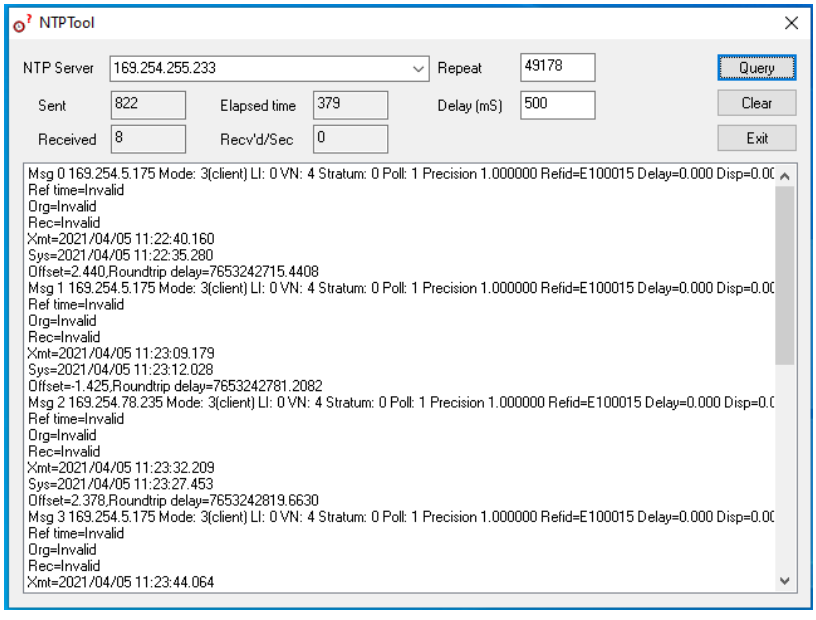
- Subprocess: realiza a requisição *ping* e a leitura da resposta *pong*.
- Threading: provê um timer em um *thread* proporcionando paralelismo.
- Mysql-connector: conecta código em Python com banco de dados MySQL.

5.1.3 Serviço Sincronizador de Horário

Para o serviço sincronizador de horário foi aproveitado o sistema operacional Windows do servidor. Nele é encontrado ferramentas internas que já conseguem prover esse serviço NTP, bastando ativá-las. Uma alternativa para isto seria modificar o Registro de Variáveis do Windows (Regedit) e o serviço do Windows 'Agente de Tempo' seguindo o mesmo modelo de diversos guias encontrados, mas por questão de facilidade foi instalado o *software* Meinberg, que já realiza estas alterações oferecendo uma opção já pronta. Após a instalação, o programa foi liberado na porta padrão 123 para protocolo NTP.

Uma ferramenta auxiliar usada para testar o servidor NTP é o Galleon NTP Server Tool, em que é possível visualizar o envio do horário do servidor periodicamente na Figura 18. Conseguimos visualizar através de seu teste que pedidos de requisição NTP estão sendo respondidos pelo servidor.

Figura 18: Teste de conexão NTP através do Galleon NTP Server Tool.



Fonte: O autor.

5.2 Módulo Cliente da Estação

O módulo do cliente foi implementado utilizando um *hardware* controlador e um monitor para apresentação de informações. No caso deste projeto foi utilizado um Raspberry Pi 3 B para o controlador e monitores comuns com entrada de Interface Multimídia de Alta Definição (HDMI) para apresentar as informações. O principal motivo do uso do Raspberry Pi é pela sua disponibilidade na empresa e porque ele tem a capacidade de prover todas as necessidades para atuar como cliente deste projeto.

O Raspberry Pi é um computador de baixo custo com tamanho de um cartão de crédito desenvolvido pela Raspberry Pi Foundation. Ele é composto por diversos circuitos integrados, sendo os principais o SoC, atuando como uma placa mãe e processador em computadores convencionais; o circuito de Wi-fi e Bluetooth para conexão sem cabo; circuito regulador de tensão de Barramento Serial Universal (USB) e de cabo Ethernet; no caso do Raspberry Pi 4 uma placa dedicada de memória RAM. Seu armazenamento é realizado através de um cartão microSD externo. As especificações do Raspberry Pi 3, usado neste projeto, estão na Tabela 6.

Tabela 6: Especificações do Raspberry Pi 3 B.

SoC (Sistema em um *Chip*)	Broadcom BCM2837
CPU (Unidade de Processamento Central)	4x ARM Cortex-A53, 1.2GHz
GPU (Unidade de Processamento Gráfico)	Broadcom Video Core IV
RAM (Memória de Acesso Aleatório)	1GB LPDDR2 (900MHz)
Rede	10/100 Ethernet, 2.4GHz 802.11n wireless
Bluetooth	Bluetooth 4.1 Classic, Bluetooth Low Energy
Armazenamento	microSD
GPIO (Entradas gerais de entrada/saída)	conector de 40 pinos
Portas	HDMI, entrada analógica de áudio e vídeo de 3,5 mm, 4 × USB 2.0, Ethernet, interface serial de câmera (CSI), interface serial de exibição (DSI)

Fonte: (FOUNDATION, [s.d.])

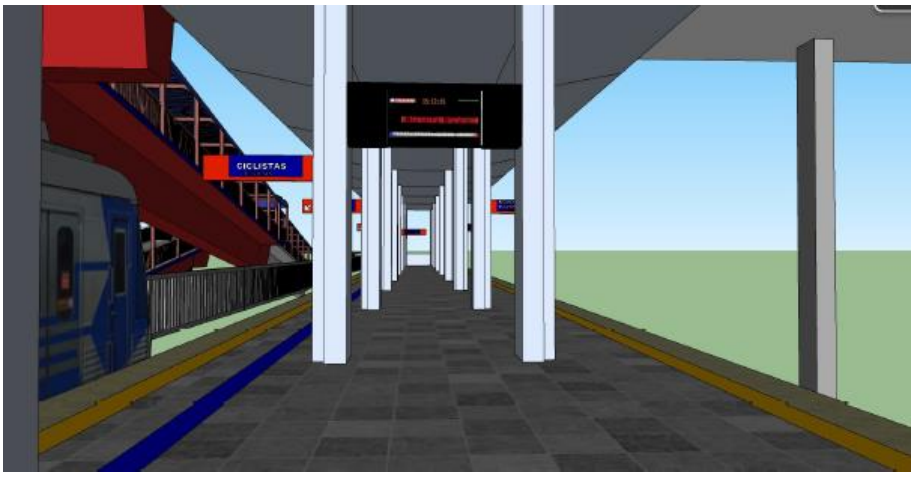
O Raspberry Pi foi posto um *case* protetor de acrílico transparente em seu entorno e dissipadores de calor nos seus *chips*. Foi instalado o sistema operacional oficial, o Raspberry Pi OS, que é um sistema baseado em Debian do Linux. Ele, por padrão, já vem com programas pré-instalados como o Python e Chromium que são utilizados neste projeto. Uma foto do Raspberry Pi 3 da Trensurb a ser utilizado está na Figura 19.

