# *Framework* Multi-LIBRAS para integração e colaboração de soluções para deficientes auditivos

Luciana Pereira Oliveira[1], Joffily Ferreira dos Santos[2], Marianna Soares Veríssimo[3]

*[1] oliveira.ifpb@gmail.com. [2] joffilyferreira@gmail.com. [3] marianna1204@gmail.com. Instituto Federal da Paraíba (IFPB), Campus João Pessoa.*

## RESUMO

Vem crescendo o número de soluções para pessoas com deficiência auditiva. Por exemplo, vêm surgindo diversas soluções de tradutores de Português para LIBRAS para facilitar a comunicação entre ouvintes e surdos. Por outro lado, muitos projetos de pesquisa estão descontinuados, não sendo possível executar os aplicativos desenvolvidos e muito menos repetir experimentos descritos nos artigos. Então, como os pesquisadores poderiam trabalhar de maneira colaborativa a construção de novos aplicativos e melhorias dos softwares existentes? Seria possível distintos grupos de pesquisa criarem um dicionário de Português e LIBRAS aberto? Objetivando-se esse ambiente colaborativo, definiu-se o *framework* Multi-LIBRAS, que oferece flexibilidade e extensibilidade para integrar qualquer solução para LIBRAS e algoritmos de avaliação. O artigo irá apresentar uma visão geral, as camadas externas e internas do *framework* bem como exemplificar o uso do Multi-LIBRAS, que dá suporte a soluções descritas em qualquer linguagem de programação.

Palavras-chave: LIBRAS. *Framework*. Língua de Sinais. Tradutor.

## ABSTRACT

*Has increased the number of solutions for people with hearing impairment. For example, there are several solutions to translate Portuguese to LIBRAS. It provides a better communication between deaf and hearing. Furthermore, many research projects are discontinued, not being able to run applications developed let alone repeat experiments described in the articles. So how could researchers work collaboratively on the construction of new applications and improvements to existing software? Would it be possible to separate research groups to create an open dictionary between Portuguese and LIBRAS? Aiming to this collaborative environment, set up the Multi- LIBRAS framework to offer flexibility and extensibility to integrate any solution for LIBRAS and evaluation algorithms. The article will present the general view, outer and inner layers from framework, and exemplify the Multi-LIBRAS of using that supports the solutions described in any programming language.*

Keywords: *LIBRAS. Framework. Sign Language. Translator.*

# 1 Introdução

No Brasil, a Língua Brasileira de Sinais (LIBRAS ou LSB) é uma linguagem gestual e visual, que foi estabelecida em 2001 pela Lei 10.432/02, como língua oficial dos surdos. Isso motivou os avanços de projetos de tradutores e outros aplicativos para surdos. Por outro lado, os resultados de Oliveira e Oliveira (2015) mostram que existe um alto índice de repetições de aplicativos. Por exemplo, em um total de 14 soluções para tradutores, apenas 30% da quantidade dessas ferramentas estão disponíveis. Também se verificou que, quando os projetos são descontinuados, não é possível executar as soluções desenvolvidas e muito menos repetir experimentos descritos nos artigos. Existem projetos que são descontinuados e disponibilizam o código ou executável da aplicação, mas são considerados aplicativos indisponíveis porque apresentam falhas no dicionário, impossibilitando o uso do software. Isso representa uma grande perda dos esforços previamente realizados.

Além disso, as soluções não se relacionam e, provavelmente, por isso, precisam de muitas melhorias, conforme se pode observar nos resultados das poucas avaliações. Por exemplo, as avaliações exploratórias de LIBRAS feitas por Pivetta, Ulbricht e Savi (2011) mostram que poucos aplicativos estão disponíveis gratuitamente. Corrêa *et al.* (2014) verificaram que os tradutores *ProDeaf*[1] e *HandTalk*[2] apresentam dicionário reduzido e, por isso, existem palavras traduzidas com datilologia (alfabeto português sinalizado), sendo necessário melhorias para uma melhor compreensão. Colling e Boscarioli (2014) e Colling (2014) verificaram que os aplicativos *Rybená*[3], *ProDeaf* e *Hand Talk* ainda precisam de melhorias para poderem ser utilizados em sala de aula com crianças surdas, mas essa pesquisa precisaria de uma maior investigação, pois utilizaram-se apenas 10 frases. Além disso, mesmo já existindo esses tradutores, foi disponibilizado, em 2014, um novo tradutor denominado de *V-Libras*[4].

Isso mostra que não existe um efetivo ambiente de interação e colaboração entre os pesquisadores em LIBRAS, principalmente, porque não existe um padrão de linguagem para a escrita dos sinais no Brasil.

