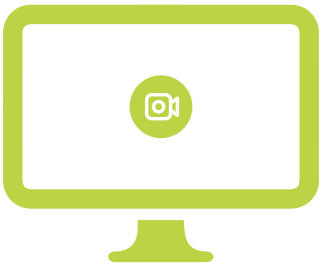

OPENMED VÍDEO: SISTEMA DE GRAVAÇÃO, EDIÇÃO E LAUDAGEM DE EXAMES EM VÍDEO INTEGRADO A SISTEMA PACS

OPENMED VIDEO: SYSTEM FOR RECORDING, EDITING AND MEDICAL REPORTS ON VIDEOS EXAMS INTEGRATED TO THE PACS SYSTEM

Thales Barros de Castro

Bacharel em Ciências e Tecnologia, engenheiro de computação e automação pela Universidade Federal do Rio Grande do Norte (UFRN) e pesquisador no Laboratório de Inovação Tecnológica em Saúde (LAIS). thalescast@gmail.com.

Prof. Dr. Hertz Wilton de Castro Lins

Doutor em Engenharia Elétrica e de Computação pela Universidade Federal do Rio Grande do Norte (UFRN), professor adjunto do Departamento de Engenharia de Comunicações da Universidade Federal do Rio Grande do Norte (UFRN), pesquisador do Grupo de Micro-ondas e Antenas – GMA. hertzw@gmail.com

Prof. Dr. Ricardo Alexsandro de Medeiros Valentim

Doutor em Engenharia Elétrica e de Computação pela Universidade Federal do Rio Grande do Norte (UFRN), professor adjunto da UFRN lotado no Departamento de Engenharia Biomédica e professor permanente do Programa de Pós-graduação em Engenharia Elétrica e de Computação (PpgEEC/UFRN), coordenador do Núcleo de Inovação Tecnológica em Saúde e do Laboratório de Inovação Tecnológica em Saúde (LAIS). ricardo.valentim@ufrnet.br

Ana Carolina Fernandes de Oliveira

Médica Otorrinolaringologista pela Hospital Universitário Onofre Lopes/Universidade Federal do Rio Grande do Norte. carol. foliveira@gmail.com.

Ana Cecília Sá Fernandes

Bacharel em Ciências e Tecnologia, engenheira biomédica e especialista em Informática na Saúde pela Universidade Federal do Rio Grande do Norte (UFRN), especialista em Engenharia Clínica pelo Instituto RTG, mestranda em Neuroengenharia pelo Instituto Internacional de Neurociências Edmond e Lily Safra (IIN-ELS) e engenheira biomédica na Secretaria de Estado da Saúde Pública do RN (SESAP/RN). aceciliasaf@gmail.com.

Marcel da Câmara Ribeiro-Dantas

Engenheiro de Computação e Automação pela Universidade Federal do Rio Grande do Norte (UFRN), especialista em Big Data pelo Instituto Metrópole Digital da UFRN, Mestre em Bioinformática pelo Programa de Pós-Graduação em Bioinformática da UFRN, aluno de doutorado na École Doctorale Informatique, Télécommunication et Électronique da Sorbonne Université e pesquisador no Institut Curie (UMR168). marcel.ribeiro-dantas@curie.fr

RESUMO

O trabalho em evidência tem como objetivo expor o desenvolvimento de um aplicativo *Desktop* que utiliza tecnologias *web* para gravação e edição de vídeos de exames de Nasofibrolaringoscopia e Laringoscopia, bem como a emissão de laudos digitais e a realização de uma integração com um servidor PACS. Nesse sentido, estuda novas tecnologias que possam facilitar e agilizar o progresso da aplicação, além de disponibilizar uma interface amigável. O *framework Electron* e tecnologias *web*, como CSS, HTML, JavaScript e Node.js, atendem bem a tais propósitos. Nesse contexto, o *OpenMed* Vídeo, nome do aplicativo apresentado, ao gravar e editar vídeos, atendeu os requisitos mínimos sugeridos pelos profissionais da saúde do serviço de otorrinolaringologia do Hospital Universitário Onofre Lopes (HUOL), local onde surgiu a demanda. O aplicativo em destaque também concedeu que os exames fossem armazenados de forma remota em um sistema de arquivamento e comunicação de imagens médicas, por meio de uma integração com um servidor PACS, ademais da realização de laudos digitais, permitindo que os profissionais do setor pudessem guardá-los, de forma mais segura, assim como os acessar de qualquer lugar, uma vez que esses se tornaram digitais. O referido projeto busca estar em consonância com a necessidade prevista pela equipe médica do HUOL e que venha a ser apropriado com êxito e com larga utilidade.

Palavras-chave: Arquivamento de imagens. Laudo digital. Otorrinolaringologia. Hospital universitário.

ABSTRACT

The project in evidence aims to expose the development of a Desktop application which uses web technologies for Nasofibrolaryngoscopy and Laryngoscopy video recording and editing exams as well as a digital result issue and an enterprise integration with a PACS server. In this respect, new technologies to facilitate and accelerate the application progress besides of providing an interface were studied. Electron framework and web technologies, such as CSS, HTML, JavaScript and Node.js, assist well for such purposes. In this regard, OpenMed Video, this application's name, records and edits videos, also it has met minimum requirements suggested by the health professionals of the Otolaryngology service in the Onofre Lopes University Hospital (HUOL), where a lawsuit arose. The Desktop application also allowed store exams in a medical imaging filing and a communication system through and integration with a PACS server, besides of allowing to make digital image exams and to give permission to health professionals to store exams, as well as to access them from anywhere, since they have become digital. Therefore, the project design is believed to be in line with the HUOL medical team forecast requirement and is expected to be successfully approved and with wide utility.

Keywords: Image archiving. Otolaryngology. Digital report. University hospital.

INTRODUÇÃO

Sabe-se que, desde o surgimento da informática, grandes volumes de dados vêm sendo gerados e armazenados. O seu armazenamento, normalmente, em um computador dá-se em discos magnéticos, conjunto de discos independentes (*Reduntant Array of Inexpensive Disks*, RAID), fitas magnéticas e memórias de estado sólido ou *Solid State Drive* (SSD). Vale ressaltar que qualquer um desses dispositivos tem espaço de memória limitado, sendo necessária a sua substituição quando os dados excedem o espaço físico do aparelho e não podem ser apagados.

Acresça-se a isso, na área médica, por exemplo, a utilização em grande escala de sistemas digitais, que tem gerado um volume de dados cada vez maior (MARQUES;

SALOMÃO, 2009), sendo, em sua totalidade, imagens geradas de exames. Para esse caso, a melhor solução a fim de gerenciar essa situação está na adoção de um Sistema de Arquivamento e Distribuição de Imagens (PACS, do inglês *Picture Archiving and Communication System*) (MARQUES; SALOMÃO, 2009).

Tratando-se do trabalho em destaque, o objeto de estudo foi o setor de otorrinolaringologia do Hospital Universitário Onofre Lopes (HUOL). Nesse setor médico, exames de Nasofibrolaringoscopia e Laringoscopia são realizados e gravados por intermédio de uma câmera Toshiba *control unit* ik cu44a, uma lente ik-M44h 85 e uma placa de captura de vídeo *PixelView Xcapture* USB, que faz a conversão de uma das saídas da câmera, s-vídeo, para dados digitais. Os exames, como dito, são gravados e armazenados em um único computador que, em pouco espaço de tempo, por limitação de memória, devem ser transferidos para discos rígidos externos.

Ainda nesse contexto, pode-se acrescentar que os exames são gravados pelo *software* proprietário da placa de captura, o *PowerDirector*, que realiza a gravação em vídeos no formato MPEG. Esse aplicativo é alvo de críticas dos médicos, que dizem ser uma aplicação que, por ter diversas funções de edição, não é intuitiva, dificultando o trabalho dos profissionais da saúde. Em uma das visitas feitas ao setor, por exemplo, pediu-se que uma das médicas recortasse algum exame de vídeo para que se tivesse uma visão das características do *software*. A profissional, por sua vez, sentiu-se desconfortável por não saber utilizar as funcionalidades fornecidas pelo programa, buscou de algumas formas descobrir como fazer, porém, não conseguiu, expondo na prática a dificuldade que é para editar os exames. Adicionando-se a essas dificuldades, o ambulatório buscava uma maneira de emitir laudos de forma digital, pois, atualmente, médicos e residentes entregam aos pacientes os laudos, que são preenchidos de forma manual e acondicionados em papéis, correndo-se o risco de perda e degradação.

Foi diante desse cenário, no setor de otorrinolaringologia do HUOL, que o *OpenMed* Vídeo surgiu com o intuito de ser uma plataforma que apresentasse os requisitos mínimos, ou seja, aqueles já fornecidos pelo aplicativo da placa de captura: gravar e editar vídeos. Além disso, procurou-se dar um caráter mais intuitivo, com poucos botões; permitir a emissão de laudagem digital; e estabelecer uma integração com um servidor PACS, de modo que o problema de armazenamento local seja solucionado. Portanto, o trabalho em evidência tem como objetivo apresentar o desenvolvimento de um *software* que grava, edita e recorta exames médicos, além de converter imagens do formato JPG para DICOM, a fim de realizar uma integração com um PACS, visto que o servidor trabalha com esse padrão.

Por fim, é importante mencionar alguns *softwares* semelhantes ao *OpenMed* Vídeo que foram encontrados. Um deles é o *VeDocs Elo,* que disponibiliza laudos e imagens de exames para pacientes e profissionais solicitantes por meio de um navegador de internet (TOUCHHEALTH, 2016). O outro é o *Clirea* ART, que permite a criação de *templates* de laudos personalizados pela própria organização ou pelo profissional de saúde, podendo armazenar as frases mais utilizadas de forma que sejam reaproveitadas no futuro (TOUCHHEALTH, 2016). O último deles é o SLC – Sistema de Laudos e Captura, que apresenta não apenas uma ou duas funções parecidas com o aplicativo em questão mas praticamente todo o *software*. Entretanto, vale ressaltar que o desenvolvedor não deixa claro trabalhar com DICOM, conversão e integração com sistema PACS, que são funções primordiais do projeto em notoriedade. Segundo seu fabricante, o SLC é um sistema completo destinado à área médica, que permite a realização de laudos médicos com captura de imagem direto do Ultrassom, Videocolposcópio, Endoscópio ou qualquer equipamento de exame por

imagem, que poderá gravar vídeos de exames e disponibilizar em CD ou *pendrive* e ainda possui a opção de editar brilho, contraste e saturação das imagens (MJR, 2016).

METODOLOGIA

Antes de começar a explicação das etapas referentes à implementação do *OpenMed,* é importante conhecer a teoria acerca dos exames realizados no serviço de Otorrinolaringologia, Nasofibrolaringoscopia e Laringoscopia, do que é um sistema PACS, comentar sobre o DICOM e as tecnologias que foram utilizadas para o desenvolvimento do projeto em evidência.

EXAMES DE NASOFIBROLARINGOSCOPIA E LARINGOSCOPIA

Pode-se afirmar que o desenvolvimento do projeto surgiu também da necessidade de gravar exames no setor de otorrinolaringologia, do Hospital Universitário Onofre Lopes (HUOL). Logo, esta seção dedicar-se-á à explicação dos exames realizados nesse hospital, Nasofibrolaringoscopia e Laringoscopia.

A otorrinolaringologia é a área da medicina responsável por abordar patologias do ouvido, nariz, garganta e pescoço. Por abordar tais regiões do organismo, faz-se necessária a utilização de lentes e fontes de luz adequadas, além de equipamentos que possibilitem a gravação e captura de imagens, a fim de encontrar anormalidades nas estruturas (VARELLA, 2016). Nesse contexto, torna-se importante a utilização da videoendoscopia que pode ser feita com fibras flexíveis (nasofibrolaringoscópio) ou rígidas (laringoscópio). As flexíveis, em geral, produzem uma imagem de menor qualidade, porém, são mais úteis para examinar cavidades nasais, como rinofaringe, orofaringe, hipofaringe e laringe, tanto em adultos como em crianças. Os diâmetros podem variar de 2,2 a 3,2 mm e alguns possuem canal de aspiração e manipulação. As fibras rígidas variam de 2,7 a 4,0 mm de diâmetro e sua ponta fixa em angulação são de 0°, 30°, 45°, 70°, 90° e 120° (PILTCHER; MAAHS; KUHL, 2015).