Figura 19: Foto do Raspberry Pi 3 da aplicação com dissipador e *case*.

Fonte: O autor.

Este Raspberry Pi foi conectado através de um cabo Ethernet à VLAN da sala de equipamentos do local em que foi instalado. Na porta HDMI deste Raspberry foi conectado um monitor que apresenta as informações. Simulações foram utilizadas para a decisão dos locais mais adequados. Na Figura 20 uma destas simulações.

Figura 20: Simulação de uma das telas.



Fonte: SEMOB - Trensurb.

Após a simulação, foi implementado as primeiras telas no saguão do Prédio Administrativo e na bilheteria da Estação Aeroporto. No caso do saguão, o Raspberry Pi foi fixado atrás da TV, em que se considera um local seguro, e na bilheteria instalado internamente a esta e somente os cabos e o monitor fixado do lado externo. Na Figura 21 e 22 imagens da apresentação.

Figura 21: Foto dos Painéis Anunciadores funcionando no saguão do Prédio Administrativo.

Fonte: O autor.

Figura 22: Foto dos Painéis Anunciadores funcionando na bilheteria da Estação Aeroporto.

Fonte: O autor.

5.2.1 Serviço de Apresentação de Informações

O cliente do serviço de Apresentação de Informações é quem realiza as requisições HTTP para o servidor, como explicado na seção 4.4 e início do capítulo5, em que deve operar com um navegador de Internet. Este navegador tem a função de realizar uma conexão ao URL do servidor, da mesma maneira que é usado para conexão a *sites* de Internet. A apresentação deve ser feita em modo *kiosk* e em tela cheia, de forma que visualmente não aparenta ser um navegador de Internet funcionando.

O navegador escolhido foi o Chromium, que é o navegador oficial do Raspberry Pi. Ele demonstrou ser o navegador leve, eficiente e com melhores níveis de resposta dentre os sete navegadores testados. Seu único revés é que foram encontrados dois *bugs* específicos da versão oficial do Chromium para o Raspberry Pi que quase impossibilitaram a sua utilização, mas através de pesquisas foi encontrado alternativas para circundar estes *bugs*.

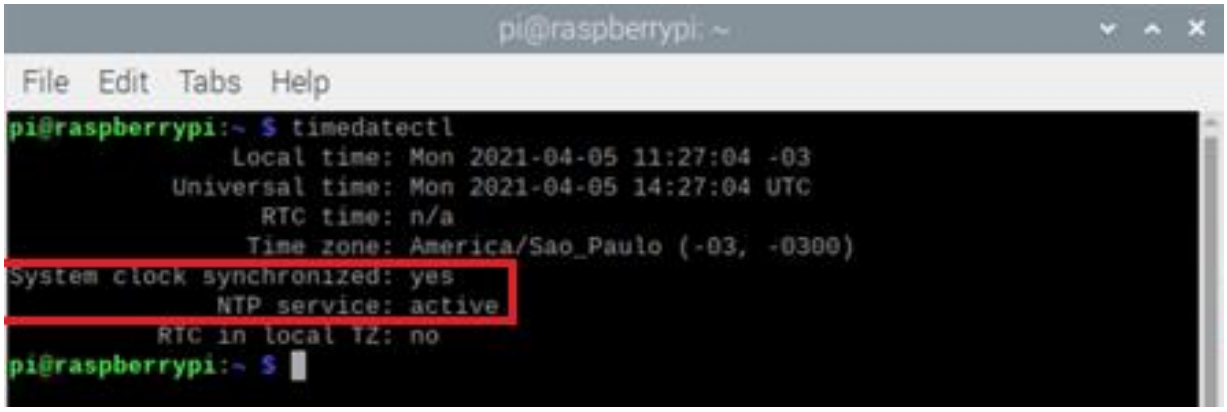
O primeiro *bug* é relativo ao alocamento de memória RAM excessivo pelo dispositivo a este programa, ocasionando travamentos intermitentes com intervalos variados podendo ocorrer entre horas ou semanas da utilização. Este erro não foi encontrado nos demais navegadores. Para contornar este *bug*, implementou-se um pequeno script que força a atualização do *browser* sempre que ocorre o travamento. A detecção do travamento é feita através da comparação da imagem reproduzida no navegador com a imagem esperada em caso de falha.

O segundo *bug* é relativo a *pop-ups* de atualização mesmo quando o Chromium já estava atualizado, ocasionando mensagens em cada inicialização perturbando a apresentação. Acredita-se que o motivo desta mensagem é porque o banco de dados do Raspberry Pi não é compartilhado com o do Chromium, em que ao consultar sua versão ele julga sempre estar desatualizado acreditando ser um *desktop* convencional. Para solucionarmos este erro utilizamos a *flag* ““--check-for-updates-interval=31536000”” na inicialização do *browser* que retarda o teste do pedido de atualização para 1000 anos depois do Raspberry Pi ser ligado. Após um milênio, acreditamos que este bug já terá sido resolvido pelo Raspberry Pi Foundation.

5.2.2 Serviço Sincronizador de Horário

No serviço sincronizador de horário foi utilizado a ferramenta pré-instalada do Raspberry Pi Timedatectl. Ela atua como um cliente NTP, que periodicamente faz requisições de atualização horária para o servidor escolhido, que neste caso é o servidor NTP descrito na subseção 5.1.3. A ferramenta também provém através do terminal um comando que comprova a conexão com o servidor NTP, mostrado na Figura 23. Nela, é possível ver que o serviço NTP (NTP service) e a sincronização do relógio do sistema (System clock synchronized) estão ativos.

Figura 23: Imagem de teste de conexão NTP através do Timedatectl.



Fonte: O autor.

