DIEGO ROBERTO ANTUNES

PROPOSTA DE UM MODELO COMPUTACIONAL PARA REPRESENTAÇÃO DE SINAIS EM UMA ARQUITETURA DE SERVIÇOS HCI-SL PARA LÍNGUAS DE SINAIS

Tese apresentada ao Programa de Pós-Graduação em Informática do Setor de Ciências Exatas da Universidade Federal do Paraná, como requisito parcial para obtenção do título de Doutor.
Orientador: Dr. André Luiz Pires Guedes
Co-orientador: Dra. Laura Sánchez García

CURITIBA

2015

DIEGO ROBERTO ANTUNES

PROPOSTA DE UM MODELO COMPUTACIONAL PARA REPRESENTAÇÃO DE SINAIS EM UMA ARQUITETURA DE SERVIÇOS HCI-SL PARA LÍNGUAS DE SINAIS

Tese apresentada ao Programa de Pós-Graduação em Informática do Setor de Ciências Exatas da Universidade Federal do Paraná, como requisito parcial para obtenção do título de Doutor.
Orientador: Dr. André Luiz Pires Guedes
Co-orientador: Dra. Laura Sánchez García

CURITIBA

2015

A636p Antunes, Diego Roberto
Proposta de um modelo computacional para representação de sinais em uma arquitetura de serviços HCI-SL para línguas de sinais/ Diego Roberto Antunes. – Curitiba, 2015.
[242] f. : il. color. ; 30 cm.

Tese - Universidade Federal do Paraná, Setor de Ciências Exatas, Programa de Pós-graduação em Informática, 2015.

Orientador: André Luiz Pires Guedes – Co-orientador: Laura Sánchez García.
Bibliografia: p. 196-206.

1. Surdos - Educação. 2. Língua de sinais. 3. Interação humano-computador. 4. Internet (Redes de computação) - Prestação de serviços. I. Universidade Federal do Paraná. II.Guedes, André Luiz Pires. III. García, Laura Sánchez . IV. Título.

CDD: 003.36350872

Ministério da Educação
Universidade Federal do Paraná
Programa de Pós-Graduação em Informática

PARECER

Nós, abaixo assinados, membros da Comissão Examinadora da defesa do(a) aluno(a) de Doutorado em Ciência da Computação, Diego Roberto Antunes, avaliamos a de tese de doutorado intitulado "Parecer de um Modelo Computacional para Representação de Sinais em uma Arquitetura de Serviços HCI-SL para Línguas de Sinais", cuja defesa pública, foi realizada no dia 23 de setembro de 2015. Após avaliação, decidimos pela:
(X) **Aprovação** do(a) candidato(a). ( ) **Reprovação** do(a) candidato(a).

Curitiba, 23 de setembro de 2015.

Prof. Dr. André Luiz Pires Guedes
PPGINF – Orientador

Profª. Dra. Laura Sanchez Garcia
PPGINF – Co-orientadora

Profª. Dra. Tanya Amara Felipe de Souza
INES – Membro Externo

Profª. Dra. Marília Abrahão Amaral
UTFPR – Membro Externo

Prof. Dr. Alexandre Ibrahim Direne
PPGINF – Membro Interno

Prof. Dr. Eduardo Todt
PPGINF – Membro Interno

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DO PARANÁ
Setor de Ciências Exatas
Coordenação de Pós-Graduação em Informática

Aos meus pais
Roberto Joaquim Antunes e Janete Kapp Antunes

AGRADECIMENTOS

Em primeiro lugar agradeço aos meus pais Roberto Joaquim Antunes e Janete Kapp Antunes pelo grande apoio, pelo esforço e pelo incentivo em todos os momentos durante os quase sete anos que trabalhei para o desenvolvimento desta pesquisa, que começou em meu mestrado em 2009. Esta conquista é para vocês!

À minha noiva, Paola, por todo o amor e a compreensão durante meu doutoramento. Agradeço de coração por ser paciente enquanto eu madrugava para terminar meu trabalho. Amo você!

Aos meus amigos e colegas de pesquisa e de trabalho que conheci durante estes anos em Curitiba, pelas conversas e as experiências que ajudaram nesta conquista.

Agradeço aos meus orientadores e amigos Prof. Dr. André Luiz Pires Guedes e Prof.ª Dr.ª Laura Sánchez García pela oportunidade, pelas lições, por todo o conhecimento, pelo grande apoio e pela infinita paciência na condução desta pesquisa. Sem a experiência e a amizade de vocês esta tese não poderia ter sido concretizada! Obrigado!

À Prof.ª Dr.ª Tanya A. Felipe, por toda a ajuda e o conhecimento que teve um papel fundamental para este trabalho.

Aos professores do Departamento de Informática da UFPR, pela amizade, pelas oportunidades e pelas discussões de corredor!

Aos secretários da Pós-Graduação, Jucélia e Rafael, que ajudaram em diversos momentos durante esta caminhada.

Por fim, quero agradecer a todas as pessoas que ajudaram para o desenvolvimento deste trabalho!

Muito Obrigado!

A simplicidade é o último grau de sofisticação!

Leonardo da Vinci

RESUMO

A falta de acesso à informação e ao conhecimento científico tem sido um grande obstáculo enfrentado por algumas comunidades minoritárias. Este é o caso das comunidades de Surdos que, historicamente, têm sofrido isolamento e exclusão social. Os surdos, no contexto desta pesquisa, têm sua identidade e sua cultura definidas pelo universo gestual-visual das Línguas de Sinais (LS), ferramentas poderosas utilizadas para as suas relações comunicacionais e para a construção de conhecimento. Neste contexto, esta tese explora as necessidades reais dos surdos e investiga as LS, sua estrutura e a forma de utilizá-las para a construção de ferramentas computacionais que auxiliem estes usuários no acesso à informação e ao conhecimento. Esta tese está inserida no contexto de uma Arquitetura de Interação Humano-Computador em Língua de Sinais (HCI-SL), que tem o intuito de criar ferramentas para o usuário final que proporcionem acessibilidade e ofereçam uma interação satisfatória por meio das LS. Um dos principais problemas que permeia todas as camadas da Arquitetura HCI-SL e seus serviços é a falta de uma estrutura formal para a representação computacional desses sinais, ou seja, uma estrutura que forneça suporte para a resolução de questões como armazenamento, indexação, recuperação, reconhecimento, reprodução, síntese, entre outras propriedades capazes de facilitar a resolução dos demais problemas computacionais associados à arquitetura. Neste contexto, esta tese apresenta a proposta de um modelo computacional para a representação de sinais das LS (CORE-SL) que considera um conjunto de propriedades formais e de condições de uso para sua aplicabilidade na HCI-SL, visando um suporte à arquitetura de hipótese para o desenvolvimento de seus contextos de usos potenciais. Adicionalmente, um estudo exploratório em relação ao uso do CORE-SL em alguns contextos de processamento computacional das LS é apresentado, gerando hipóteses de pesquisa, metodologias, *frameworks* e novas abordagens que constituem uma base para o desenvolvimento de serviços da HCI-SL, tais como o reconhecimento e a síntese automática de sinais, o processamento de linguagem natural para tradução, um banco de dados para armazenamento de sinais, entre outros.