SISTEMA PACS

Um dos objetivos principais do trabalho em destaque é solucionar o problema de armazenamento dos exames, Nasofibrolaringoscopia e Laringoscopia, em forma de vídeo. Para isso, pensou-se em uma aplicação que pudesse estabelecer uma integração com um sistema PACS. Por esse motivo é que esta seção se destina à explicação dessa ferramenta, proporcionando um melhor entendimento da temática do aplicativo.

PACS (do inglês, *Picture Archiving and Communication System*) é um sistema de arquivamento e comunicação voltado para o diagnóstico por imagem e que permite o pronto acesso às imagens médicas em formato digital em qualquer setor de um hospital (MARQUES; SALOMÃO, 2009). Espera-se com um PACS, que é um sistema de informação, uma diminuição nas filas de espera, aumento na produtividade, eficiência e melhorias no atendimento aos pacientes (FIRMINO; PEREIRA; VALENTIN, 2012).

A arquitetura de um sistema PACS pode ser compreendida por equipamentos que realizam aquisição, armazenamento e exibição de imagens médicas. A Figura 1 a seguir ilustra os componentes básicos para a implementação dessa plataforma. Nesse caso, o termo modalidades se refere aos equipamentos de aquisição de imagens possíveis, como, por exemplo, tomografia computadorizada ou ressonância magnética.

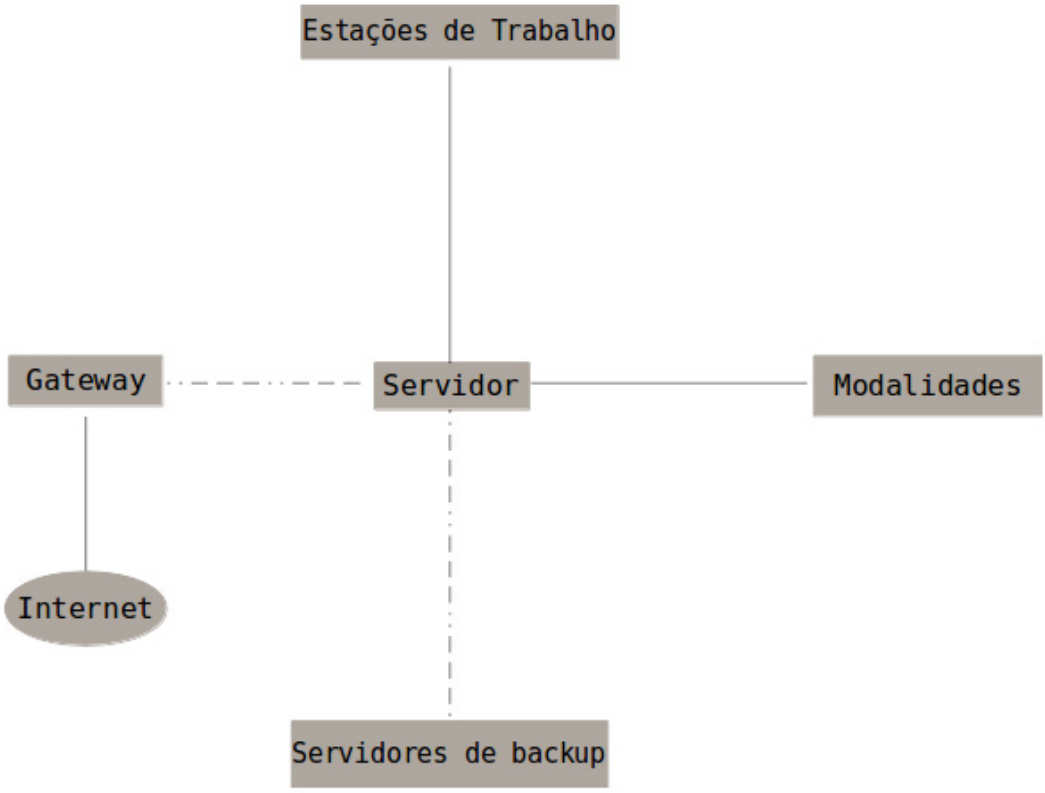

Figura 1 - Arquitetura de um Sistema PACS com *gateway* de Internet e Servidores de ***backup.***
Fonte: Autoria própria (2018).

O servidor PACS realiza o arquivamento das imagens, controlando a comunicação e todo o fluxo de dados de modo a garantir sua integridade. Por sua vez, os servidores de *backup* são úteis para a recuperação de dados, mantendo o sistema disponível quando inativo. O terceiro componente, que são as estações de trabalho, permitem aos profissionais de saúde a visualização das imagens médicas. Por último, a imagem apresenta um componente opcional, *Gateway* de *Internet*, que tem como função permitir a telerradiologia, ou seja, um diagnóstico a distância (telediagnóstico) ou uma segunda opinião especializada (teleconsultoria), por meio do envio digital das imagens (FIRMINO; PEREIRA; VALENTIN, 2012).

Tantos os servidores quanto as estações de trabalho possuem *softwares* aplicativos específicos para o sistema PACS. Esses programas são fundamentais para que esses componentes executem suas tarefas de forma adequada. Atualmente, existem vários *softwares*: DCM4CHEE, ConQuest e ClearCanvas PACS Server. Entre eles, o DCM4CHEE mostra-se melhor em relação à qualidade e requisitos DICOM. Diante disso, o Laboratório de Inovação Tecnológica em Saúde (LAIS) implantou um PACS, o OpenPACS, no Hospital Universitário Onofre Lopes, utilizando o DCM4CHEE.

DICOM

Não foi apresentado anteriormente, quando da explicação das modalidades, mas as funções principais delas são: aquisição de imagens de forma rápida e confiável, conversão dos dados para um formato padrão, chamado de DICOM, e o envio das imagens para o servidor (FIRMINO; PEREIRA; VALENTIN, 2012). Remetendo à conversão dos dados, pode-se dizer que o DICOM (do inglês, *Digital Imaging and Communications in Medicine*) é um formato de arquivo. Porém, é um engano pensar que seja somente isso, já que o formato é apenas um dos serviços que ele oferece.

Define-se o DICOM como um padrão global para transferência de imagens médicas e outras informações entre computadores (FIRMINO; PEREIRA; VALENTIN, 2012). Finalizando, vale ressaltar que ele foi desenvolvido pelo Colégio Americano de Radiologia (*American College of Radiology*, ACR) em conjunto com a Associação Americana de Equipamentos Elétricos (NEMA) e que, devido ao seu impacto mundial, agora é mantido e atualizado por um comitê multidisciplinar internacional.

TECNOLOGIAS UTILIZADAS

Node.js

O *Node.js* é um servidor *web* que está em contraste com o modelo de concorrência mais comum nos dias de hoje, Figura 3, em que *threads* são alocadas pelo sistema operacional (NODE.JS FOUNDATION, 2016). Diz-se que *threads* são processos "leves", mas na verdade são partes de um processo, assim como estes são fragmentos de um programa (PENHA; CORREA; MARTINS, 2002). Essa estrutura atual pode ser vista na Figura 2, observando-se que os recursos são reservados à medida que as requisições de usuário chegam.

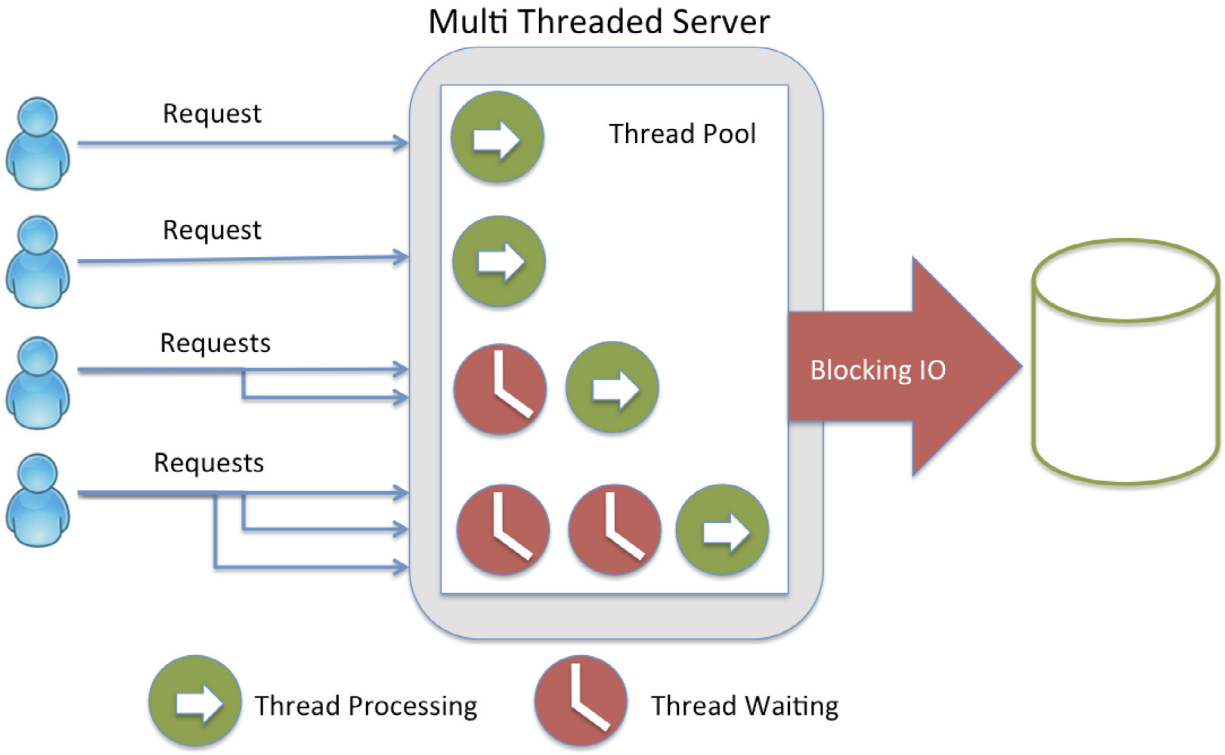

Figura 2 – Estrutura de um servidor de múltiplas *threads*.
Fonte: StrongLoop (2016).

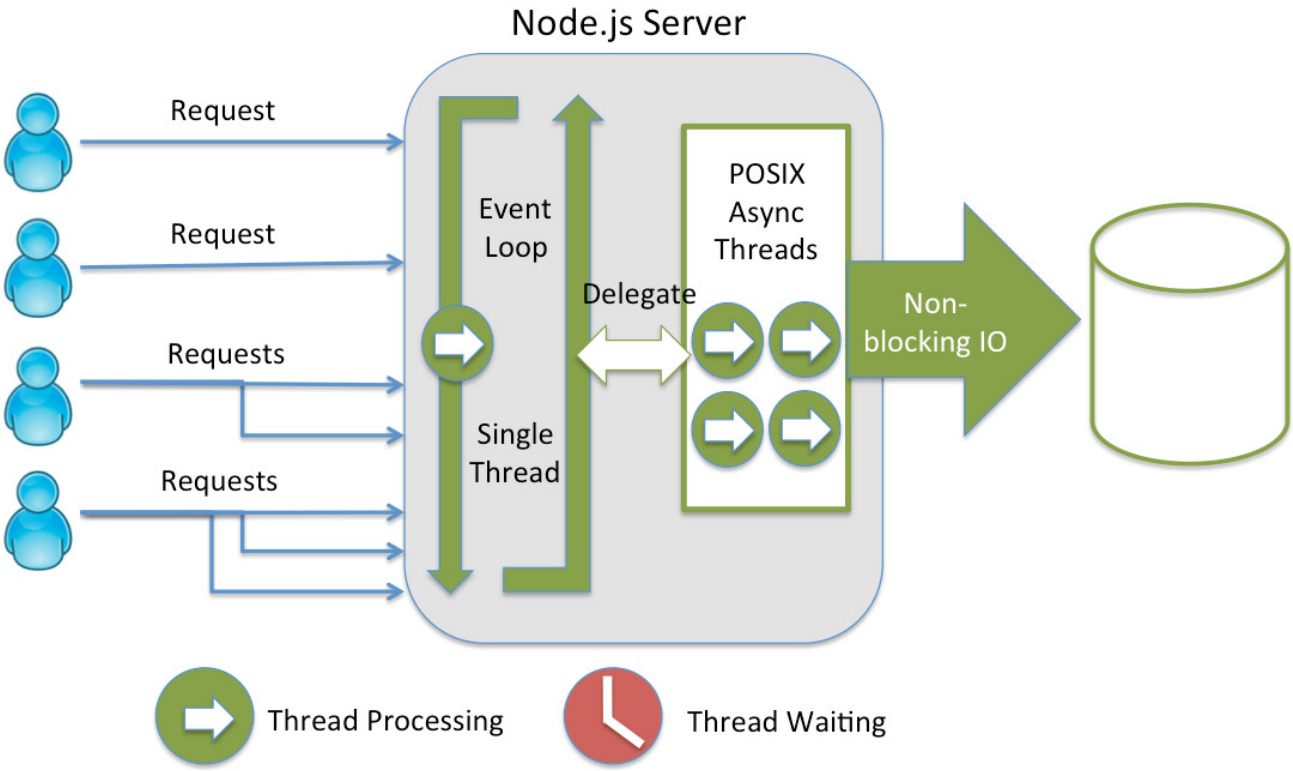

Figura 3 – Estrutura do ***node.js.***
Fonte: StrongLoop (2016).