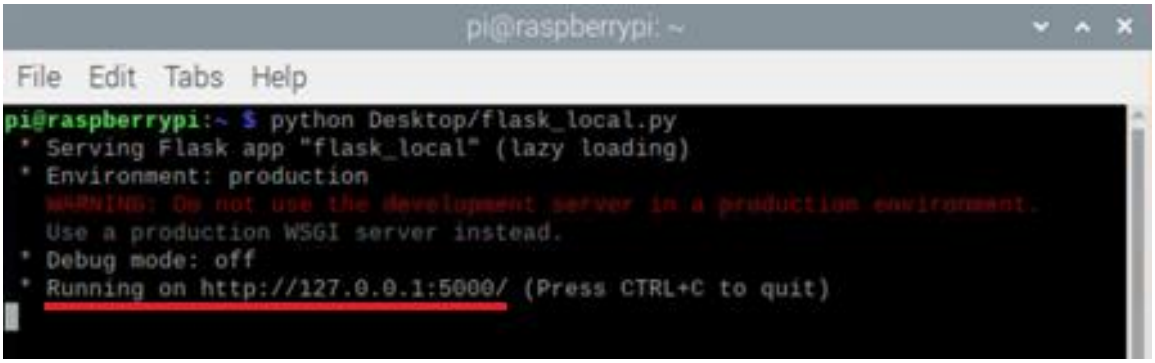
5.2.3 Serviço de Servidor Local de *Backup*

No uso do serviço de servidor local de *backup*, foi desenvolvido uma aplicação em Python que é capaz de prover todos os benefícios da linguagem descrita no início do capítulo 5. Este servidor local de *backup* HTTP foi implementado usando Flask de forma bastante similar ao servidor, descrito na subseção 5.1.1, com a diferença de que este é acessível somente através do IP local 127.0.0.1. Neste servidor local, apenas é apresentado informações de interesse já gravadas no cartão microSD do Raspberry Pi, garantindo mesmo que de uma maneira limitada o fluxo contínuo de informações à tela. Quando a conexão for restabelecida novas informações podem ser enviadas ao Raspberry Pi.

O uso do serviço de servidor local de *backup* fica bastante interligado com o serviço de apresentação de informações do cliente, descrito na subseção 5.2.1. Com

auxílio de jQuery[2] nas páginas apresentadas, em caso de não recebimento da resposta quando requisitado o URL do servidor geral da aplicação, o navegador ao invés de apresentar a tela de erro é redirecionada à página do servidor local de *backup*. Enquanto o servidor local de *backup* apresenta as informações, são feitos testes utilizando o jQuery que tentam se reconectar ao servidor, que em caso de sucesso retomam o serviço de apresentação de informações, retornando ao URL ao do servidor geral. Na Figura 24 é mostrado a inicialização do servidor local de *backup*, com o IP de *localhost*.

Figura 24: Servidor local de *backup* sendo inicializado.



Fonte: O autor.

[2] biblioteca do JavaScript, desenvolvida para simplificar passagem e manipulações no HTML, bem como tratamento de eventos entre outras funcionalidades.

5.3 O Módulo de Administração

Para a implementação do módulo de administração foi utilizado um computador *desktop* igual ao servidor descrito na seção 5.1, do tipo de escritório com o sistema operacional Windows. A intenção inicial do projeto era que diversos computadores da Intranet da Trensurb conseguissem ter acesso ao módulo de administração, mas este processo está ainda em alinhamento interno de setores, porém com pretensão futura de funcionamento desta forma. Este computador está conectado via cabo Ethernet ao bastidor da VLAN da Sala de Equipamentos do Prédio Administrativo.

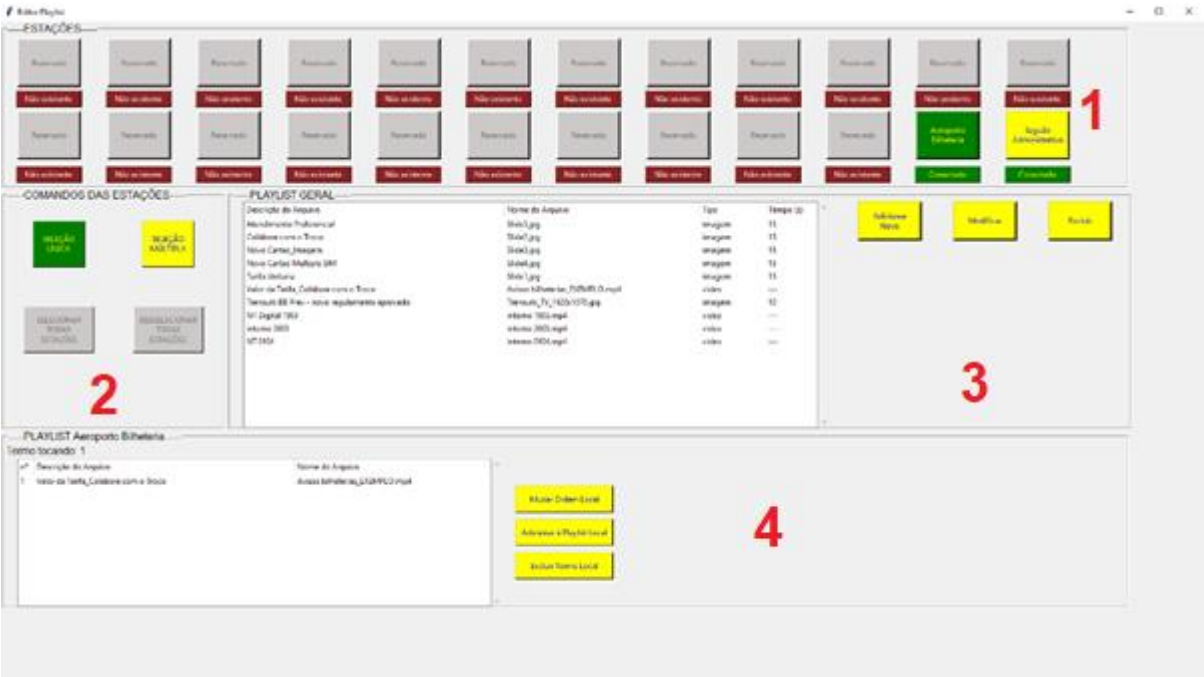
Neste computador foi instalado o Python e diversas bibliotecas para realizar os serviços neste módulo. Este computador, apesar de tecnicamente ser um cliente do servidor, ele possui permissão de administração, com acesso ao banco de dados MySQL e arquivos locais do módulo servidor.

5.3.1 Serviço de Apresentação de Informações

No serviço de apresentação de informações foi necessário desenvolver uma GUI e uma conexão diretamente com o módulo do servidor para obter e enviar informações atualizadas. Neste serviço foi escolhido o Python como linguagem de programação para implementação.

Para desenvolver a GUI foi utilizada a biblioteca Tkinter que fornece suporte a esta implementação. Apesar de ele ser conhecido por não ser esteticamente bonito, o Tkinter oferece todas as funcionalidades de uma biblioteca TK, provendo portabilidade e acessibilidade ao estar incluído diretamente na distribuição do Python. Uma imagem da GUI está na Figura 25. No momento, a interface gráfica provém alterações na apresentação de imagens e vídeos com a apresentação das páginas HTML automatizada. A funcionalidade de escolher as páginas HTML é uma funcionalidade que pretende ser adicionada futuramente.