Palavras-chave: Modelo Computacional, Comunidades de Surdos, Línguas de Sinais, Design de Interação Natural, Arquitetura de Serviços para Internet, Usabilidade.

ABSTRACT

The lack of access to information and scientific knowledge has been a major obstacle faced by some minority communities. This is the case of the Deaf Communities that, historically, have experienced isolation and social exclusion. Deaf people, in the context of this research, have their identity and their culture defined by the gestural-visual universe of Sign Languages (SL), powerful tools used for their communicational relationships and to building of knowledge. In this context, this thesis explores the real Deaf's needs and research the SL, its structure and the manner of use to building computational tools that assist Deaf people in the access to information and knowledge. This thesis fits into the context of a Human-Computer Interaction Architecture by Sign Language (HCI-SL), which aims to create tools for the end-user to provide accessibility and a satisfactory interaction via the SL. A key problem that have intersection with all layers of the HCI-SL Architecture and their services is the lack of a formal structure for the computational representation of signs, i.e., a structure that provides support to solve some problems like storage, indexing, retrieval, recognition, reproduction, synthesis, among other properties able to facilitate the resolution of other computational problems associated with this architecture. In this context, this thesis presents a proposal of a computational model for the representation of SL signs (CORE-SL), which considers a set of formal properties and conditions of use for their applicability in the HCI-SL, aiming a support for the architecture hypothesis to develop their potential contexts of uses. Additionally, an exploratory study regarding the use of the CORE-SL in some contexts related to SL computer processing is presented, generating research hypothesis, methodologies, frameworks and new approaches that provide a basis for the development of HCI-SL services, such as automatic sign recognition and synthesis, natural language processing for translation, databases for sign storage, among others.

Keywords: Computational Model, Deaf Communities, Sign Languages, Design of Natural Interaction, Services Architecture for Internet, Usability.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AM	Articulador Manual
ANSI	American National Standards Institute
API	Application Programming Interface
ASL	American Sign Language
CM	Configuração de Mão
ENM	Expressão Não-Manual
FLS	Fonologia das Línguas de Sinais
HCI-SL	Human-Computer Interaction in Sign Language
IHC	Interação Humano-Computador
ISL	Israeli Sign Language
JSON	Java Script Object Notation
LMG	Linguagem de Marcação Gestual
LML	Localização-Movimento-Localização
LS	Língua(s) de Sinais
LOC	Localização
MBP	Modelo Baseado em Parâmetros
MBS	Modelo Baseado em Segmentos
MD	Modelo de Dependência
MFV	Modelo da Fonologia Visual
MHT	Modelo Hand Tier
MM	Modelo Moraico
MMS	Modelo Movimento-Suspensão
MOV	Movimento
MP	Modelo Prosódico
OP	Orientação da Palma
PA	Ponto de Articulação
PLN	Processamento de Linguagem Natural
SELS	Sistema de Escrita de Línguas de Sinais
SPARC	Standards Planning and Requirements Committee
TIC	Tecnologias de Informação e Comunicação
UFPR	Universidade Federal do Paraná
VC	Visão Computacional
XML	Extensible Markup Language

SUMÁRIO

CAPÍTULO 1

INTRODUÇÃO

A língua natural é um recurso imprescindível no cotidiano do ser humano, pois tem a função de mediar as interações sociais e a comunicação interpessoal, sendo um meio de acesso a todo o conhecimento cultural e científico da sociedade. Para Chomsky (1986) [20] a aprendizagem e o domínio de uma língua natural tem um papel fundamental no desenvolvimento intelectual do indivíduo.

A língua é um meio para auxiliar o exercício da cidadania plena, definida pela composição de elementos como *"democracia, participação popular nos destinos da coletividade, soberania do povo, liberdade do indivíduo"* [60].

Dallari [28] defende que *"[...] a cidadania expressa um conjunto de direitos que dá a pessoa a possibilidade de participar ativamente da vida e do governo de seu povo. Quem não tem cidadania está marginalizado ou excluído da vida social [...]"*.

Um instrumento poderoso que pode permitir a aquisição de aspectos culturais, a inclusão do indivíduo na sociedade e o exercício de sua cidadania é a língua, um *"elemento mediador fundamental para o acesso ao conhecimento científico"* [51].

O indivíduo tem o direito de aprender, de expressar-se e de comunicar-se em sua língua natural. É também direito linguístico fundamental do indivíduo receber informação e conhecimento em uma língua oficial da nação ou do local onde reside. Quando estes direitos ao uso da língua natural e ao caráter oficial dessa língua são negados, o exercício da cidadania também é negado ao indivíduo [43].

1.1 Motivação e Justificativa

A falta de acesso à informação e ao conhecimento científico tem sido um grande obstáculo enfrentado por comunidades minoritárias. Este é o caso das comunidades de surdos, indivíduos que, historicamente, têm sofrido isolamento e exclusão da sociedade devido ao não acesso à língua majoritária e ao desrespeito de sua língua patrimonial [43].

Ao longo dos anos, em especial na década de 1980, as comunidades de surdos pelo mundo passaram a lutar e a revindicar as línguas de sinais como o meio de comunicação e acesso ao conhecimento nos diferentes segmentos sociais, um direito fundamental para o reconhecimento de sua cidadania [51].

No Brasil, desde a década de 80, os surdos passaram a ser vistos como membros de comunidades minoritárias, nas quais a suas culturas e identidades perpassam a sua língua de sinais, e cuja inclusão na sociedade agrega dimensões sociais, políticas e de cidadania [100]. Neste sentido, os surdos não são mais definidos por uma deficiência, mas como

integrantes de comunidades linguísticas e culturais minoritárias que utilizam as línguas de sinais para a comunicação e para suas relações sociais [51].