Em um ambiente de muitas requisições a operações de Entrada e Saída (E/S), é inevitável que haja E/S bloqueante, fazendo com que as *threads* esperem, desperdiçando memória, por exemplo. Por sua vez, o *Node.js*, com uma proposta diferente, tem o sistema de concorrência, apresentado na Figura 3, de modo que as requisições ao servidor são tratadas pelo o *Event Loop*, que é um *Single Thread* que as recebe e delega para um *Pool* de *Threads* fazer operações de Entrada e Saída (NODE.JS FOUNDATION, 2016). Como pode ser visto, o *Node* procura otimizar os recursos computacionais com uma estrutura diferente da dos demais servidores, como o caso do *Apache*, por exemplo, que para cada requisição uma nova *Thread* é alocada concorrentemente. Dessa maneira, dá a entender que por ele usar um único *thread,* não existe escalabilidade, porém, percebe-se que isso não é verdade pelo fato de existir uma piscina de *threads* em que realizam suas tarefas de forma não bloqueante. Ele foi criado com a ideia de se ter essa estrutura diferente, contudo, é mais conhecido por possibilitar a utilização de *javaScript* do lado servidor, sendo isso uma consequência e não um dos objetivos.

Finalizando esse contexto, é importante ressaltar que o *Node.js* também disponibiliza um sistema simples de carregamento de módulos, de forma que sua utilização permite incluir outros arquivos *javaScript* (MOREIRA, 2016). Esses módulos são carregados por meio do comando *npm*, que é o gerenciador de pacotes do *node* e o maior ecossistema de bibliotecas de código aberto do mundo inteiro (NODE.JS FOUNDATION, 2016).

Electron

O *Electron* é um *framework* para aplicações *Desktop*, que permite a utilização de ferramentas *web* (VARELLA, 2016), como HTML, CSS e JavaScript. Desenvolvido pelos profissionais do *GitHub*, tem a seguinte estrutura:

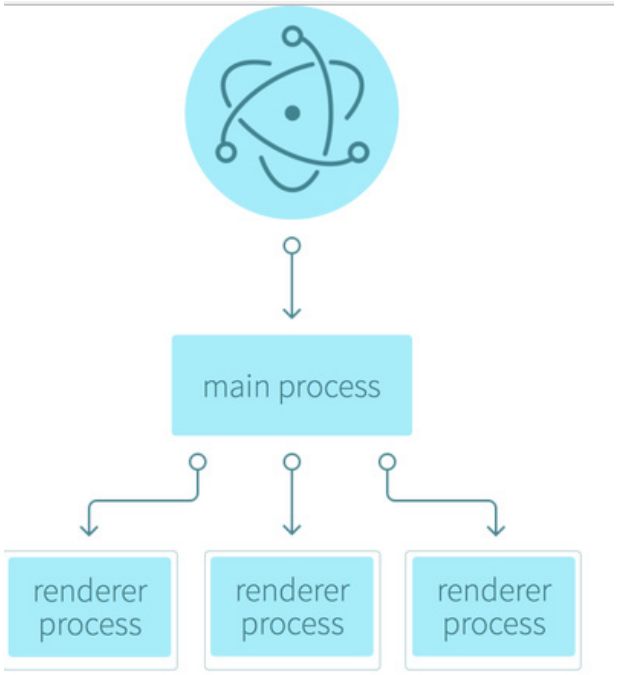

Figura 4 – Estrutura do Electron.
Fonte: ComSysto (2016).

Antes de explicar a Figura 4, faz-se necessário um breve comentário do que é um *framework*. Trata-se de um conjunto de algoritmos (*micro-frameworks*) ou sistemas inteiros úteis no desenvolvimento de aplicações *web* (DANTAS, 2016).

No que diz respeito à Figura 4, podem-se observar três camadas: o *Chromium* que é a versão aberta do *Google Chrome*; o *Main Process* em que a aplicação é criada e monitorada; e o *Renderer Process*, ou novos processos, que nesse contexto, são simples abas do navegador. O *Chromium* é a ferramenta em que a aplicação é executada e que utiliza o interpretador de código para javaScript, desenvolvido pela *google*, o V8. O *Main* monitora o ciclo de vida da aplicação e possibilita a utilização dos recursos do *Node.js*. Ressalta-se ainda que o processo *Renderer* permite que tecnologias *web* sejam utilizadas como em um *website* qualquer (VARELLA, 2016).

Tecnologias *web*

Como dito anteriormente, o *Electron* é uma plataforma que permite criar aplicativos *desktop* por meio de tecnologias *web*. Pode-se dizer que é umas das vantagens desse *framework*, pois além desses tipos de tecnologia fornecerem um ágil desenvolvimento ao programador, também permitem que a interface dos aplicativos seja customizável com o possível uso do *CSS*, por exemplo.

As tecnologias *web* utilizadas neste estudo foram o HTML, o CSS e *javaScript*. Pode-se afirmar que as duas primeiras são utilizadas apenas no *front-end* da aplicação, enquanto que o *javaScript* pode ser usado tanto no *back-end* quanto no *front*. Isso porque, como já foi mencionado na subseção *Node.js*, com o advento do *node*, essa linguagem pode ser trazida para o lado servidor.

Entende-se por *front-end* e *back-end* camadas entre o *hardware* e o usuário final que, no contexto de desenvolvimento *web*, são as duas que mais se destacam. O *front-end* é uma abstração para algumas funções do sistema que terão como objetivo a interação direta com o usuário e o *back-end* processa vários de tipos de informações para essa primeira camada (DANTAS, 2016). Vale ressaltar que mesmo não sendo uma linguagem de programação considerada em sistemas *web*, o C juntamente com uma ferramenta de programação paralela, *openMP*, foi utilizada nessa aplicação, como será dito mais adiante, mostrando a grande flexibilidade que o *Node.js* traz.

GRAVAÇÃO DOS EXAMES

As seções anteriores tiveram como finalidade apresentar os exames gravados no setor, bem como o sistema PACS, o DICOM e as tecnologias utilizadas para o desenvolvimento da aplicação. Esta seção e as seguintes remetem às etapas referentes à implementação do projeto em destaque.

O *OpenMed* Vídeo foi desenvolvido para ser um aplicativo *Desktop* a fim de gravar e editar exames de Nasofibrolaringoscopia e Laringoscopia. Para isso, foram estudadas

novas tecnologias que pudessem facilitar e agilizar o desenvolvimento da aplicação bem como disponibilizar uma interface amigável. O *Electron* e tecnologias *web*, como CSS, HTML, JavaScript e *Node.js*, mencionados anteriormente, atendem bem a esses propósitos.

Após a escolha das tecnologias a ser utilizadas, caminhou-se em direção à implementação do projeto e uma das primeiras etapas foi a gravação dos exames. De posse das informações de como o vídeo é capturado, citadas logo no início deste artigo, a ideia é de que, quando a placa de captura de vídeo for conectada no computador via USB, o aplicativo a reconheça como uma *webcam*.

A transmissão do vídeo é feita por meio de um envio contínuo de pacotes de dados da *webcam*, conhecido por *stream*. Para obter esse fluxo de dados, optou-se em utilizar o WebRTC, que é um projeto de código aberto, que não necessita de *plugins* para realizar a comunicação em tempo real de áudio, vídeos e dados na *web* (WEBRTC, 2016). Essa plataforma, desenvolvida pelo grupo de pesquisadores da *World Wide Web Consortium*, disponibiliza várias APIs em *javaScript*, sendo duas delas para captura e gravação de áudio e vídeo, a saber, *getUserMedia* e *MediaRecorder*, respectivamente. Vale ressaltar que uma API (do inglês, Application Programming Interface) é, em resumo, uma documentação de como se pode realizar uma determinada tarefa a partir de uma biblioteca.

O *getUserMedia* fornece uma maneira de acessar o fluxo da câmera e do áudio local do computador. O primeiro parâmetro que essa função recebe indica quais mídias se deseja acessar, como, por exemplo, áudio e vídeo (WEBRTC, 2016). A partir daí, a captura do fluxo da *webcam* é realizada dentro da aplicação e, após isso, cria-se, por meio do código desenvolvido, um caminho interno para o fluxo de vídeo da câmera.

Com o acesso às mídias, o problema agora seria como conseguir gravá-las, o que seria a parte mais importante dessa primeira etapa. Isso foi conseguido com a utilização da outra API, fornecida pelo WebRTC, a *MediaRecorder*. Essa interface cria um objeto, recebendo o *stream* de vídeo e algumas opções de formato, que no caso do projeto, o escolhido foi o WebM. Por fim, o objeto *MediaRecorder* disponibiliza as funções *start* e *stop* para iniciar a gravação e finalizá-la, respectivamente.

ABRIR E EDITAR OS VÍDEOS

Além da necessidade de gravar, foi observada também a necessidade de poder abrir o exame gravado e editá-lo. A ideia inicial, proveniente do setor de otorrinolaringologia, do Hospital Universitário Onofre Lopes (HUOL), era de poder tirar uma foto em um instante de tempo do vídeo, poder desenhar com o cursor do mouse e colocar textos, indicando alguma anomalia nos exames.

Com apenas quatro botões, todo o objetivo é alcançado. Um único botão serve para tirar a foto e limpar o que foi desenhado, como poderá ser observado na Figura 11, na seção Abrir e Editar mais a frente. No momento em que a foto é capturada, ilustrado na Figura 12 adiante, na seção subsequente, três outros botões aparecem: um para mudar a cor da linha, a espessura e outro para realizar o laudo com a imagem editada. Nenhum botão existe para colocar textos nas imagens já que, apenas com dois cliques, abre-se uma janela para o usuário inserir alguma observação.

RECORTAR VÍDEOS GRAVADOS

Na gravação dos exames, algumas partes dos vídeos são dispensáveis, na maior parte das vezes, seu início e seu fim, pois é quando a lente é introduzida e retirada do paciente, não gravando informação pertinente. Com o objetivo de ter vídeos menores para se obter um melhor desempenho de outras

funcionalidades do aplicativo, bem como um melhor armazenamento, surgiu a necessidade de deixar só aquilo que é válido, ou seja, oferecer ao usuário a opção, quando necessário, de recortar o vídeo.

Na página principal da aplicação, existem quatro botões, Figura 6, destacando-se até este momento: Gravar Exame e Abrir e Editar Exames. Podia-se fazer essa etapa dentro de Abrir e Editar exames, no entanto, optou-se por criar um botão com o nome Recortar Exame.

Apesar da existência desses quatro botões, pode-se dizer que os botões principais são os de recortar o início e o fim do vídeo. Quando o usuário clica no primeiro botão, o do início, por exemplo, uma caixa de diálogo é mostrada com dois campos, um para o tempo inicial, ou melhor explicando, o tempo que se deseja de início do corte e o outro para a duração, isto é, para até onde vai o corte. Quando o usuário coloca os valores de tempo, o sistema é forçado a colocar no formato 00:00:00. Quando o usuário clica no botão para recortar o fim, um processo semelhante acontece. Só que dessa vez é só a duração já que o tempo inicial é configurado para 0, isto é, começa-se do início do vídeo e vai até o tempo informado, realizando o corte final.

Nessa etapa, existem algumas possibilidades para as quais o arquivo deve ser recortado. A primeira delas é quando o usuário deseja apenas retirar o fim do vídeo e tudo é feito em cima do arquivo original, e a segunda é quando o usuário deseja recortar o início e o fim. Uma possível desvantagem é que o usuário não pode alterar a ordem e cortar primeiro o fim e depois o início, ou ele corta apenas o início ou apenas o fim, ou ambos.

Para finalizar, existe o botão de Substituir arquivo, que, após o exame ser editado, aplicam-se as alterações. Quando o botão é clicado, uma função é chamada para verificar qual parte do vídeo foi recortado, melhor dizendo, qual saída será salva, se é a do início ou a do fim do vídeo. O arquivo editado deve ser renomeado para o nome do arquivo inicialmente baixado pelo usuário, a fim de que a substituição seja realizada com sucesso.