Dessa forma, sabendo-se que um *framework* é uma abstração que une soluções para atingir uma funcionalidade específica ou resolver um determinado problema (MATTSSON,1996), este artigo tem o objetivo de apresentar um *framework*, para unir as soluções comerciais e acadêmicas em LIBRAS de forma flexível, denominado de Multi-LIBRAS, e, especificamente, busca responder aos os seguintes questionamentos:

- Como pesquisadores podem trabalhar de maneira colaborativa a construção de novos aplicativos e melhorias dos *softwares* existentes?
- Poderiam distintos grupos de pesquisa criarem um dicionário de Português e LIBRAS aberto, com auxílio de projetos acadêmicos e comerciais?

Esses são questionamentos aos quais buscaremos responder por meio da compreensão da especificação do *framework* Multi-LIBRAS e de um exemplo de caso de como utilizá-lo. A seção de trabalhos relacionados apresenta alguns exemplos de padrões para linguagem dos sinais e também um único *framework* na área de LIBRAS, encontrado, em termos de colaboração. Na ocasião, mostraremos por que esses trabalhos não respondem a nossos questionamentos. A seção de material e métodos apresenta a definição do procedimento metodológico da pesquisa para especificação do *framework*. Posteriormente, será apresentado o Multi-LIBRAS, em termos de visão geral, arquitetura e um exemplo de caso de uso, em que se realiza avaliação automatizada com mais de 100 mil palavras em dois tipos de tradutores disponíveis na Web. Por fim, a descrição da conclusão e possibilidades de trabalhos futuros.

# 2 Trabalhos relacionados

## 2.1 Condições gerais

O Dicionário de LIBRAS (on-line[5]) pode ser visualizado como primeiro passo de um ambiente de colaboração, onde se podem encontrar diversos aplicativos e um dicionário com 2550 palavras com correspondência em LIBRAS. Repetições, no entanto, também existem – a palavra Deus, por exemplo,

1 ProDeaf - http://www.prodeaf.net/

2 HandTalk - http://www.handtalk.me/

3 Rybená - http://www.rybena.com.br/

4 V-Libras - http://vlibrasplayer.lavid.ufpb.br/

5 Dicionário de LIBRAS - http://www.dicionariolibras.com.br

possui 3 correspondências semelhantes. Não existe um padrão de intérpretes nos aplicativos. Não há um *framework* em que se possam integrar os aplicativos ou um padrão para os disponibilizar.

A falta de padronização no dicionário de LIBRAS talvez se dê, provavelmente, pela também falta de padronização da modelagem da sinalização de maneira geral no Brasil. Conforme Quevedo, Vanzin e Ulbricht (2014), ainda não existe um padrão para escrita definido.

Enquanto isso, diversos países já adotaram o *SignWriting*[6], padrão de escrita dos sinais que foi criado em 1974, nos Estados Unidos, por Sutton (2008). De acordo com Sutton e Frost (2008), a Nicarágua, Alemanha, Estados Unidos, Canadá, Bélgica e Espanha adotaram o *SignWriting* para o ensino da escrita da linguagem dos sinais. Em relação ao Japão, Matsumoto, Kato e Ikeda (2009) propõem a ferramenta *JSPad*[7] em *SignWriting*; na Tunísia, foi proposto por Bouzid e Jemni (2014) o *Avatar TuniSigner*, para automatizar a linguagem dos sinais definida em *SigWriting*; na Grécia, foi proposto por Papadogiogarki *et al.* (2006) um sistema para gerar VRML a partir do *SignWriting*.

Mesmo com a Associação de Surdos do Brasil tendo votado em se aceitar o *SignWriting* como o padrão e existindo um dicionário com 6.799 palavras em português mapeadas em *SignWriting,*[8] no site *BankSign*, o trabalho de Silva *et al.* (2014) propõe o uso do tradutor Librol com EliS[9] (BARROS, 2008). Este é um padrão para escrita dos sinais com 94 símbolos, criado oficialmente em 2008. Adicionalmente, ainda em 2015, se discute o uso do padrão EliS, *Signwriting* e outros, de acordo com Aguiar e Chaibue (2015).

O trabalho de Fusco (2004) propõe uma modelagem de dados para padronizar a correspondência entre LIBRAS e a linguagem dos sinais em XML[10], que é uma linguagem de marcação recomendada para a criação de documentos com dados organizados hierarquicamente. Desta forma, este trabalho apresenta uma nova abordagem para permitir o armazenamento dos sinais em forma de texto ao invés de vídeo. A proposta não segue, no entanto, o padrão *SignWriting*, nem o padrão EliS.

O Quadro 1 apresenta um resumo dessas soluções relacionando os padrões utilizados e o país para o qual o trabalho direcionou a solução, sendo possível observar esta falta de padronização no Brasil.