Segundo a Federação Mundial de Surdos *(WFD - World Federation of the Deaf)* existe uma estimativa de 70 milhões de surdos no mundo, indivíduos que encontram barreiras sociais e tecnológicas para o uso das línguas de sinais nas relações interpessoais mesmo em países desenvolvidos [111]. No Brasil, segundo dados do Censo do IBGE 2010 [69], existem mais de 9 milhões de brasileiros com algum grau de surdez [30], dentre os quais estão incluídos os membros das comunidades de surdos que se comunicam por meio das línguas de sinais.

A língua natural para os surdos é a língua de sinais, um sistema linguístico autônomo que possui regras de organização gramatical diferente das línguas orais (e.g. Língua Portuguesa) e uma complexidade estrutural que, como toda língua natural, possui seus níveis fonológico, morfológico, sintático, semântico e discursivo concretizados nos enunciados, em enunciações específicas, de seus usuários enquanto interlocutores [44], [46], [47] e [30].

As línguas de sinais são sistemas linguísticos que podem representar até mesmo conceitos abstratos, como qualquer língua natural. Neste sentido, é importante desmistificar pré-concepções sociais de que a comunicação dos surdos é feita por meio de mímicas, gestos ou apenas sinais icônicos. Cabe ressaltar, também, que cada país possui a sua língua de sinais, conforme acontece com as línguas orais em países diferentes, mas que possuem em sua modalidade gestual-visual uma característica comum em relação a estrutura fonética dos aspectos articulatórios que formam os sinais [51] [30] [4] [46] [47].

A partir dos diversos estudos linguísticos e das mobilizações de organizações, das federações e dos núcleos ligados às comunidades surdas para divulgação da língua, os surdos brasileiros conquistaram a oficialização da Libras (Língua Brasileira de Sinais) pelo decreto de Lei n. 10.436/2002 [30] [12]. Fernandes (2011) [51] e outros autores como [44], [46], [48], [30] reforçam o potencial visual dos surdos e o fato de a língua de sinais ser um produto cultural representativo e um recurso poderoso para melhorar a comunicação, a educação e o acesso ao conhecimento para essas comunidades.

Todavia, a língua majoritária (no Brasil, o Português) ainda domina os veículos de acesso à informação e ao conhecimento na sociedade. Assim, a maioria dos surdos acabam sendo excluídos do processo comunicacional e têm suas interações sociais e culturais restritas, devido à dificuldade de acesso à língua majoritária da sociedade em questão e por sua língua patrimonial (no Brasil, a Libras) ser desconsiderada [51].

Adicionalmente, observa-se que a sociedade (da língua oral) tem pouco ou nenhum conhecimento das línguas de sinais, o que ocorre, em particular, com a Libras [49]. Neste cenário, os surdos encontram dificuldades para a comunicação e a interação cotidiana na sociedade, pois todo o conhecimento é disponibilizado em outra língua (não natural para eles [62]), entre outros fatores e concepções geradas do desconhecimento pela sociedade dos aspectos culturais e linguísticos ligados às comunidades de surdos.

Portanto, o uso da língua de sinais é uma questão, um princípio fundamental para essas comunidades, no sentido de melhorar suas interações sociais, sua produção/acesso ao conhecimento e proporcionar as condições necessárias para o seu desenvolvimento intelectual.

Neste sentido, destaca-se que esta tese está inserida no contexto da surdez como uma diferença na experiência humana, na qual o indivíduo surdo está inserido em comunidades minoritárias onde os aspectos culturais são definidos pelo universo gestual-visual da língua de sinais. Adicionalmente, cabe chamar a atenção para o fato de que a tese também busca um entendimento adequado das necessidades dos surdos, principalmente no que concerne às línguas de sinais e a como utilizá-las para o desenvolvimento de ferramentas que apoiem de fato estas comunidades em sua inclusão social.

Com base no contexto das comunidades de surdos e as línguas de sinais, esta tese tem como motivação utilizar a tecnologia para desenvolver ferramentas que atendam às necessidades do usuário, no que tange à liberdade e à forma natural de interagir com o mundo por meio das línguas de sinais, e que esses artefatos tecnológicos possam impulsionar o acesso ao conhecimento, à inclusão e ao exercício da cidadania.

As Tecnologias de Informação e Comunicação (TIC), que transformaram a forma como a sociedade produz e consome o conhecimento [62], tem um papel essencial na construção de ferramentas que dêem o suporte necessário ao acesso à informação e tragam uma melhoria as interações sociais e comunicacionais dos surdos.

Entretanto, as TICs, ao invés de resolverem o problema em questão, acabam impondo diversas barreiras de acesso e de uso, principalmente, pelo fato de os desenvolvedores não terem um entendimento claro sobre o contexto: as comunidades de surdos, suas necessidades comunicacionais e de acesso à informação e um tratamento computacional adequado das língua de sinais [4], [62], [5].

A Interação Humano-Computador (IHC) é uma área que busca, entre outros objetivos, projetar e desenvolver artefatos computacionais centrados em solucionar as necessidades dos usuários finais, construindo formas de interação mais naturais e que proporcionem uma melhor experiência às pessoas durante o uso [93].

Dentre os conceitos abordados pela IHC estão as propriedades de **usabilidade** que descreve métodos de *design*, modelagem e teste para que sistemas e tecnologias sejam fáceis de usar, de aprender, de lembrar e que tragam satisfação ao usuário durante a interação; de **comunicabilidade**, pela qual o sistema deve permitir que o usuário entenda os objetivos do projetista e consiga utilizar todo o potencial da solução a partir da interação com o próprio sistema; e de **acessibilidade** que descreve as regras e métodos para tornar uma ferramenta acessível a todos os usuários, independentemente da tecnologia, da plataforma ou de quaisquer limitações [4] [98].

Porém, as TIC têm apresentado poucas soluções que contemplem as necessidades do usuário final e incluam de forma adequada os conceitos da IHC para disponibilizar uma

melhor interação para o público-alvo, os surdos. A interação com o usuário destas tecnologias não é natural para os surdos pois, como já foi dito, a interação com a ferramenta e a representação da informação não é mediada por uma língua de sinais [4] [5] [6].

Como exemplo, é possível citar a utilização, pelos surdos, de tecnologias de chamadas por vídeo e aplicativos móveis de mensagens textuais como solução para comunicação pessoa-pessoa. Porém, para iniciar uma conversação com outra pessoa ainda é necessário que os usuários (surdos) compreendam a língua utilizada para representação da informação e dos comandos na interface (no Brasil, o Português). No caso de aplicativos de mensagens textuais o surdo ainda precisa conhecer a língua majoritária de modalidade escrita.