CONVERSÃO DO VÍDEO WEBM EM DICOM

Os exames gravados precisam ser armazenados e, atualmente, o setor de otorrinolaringologia armazena em discos magnéticos, que excedem o espaço em pouco tempo, pela alta demanda de exames. Uma das ideias foi de armazenamento em um servidor *web*, porém, não poderia ser em qualquer um, uma vez que se trata de exames médicos. Para esse caso, a melhor solução seria na adoção de um sistema PACS, visto que é um servidor voltado para transmissão e armazenamento de imagens médicas (URTIGA *et al*., 2004).

PACS (do inglês, Picture Archiving and Communication System) é um sistema de arquivamento, como mencionado anteriormente, e comunicação, voltado para o diagnóstico por imagem que permite o pronto acesso às imagens médicas (MARQUES; SALOMÃO, 2009). Em um sistema como esse, as imagens são armazenadas em formato DICOM (do inglês, Digital Imaging and Communications in Medicene), que tem como objetivo promover a comunicação de imagens digitais, fazer com que os diversos fabricantes de aparelhos gerem as imagens em um formato único, entre outros. Uma das desvantagens do sistema PACS é não armazenar vídeos, sendo necessária a conversão em vários frames de DICOM.

Logo, como o aplicativo grava exames em formato WebM, precisava convertê-lo para imagens DICOM a fim de fazer a integração com o sistema de arquivamento. A primeira tentativa foi com o ffmpeg, que é um programa em linha de comando utilizado para manipular vídeos. Pegava-se o vídeo e se tentava, com o Child Process (módulo do node que permite rodar comandos do terminal) executar o

programa. O comando era executado com sucesso, entretanto, o ffmpeg não convertia corretamente para DICOM.

A segunda tentativa foi usar o dcm4che, que é uma caixa de ferramentas para trabalhar com DICOM (DCM4CHEE.ORG, 2016). Uma dessas ferramentas fornece um arquivo na linguagem java que permite converter imagens no formato JPG para DICOM. Nessa solução, ainda iria precisar do ffmpeg para converter o vídeo WebM em imagens JPG, já que o dcm4che não disponibiliza essa funcionalidade. Mais uma vez, fez-se uso do Child Process do node para executar o programa em linha de comando.

Entretanto, essa solução ainda não parecia viável. Com um vídeo de 60 segundos, mais ou menos 30 frames por segundo (mil e oitocentas imagens no formato JPG), o computador ficou travado por mais de 15 min. A fim de encontrar uma solução para esse problema de desempenho, tentou-se criar um programa na linguagem C, com sua primeira versão serial, que conseguisse percorrer o diretório no qual estariam as imagens convertidas pelo ffmpeg, e executar o comando do dcm4che para convertê-las em DICOM.

Assim, o programa serial criado mostrou um desempenho melhor que aquele usando Child Process, isso porque o Node.js estava criando centenas de subprocessos para cada imagem a ser convertida, fazendo com que o sistema operacional gastasse tempo em criar e apagar esses processos. Em suma, o computador não travou e o tempo foi reduzido pela metade.

Tendo em vista a melhoria significativa de utilizar o programa serial em C, imaginou-se que um em paralelo teria uma resposta superior, criando-se *threads* para as diversas operações. Primeiramente, o código deveria estar adequado para aplicar uma ferramenta de programação paralela, que vinha a ser o OpenMP por ser para memória compartilhada. Nessa ferramenta, existem diretivas que indicam o ponto em que o programa deve ser paralelo. No entanto, o OpenMP não fornece paralelização de loops *Whiles*. Logo, no código, criou-se uma função apenas para varrer o diretório, contar e armazenar o nome das imagens, a fim de se criar um *for* e aplicar o *system* a cada imagem. Para finalizar, utilizando a diretiva *pragma omp parallel for*, o código, aparentemente, apresentou um tempo reduzido daquele do programa serial, apesar de ser um código *i/o bound*, ou seja, que apresenta diversas operações de Entrada e Saída, como ler e retornar arquivos de um diretório.

INSERINDO INFORMAÇÕES DO PACIENTE NO DICOM

Com a parte de converter para DICOM funcionando, a preocupação foi inserir as informações do paciente nas imagens convertidas que são exigidas neste formato. Para tanto, no ato da conversão, o dcm4che possibilita que seja passado um xml com as informações. O node disponibiliza diversos módulos, um deles, é o xmlbuilder. Com ele, foi possível criar o modelo do xml a ser passado ao comando dentro do arquivo em C.

As informações do paciente para a construção desse arquivo são obtidas por um formulário dentro da aplicação, Figura 5. Nele, apresentam-se os campos com o nome do paciente, idade, ano de nascimento, sexo, registro, modalidade, anestesia, entre outros. Toda essa etapa foi conseguida com uma biblioteca em javaScript, jquery, para pegar as informações de cada campo e guardar em uma lista para ser usado na função do xmlbuilder.

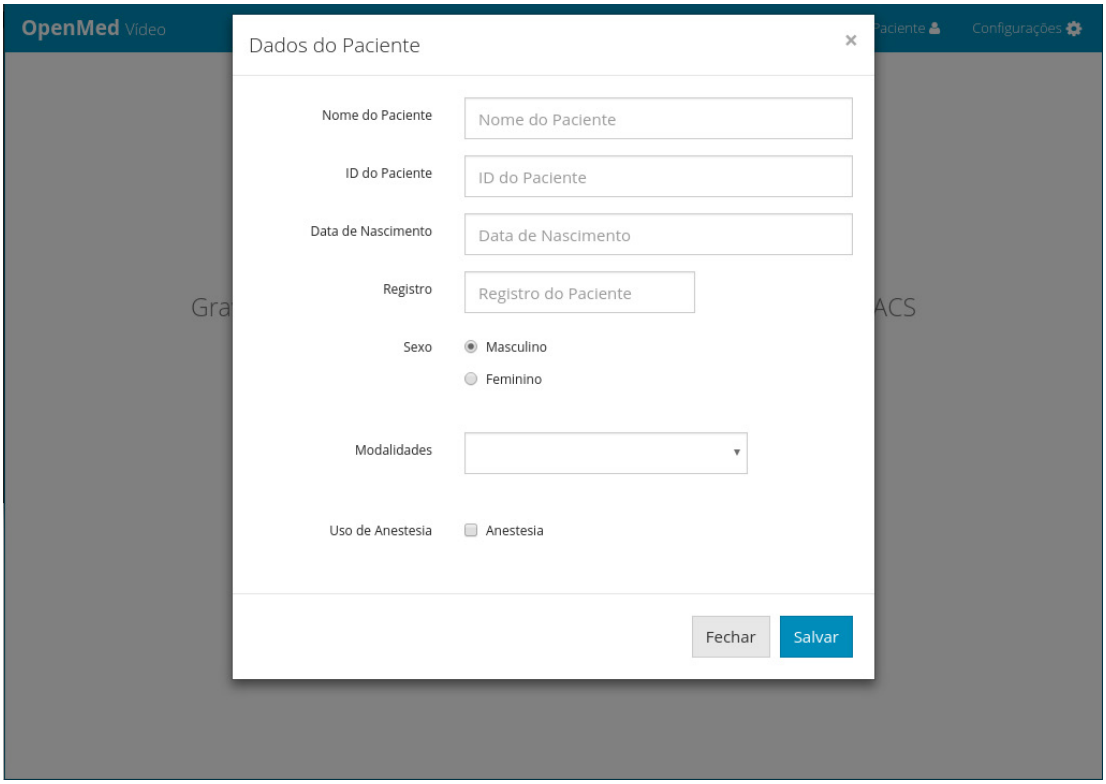

Figura 5 – Formulário para obtenção das informações do paciente e do exame.
Fonte: Autoria própria (2018).

LAUDAGEM

O laudo, nos mais diversos hospitais, dá-se, na maior parte das vezes, de forma manual, isto é, escrevendo em papéis que podem ser perdidos e que se desgastam com o tempo. O OpenMed Vídeo procurou dar uma opção para o usuário de ter esse laudo digital em que nem o tempo nem o ambiente podem estragá-lo, entre outras vantagens.

É possível visualizar, na Figura 14, o formulário desenvolvido, que será o laudo propriamente dito com a imagem editada. Ao concluir as observações e informações do paciente no formulário, o médico tem a opção de criar um PDF, com um botão no canto inferior direito da página.

A criação do PDF é feita utilizando-se dos módulos *File System* (fs), que permite a manipulação de arquivos de diretórios no sistema operacional, e *Operating System* (os), que, como o próprio nome sugere, é um módulo que fornece métodos úteis relacionados ao sistema operacional, como, por exemplo: path, que promove uma interface para trabalhar com arquivos e diretórios; e *shell*, que fornece funções relacionadas à integração com o Desktop (ELECTRON, 2016). Porém, a propriedade mais importante dessa etapa é o *webContents* do *BrowserWindow*, que permite, a partir do método *printToPdf*, gerar o arquivo pdf e chamar um *callback* para escrever, utilizando-se do fs, o arquivo no sistema de diretórios. Finalizando o assunto, o arquivo ainda é aberto automaticamente, por meio do método *openExternal*(), do módulo *shell*.

INTEGRAÇÃO COM UM SISTEMA PACS

Com o vídeo WebM sendo convertido para frames DICOM e por meio do preenchimento dos dados por um formulário, Figura 5, permitindo a inserção dos dados nas imagens, pôde-se facilmente desenvolver a etapa em destaque. A integração com um sistema PACS, como dito em mais de

uma seção, é um dos objetivos mais importantes do trabalho, pois permitirá que os exames gravados sejam armazenados em um servidor, solucionando o problema de armazenamento local dos computadores do setor de otorrinolaringoscopia.

Vale ressaltar que o DICOM fornece diversos serviços que permitem executar determinada tarefa a um objeto, como, por exemplo, armazená-lo ou mostrá-lo. Um serviço também é conhecido como Classe de Serviço por causa da natureza orientada a objetos do seu modelo de informação (MARQUES; SALOMÃO, 2009). Destacando-se ainda que, quando um dispositivo fornece um serviço, ele pertence a uma Classe Provedora de Serviço (*Service Class Provider*, SCP) e, no momento que solicita, pertence a uma Classe de Usuário de Serviço (*Service Class User*, SCU). Para o desenvolvimento da integração, o aplicativo assumiu a condição de SCU, uma vez que necessita armazenar as imagens; e o servidor, a função de SCP, já que fornece uma funcionalidade. Para isso, cada imagem convertida é enviada por meio do serviço DICOM C-STORE.

RESULTADOS

O *OpenMed* Vídeo passou por avaliações dos profissionais do setor de otorrinolaringoscopia, aos quais foram apresentadas todas as funcionalidades do sistema, podendo opinar e fazer sugestões. Ao longo do capítulo, pode-se perceber que as funções requeridas pela equipe médica foram atendidas. Seguindo o mesmo contexto, as avaliações não pararam por aí já que, em semanas consecutivas, foram realizadas reuniões com a equipe do OpenPacs, (MARQUES; SALOMÃO, 2009), a quem os médicos recorreram quando surgiu a necessidade de armazenamento e de uma aplicação mais intuitiva, objetivando analisar se os requisitos estavam sendo atendidos e como estava ocorrendo a evolução do projeto em evidência.

É importante ressaltar que não foi possível realizar alguns testes, ou seja, algum exame real realizado pela aplicação, uma vez que a placa de captura de vídeo, utilizada no setor médico, mencionada na Introdução, não era compatível com o sistema operacional *Linux*, no qual todo o aplicativo foi desenvolvido. Uma possível solução seria mudar para o sistema *Windows*.

Acredita-se que os testes, que seriam realizados no setor com algum paciente, iriam acrescentar muito a este trabalho. Apesar disso, o que será mostrado ao longo do capítulo, estará de acordo com o objetivo deste estudo, que é mostrar que o OpenMed Vídeo atendeu bem todos os requisitos que no início foram propostos. Isso pode ser comprovado a seguir, visto que serão apresentados todos os componentes do sistema, as telas, assim como o resultados de suas funções, como, por exemplo, a conversão realizada e a integração com o PACS.

Tela Inicial

A tela inicial, mostrada na Figura 6, a seguir, propõe quatro páginas inicialmente. A primeira tela, quando clicada no botão "Gravar Exame", oferecerá ao usuário a possibilidade de gravar o exame no formato WebM e salvar em qualquer lugar do seu computador. O "Recortar Exame" possibilitará, em uma tela que será mostrada mais adiante, figura 9, o corte inicial e final do vídeo gravado. Por sua vez, o "Abrir e Editar", em uma outra página, figura 11, permite que o vídeo seja aberto e editado. Para finalizar, observando-se a imagem a seguir, figura 6, vê-se o botão "Converter", que proporcionará a conversão do vídeo em imagens DICOM e a integração com o sistema PACS.