Figura 25: Imagem da GUI desenvolvida.



Fonte: O autor.

Usando os nomes da documentação dos objetos da biblioteca Flask, na interface é possível visualizar os diferentes blocos de LabelFrame separados em um sistema de *grid* que em conjunto compõem a interface. Os LabelFrame's encontrados são: em **1** "estações"; em **2** "comando de estações"; em **3** "playlist geral"; e em **4** "playlist específica". Cada um desses LabelFrame's é subdividido em *grids* internos com diversos objetos, compostos por Button's, Label's, Treeview's, Combobox's, Scrollbar's e Entry's. Para a criação da GUI foi essencial o conhecimento de orientação a objetos, pois foram utilizados dois dos seus conceitos fundamentais como a herança e a abstração. Foi criada uma convenção de código de cores para facilitar a utilização de acordo com a Tabela 7.

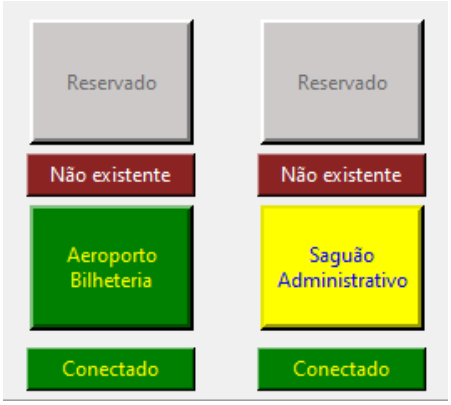
Tabela 7: Convenção do código de cores da GUI.

Verde	O botão está selecionado. Clique para desselecioná-lo.
Amarelo	O botão está desselecionado. Clique para selecioná-lo.
Laranja	Botão está informando algum alerta.
Cinza	Botão está desativado devido a alguma restrição.

Fonte: O Autor.

Começando pelo bloco **1** superior chamado “Estações”, podemos escolher o local que desejamos modificar a *playlist* (no momento somente o saguão do Prédio Administrativo e a bilheteria da Estação Aeroporto) e visualizar o nível de conexão do servidor com esses. No momento em que selecionamos um dos locais, o bloco inferior de “Playlist Local” é atualizado para os seus termos correspondentes. Na Figura 26 temos uma aproximação no bloco superior.

Figura 26: Bloco “Estações”.



Fonte: O autor.

Para a apresentação do nível de conexão entre o servidor e o local é possível ter as combinações da Figura 27.

Figura 27: Níveis de conexão possíveis do bloco “Estações”.

Fonte: O autor.

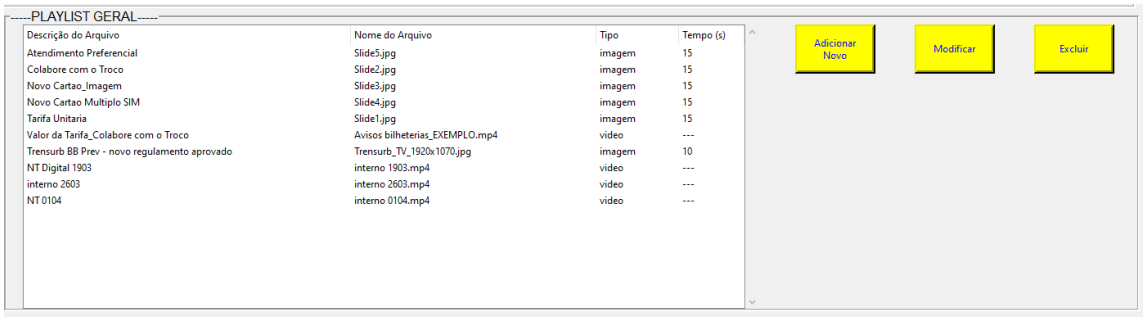
À esquerda da GUI temos o bloco **2** “Comandos das Estações”, com uma aproximação na Figura 28. Ele possibilita a seleção de uma estação por vez ou múltiplas, com um atalho para selecionar ou desselecionar todas.

Figura 28: Bloco “Comandos das Estações”.

Fonte: O autor.

No meio da GUI temos o bloco **3** “Playlist Geral”, em que uma Treeview apresenta todos os termos já adicionados nela. À direita temos uma visão das funcionalidades possíveis de “Adicionar Novo”, “Modificar” e “Excluir” da “Playlist Geral”. Na Figura 29 temos uma aproximação deste bloco.

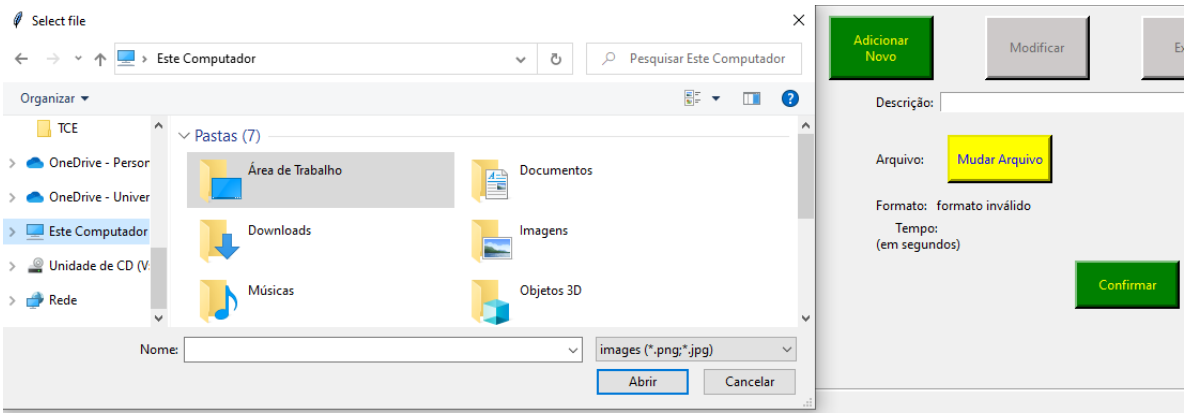
Figura 29: Bloco “Playlist Geral”.



Fonte: O autor.

Selecionando a funcionalidade “Adicionar Novo”, uma aba é aberta no computador requisitando o novo termo a ser escolhido, e um campo colocando as opções desejadas. Testes são feitos para verificar se o termo escolhido é um vídeo, e neste caso o tempo é automaticamente preenchido com o tempo do vídeo. Quando o botão “Confirmar” é selecionado, é gerado uma cópia do termo escolhido no servidor, são atualizadas as informações no banco de dados MySQL e é atualizado a Treeview de apresentação da “Playlist Geral”. A apresentação desta funcionalidade está na Figura 30.