Outro exemplo é o aplicativo Fone Fácil [7], que visa facilitar a comunicação entre surdos e ouvintes por meio de uma arquitetura de tradução texto/mensagem do celular (surdo) para voz (ouvinte) e vice-versa. O aplicativo é destinado aos surdos com o objetivo de promover acessibilidade, mas uma limitação evidente é a exigência de o surdo dominar o Português na modalidade escrita, exigência não realista no contexto [51] [44] [74].

Adicionalmente, podemos citar um cenário de um surdo que está assistindo a um vídeo em uma língua de sinais. Ao perceber no discurso um sinal que ele não conhece, o surdo busca um dicionário online para pesquisar o significado daquele sinal. Se o sistema não permite uma interação via língua de sinais, a tarefa de pesquisa não pode ser executada.

Neste sentido, houve uma iniciativa com o Dicionário Digital da Libras [45], no qual os surdos podem escolher uma configuração de mão para tentar encontrar o sinal desejado. Todavia, este dicionário não resolveu a questão da grande quantidade de resultados em uma busca, visto que possui um grande número de sinais (3.000 na versão 1 e 8.000 na versão 2) e utiliza somente a configuração de mão como critério de busca.

Estes exemplos reafirmam a necessidade, colocada anteriormente, de desenvolver ferramentas computacionais com ambientes de interface e interação que, por meio da língua de sinais, proporcionem acessibilidade plena e facilidade de uso às comunidades surdas.

Esta necessidade, por sua vez, determina a exigência da construção de artefatos e de processos computacionais para o desenvolvimento de uma Interação Humano-Computador em Língua de Sinais (HCI-SL) que seja natural aos surdos e passível de aplicação em diversas ferramentas e contextos.

Todavia, como apresentado nas pesquisas de [4] e [5], embora a área da Ciência da Computação tenha desenvolvido durante anos diversas soluções algorítmicas para diversos problemas envolvendo essa HCI-SL, não têm sido apresentados sistemas aplicados ao usuário final em contextos reais de uso, com base em um processamento computacional que contemple a estrutura das Língua de Sinais (Libras).

1.2 Contexto da Pesquisa

Com o intuito de desenvolver tecnologias que considerem as necessidades dos surdos e partam de um claro entendimento das línguas de sinais, esta tese trabalha no contexto da Arquitetura HCI-SL, um projeto de pesquisa que visa o desenvolvimento e a implementação desta estrutura computacional, dos seus serviços e dos seus aplicativos utilizando como base os conceitos da IHC e uma abordagem centrada no usuário, sob um prisma culturalmente informado capaz de determinar as necessidades deste perfil de usuário alvo e os decorrentes requisitos do sistema [56].

Este projeto de pesquisa tem sido desenvolvido pelo Grupo de Design de Interação para Inclusão e Desenvolvimento Social da Pós-Graduação em Informática da UFPR, que conta com diversos trabalhos interdisciplinares de mestrado e de doutorado nas sub-áreas da Ciência da Computação, da Linguística e da Educação. Para este desenvolvimento, o grupo tem estabelecido parcerias estratégicas com Linguistas, programas de Graduação em Letras/Libras e uma interação direta com algumas comunidades de surdos no Paraná.

A hipótese adotada nesta pesquisa é de que o desenvolvimento desta arquitetura e a solução dos seus sub-problemas computacionais e linguísticos é um requisito para a construção de aplicativos e de sistemas para o usuário final que promovam o acesso à informação e ao conhecimento por meio de uma língua de sinais, alavancando a inclusão destas comunidades na sociedade [56]. Uma abstração da Arquitetura HCI-SL pode ser vista na Figura 1.1.

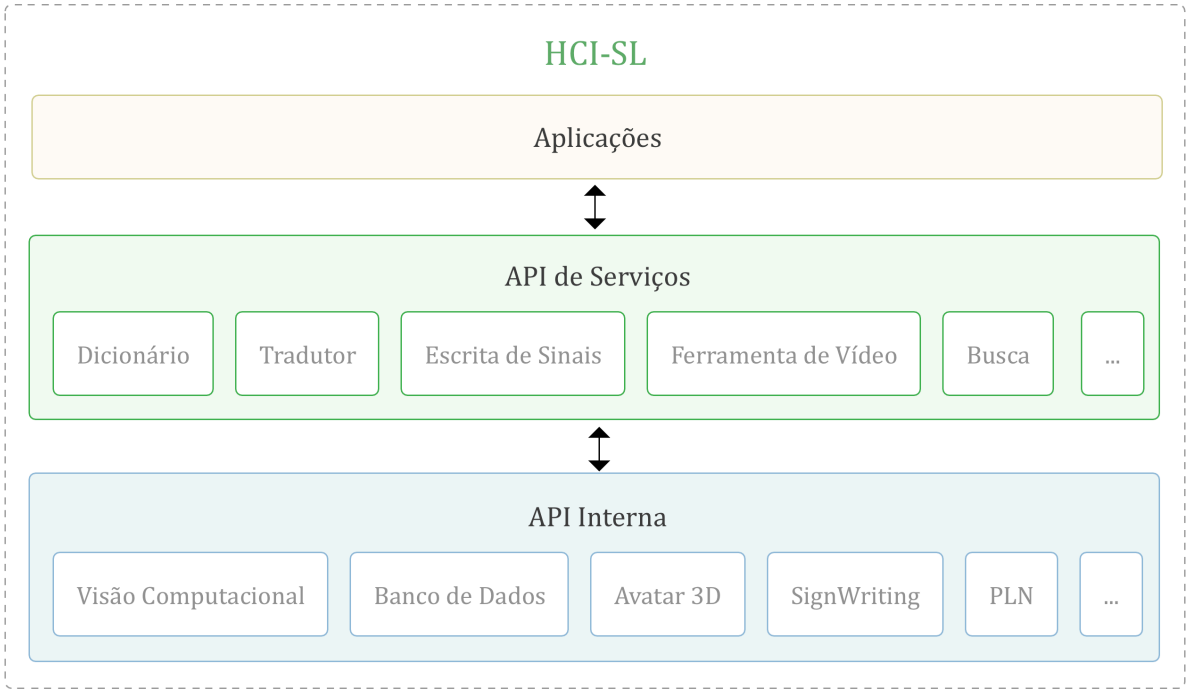


Figura 1.1: Esquema Conceitual da Arquitetura HCI-SL
Fonte: Modificado pelo autor (2015) [56]

Na abstração da arquitetura (Figura 1.1), podemos notar diversas camadas de serviços que representam sub-problemas específicos da Ciência da Computação a serem resolvidos, tais como a entrada de sinais via câmera (Visão Computacional), a tradução Português-Libras (Processamento de Linguagem Natural - PLN), a geração da Libras como saída do sistema (Síntese Automática de Sinais e SignWriting), o armazenamento e a recuperação de sinais (Banco de Dados), entre outros. Um estudo panorâmico sobre estas questões computacionais é apresentado nos Capítulos 5 e 6.