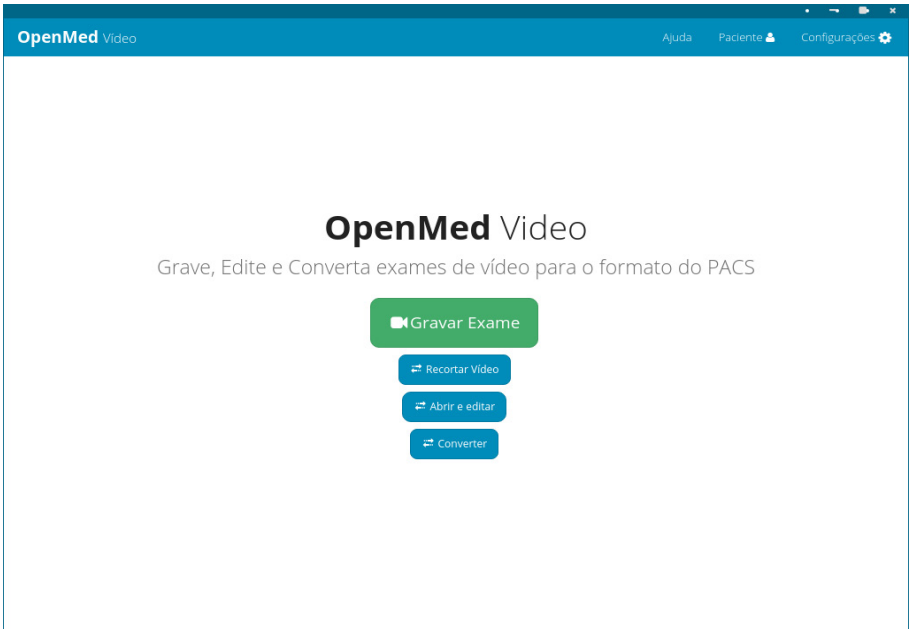

Figura 6 – Tela inicial do *OpenMed* Vídeo.
Fonte: Autoria própria (2018).

Gravar Exame

A Figura 7 possibilita a observação da página em que os vídeos são gravados. Com a finalidade de buscar uma aplicação intuitiva, percebe-se que existe apenas um botão. Apesar disso, o aplicativo grava, marca o tempo no canto inferior esquerdo e salva.

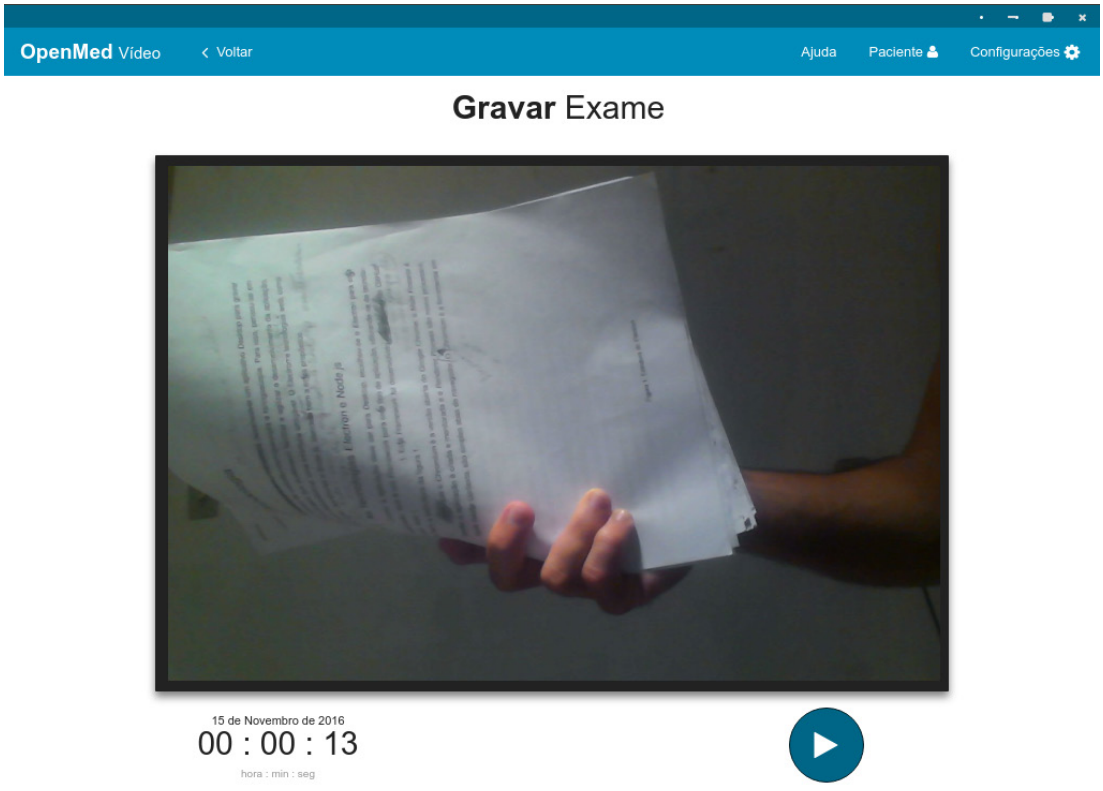

Figura 7 – Página responsável por gravar exame.
Fonte: Autoria própria (2018).

Recortar Exame

A seguir, serão apresentadas as figuras referentes à etapa de recortar o exame. A primeira, Figura 8, mostra a página em que o vídeo será recortado tal como os botões para iniciar, adiantar ou atrasar, recortar no início ou no fim, um de recomeçar e o de salvar a edição. As duas que seguem (Figuras 9 e 10) mostram as janelas para a obtenção dos tempos na realização dos cortes de começo e final, respectivamente.

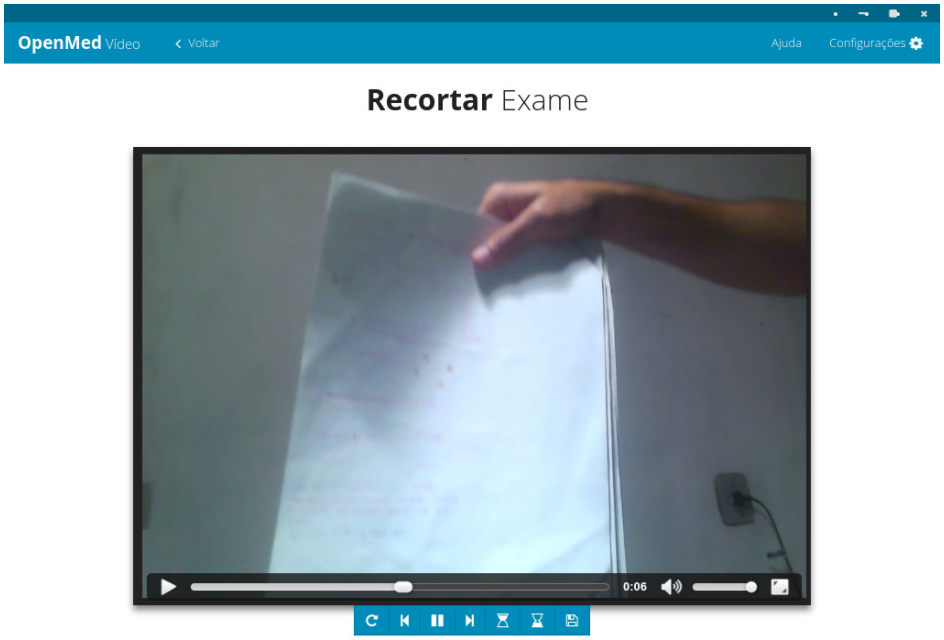

Figura 8 – Página inicial para recortar vídeo.
Fonte: Autoria própria (2018).

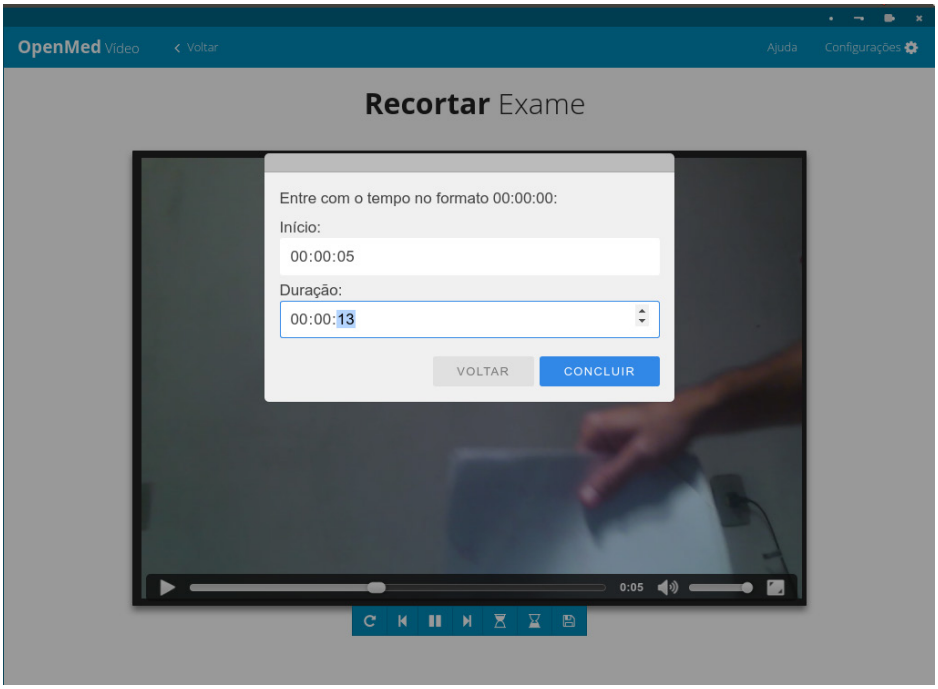

Figura 9 – Janela para se obter os tempos de corte.
Fonte: Autoria própria (2018).

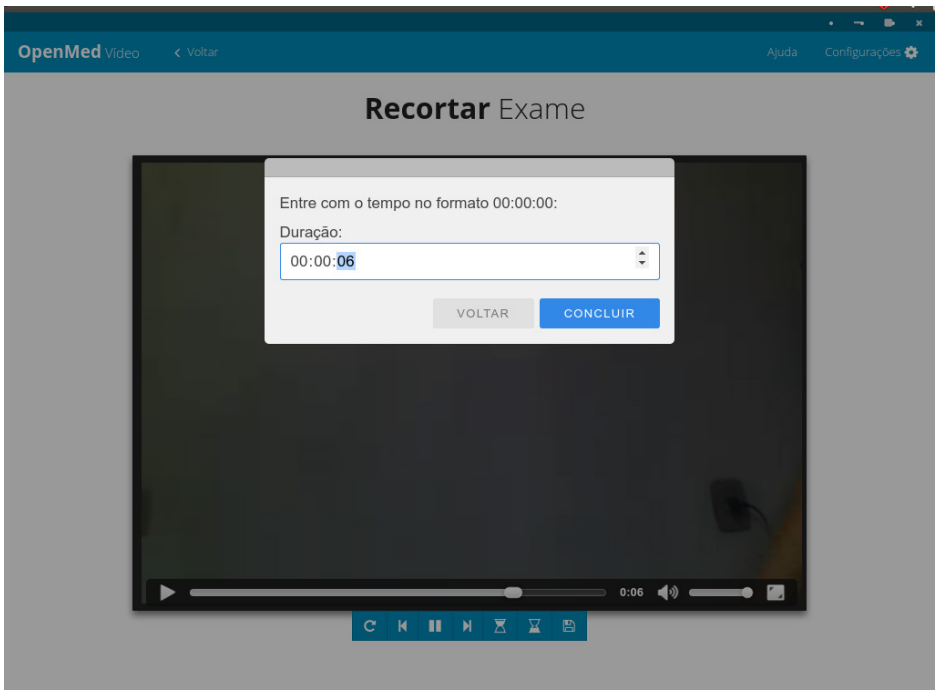

Figura 10 – Janela para obtenção do tempo do fim.
Fonte: Autoria própria (2018).

Percebe-se que, para retirar parte do fim do vídeo, é necessário apenas que o usuário informe até onde ele quer que o vídeo vá, ou seja, somente a duração final.