**Quadro 1 –** Resumo de trabalhos relacionados à padronização

| Solução | Padrão | País |
|---|---|---|
| Dicionário de LIBRAS online | Nenhum | Brasil |
| JSPad | SignWriting | Japão |
| Avatar TuniSigner | SignWriting | Tunísia |
| Papadogiorgaki *et al.* (2006) | SignWriting | Grécia |
| BankSign | SigWriting | Brasil e outros |
| Librol | ELiS | Brasil |
| Fusco | Novo padrão com XML | Brasil |

*Fonte:* autoria própria.

Consequentemente, isso motiva a definição de um *framework* integrador. Este precisa ser flexível e não deve ser restrito a um único padrão para representar LIBRAS.

No Brasil, só foi encontrada uma proposta de *framework*, por Trindade (2013). Esta considera o *framework* como sendo um conjunto de classes que implementam um determinado conjunto de serviços para permitir ambientes colaborativos para LIBRAS. Essa proposta, no entanto, não específica como integrar novas soluções a este ambiente colaborativo, sendo, então, um *framework* que apenas oferece o serviço de videoconferência, com gerenciamento administrativo para incluir surdos, ouvintes e intérpretes para auxiliar a conversação em uma conferência em tempo real.

Dessa forma, existe a necessidade da definição de um *framework* para LIBRAS, altamente flexível, de forma a permitir o uso de mais de uma linguagem (*SingWrinting*, Elis e outras). Para isso, é de grande importância a consideração de *framework* defendida por Mattsson (1996) – uma abstração que une soluções para atingir uma funcionalidade específica ou resolver um determinado problema.

Adicionalmente, deve-se considerar que um *framework* deve ser especificado com definições de *frozenspots* e *hotspots* (Fontoura, 1999), para que

6 SignWriting - http://www.signwriting.org/brazil/

7 Sistema de escrita prático para línguas de sinais.

8 Bank Sign - http://www.signbank.org/

9 Escrita das línguas de sinais.

10 Linguagem de marcação.

existam pontos específicos de integração e extensibilidade da arquitetura. Os *frozenspots* são partes fixas de um *framework* e fazem chamadas indiretas ao *hotspots*. Estes últimos são partes flexíveis e extensíveis que podem ser classes abstratas ou interfaces, ou seja, componentes que necessitam ser codificados, em geral, como *plugins* que recebem mensagens dos *frozenspots*.

Pode-se verificar, então, que a construção de um *framework,* na abordagem de Mattsson (1996), poderá oferecer a possibilidade de integrar uma combinação de soluções em LIBRAS, aumentando a colaboração dos pesquisadores.

## 3 Material e métodos

Os trabalhos apresentados na seção anterior correspondem ao resultado da pesquisa exploratória, no período de maio a setembro de 2015, com levantamento bibliográfico de artigos que descrevem sobre aplicativos relacionados a LIBRAS, padrão de linguagens para escrita dos sinais e definição de *framework*.

Esse levantamento, entretanto, mostrou que nenhum trabalho é capaz de responder aos questionamentos levantados na introdução, sendo necessário um novo *framework* flexível para integrar as existentes soluções em LIBRAS bem como permitir uma maior colaboração dos projetos e fácil acoplamento de novas soluções. Para atingir esse objetivo, foi definido o *framework* Multi-LIBRAS de código aberto, suportando qualquer padrão de linguagem dos sinais e qualquer linguagem de programação. O código do Multi-LIBRAS, arquivos de configuração e tutorial estão disponíveis no sourceforge.net.

O *framework* foi construído de maneira modular, por meio de uma arquitetura composta de pontos fixos e flexíveis, de maneira que os colaboradores possam adicionar novos plugins referentes a aplicativos e analisadores de LIBRAS.

Ele foi analisado em termos de modularidade, flexibilidade e linguagem de programação. Para exemplificar a modularidade, foram construídos dois plugins para os aplicativos ProDeaf e V-Libras, demonstrando o fácil acoplamento de plugins ao *framework*. Para a flexibilidade, foram construídos e exemplificados plugins escritos em Java, C, C++ e Python. Para análise da linguagem de programação, foram realizadas medições entre a comunicação do *framework* Multi-LIBRAS e cada plugin escrito nas linguagens Java, C, C++ e Python.

## 4 *Framework* Multi-LIBRAS

### 4.1 Visão geral

Multi-LIBRAS é um *framework* composto de quatro camadas, possui código aberto e contém um conjunto de mensagens bem definidas para acoplamento de aplicativos e algoritmos para análise de soluções em LIBRAS. Cada camada do *framework* pode ser executada em apenas uma máquina ou de maneira distribuída.

Esse *framework* tem como principal finalidade, unir tecnologias de LIBRAS em uma estrutura simplificada, para que o usuário possa trabalhar de maneira colaborativa, realizar aprimoramentos globais dos aplicativos e até mesmo ampliar os atuais dicionários de LIBRAS a partir de pequenas pesquisas. A grande vantagem é que tudo isso pode ser realizado sem a necessidade de conhecimentos avançados de ambas as áreas (desenvolvimento e análises de *software*).