1.3 Descrição do Problema

Na Arquitetura HCI-SL duas características mostram-se relevantes: 1 - a dependência entre algumas ferramentas e serviços, e 2 - a necessidade de combinar soluções específicas para a construção de aplicações para o usuário final.

Como exemplo, pode-se perceber que para a construção de um Ambiente de Ensino a Distância (EAD) (**Camada de Aplicação**) é necessário um conjunto de ferramentas (**Camada de Serviços**) que incluem um dicionário (pesquisa de termos técnicos e sinais desconhecidos), um tradutor (para mediação e tradução da comunicação entre surdos e não-surdos), entre outros. A construção desses sistemas, por sua vez, exige que sejam utilizados os serviços da **Camada Interna** da arquitetura: um sistema de reconhecimento automático de sinais (*input*), um sistema para síntese de sinais, entre outros [56].

Neste contexto, percebeu-se que uma questão que tem intersecção em todas as camadas da Arquitetura HCI-SL, bem como em seus serviços, é a necessidade de uma estrutura para representação computacional dos sinais das LS. A abordagem arquitetural dos serviços deve ser baseada em um mesmo núcleo de software, um modelo robusto que permita um tratamento computacional eficiente das LS para resolver questões como armazenamento, indexação, recuperação, reconhecimento, síntese, entre outras propriedades de maneira a viabilizar o desenvolvimento dos demais serviços e aplicações previstos na arquitetura [4].

Como apresentado na revisão de literatura, as modelagens computacionais desenvolvidas para este fim têm apresentado soluções pontuais para problemas muito específicos e não têm realizado estudos mais amplos relacionados à solução do problema global: o processamento computacional das LS no contexto da Arquitetura HCI-SL.

Entre outras fragilidades, nota-se que a maioria dos modelos apresentados na literatura (e.g. [34] [22] [25] [115] [36] [35]) não fazem um estudo em relação à unicidade de representação, à usabilidade, à indexação e busca eficiente dos sinais (e.g. visto que deve ser utilizado um conceito de busca por similaridade[1]), à expressividade dos aspectos da LS, entre outros.

[1] O problema de Busca por Similaridade, no contexto desta tese, será discutido no Capítulo 5.

No Sistema *Web* da ProfDeaf [95] é disponibilizada uma interface para a criação de sinais com base na interação e na manipulação de um avatar 3D. A Figura 1.2 apresenta três momentos na modelagem de um novo sinal no sistema, nos quais o modelo permite e valida rotações de pulso e trajetórias de movimentos que não são anatomicamente possíveis. Este é um exemplo que deixa clara a relevância de um modelo computacional para representar os sinais em uma estrutura formal e com regras bem definidas.

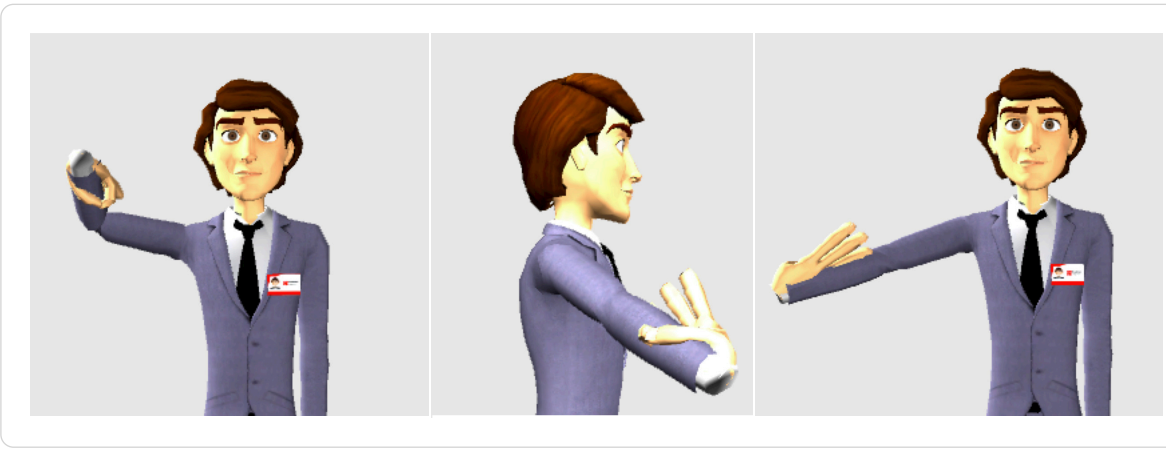


Figura 1.2: Avatar 3D com Erros na Representação do Sinal
Fonte: [95]

Como apresentado no Capítulo 2, a maioria dos modelos também é específico para problemas relacionados à síntese (*output* dos sinais), mas não fazem um estudo das propriedades formais necessárias para que um modelo computacional possa ser utilizado eficientemente para dar suporte à resolução dos problemas associados ao reconhecimento e à síntese de sinais, à consulta ao dicionário, à tradução, entre outros.

Adicionalmente, os trabalhos existentes não abordam um processo para a extensão e a melhoria das regras e da organização estrutural desses modelos - uma característica importante na Arquitetura HCI-SL devido à aplicabilidade do modelo computacional em seus contextos e aplicações.

Um problema comum dos trabalhos relacionados ([34] [22] [25] [115] [36] [35]) é a falta de uma documentação adequada para facilitar o aprendizado, o entendimento e a aplicação dos modelos em outras pesquisas. Na maioria das vezes os autores utilizam exemplos repletos de códigos numéricos e textuais que não são documentados em relação às LS. Por exemplo, se um movimento é descrito como rápido, este conceito deve ser definido para o usuário do modelo.

Portanto, o desenvolvimento deste modelo é uma condição *sine qua non* para contribuir na solução de grande parte dos problemas associados ao desenvolvimento dos serviços e aplicações previstos pela arquitetura de hipótese e, consequentemente, para o desenvolvimento de ferramentas ao usuário final que incluam este recurso de interação via língua de sinais [5][6][56].