Abrir e Editar

Um dos requisitos mínimos do sistema era que apresentasse, de forma intuitiva, uma possibilidade de editar as imagens dos exames. A Figura 11, a seguir, ilustra um vídeo sendo carregado e apenas um botão para capturar a foto sendo exibido.

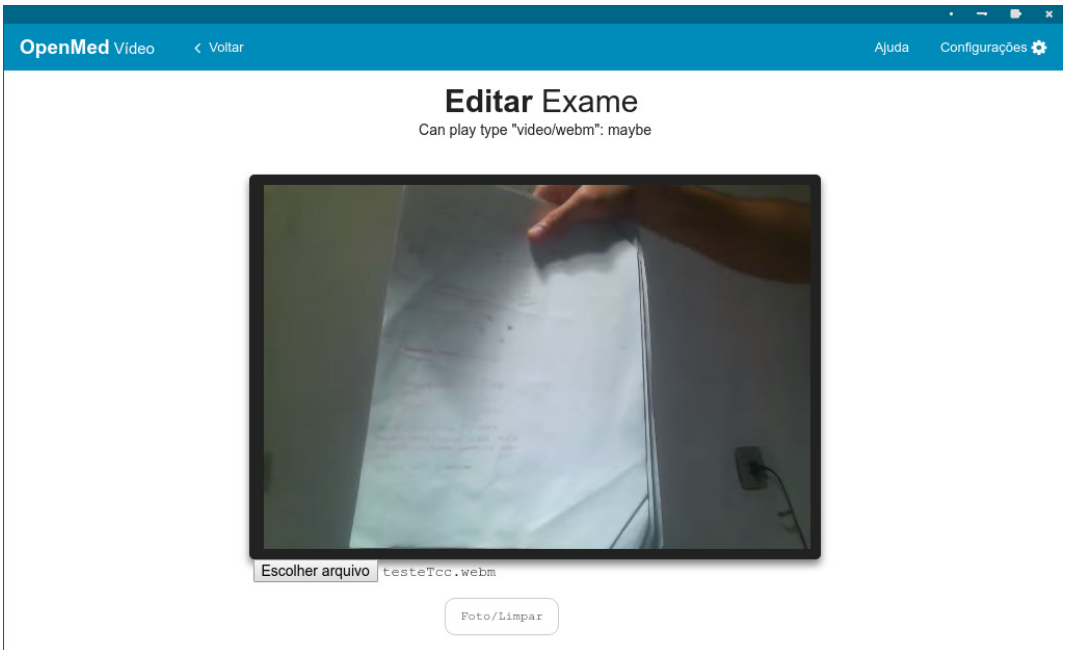

Figura 11 – Carregamento do vídeo para ser editado.
Fonte: Autoria própria (2018).

No momento do clique, outros três botões aparecem, assim como a foto correspondente no dado instante, como pode ser observado na Figura 12. Além disso, pode-se perceber que há duas imagens, sendo a de cima o vídeo pausado e a de baixo a foto retirada.

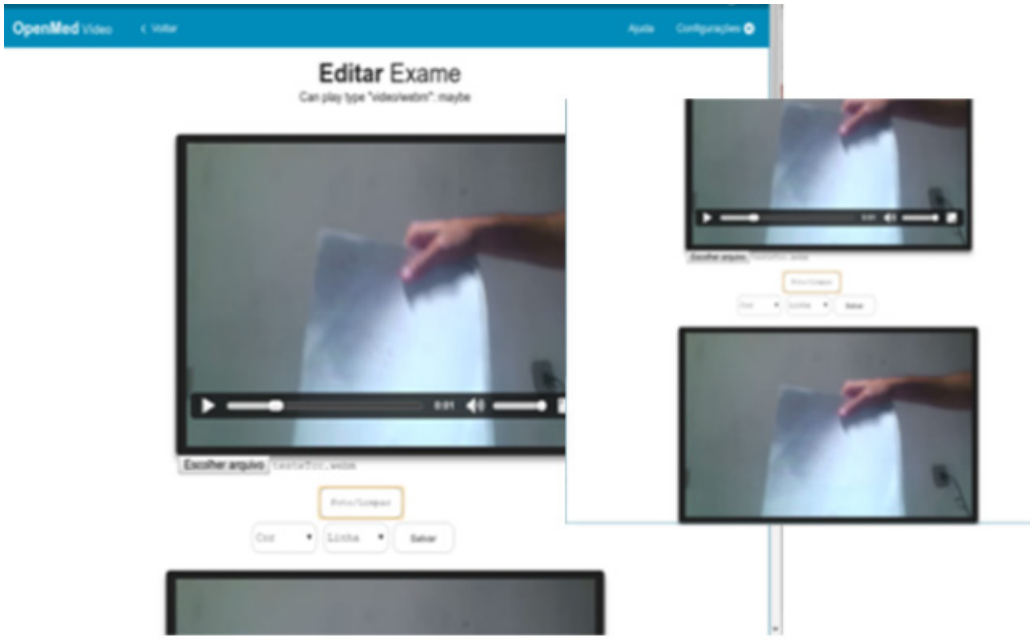

Figura 12 – Momento em que o botão é clicado e a foto é retirada.
Fonte: Autoria própria (2018).

Na foto retirada do vídeo, existe a possibilidade de realizar desenhos e colocar textos (Figura 13) finalizando as funções de edição dos exames.

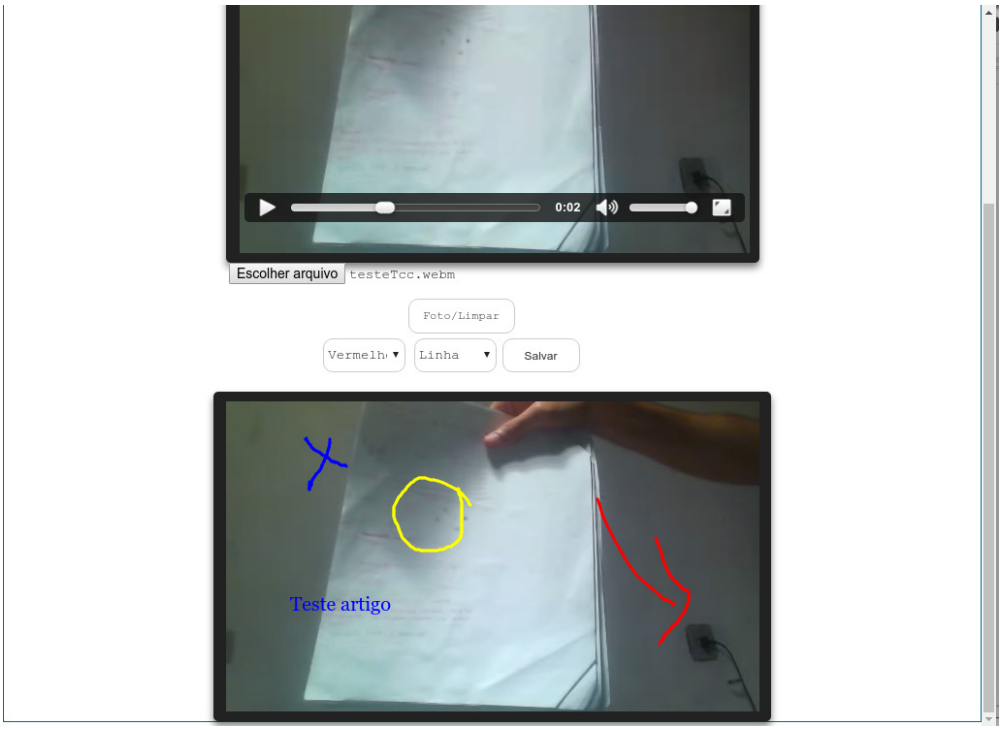

Figura 13 - Edição na imagem por meio de desenhos e textos.
Fonte: Autoria própria (2018).

Laudagem

Outro objetivo do *OpenMed* Vídeo era que laudos digitais pudessem ser realizados no setor de otorrinolaringologia. Nesse sentido, a ferramenta possibilita a laudagem digital, após a edição do exame, clicando apenas no botão "Salvar". Com isso, uma nova página é aberta (Figura 14, junção de dois *prints*). Nela, o profissional de saúde preenche os campos com o nome do paciente, idade, data, sexo e faz algumas observações.

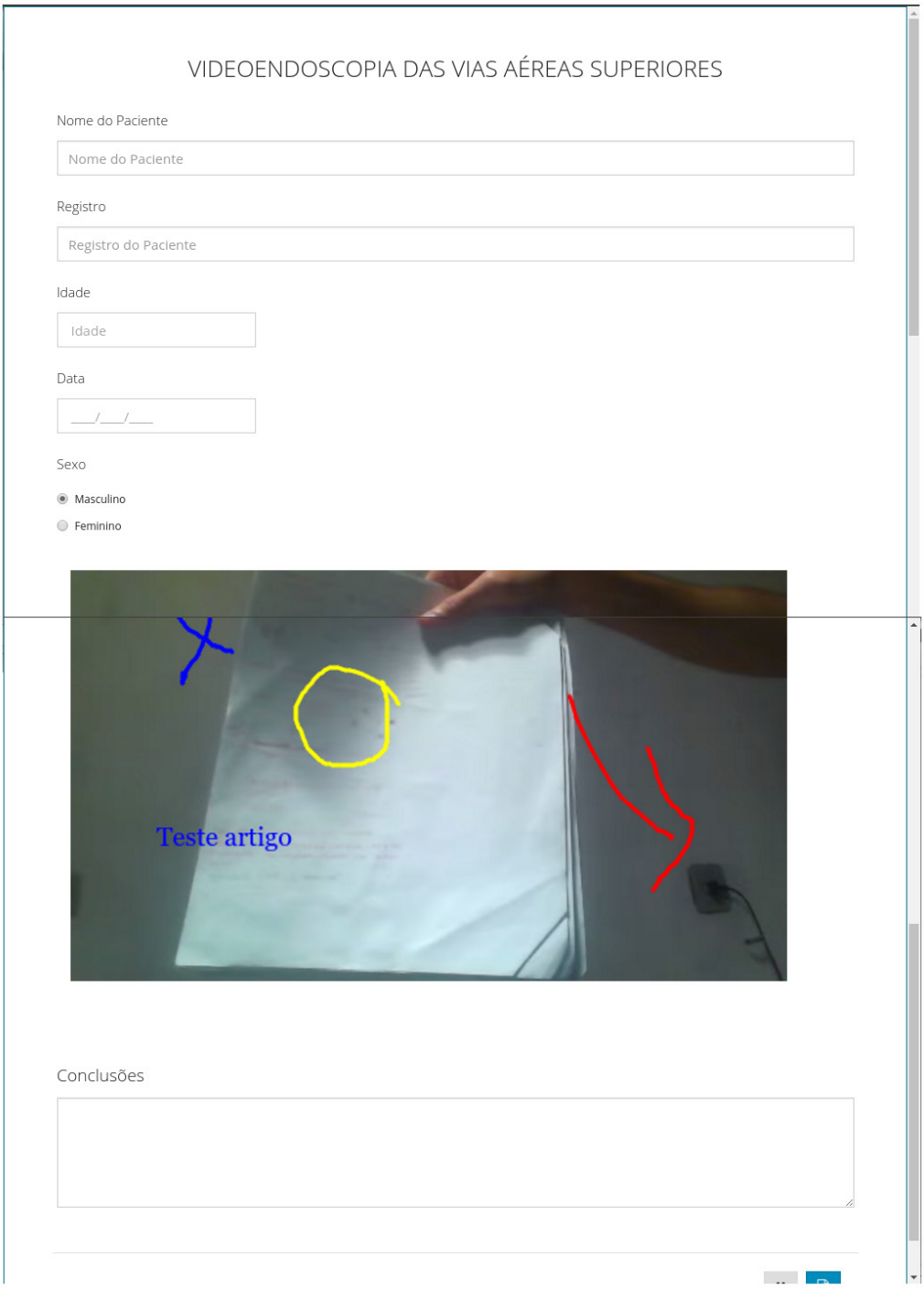

Figura 14 – Laudo digital realizado pelo *OpenMed* Vídeo.
Fonte: Autoria própria (2018).

Converter

A apresentação, que segue em forma de figuras, é referente ao objetivo mais importante do *OpenMed* Vídeo, a integração com o PACS, que consiste na conversão do vídeo WebM em *frames* JPG, JPG em DICOM e, por fim, no envio das imagens desse último formato ao servidor DCM4CHEE.