Isso é possível porque um usuário sem tal conhecimento utilizará componentes já desenvolvidos por dois tipos de usuários do *framework*: colaboradores e investigadores, pois a padronização das camadas e sinalização da comunicação entre os componentes permite a colaboração desses usuários.

Os colaboradores do *framework* poderão contribuir no desenvolvimento de *plugins* para ativar as soluções acadêmicas ou comerciais de aplicativos para LIBRAS.

Os investigadores poderão construir novos algoritmos para automatizar a análise de aplicativos para LIBRAS e encapsular tal solução em um *plugin* a ser acoplado ao *framework*.

Estando os *plugins* dos colaboradores e *plugins* dos investigadores acoplados ao Multi-LIBRAS, é possível realizar avaliações de um maior número de aplicativos ao mesmo tempo.

Quanto maior o número de *plugins* de algoritmo de análise de LIBRAS, maior o total de resultados das análises para os desenvolvedores dos aplicativos que estão acoplados no *framework*.

Dessa forma, cada projeto de pesquisa pode focar em uma área específica e de maneira automatizada interagir com outros projetos já acoplados ao *framework*, facilitando a continuidade do projeto como um todo.

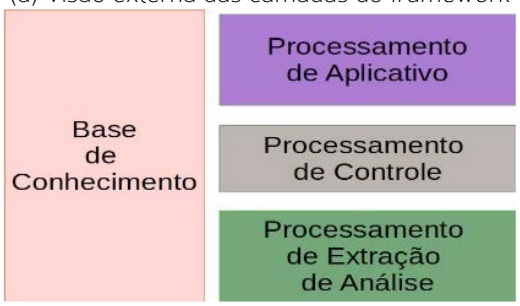
## 4.2 Camadas e estrutura interna

O Multi-LIBRAS foi projetado com a linguagem Java, já que, de acordo com CASS (2015), esta é a linguagem de programação mais utilizada, pois oferece grande flexibilidade para ser executada em qualquer sistema operacional com suporte à máquina virtual Java. Apesar do *framework* ser projetado em Java, ele não está restrito a *plugins* escritos nessa linguagem computacional. Ele oferece ainda maior flexibilidade do suporte a qualquer linguagem de programação, porque a comunicação entre as camadas ocorre através de interface de rede.

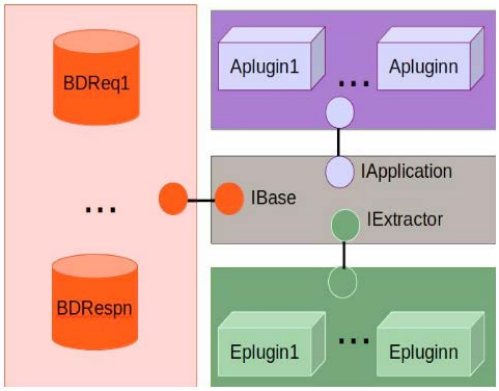
O Multi-LIBRAS tem quatro camadas (BC – Base de Conhecimento; PC – Processamento de Controle; PEA – Processamento de Extração de Análise; e PA – Processamento de Aplicativo) apresentadas na Figura 1(a).

**Figura 1 –** Visão Geral do *Framework* Multi-LIBRAS

(a) Visão externa das camadas do *framework*




(b) Visão interna das camadas do Multi-LIBRAS




*Fonte:* autoria própria.

A camada PEA (Processamento de Extração e Análise) permite o rápido acoplamento de Eplugin ao *framework*, que corresponde a um código em qualquer linguagem de programação contendo um algoritmo para análise de aplicativos em LIBRAS. Um exemplo que está no site do projeto Multi-LIBRAS é um EpluginOpenCV, codificado em Python com a biblioteca OpenCV, para calcular o tempo de início e fim de cada tradução de um aplicativo acoplado ao Multi-LIBRAS. Dessa forma, o controlador pode solicitar extrações de vários conhecimentos, de acordo com o número total de Eplugin e a capacidade de cada um deles.

A camada PA (Processamento de Aplicativo) permite o rápido acoplamento de Aplugin ao *framework*, que corresponde a aplicativos desenvolvidos no contexto de LIBRAS. Um exemplo que está no site do projeto Multi-LIBRAS é um ApluginVLibras, codificado em Java, que utiliza a classe Robot para interagir de maneira automatizada com o aplicativo V-Libras. Outro exemplo também disponível no site do projeto Multi-LIBRAS é o ApluginProDeaf codificado em HTML, JavaScript e Java, para interagir com o tradutor ProDeaf.

Dessa forma, um usuário do Multi-LIBRAS já pode solicitar ao controlador extrações do tempo de tradução dos aplicativos ProDeaf e V-Libras através dos plugins EpluginOpenCV, APluginProDeaf e APluginVLibras.