Desta maneira, o problema de pesquisa está centrado no tratamento computacional das LS, particularmente no sentido de estudar como representar computacionalmente os sinais das LS em uma estrutura formal passível de aplicação em diversos contextos de uso, para servir de base para a resolução dos problemas técnicos associados à arquitetura e que disponibilize um *framework* para permitir sua extensão e melhoria colaborativa pela própria comunidade de pesquisa.

A questão principal que esta tese buscou responder foi: *como construir um modelo computacional para a representação dos sinais das LS que contenha propriedades específicas (e.g. unicidade, indexabilidade, recuperabilidade, completude, etc.), que permitam sua aplicação nos serviços de processamento computacional das LS visando um suporte para a construção de aplicações integradas na Arquitetura HCI-SL?*

O pressuposto adotado na tese é de que a solução deste problema em particular constitui um dos requisitos necessários para o desenvolvimento de um conjunto de ferramentas para disponibilizar essa interação mediada pela LS.

A solução deste problema, de representação computacional dos sinais de uma LS que considere um estudo de propriedades formais internas e externas (condições de uso) visando a aplicação nos contextos de uso da Arquitetura HCI-SL [56] ainda é pouco explorado na área.

Em geral, a maioria dos modelos existentes, apresentados no Capítulo 2, resolvem questões muito específicas (e.g. modelos que representam gestos) e não exploram seu uso na solução de outros problemas. Nesta situação, o problema levou à determinação de uma série de contextos de aplicação e de desafios respectivos, apontados a seguir:

- Como os sinais devem ser armazenados e indexados computacionalmente com o intuito de permitir a recuperação (busca) a partir da entrada (*input*) em LS?
- Como determinar uma quantidade representativa e qualitativa de sinais, considerando um léxico em desenvolvimento e não-finito?
- Quais estratégias podem ser utilizadas para a construção de um mecanismo de busca para recuperar esses sinais de forma eficiente, considerando o grande léxico das LS (em constante expansão) e o alto índice de similaridade entre diversos pares de sinais?
- Como o modelo pode ser aplicado na construção de um sistema de reconhecimento automático (via câmera) de sinais? Quais dentre as características gestuais, visuais e espaciais (padrões) necessárias computacionalmente e que devem ser consideradas na fase de reconhecimento?
- Como construir uma base de sinais baseada em contextos reais de uso da Libras pelos surdos que possa ser usada no treinamento e no teste de algoritmos de reconhecimento, em estudos linguísticos, entre outros?

- Como utilizar o modelo computacional como base de representação para sistemas de escrita de sinais, como o *SignWriting*?

- No contexto da Arquitetura HCI-SL, como os sinais podem ser sintetizados para o usuário final (Avatares 3D, SignWriting, entre outros)? É possível fazer o *parsing* do modelo computacional para diferentes saídas?

Os contextos de aplicação considerados levaram às seguintes questões de pesquisa que esta tese buscou responder:

Qual a estrutura e quais os elementos que constituem os sinais das LS? Como o objetivo macro é a construção de um modelo computacional para representar os sinais das LS, foi necessário um estudo específico para entender a organização e quais são os traços gestuais-visuais utilizados na formação e na distinção entre os sinais.

O ponto de partida para esta pesquisa foi a revisão de literatura sobre as LS para entender as formas pelas quais os linguistas trataram da composição dos sinais.

Entretanto, cabe ressaltar que, embora o conhecimento linguístico possa ser visto como insumo do nosso processo de pesquisa, esta tese não se limitou a determinar e utilizar um único sistema fonético-fonológico (traços distintivos), pois considerou-se que, para maximizar a completude de representação computacional para quaisquer sinais, seria necessário estudar, compilar, adaptar e incluir as melhores características de representação de cada sistema linguístico em uma estrutura formal.

Como construir um modelo computacional para representar os sinais? A partir de uma análise dos sistemas linguísticos que descrevem a estrutura dos sinais das LS, bem como de um estudo de sistemas e modelagens computacionais da literatura da Ciência da Computação, foi necessário analisar como adaptar todo este conhecimento para desenvolver um modelo formal que contivesse todos os traços distintivos necessários à representação de quaisquer sinais.

Este processo para o desenvolvimento do modelo computacional é uma das contribuições mais importantes desta tese, que tem como pressuposto a melhoria colaborativa do modelo. Neste sentido, o modelo deve possibilitar à comunidade de pesquisa uma forma de incluir eventuais novos elementos constituintes dos sinais e novas regras quando necessário (e.g. para a inclusão de outros níveis gramaticais).

Assim, quando for necessário realizar uma extensão ou uma complementação no modelo, este *framework* poderá e deverá ser utilizado de forma a manter a padronização e os conceitos já formalizados. Isto é condição necessária para a garantia do funcionamento correto dos serviços que utilizem como base o modelo proposto.

Como descrever as regras formais de produção dos sinais? O estudo de metalinguagens e de sistemas de modelagem foi necessário para determinar:

1. a forma de representação da estrutura conceitual do modelo, composta pelos parâmetros intrínsecos à composição dos sinais e às relações entre eles;

2. a maneira de formalizar este modelo conceitual em um sistema de regras de representação não-ambíguas e que facilite a sua implementação em qualquer linguagem de programação;

3. a forma de documentar este modelo para maximizar a facilidade de uso e de aprendizado das regras, visto que ele envolve elementos como movimentos e expressões faciais.

Quais propriedades este modelo deve considerar? O processo de construção deste modelo computacional ainda precisou investigar quais propriedades do modelo são necessárias à sua aplicabilidade nos serviços e aplicações previstos na Arquitetura HCI-SL. O modelo deve servir como uma *API*[2] para os serviços e, portanto, deve definir quais os dados, os formatos e os métodos de uso.

Para este estudo, foram considerados três grupos de propriedades para o *framework*: internas (relacionadas à formalização), externas (condições para o uso nos serviços) e de qualidade (relacionada à facilidade de uso e ao aprendizado do modelo).

O desenvolvimento deste modelo computacional, juntamente com o estudo exploratório das propriedades que um modelo deve ter para dar suporte à viabilização dos serviços da Arquitetura HCI-SL e a proposição de métodos, abordagens e *frameworks* para apoiar a resolução do problema global são as maiores contribuições desta tese.