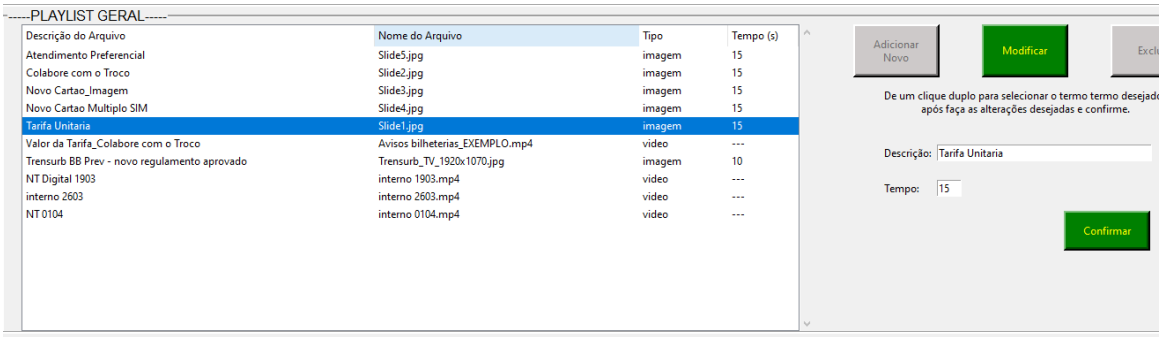
Figura 30: Funcionalidade “Adicionar Novo” do bloco “Playlist Geral”.



Fonte: O autor.

Na funcionalidade “Modificar”, o usuário seleciona o termo diretamente na “Playlist Geral” e os campos são automaticamente preenchidos com os valores atuais daquele termo, deixando o usuário fazer as alterações desejadas. Igual a funcionalidade de adicionar novo, quando apertado o botão “Confirmar”, são atualizadas as informações no banco de dados MySQL e é atualizado a Treeview de apresentação da “Playlist Geral”. Na figura 31 está demonstrada a funcionalidade “Modificar”.

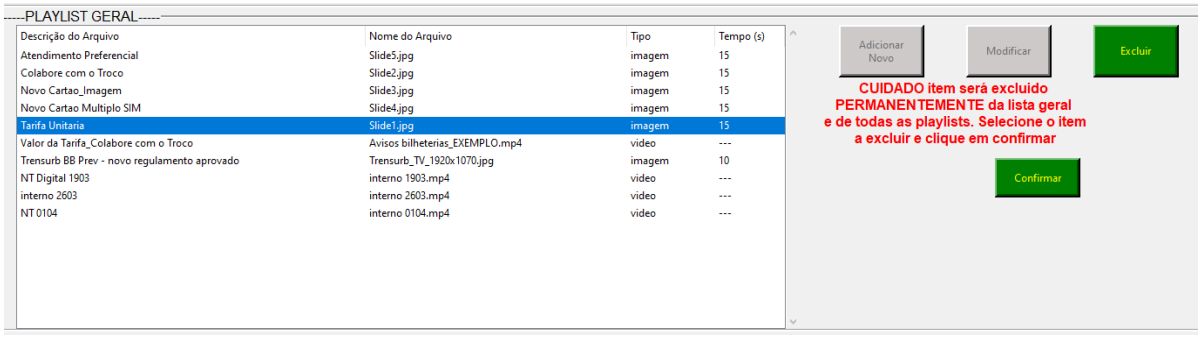
Figura 31: Funcionalidade “Modificar” do bloco “Playlist Geral”.



Fonte: O autor.

Como última funcionalidade deste bloco, foi criado o “Excluir”. Ao usar a funcionalidade, o usuário seleciona o termo desejado na “Playlist Geral”, e similar as duas funcionalidades anterior, ao confirmar é excluído o termo escolhido no servidor, são atualizadas as informações no banco de dados MySQL e é atualizado a Treeview de apresentação da “Playlist Geral”. Na figura 32 a demonstração da funcionalidade “Excluir”.

Figura 32: Funcionalidade “Excluir” do bloco “Playlist Geral”.



Fonte: O autor.

Para garantir o uso correto da GUI, diversos testes são executados, descritos na Tabela 8.

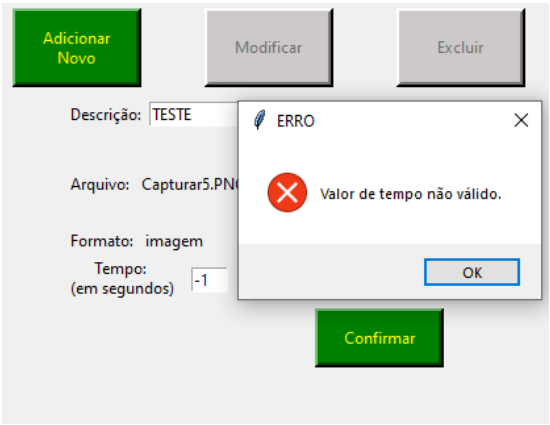
Tabela 8: Testes do bloco “Playlist Geral”.

Teste	Descrição	Resultado se falha
Conexão com módulo servidor	Código em Python antes de inicializar GUI testa conexão com banco de dados do servidor e os seus arquivos.	Informa ao usuário a falha de conexão e GUI não inicializa ou a fecha se já aberta.
Arquivo a adicionar válido	Quando um termo for adicionado à “Playlist Geral”, verifica-se o formato do arquivo.	O arquivo não é adicionado e uma mensagem de erro correspondente é mostrada, informando ao usuário.
Arquivo a adicionar duplicado	Quando um termo for adicionado à “Playlist Geral”, verifica-se se ele já está presente.	O arquivo não é adicionado e uma mensagem de erro correspondente é mostrada, informando ao usuário.
Arquivo a adicionar com descrição válida	Quando um termo for adicionado à “Playlist Geral”, verifica-se que existe uma descrição.	O arquivo não é adicionado e uma mensagem de erro correspondente é mostrada, informando ao usuário.
Arquivo a adicionar com tempo válido	Quando uma imagem for adicionada à “Playlist Geral”, verifica-se que existe um tempo preenchido e que ele é um número inteiro (em segundos) e positivo.	O arquivo não é adicionado e uma mensagem de erro correspondente é mostrada, informando ao usuário.

FONTE: O autor.

Na Figura 33 é apresentado a demonstração de um destes testes ocasionados pelo preenchimento incorreto do tempo, ocasionando uma notificação ao usuário.