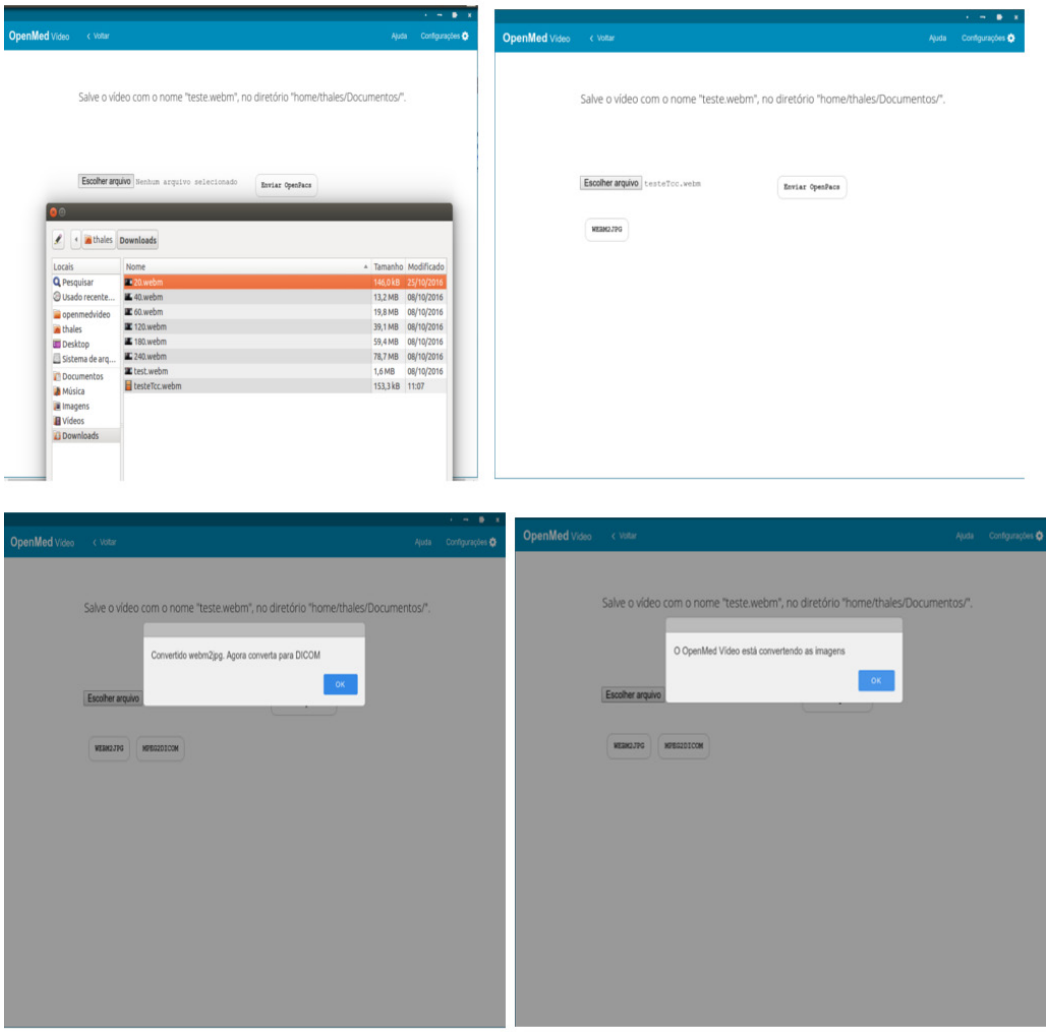

Figura 15 – Telas de conversão WebM para DICOM.
Fonte: Autoria própria (2018).

A Figura 15 ilustra as etapas realizadas para conversão do vídeo WebM em DICOM. Percebe-se que a imagem é formada por outras quatro, cada uma representando um momento distinto, a saber:

- a escolha do vídeo a ser convertido (canto superior esquerdo);
- o surgimento de um terceiro botão, logo abaixo do "Escolher arquivo", responsável por converter o vídeo em frames JPG (canto superior direito);
- a realização da operação anterior bem sucedida além do aparecimento do botão responsável por converter as imagens JPG em DICOM (canto inferior esquerdo);
- mensagem informando que as imagens foram transformadas em DICOM com sucesso (canto inferior direito);

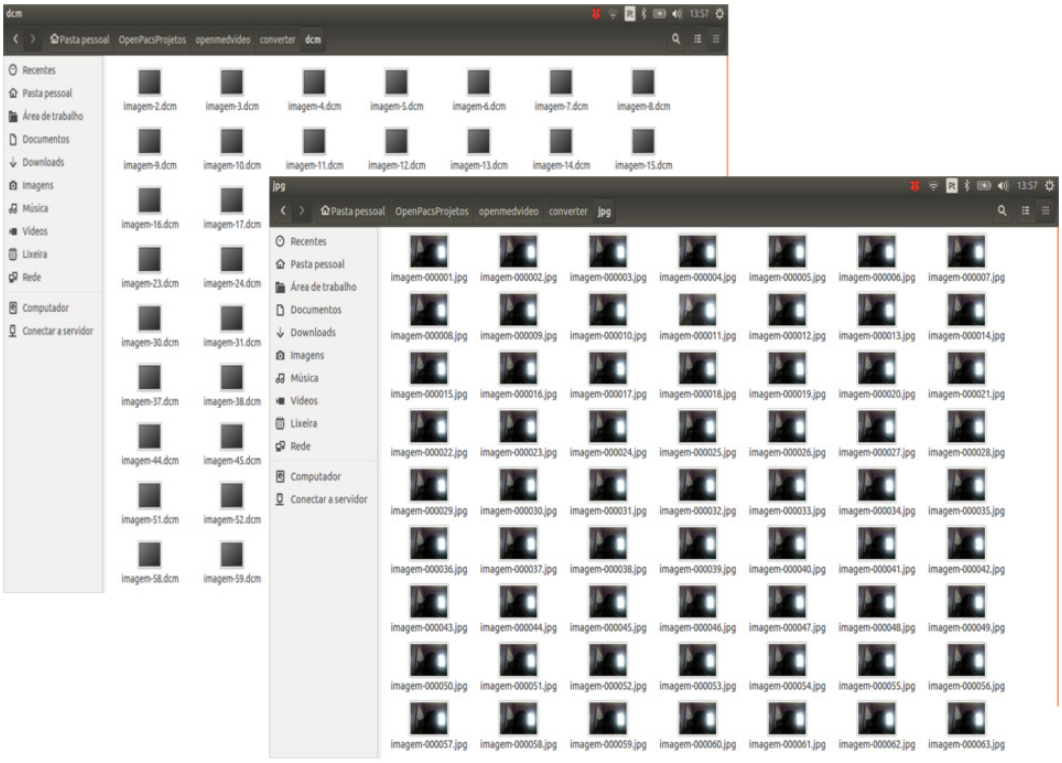

Figura 16 – Imagens JPG e DCM.
Fonte: Autoria própria (2018).

Já na Figura 16, observam-se as imagens, DCM e JPG, em diretórios diferentes, comprovando que as etapas anteriores foram realizadas com êxito.

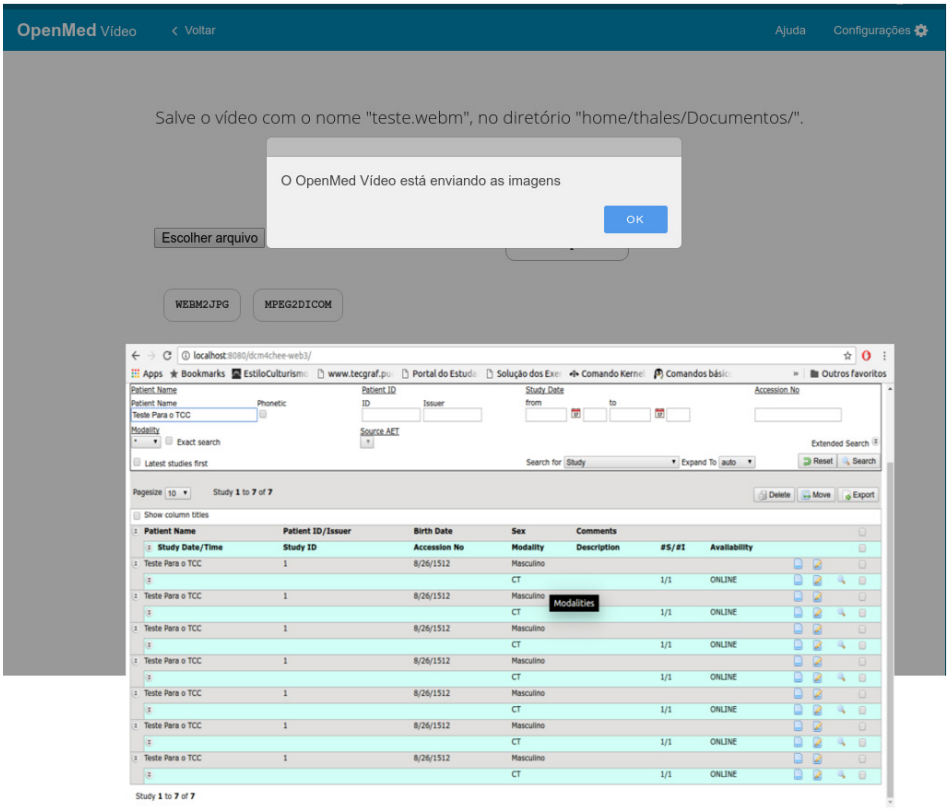

Figura 17 – Imagens sendo enviadas para o sistema PACS.
Fonte: Autoria própria (2018).

E para finalizar, a Figura 17 destaca o envio e o recebimento das imagens ao servidor PACS.

DISCUSSÕES

Atualmente, o vídeo do exame não está podendo ser gravado, pois não há compatibilidade da placa de captura utilizada no setor com o sistema operacional *Linux,* onde o *software* foi desenvolvido, como mencionado anteriormente. Pode-se dizer que o intuito inicial era utilizar a plataforma desenvolvida, levar para o ambulatório de otorrinolaringologia e, por meio de algum paciente voluntário, testar todas as funções do sistema. No entanto, diante dessa dificuldade, a função de gravar exame foi impossibilitada de ser realizada. Por conseguinte, para que os demais testes pudessem ser realizados, foi necessário o fornecimento de um exame já gravado anteriormente pela ferramenta utilizada no setor.

No ambiente pesquisado, o *software* que os profissionais utilizam para gravar e editar os exames é o *PowerDirector*, fornecido pelo fabricante da placa de captura, como já foi dito em outras oportunidades no trabalho em evidência. Esse aplicativo grava os exames em MPEG, formato no qual o *OpenMed* Vídeo não trabalha, pois ele grava os vídeos em WebM e todas as suas funções são baseadas em ler esse formato.

Seguindo esse contexto, foi necessário que o vídeo fornecido em MPEG fosse convertido ao formato padrão do *OpenMed*, o WebM para que se pudesse examinar as demais aplicabilidades. A seguir, na Figura 18, é mostrado o exame original e o recortado na tela de "Recortar Exame".

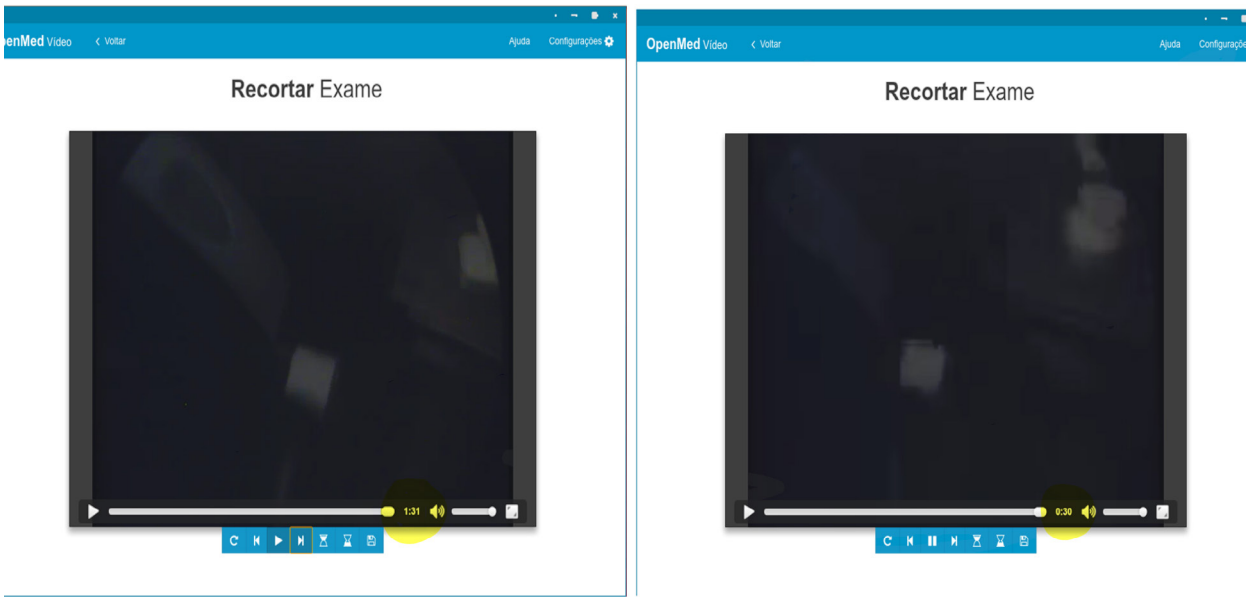

Figura 18 – Corte em um exame real.
Fonte: Autoria própria (2018).