A camada BC (Base de Conhecimento) suporta diferentes modelagens de informações em LIBRAS, não restringindo o padrão ELiS, língua portuguesa, SignWriting e outros. Essa camada pode ser visualizada como subdividida em dois tipos de bases: uma com informações a serem encaminhadas para os aplicativos e outra para armazenar os dados coletados na camada PEA. A atual restrição é que as informações estejam estruturadas em um modelo de banco de dados com suporte à linguagem de consulta SQL, tal como o banco de dados MySQL, que suporta dados no formato de texto, vídeo e imagem.

A camada PC (Processamento de Controle) é o ponto fixo do *framework*. Esta tem o objetivo de prover flexibilidade para realizar o acoplamento e acesso de diversas bases de dados, algoritmos de extração de análise e aplicativos na área de LIBRAS. Ela foi implementada em Java, mas não restringe a linguagem de programação dos aplicativos e algoritmo de extração, porque a comunicação do controlador com as camadas ocorre pela troca de mensagens de textos por meio da camada de rede de qualquer sistema operacional.

Esta camada PC, coordena e executa a integração das camadas BS, e dos *plugins* da Figura 1 (b),

desde que o *APlugin* já tenha sido implementado pelo usuário colaborador e o *EPlugin* pelo investigador.

Além disso, a camada PC tem três pontos flexíveis – *IExtractor*, *IData* e *IApplication* – usados para integrar, respectivamente, diferentes *EPlugin*, base de dados e *APlugin* ao *framework*. A coordenação de PC ocorre de forma que acessa a base de dados para submeter informação ao *APlugin,* e, em seguida, solicita do *EPlugin* analisar o comportamento dos aplicativos acoplados e receber a resposta que, em seguida, é armazenada na base de conhecimento.

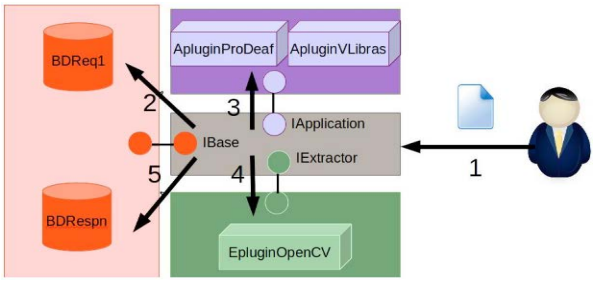
Em resumo, este *framework* consiste de: a) pontos flexíveis e fixos que permitem o acoplamento de base de dados pela associação da interface *IBase* bem como aplicativos e algoritmos de análise de soluções em LIBRAS pela inserção de *plugins* (*APlugin e EPlugin*), que estão associados às interfaces (*IAplication e IExtractor*); b) O *plugin APlugin* e o *IAplication* oferecem os pontos flexíveis que permitem a manipulação automatizada, sendo gerados pelos colaboradores para inserção de novos aplicativos ; c) O *EPlugin* e o *IExtractor* são pontos flexíveis que processam e coletam dados referentes às análises dos aplicativos na área de LIBRAS, que são armazenados na camada BC; e d) A interface *IBase* é outro ponto flexível acessado pela camada PC para obter dados a serem utilizados na manipulação dos *Aplugin* e para armazenar os dados gerado pelo *Eplugin*, não restringindo o tipo de modelagem de informações a ser armazenado.

## 4.3 Exemplificando o uso do Multi-LIBRAS

O cenário de exemplificação considera o uso do Multi-LIBRAS para realizar o processo de medição da tradução de cada palavra por cada tradutor, por meio de um mecanismo automatizado, com auxílio de um *Aplugin* para interagir com *ProDeaf* e um *Eplugin* com um algoritmo de extração de análise dos tradutores com uso do *OpenCV*[11]. Utiliza também uma câmera para detectar quando o avatar[12] do tradutor inicia um movimento e quando este voltou ao repouso.

Na Figura 2, um usuário sem conhecimento em programação pode executar o cenário descrito anteriormente, a partir da interação com a camada de Processamento de Controle. Para isso, esse usuário deve, primeiramente, informar ao PC um arquivo de configuração que especifica quais os *Aplugins*, *Eplugins* e bases de dados a serem utilizadas bem como a sequência e a repetição de uso desses programas, nesse caso, especificando o *ApluginProDeaf* (*plugin* para *ProDeaf*), *ApluginVLibras*(*plugin* para o V-Libras) e *EpluginOpenCV* (algoritmo de análise através do *OpenCV*).

**Figura 2 –** Exemplificando o uso do *framework* Multi-LIBRAS




*Fonte:* autoria própria.