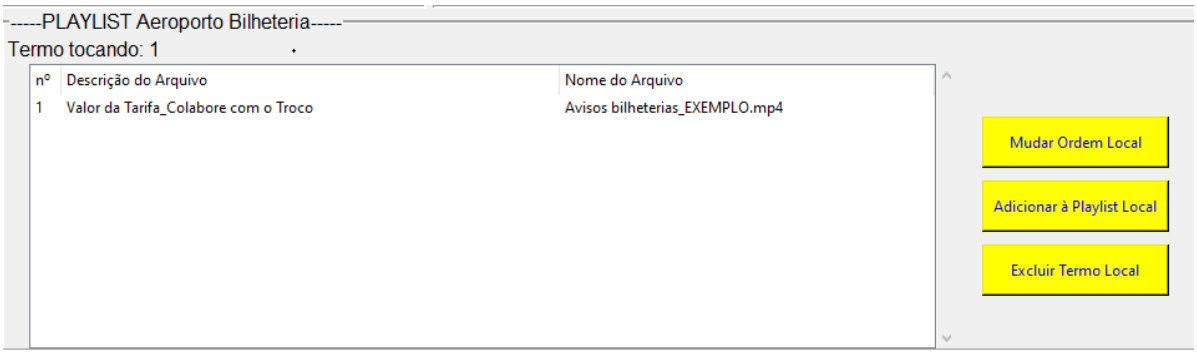
Figura 33: Teste de “arquivo a adicionar com tempo válido”.



Fonte: O autor.

Como último bloco em **4**, na parte inferior temos a “Playlist Local”, em que seu nome e informações dispostas são modificadas dependendo da estação selecionada ou das estações selecionadas. Neste bloco estão presentes os termos que estão tocando no local, o número do termo sendo apresentado, a funcionalidade de “Mudar Ordem Local”, “adicionar à Playlist Local” e “Excluir Termo Local”. Na Figura 34 temos uma aproximação para este bloco da GUI.

Figura 34: Imagem do bloco “Playlist Local”.



Fonte: O autor.

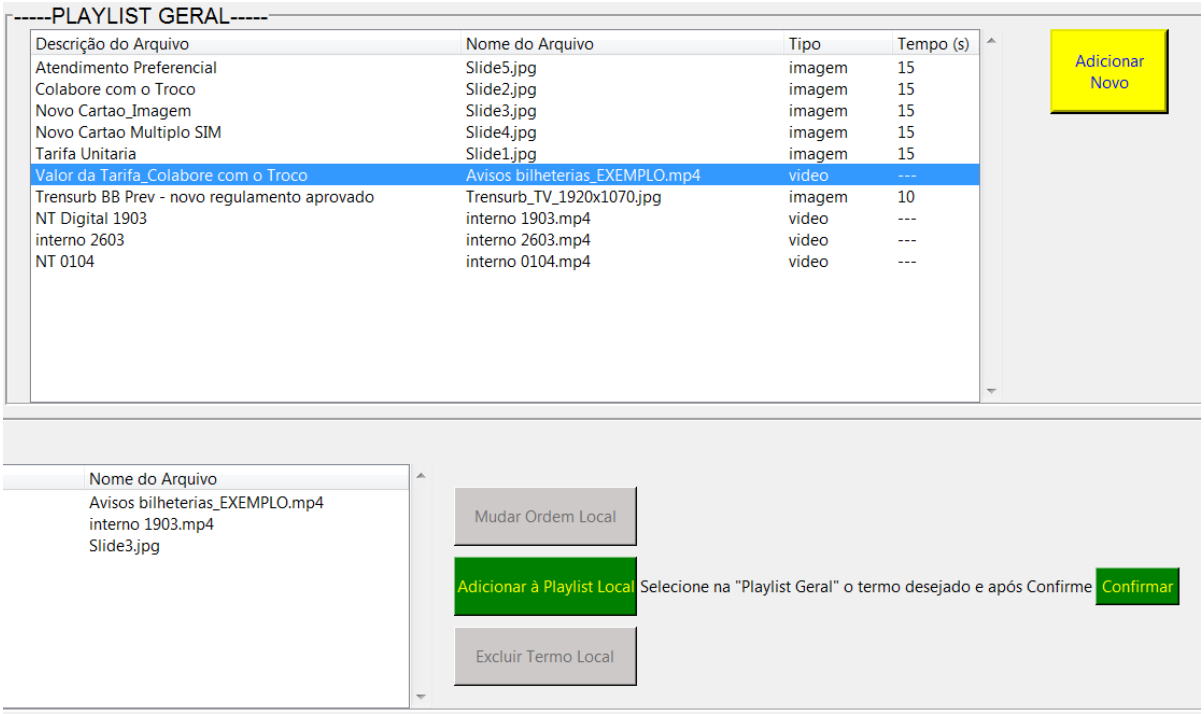
Neste bloco, ao selecionarmos a funcionalidade “Mudar Ordem Local”, podemos selecionar o termo desejado de mudar de lugar, escolher a nova posição desejada através do Combobox. Ao confirmar, será realizada a atualização da lista de reprodução. Na Figura 35 a apresentação desta funcionalidade.

Figura 35: Funcionalidade “Mudar Ordem Local” do bloco “Playlist Local”.

Fonte: O autor.

Ao selecionarmos a funcionalidade “Adicionar à Playlist Local”, selecionamos o termo desejado da “Playlist Geral” e após confirmamos para adicioná-la a “Playlist Local”. Na Figura 36 uma exposição desta funcionalidade.

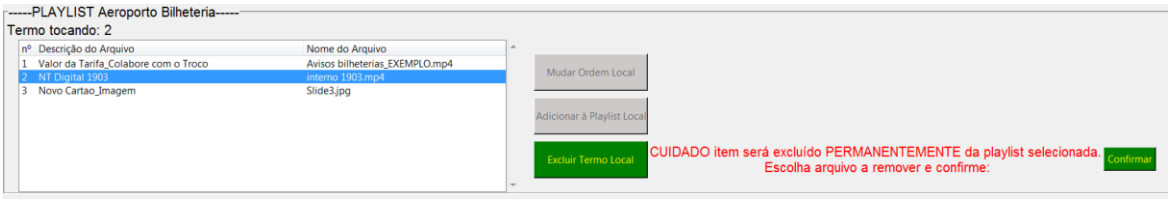
Figura 36: Funcionalidade “Adicionar à Playlist Local” do bloco “Playlist Local”.



Fonte: O autor.