Devido às características inerentes às LS, bem como ao seu desenvolvimento contínuo, os desafios ainda persistem e estão associados a diversas questões, dentre as quais:

- o léxico crescente e não-finito;

- a existência de bases de sinais (dicionários) com sinais incorretos quanto às regras de formação;

- o grande conjunto de sub-unidades fonéticas;

- as diversas possibilidades combinatórias para a formação de sinais;

- o grau de complexidade e o nível de detalhamento da representação das LS para o processamento computacional (por exemplo, a complexidade de reconhecer detalhes muito particulares dos sinais como movimentos locais, configurações de mão muito similares e uma diversidade de micro expressões faciais que são inerentes à execução do sinal e de seu correto entendimento no discurso);

[2]Application Programming Interface - Interface de Programação de Aplicações

- e a interdisciplinaridade envolvida no âmbito deste modelo computacional inerente aos serviços da arquitetura.

Esta tese é uma continuação da pesquisa de mestrado do autor (Antunes, 2011) [4], que realizou um estudo dos aspectos linguísticos das LS e desenvolveu uma modelagem no formato XML para a representação de sinais baseado em um sistema fonético-fonológico.

Nessa dissertação [4] foi analisado um conjunto de sinais para estudar os problemas computacionais na representação, tais como a falta de uma forma canônica de representação, de regras de produção bem definidas e de um processo para a incorporação de novos elementos para melhorar o nível de detalhamento das descrições.

A pesquisa de mestrado [4] possibilitou a construção de uma modelagem inicial que, quando analisada em relação à Arquitetura HCI-SL, levou a uma série de questionamentos, hipóteses e limitações, que explicitaram a necessidade de um estudo específico que, mesmo mantendo seu caráter exploratório, foi concretizado nesta tese.

Cabe ressaltar que o objetivo desta tese consistiu, então, em fazer um estudo exploratório com vistas ao desenvolvimento de uma base computacional sólida e de um framework para a construção de um modelo de representação dos sinais para fins computacionais na arquitetura proposta. Esta tese não tem o intuito de determinar um uso específico, mas construir uma base computacional por meio de modelos, metodologias e *frameworks* conceituais capaz de permitir o desenvolvimento dos serviços e aplicações associados à arquitetura de hipótese.

Assim, os objetos gerados nesta tese são destinados aos especialistas e aos pesquisadores de Ciência da Computação que tem a responsabilidade de implementação dos serviços computacionais da Arquitetura HCI-SL. Portanto, os resultados documentados nesta tese ainda não são destinados aos vários perfis de usuário-final, mas tem o intuito de impulsionar o desenvolvimento de ferramentas para eles.

É importante destacar que esta tese não teve como objetivo criar um novo sistema fonético ou fonológico para as LS, mas, sim, investigar na literatura (Computação, LS, entre outras) os elementos necessários ao desenvolvimento de um modelo para a representação de sinais que explorasse problemas computacionais críticos como a indexação, o armazenamento, a busca eficiente, entre outros.

1.4 Objetivo Geral

Propor um modelo computacional para a representação de sinais das LS considerando propriedades formais e condições de uso para sua aplicabilidade na Arquitetura HCI-SL.

1.5 Objetivos Específicos

- Especificar um framework para a construção de um modelo computacional para a representação de sinais de LS;
- Analisar os contextos de uso de um modelo computacional de representação de sinais na Arquitetura HCI-SL;
- Levantar e analisar propriedades formais para o modelo proposto em relação à aplicabilidade na Arquitetura HCI-SL.

1.6 Metodologia

Esta tese consiste de uma pesquisa exploratória que visa um aprofundamento em relação à representação computacional dos sinais e tem o intuito de explicitar hipóteses em relação ao uso do modelo proposto no tratamento computacional das LS nos contextos de uso da Arquitetura HCI-SL.

Por meio de uma abordagem qualitativa, esta tese descreve os processos e as metodologias utilizadas para a compreensão do contexto e dos aspectos computacionais relacionados ao problema, visando apoiar a consecução dos objetivos.

Os seguintes passos metodológicos foram utilizados:

1. Uma pesquisa na Linguística das LS sobre os sistemas que descrevem a estrutura e as características gestuais-visuais que constituem os sinais, com o intuito de criar uma base conceitual para o modelo computacional;
2. Estudo das modelagens existentes na literatura da Ciência da Computação para a representação de LS e de movimentos humanos, com o intuito de determinar parâmetros ou traços distintivos que não estivessem inclusos nos sistemas fonéticos, mas que poderiam complementar o modelo computacional aumentando o nível de detalhamento da representação;
3. Formalização do framework conceitual com vistas a subsidiar a construção de um modelo computacional para a representação de sinais, bem como investigação de meta-linguagens e estruturas para a formalização deste modelo e de suas regras;
4. Estudo e descrição de um padrão para a documentação do modelo computacional, visando a facilidade de uso, de entendimento e de aprendizado, de maneira a permitir que o modelo pudesse ser aplicado em diversos contextos reais;
5. Estudo e formalização das propriedades internas (formalização), externas (condições de uso) e de qualidade (facilidade de uso e de aprendizado) do modelo proposto,

com o intuito de permitir sua utilização no suporte ao desenvolvimento dos serviços computacionais que compõem a arquitetura de hipótese;

6. Estudo do problema de indexação e recuperação de sinais na arquitetura, com o objetivo de: a) clarificar a problemática e a complexidade da busca de sinais por similaridade; b) levantar hipóteses para auxiliar no desenvolvimento de uma função de distância (métrica) para calcular o índice de similaridade de uma entrada (sinal pesquisado) e os sinais da base; e c) discutir algumas estratégias algorítmicas para apoiar o desenvolvimento de uma estrutura de indexação eficiente dos sinais;

7. Explorar a aplicabilidade do modelo proposto na Arquitetura HCI-SL, com o intuito de analisar um conjunto de serviços específicos e desafios computacionais associados e elaborar hipóteses que descrevessem como o modelo poderia ser utilizado para apoiar a resolução de tais problemas, bem como a melhoria de processos de desenvolvimento utilizados na literatura.

8. Disseminar, por meio de publicações da área, o conhecimento construído visando impulsionar pesquisas futuras relacionadas às LS com metodologias, abordagens técnicas e *frameworks*;

1.7 Contribuições da Pesquisa

O processo de pesquisa desta tese teve o intuito de contribuir com um estudo para o desenvolvimento de um modelo computacional para a representação de sinais, e gerou como resultados concretos: um modelo formal que buscou uma estrutura canônica para a representação, um *framework* conceitual que pode ser utilizado para extensões e melhorias do modelo mantendo sua estrutura formal, podendo também ser adaptado como processo de teste de qualidade para outros modelos.