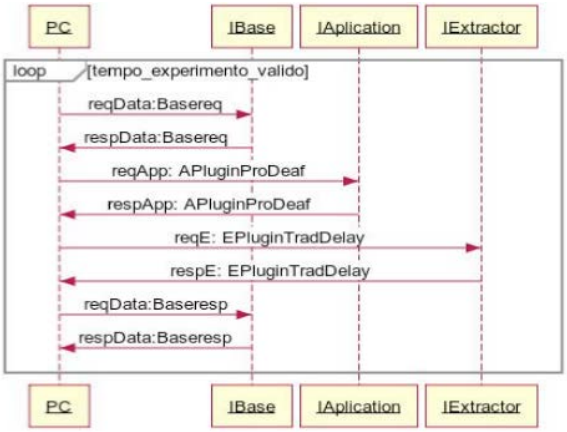
Então, o PC processa o arquivo de configuração que determina qual sequência executar, ou seja, no passo 2 da Figura 2, PC deve acessar uma base de dados. No passo 3 da Figura 2, PC envia uma das palavras da BaseReq para o *ApluginProDeaf* através da interface *IAplication*. No passo 4 da Figura 2, PC notifica o *EpluginOpenCV* para analisar o *ProDeaf* e aguardar resposta desse *EpluginOpenCV* através da interface *IExtractor*. Quando a camada de Processamento de Controle recebe a resposta pela *Iextractor*, então, no passo 5 da Figura 2, PC utiliza a interface *Ibase* para armazenar o resultado desta primeira análise no banco BaseResp. Depois, executa o mesmo passo a passo para o *ApluginVLibras* e sucessivamente para *ApluginProDeaf* até finalizar o tempo do experimento também determinado no arquivo de configuração.

De acordo com os passos anteriores, temos o seguinte diagrama de sequência apresentado na Figura 3:

11 OpenCV - http://opencv.org/

12 Representação gráfica de um indivíduo.

**Figura 3 –** Diagrama de sequência do processamento

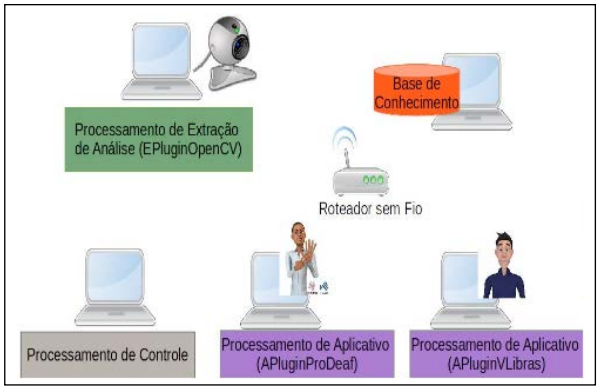



*Fonte:* autoria própria.

Além disso, é importante destacar que o *framework* pode ser executado em modo centralizado ou distribuído em rede, pois a comunicação dos módulos se dá por meio de mensagens trocadas por *sockets*.

No primeiro modo, todos os componentes estão em uma única máquina. No modo distribuído, conforme exemplo da Figura 4, o processamento é realizado por computadores distintos e conectados em rede, permitindo distribuição de carga, e, consequentemente, reduzindo o tempo de processamento.

**Figura 4 –** Exemplo do *framework* Multi-LIBRAS sendo configurado para executar no modo distribuído




*Fonte:* autoria própria.

Por fim, é importante esclarecer que o cenário demonstrado anteriormente não detalha a implementação de componentes, protocolos, algoritmos e arquivo de configuração, por razões de simplicidade, principalmente porque esse material pode ser encontrado no site do projeto do *framework* Multi-LIBRAS [13].

## 5 Avaliação

A técnica de medição foi escolhida para a avaliação do Multi-LIBRAS, a fim de demonstrar que *plugins* codificados em linguagens distintas podem ser acoplados ao *framework*, sendo identificado o custo real da comunicação entre a camada de processamento de controle e um *plugin* acoplado a esse *framework*.

O cenário de medição foi configurado de maneira centralizada, pois é um ambiente mais controlável do que o processamento distribuído, que pode ser influenciado por número de usuário, tráfego da rede e outros.

Nesse cenário, todo o processamento foi executado em um sistema operacional Linux, em um netbook com processador Intel Atom N555 dual core e 2GB de memória.

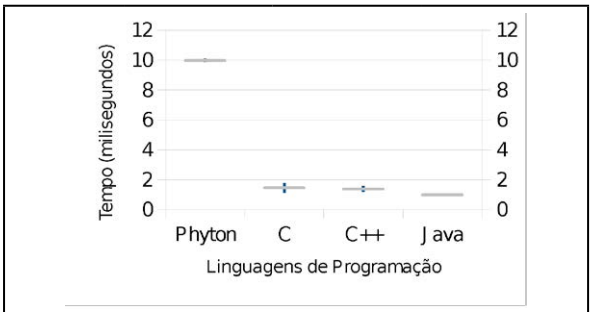
Para coleta dos dados, a métrica utilizada foi o tempo de resposta (intervalo entre a camada PC enviar uma solicitação e receber a resposta gerada pelo *plugin*); o parâmetro foi o tipo de linguagem utilizada na codificação do *plugin*, sendo verificado se esta afetaria o tempo de resposta.