Ao selecionarmos a funcionalidade “Excluir Termo Local”, selecionamos o termo que desejamos excluir da Treeview da “Playlist Local” e após confirmamos para realizar a alteração. Na Figura 37 a exposição desta funcionalidade.

Figura 37: Funcionalidade “Excluir Termo Local” do bloco “Playlist Local”.



Fonte: O autor.

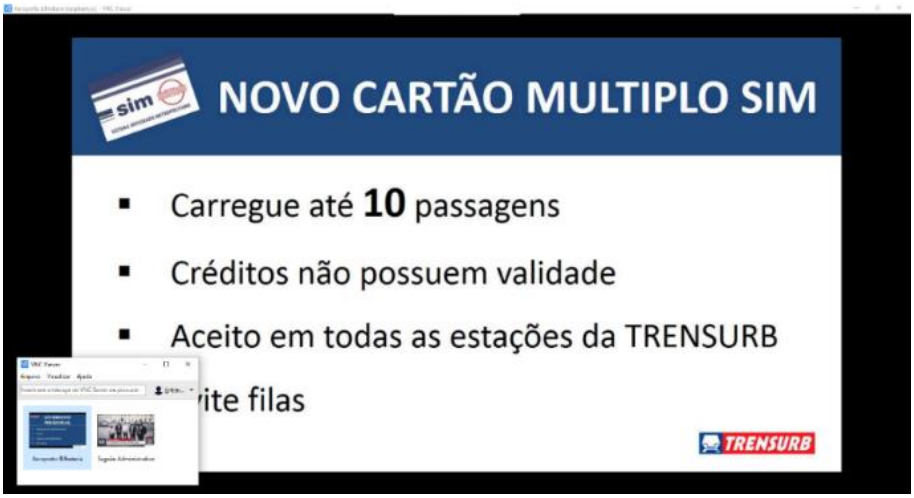
Todas as informações dispostas e que se deseja atualizar é feita diretamente no módulo servidor em seus arquivos e no banco de dados MySQL localizado neste. Isto é feito de maneira automatizada através do mesmo código em Python da GUI. A seguir um resumo das bibliotecas externas do Python sendo utilizadas para este serviço:

- Tkinter: cria a interface GUI.
- Shutil: copia o vídeo, imagem ou tela do computador do módulo de administração para o módulo do servidor quando termo adicionado à “Playlist Geral”.
- Threading: implementa *timers* em paralelo ao funcionamento da GUI que atualizam informação do e para o banco de dados do módulo do servidor.
- Moviepy: realiza a leitura do tempo de vídeos de maneira automatizada.
- Mysql-connector: conecta código em Python com banco de dados MySQL do módulo do servidor.

5.3.2 Serviço de Visualização e Atualização Remota

Para o serviço de visualização e atualização remota utilizando o protocolo VNC optamos por utilizar o programa VNC Viewer. Como a distribuição padrão do OS do Raspberry Pi, usado no módulo do cliente descrito na seção 5.3, já possui pré-instalado o programa VNC Viewer, bastando somente habilitá-lo, optou-se pela utilização deste programa em todos os módulos. O programa funciona de maneira que o computador do módulo de administração escolhe o dispositivo de computador que deseja conectar usando o IP deste e a senha correspondente, e logo após obtém acesso a tela atualmente apresentada e total controle sobre seus arquivos internos de forma remota. Na Figura 38 é possível visualizar a tela da Estação Aeroporto através do VNC Viewer.

Figura 38: Tela do monitor da Estação Aeroporto visualizada pelo módulo de administração.

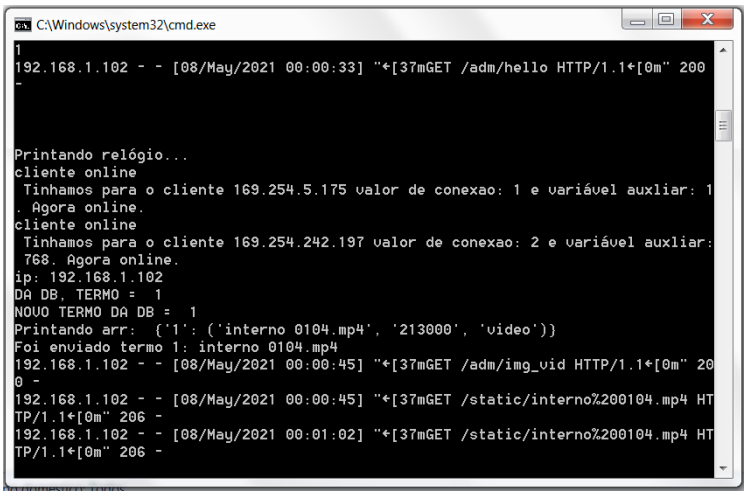


Fonte: O autor.

5.4 Integração dos Módulos

Com o desenvolvimento dos módulos do sistema, realizamos a interconexão de todos com o intuito de testar a eficácia de todo o sistema. Para isto seguimos os passos descritos nas seções 5.1, 5.2 e 5.3. Após inicializamos o servidor, atualizamos a lista de reprodução desejada nos clientes das estações e conectamos o Raspberry Pi ao sistema. O sistema demonstrou-se eficaz e robusto a garantir funcionamento e manter-se desde novembro de 2020 até o momento de entrega deste documento. Na Figura 39 o servidor recebendo requisições dos clientes. Na Figura 40 a apresentação das informações ocorrendo em dois clientes distintos ocorrendo de forma simultânea.

Figura 39: Apresentação do servidor recebendo requisições dos clientes.



Fonte: O autor.

Figura 40: Apresentação de dois monitores em locais distintos apresentando informações diferentes.

Fonte: O autor.

6. CONCLUSÃO E TRABALHOS FUTUROS

O objetivo deste trabalho foi desenvolver e implementar um sistema de painéis anunciadores de informação para as estações da Trensurb, com uma reprodução em formato de lista de reprodução e com um módulo de administração em outro local em que é possível modificar e visualizar as informações em tempo real. Os testes e as primeiras implementações mostram que o sistema desenvolvido consegue atender aos requisitos propostos, apesar de ainda não estar totalmente finalizado e sem sua implementação final em grande escala.

Como determinado na especificação, é possível adicionar e apresentar ao sistema arquivos de formatos de vídeo, imagem ou telas HTML para todas as telas conectadas, em que ficam armazenadas em um computador central. Foi desenvolvido um sistema expansível e modular, que permite a adaptação e implementação de novas funcionalidades de maneira fácil. Além disso, o sistema desenvolvido garante plenitude de funcionamento através dos serviços:

- O serviço de apresentação de informação aos painéis anunciadores de informação;
- O serviço de teste de conexão;
- O serviço de servidor local de *backup*;
- O serviço de sincronização horária das telas;
- O serviço de acesso remoto dos painéis.