Pode-se perceber, pela imagem acima, lado esquerdo, que o vídeo original tinha 1 minuto e 31 segundos de duração e que, após ser editado, ficou com 30 segundos, retirando alguma informação não relevante do seu início e do fim. O teste em questão foi realizado pela residente, Ana Carolina Fernandes, que pôde identificar o ponto em que o vídeo deveria ser recortado, e que também editou e emitiu o laudo (Figura 19) com as informações reais do exame em destaque.

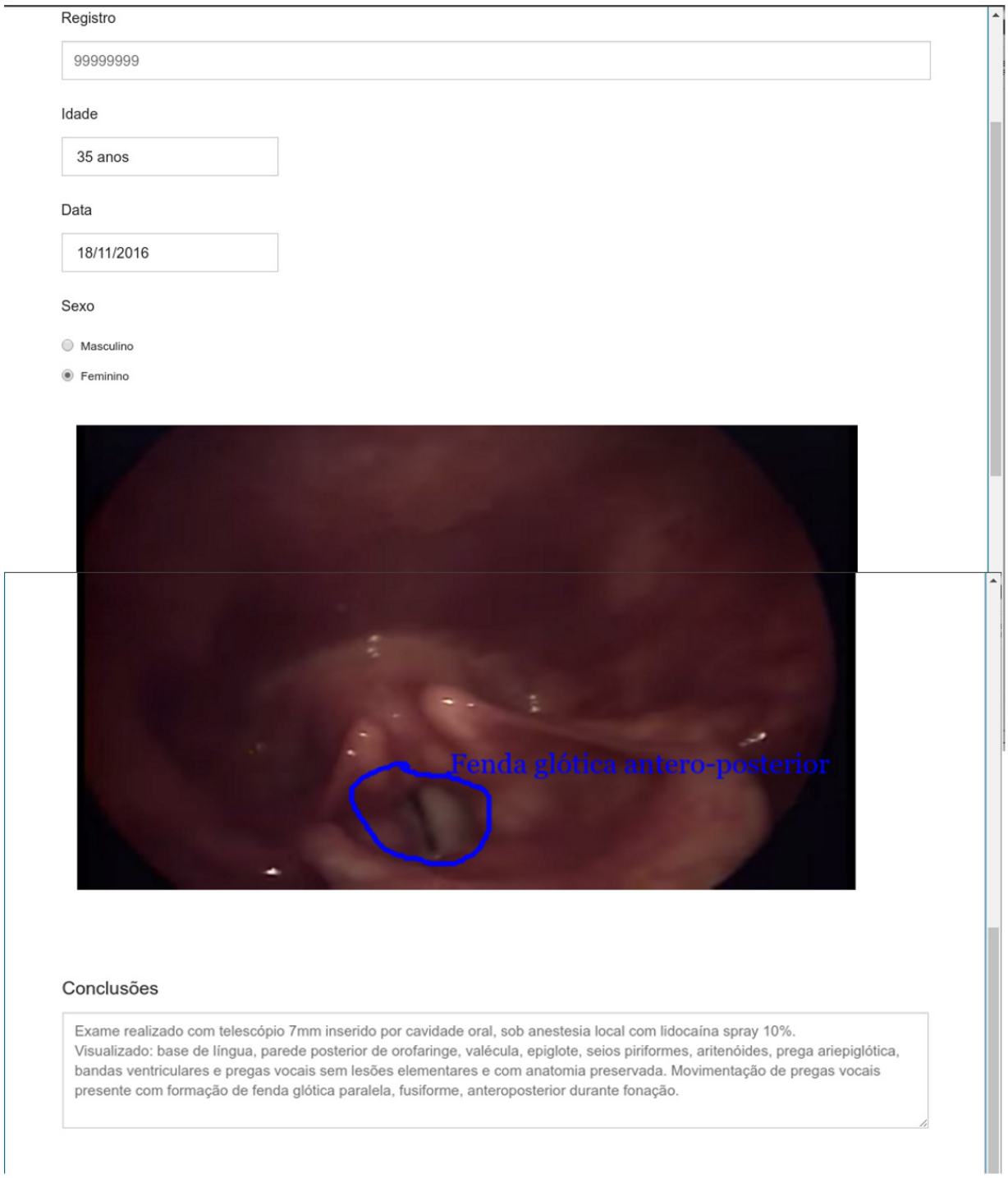

Figura 19 – Informações reais preenchidas no laudo do *OpenMed*.
Fonte: Autoria própria (2018).

Percebe-se que a imagem editada está posicionada no centro de um formulário, a saber, o laudo digital, em que constam os campos preenchidos bem como as conclusões editadas pela médica. Nele, pode-se visualizar, durante a movimentação das pregas vocais, no momento da edição, a presença de uma fenda glótica, paralela, fusiforme, antero-posterior durante a fonação da paciente.

A médica também percebeu que a imagem perdeu qualidade, sendo isso de extrema importância para o seu trabalho, uma vez que, quanto maior é a qualidade, mais acurácia se tem no exame. Apesar de não ter havido qualquer teste de comparação entre esses dois formatos, sabe-se que o MPEG é um pouco superior em relação a essa característica perdida. Vale salientar que o formato WebM foi escolhido por ser

um dos padrões suportados pelo HTML5 (uma das tecnologias utilizadas para o desenvolvimento da aplicação) e poder ser representado por um tamanho menor.

CONCLUSÕES

Diante das necessidades apresentadas pelos otorrinolaringologistas do Hospital Universitário Onofre Lopes, concluiu-se que seria de extrema importância a criação de uma ferramenta computacional que pudesse solucionar o problema de armazenamento dos exames gravados do setor e que conseguisse ser de fácil utilização ao mesmo tempo, de modo que esses profissionais de saúde não desperdiçassem tempo para editar as imagens dos vídeos e emitir os laudos médicos. Partindo dessa necessidade, o *OpenMed* Vídeo foi desenvolvido, em parceria com o Laboratório de Inovação Tecnológica em Saúde, ao qual a equipe médica recorreu quando da carência desse produto. Seu desenvolvimento possibilitou que os exames gravados pudessem ser armazenados em forma de imagens DICOM em um sistema PACS, visando obter um armazenamento não local.

Além disso, por ser voltado para a problemática dos médicos, buscou ser uma ferramenta mais específica e intuitiva para o seu público-alvo. Ainda nessa conjuntura, pode-se afirmar que o aplicativo em destaque atende as características mínimas apresentadas pelos *softwares* atualmente utilizados pelos profissionais do setor, que são, normalmente, aqueles dos fabricantes da placa de captura de vídeo, uma vez que grava e edita os exames.

Sabe-se também que o *OpenMed* Vídeo não pôde ser testado por completo por causa da incompatibilidade da placa de captura ao sistema operacional *Linux*. Apesar disso, as demais funcionalidades foram postas à prova e algumas dificuldades puderam aparecer. A que chamou maior atenção foi a perda de qualidade do vídeo, transformado de MPEG para WebM. Para isso, propõe-se um estudo mais apurado no que se refere a tamanho *versus* qualidade, projetando uma possível troca do formato atual do aplicativo para o MPEG.

Sugere-se, ainda, uma versão compatível para *Windows*, de modo que mais testes possam ser realizados no setor de otorrinolaringologia, a fim de obter, por exemplo, um laudo com quatro imagens proposto há pouco tempo pelos profissionais do serviço. Também é possível propor melhorias no *design* e na usabilidade, acréscimo de formas geométricas pré-definidas para auxiliar na edição e o incremento de outras funções que venham a surgir, conforme a necessidade de seus usuários. Para isso, é necessário que o projeto em evidência venha a ser apropriado com êxito e com larga utilidade.

REFERÊNCIAS

COMSYSTO. **Building a desktop application with eletcron**. Disponível em: https://comsysto.com/blog-post/building-a-desktop-applications-with-electron. Acesso em: 3 nov. 2016.

DANTAS, Marcel da Câmara Ribeiro. **Sistema de Telemonitoramento para pacientes com Esclerose Lateral Amiotrófica**. 2016. 44 f. Trabalho de Conclusão de Curso (Graduação em Engenharia da Computação) – Departamento de Engenharia de Computação e Automação, Universidade Federal do Rio Grande do Norte, Natal, 2016.

GITHUB. **Electron**. Disponível em: https://electron.atom.io/. Acesso em: 19 jul. 2016.

HUBSPOT. **Vex Documentation**. Disponível em: https://github.hubspot.com/vex/docs/welcome/. Acesso em: 19 jul. 2016.

IEEE STD. **The Open Group Base Specifications Issue 6**. Disponível em: https://pubs.opengroup.org/onlinepubs/9699919799/. Acesso em: 25 nov. 2016.

MACÊDO FIRMINO, Sheila Pereira; VALENTIM, Ricardo Valentin. PACS – Sistema de Comunicação e Arquivamento de Imagens Médica: Visão Introdutória e Usabilidade no Sistema de Saúde Brasileiro. In: CONGRESSO NORTE-NORDESTE DE PESQUISA E INOVAÇÃO. 1., 2012. **Anais** [...]. Natal, 2012. p. 2.

MARQUES, Paulo Mazzoncini de Azevedo; SALOMÃO, Samuel Covas. PACS: Sistemas de Arquivamento e Distribuição de Imagens. **Revista Brasileira de Física Médica**, v. 3, n. 1, p. 131-139, 2009.

MJR, S. **SLC – Sistemas de Laudos e Captura**. 2016. Disponível em: http://www.marjr.com.br/sistemas_slc.asp/. Acesso em: 3 nov. 2016.

MOREIRA, Rafael Henrique. **Módulos em Node.js**. Disponível em: https://nodebr.com/modulos-em-nodejs/. Acesso em: 25 nov. 2016.

NODE.JS FOUNDATION. **Node.js**. Disponível em: https://nodejs.org/en/. Acesso em: 19 jul. 2016.

NODEJS.ORG. **Node.js v7.2.0 Documentation**. Disponível em: https://nodejs.org/api/fs.html. Acesso em: 25 nov. 2016.

NODEJS.ORG. **Node.js v7.2.0 Documentation**. Disponível em: https://nodejs.org/api/child_process.html. Acesso em: 3 nov. 2016.

PENHA, Dulcineia Oliveira da; CORREA, João Batista Torres; MARTINS, Carlos Augusto Paiva S. **Análise Comparativa do Uso de Multi-Thread e OpenMP Aplicados a Operações de Convolução de Imagem**. 2002. Disponível em: http://www.lbd.dcc.ufmg.br/colecoes/wscad/2002/0018.pdf. Acesso em: 15 abr. 2019.

PILCHER, Otavio B.; MAAHS, Gerson Schulz; KUHL Gabriel. **Rotinas em otorrinolaringologia**. Porto Alegre: Artmed, 2014.

STRONGLOOP. **What Makes Node.js Faster Than Java?** Disponível em: https://strongloop.com/strongblog/node-js-is-faster-than-java/. Acesso em: 3 nov. 2016.

TOUCHHEALTH. **Software para quem cuida da saúde**. 2016. Disponível em: http://touchhealth.com.br/. Acesso em: 8 nov. 2016.

URTIGA, Keylla Sá *et al*. **Mini-WEBPACS**: Sistema Compacto para Armazenamento e Distribuição de Imagens Médicas em Ambientes Clínico-Hospitalares. 2004. Disponível em: http://telemedicina.unifesp.br/pub/sbis/CBIS2004/trabalhos/arquivos/156.pdf. Acesso em: 15 abr. 2019.

VARELLA, Drauzio. **Exame de Nasofibrolaringoscopia**. Disponível em: https://drauziovarella.com.br/videos-3/nasofibrolaringoscopia/. Acesso em: 17 nov. 2016.

WEBRTC. **World Wide Web**. Disponível em: https://webrtc.org/. Acesso em: 19 ago. 2016.