A decisão da escolha dos fatores para esse parâmetro foi baseada no artigo de CASS (2015), que aponta Java, C, C++ e Python como as quatros linguagens mais populares, nesta ordem. Então, essas quatro linguagens foram utilizadas para codificar os *plugins* conectados ao *framework*.

Após obtenção de 5.000 amostras para comunicação entre PC e o tipo de *plugin*, observou-se que 900 amostras seriam suficientes, pois o desvio padrão está abaixo de 0,02 milissegundos. Então, considerando essas amostras, calculou-se a média e o desvio padrão populacional para então calcular os limites inferior e superior da média, com intervalo de confiança de 95%, obtendo-se o resultado da Figura 5, apresentada a seguir.

13 Projeto Multi-LIBRAS- https://sourceforge.net/projects/multi-libras

**Figura 5 –** Resultado da medição da comunicação entre o Multi-LIBRAS e plugins acoplados ao *framework*




| Linguagem | Média | Limite Inferior | Limite Superior |
|---|---|---|---|
| Python | 9.9766666667 | 9.83220086 | 10.12113247 |
| C | 1.4577777778 | 1.1231404448 | 1.7924151108 |
| C++ | 1.3788888889 | 1.1610915241 | 1.5966862537 |
| Java | 0.9922222222 | 0.9222295979 | 1.0622148466 |

*Fonte:* autoria própria.

Os resultados permitem verificar que o tipo de linguagem escolhida para codificar um *plugin* para o Multi-LIBRAS afeta o desempenho da solução final.

Deve-se evitar a escolha de *Python* para construção de *plugins* para o Multi-LIBRAS, pois é a quarta linguagem mais popular e apresentou o menor tempo de resposta.

Os *plugins* escritos em Java, C e C++ apresentaram tempos de respostas próximos, no entanto, C e C++ não são linguagens portáveis e requerem a recompilação de código.

Então, usuários colaboradores do Multi-LIBRAS devem escolher a linguagem Java para construção de *Aplugins* ou *Eplugins,* pois apresentaram menor tempo de resposta, e, adicionalmente, Java é a linguagem mais popular e portável.

## 6 Conclusões e trabalhos futuros

Este artigo mostrou que, no Brasil, a falta de padrões na área de LIBRAS dificulta a colaboração e integração de projetos, sendo possível identificar um alto índice de projetos descontinuados e de aplicativos que ainda precisam de muitas melhorias.

Por isso, o *framework* Multi-LIBRAS foi proposto e publicado no site da Sourceforge 14 com código aberto. Neste mesmo site está disponível o manual de uso e um exemplo de como analisar aplicativos de tradução, sendo exemplificados Aplugins para os softwares V-LIBRAS e ProDeaf. Esses exemplos motivam a extensão do Multi-LIBRAS para receber novos tradutores, sendo possível estender o *framework* para suportar HandTalk e Rybená, desde que seja criado um Aplugin para essa plataforma e esse aplicativo, respectivamente.

Como trabalhos futuros, pretende-se realizar as seguintes atividades de pesquisa e desenvolvimento:

- Adicionar ao Multi-LIBRAS um plugin de extração de conhecimento para contribuir com a análise automatizada de expressão facial e movimentos corporais dos avatares, dessa maneira, proporcionando uma análise mais detalhada dos movimentos.
- Utilizar o *framework* para analisar de maneira automatizada um grande volume de palavras para identificar e contabilizar a quantidade de palavras que utilizaram datilologia.
- Submeter os resultados das avaliações para os autores de cada um dos tradutores acoplados ao *framework,* a fim de que se possa divulgar o Multi-LIBRAS e colaborar com melhorias dos aplicativos.
- Gerar uma base padronizada de conhecimentos sobre tradução de Português para LIBRAS, a partir do conhecimento adquirido por meio dos atuais e novos tradutores que venham a ser acoplados ao *framework*.

Dessa forma, o *framework* Multi-LIBRAS pode auxiliar em uma possível criação de dicionário geral e aberto de Português e LIBRAS.

14 Site do Sourceforge - https://sourceforge.net

## REFERÊNCIAS

AGUIAR, T. C.; CHAIBUE, K. Histórico das Escritas de Línguas de Sinais. Revista Virtual de Cultura Surda, n. 15, 2015. Disponível em: <http://editora-arara-azul.com.br/site/admin/ckfinder/userfiles/files/3° Artigo para REVISTA 15 de THIAGO AGUIAR e KARIME CHAIBUE.pdf>. Acesso em: 20 set. 2015.