O projeto desenvolvido foi baseado na divisão de três módulos na sua constituição, o módulo do cliente nas estações, o módulo do servidor e módulo de administração na sede da Trensurb. As informações são armazenadas em arquivos e no banco de dados do servidor. Os testes e protótipos iniciais funcionaram com sucesso tanto no saguão do Prédio Administrativo quanto para a bilheteria da Estação Aeroporto. A Trensurb, em reconhecimento ao projeto e por motivos de divulgação interna da empresa, publicou uma notícia sobre o sistema e o autor deste documento, que pode ser consultado em (SOUZA, 2021).

Entre as dificuldades que encontramos durante a realização do trabalho, destaca-se a limitada disponibilidade de ferramentas encontradas em ambientes de trabalho; a necessidade de aprender diversos conhecimentos referentes a Engenharia da Computação e saber aplicá-los de maneira eficaz; e dois *bugs* encontrados na

distribuição oficial do navegador Chromium do Raspberry Pi. No entanto, todos estes problemas foram resolvidos ou contornados de alguma forma, em que através de diferentes adaptações e iterações, conseguimos chegar na versão atual e funcional.

Como trabalhos futuros, algumas das pretensões são:

- Apresentar mais painéis em outros locais;
- Criar um modelo de *feedback* e avaliação dos passageiros e outros funcionários;
- Criar uma tela informativa do intervalo entre os trens para a plataforma;
- Criar uma tela informativa do tempo de chegada do trem para a plataforma;
- Atrelar ao sistema já existente de avisos sonoros das estações para apresentação síncrona de som e imagem;
- Criar e automatizar um modelo para apresentação prioritária de alertas no CCO;
- Transformar o servidor físico em uma máquina virtual;
- Programação prévia do horário de aparecimento de páginas (como o fechamento e abertura das estações);
- Expandir a interface gráfica para englobar mais funcionalidades e formatos de arquivos de apresentação;
- Otimizações no modelo de apresentação, como transferência de arquivos para diminuir a largura da banda da rede necessária;
- Desenvolvimento de protocolos de segurança;
- Possibilidade de migração do módulo de administração para uma página web.

Apesar da aplicação presente ainda ser limitada, o trabalho mostra a sua eficácia ao demonstrar suas aplicações atuais e possíveis expansões planejadas.

7. REFERÊNCIAS

BALENA. **Build a Raspberry Pi powered live train station sign for your desk**. Disponível em: <https://www.balena.io/blog/build-a-raspberry-pi-powered-train-station-oled-sign-for-your-desk/>. Acesso em: 26 abr. 2021.

FLASK. **Welcome to Flask — Flask Documentation (1.1.x).** Disponível em: <https://flask.palletsprojects.com/en/1.1.x/>. Acesso em: 7 maio. 2021.

FLASK-SQLALCHEMY. **Flask-SQLAlchemy Documentation (2.x)**. Disponível em: <https://flask-sqlalchemy.palletsprojects.com/en/2.x/>. Acesso em: 27 abr. 2021.

FOUNDATION, T. R. P. **Buy a Raspberry Pi 3 Model B**. Disponível em: <https://www.raspberrypi.org/products/raspberry-pi-3-model-b/>. Acesso em: 26 abr. 2021.

GALLEON SYSTEMS. **NTP Server Tool**. Disponível em: <https://www.ntp-time-server.com/ntp-server-tool.html>. Acesso em: 4 maio. 2021.

GURU. **TCP/IP vs OSI Model: What's the Difference?** Disponível em: <https://www.guru99.com/difference-tcp-ip-vs-osi-model.html>. Acesso em: 26 abr. 2021.

LUMINATOR. **How Transit Can Leverage Information Technology to Respond to and.** Disponível em: <https://www.linkedin.com/feed/update/urn:li:activity:6754816625821777920>. Acesso em: 7 maio. 2021.

MEINBERG. **Meinberg NTP Software Downloads.** Disponível em: <https://www.meinbergglobal.com/english/sw/ntp.htm>. Acesso em: 27 abr. 2021.

MOBITEC. **A Luminator Technology Group Company**. Disponível em: <https://www.mobitec.com.br/en/>. Acesso em: 26 abr. 2021.

REALVNC. **Download VNC Viewer | VNC® Connect.** Disponível em: <https://www.realvnc.com/en/connect/download/viewer/>. Acesso em: 7 maio. 2021.

SERPANOS, D.; WOLF, T. **Architecture of Network Systems.** Saint Louis: Elsevier Science, 2014.

SNCFWORLD. **BIENVENUE SUR SNCFWORLD, dezembro 2016.** Disponível em: <http://www.sncfworld.fr.nf/>. Acesso em: 7 maio. 2021.

RUSSEL. **OSI: The Internet That Wasn't, janeiro 2011.** Disponível em: < https://spectrum.ieee.org/tech-history/cyberspace/osi-the-internet-that-wasnt>. Acesso em: 8 junho. 2021.

SOUZA, Ingrid. **Notícias Trensurb - Estagiário Desenvolve Projeto para Auxiliar a Divulgação de Informações nas Estações.** Disponível em: <https://mail.trensurb.gov.br/noticiastrensurb.nsf/5ff76070b38e397183257b1e006900

2a/1c406214a6dda4a50325867f0075b3c5?OpenDocument>. Acesso em: 27 abr. 2021.

TANENBAUM, A. S.; WETHERALL, D. **Computer networks.** 5th ed ed. Boston: Pearson Prentice Hall, 2011.

TAVARES, ALEXEI. **Entendendo o modelo OSI para melhorar sua capacidade de resolver problemas em uma rede Cisco**. **DlteC do Brasil**, 20 out. 2011. Disponível em: <http://www.dltec.com.br/blog/cisco/entendendo-o-modelo-osi-para-melhorar-sua-capacidade-de-resolver-problemas-em-uma-rede-cisco/>. Acesso em: 26 abr. 2021

TIMEDATECTL. **Controle the system time and date.** Disponível em: <https://www.freedesktop.org/software/systemd/man/timedatectl.html>. Acesso em: 7 maio. 2021.

TKINTER. **Python interface to Tcl/Tk — Python 3.9.5 documentation.** Disponível em: <https://docs.python.org/3/library/tkinter.html>. Acesso em: 7 maio. 2021.

TRENSURB. **Empresa de Trens Urbanos de Porto Alegre S.A.** Disponível em: <http://www.trensurb.gov.br/paginas/paginas_detalhe.php?codigo_sitemap=3>. Acesso em: 26 abr. 2021.