Barros, M. E. Escrita das Línguas de Sinais: proposta teórica e verificação prática. Tese (Doutorado em Linguística) – Universidade Federal de Santa Catarina, UFSC, Florianópolis, 2008.

BOUZID, Y; JEMNI, M. TuniSigner: A Virtual Interpreter to Learn Sign Writing. 2014 Ieee 14th International Conference On Advanced

Learning Technologies, [s.l.], jul. 2014. Institute of Electrical & Electronics Engineers (IEEE). DOI: 10.1109/icalt.2014.176.

CASS, S. The 2015 Top Ten Programming Languages. IEEE Spectrum, 20 jul. 2015. Disponível em: <http://spectrum.ieee.org/computing/software/the-2015-top-ten-programming-languages>. Acesso em: 30 set. 2015.

COLLING, J. P. Requisitos para inserção de libras em softwares educacionais para os anos iniciais. 2014. 55 f. Monografia (Especialização em Ciência da Computação) –Unioeste - Universidade Estadual do Oeste do Paraná, Cascavel, 2014.

COLLING, J. P; BOSCARIOLI, C. Avaliação de tecnologias de tradução português-libras visando o uso no ensino de crianças surdas. Revista Novas Tecnologias na Educação, Porto Alegre, v. 12, n. 2, p. 1-100, dez. 2014.

CORRÊA, Y. *et al*. Tecnologia Assistiva: a inserção de aplicativos de tradução na promoção de uma melhor comunicação entre surdos e ouvintes. In: CICLO DE PALESTRAS NOVAS TECNOLOGIAS NA EDUCAÇÃO, 13., 2014, Porto Alegre. Anais... Porto Alegre: Cinted, 2014. p. 1-10.

FUSCO, E. X-LIBRAS: um Ambiente Virtual para Língua Brasileira de Sinais. 2004. 156 f. Dissertação (Mestrado em Computação) – Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares da Rocha, Marília, 2004.

MATTSSON, M. Object-Oriented *Framework*s: a Survey of Methodological Issues, M.Sc. Dissertation (Master of Computer Science and Business Administration) – Department of Computer Science and Business Administration, University College of Karlskrona/Ronneby, LU-CS-96-197, 1996.

MATSUMOTO, T.; KATO, M.; IKEDA, T. JSPad. Proceedings Of The 3rd International Universal Communication Symposium On - Iucs '09, [s.l.], 2009. Association for Computing Machinery (ACM). DOI: 10.1145/1667780.1667855.

OLIVEIRA, L. P.; OLIVEIRA, G. A. Estudo de softwares em termos de disponibilidade e de maturidade para ensinar libras. In: CONGRESSO NORTE-NORDESTE DE PESQUISA E INOVAÇÃO, 10., Anais... Rio Branco: CONNEPI, 2015. p. 3-30.

PAPADOGIOGARKI, M. *et al*. Gesture synthesis from sign language notation using MPEG-4 humanoid animation parameters and inverse kinematics Intelligent Environments, 2006. 2nd IET International conference on.

PIVETTA, E. M.; ULBRICHT, V.; SAVI, R. Tradutores automáticos da linguagem português oral e escrita para uma linguagem visual-espacial da língua brasileira de sinais. 2011. Disponível em: <http://wright.ava.ufsc.br/~alice/conahpa/anais/2011/papers/4.pdf>. Acesso em: 21 jun. 2015.

QUEVEDO, S. R. P. de; VANZIN, T.; ULBRICH, V. R. Ambientes virtuais de aprendizagem bilíngues para surdos em EAD. Revista Brasileira de Aprendizagem Aberta e a Distância, São Paulo, v. 13, 2014.

SILVA, I. Q. *et al*. Avaliação da Compreensão de Textos Jornalísticos em Português, em Librol e em LIBRAS por Estudantes Surdos. In: II ENCOMPIF - ENCONTRO NACIONAL DE COMPUTAÇÃO DOS INSTITUTOS FEDERAIS, II., 2014, Brasília. Anais... Brasília: DF, XXXIV CSBC, 2014. p. 718-721

SUTTON, V.; FROST, A. SignWriting: Sign Languages Are Written Languages. 2008. Disponível em: <http://www.signwriting.org/archive/docs6/sw0523-US-SignWritingVAILConference2008.pdf>. Acesso em: 20 set. 2015.

Fontoura, M. F.; Uma Abordagem Sistemática para o Desenvolvimento de *Framework*s. Tese (Doutorado em ) – Departamento de. Informática, PUC-Rio, 1999.

TRINDADE, D. de F. G. Incop: um *framework* conceitual para o design de ambientes colaborativos inclusivos para surdos e não surdos de cultivo a comunidades de prática. 2013. Tese (Doutorado em Informática) – Universidade Federal do Paraná, Curitiba, 2013.