Adicionalmente, foi apresentado um estudo exploratório sobre a aplicabilidade do modelo formal na Arquitetura HCI-SL, um problema crítico que é desconsiderado em diversos trabalhos correlatos (Capítulo 2), tendo estudado hipóteses para auxiliar na resolução de diversos problemas computacionais associados aos serviços e aplicações previstos na arquitetura em questão. Neste sentido, esta tese apresenta e formaliza um conjunto de *frameworks*, abordagens técnicas e metodologias que podem impulsionar o desenvolvimento de serviços para a interação usuário-computador em língua de sinais em contextos reais de uso.

Destaca-se que os resultados obtidos nesta tese estão sendo aplicados em pesquisas de mestrado e doutorado em diversas sub-áreas da Computação que buscam desenvolver os serviços descritos na Arquitetura HCI-SL. Ao final deste texto são apresentados os resultados desses trabalhos.

Portanto, o estudo das propriedades formais do modelo computacional, as hipóteses de pesquisa levantadas e as metodologias e processos desenvolvidos consistem de contribuições para a área de pesquisa, que podem ser divididas em: Científicas (Ciência da Computação, Educação e Linguística) e Sociais. Algumas contribuições específicas incluem:

- Disponibilização de um modelo computacional para dar suporte à resolução de problemas específicos do processamento de LS na Arquitetura HCI-SL;
- Contribuição em sub-áreas da Ciência da Computação entre as quais Visão Computacional, Computação Gráfica, Processamento de Linguagem Natural, e Linguística, com um modelo computacional passível de aplicação no auxílio ao processo de resolução dos problemas associados ao desenvolvimento de ferramentas em LS;
- Disponibilização à comunidade científica de hipóteses de pesquisa, bem como de métodos e processos, relativos ao problema do tratamento computacional das LS;
- Contribuição para a discussão nas áreas de Educação e de Linguística sobre a importância das propriedades fonético-fonológicas na formação e na execução dos sinais, contribuindo para a conscientização do usuário em relação à importância das regras fonético-fonológicas intrínsecas à articulação dos sinais.

1.8 Organização da Tese

No Capítulo 2 é apresentada a fundamentação teórica, que consiste em um estudo sobre os modelos fonéticos que descrevem a estrutura e os elementos que formam os sinais das LS, com o objetivo de compreender quais modelos linguísticos poderiam ser utilizados na construção deste trabalho. Adicionalmente, são apresentados os modelos da Ciência da Computação que tem a capacidade de representar sinais, buscando entender as características que poderiam complementar o modelo proposto.

A partir do Capítulo 3 é apresentado o estudo realizado para a construção do modelo computacional, investigando estruturas formais, as propriedades intrínsecas ao modelo e a discussão sobre um padrão de documentação com usabilidade.

No Capítulo 4 é apresentado um estudo sobre as propriedades externas do modelo, com o intuito de considerar as condições de uso necessárias para auxiliar na resolução de outros sub-problemas da arquitetura.

O Capítulo 5 é dedicado à descrição do estudo sobre o problema de indexação e busca de sinais, analisando estratégias algorítmicas que poderiam ser utilizadas no desenvolvimento de um mecanismo de busca eficiente.

O Capítulo 6 apresenta um estudo exploratório que discute a aplicação do modelo em alguns contextos da Arquitetura HCI-SL e descreve algumas hipóteses capazes de apoiar o desenvolvimento de ferramentas com base no modelo.

No Capítulo 7 são apresentados alguns trabalhos em andamento que estão sendo apoiados pelo conhecimento gerado nesta tese.

Para finalizar, no Capítulo 8 são apresentadas as conclusões e as próximas etapas desta pesquisa.

CAPÍTULO 2

FUNDAMENTAÇÃO TEÓRICA

O presente capítulo tem o intuito de apresentar o estado da arte do problema tratado por esta tese, por meio da fundamentação teórica e da consequente análise crítica dos trabalhos relacionados.

Esta revisão bibliográfica foi dividida em quatro eixos (Figura 2.1): a) uma introdução às línguas de sinais, b) um estudo de sistemas fonético-fonológicos da Linguística para fundamentar as bases conceituais desta pesquisa, c) uma revisão das principais modelagens computacionais baseados em gestos e movimentos humanos, em escrita de sinais e na fonética das LS, e d) um levantamento das meta-linguagens utilizadas para a formalização deste tipo de conhecimento.

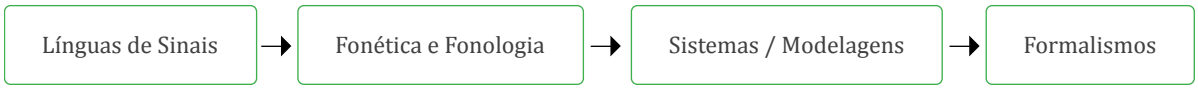


Figura 2.1: Base teórica e trabalhos correlatos revisados
Fonte: O autor (2015)

2.1 Línguas de Sinais

As Línguas de Sinais são línguas naturais de modalidade gestual-visual, pois a comunicação é realizada pelo interlocutor por meio de movimentos sinalizados (mãos, braços e antebraços) e de expressões não-manuais (expressões faciais e outros movimentos corporais feitos pela cabeça e pelo tronco) que são percebidos pelo receptor por meio do canal visual [4] [51] [44] [46] [42].

Além disso, as LS dispõem de outros recursos e propriedades que as diferenciam dos demais sistemas de comunicação, definindo-as como sistemas linguísticos legítimos e naturais que têm a habilidade de fornecer aos surdos um meio adequado para o desenvolvimento de todas as suas potencialidades linguísticas [30].

Para Fernandes [49] [51], as LS e, em particular a Libras, são sistemas de comunicação autônomos, organizados *"lexicalmente (vocabulário), gramaticalmente (regras) e funcionalmente (usos)"* [4] e têm a capacidade de expressar conceitos concretos e abstratos, sentimentos e processos de raciocínio [30], ou seja, *"possibilitam a expressão de qualquer conceito ou referência de dados da realidade"* [50][4].

Segundo Fernandes (2011) [51] é fundamental destacar que as LS são sistemas linguísticos que dispõem de uma estrutura definida, organizada em todos os níveis gramaticais, tais

como fonético-fonológico, morfológico, sintático, semântico e pragmático.

As LS, como objeto de pesquisa na Ciência da Computação, proporcionam um grande desafio em relação ao Processamento de Linguagem Natural, pois, além da complexidade da modalidade gestual-visual e da quantidade de propriedades articulatórias e suas combinações na formação dos sinais, também apresentam aspectos inerentes a qualquer língua natural, entre os quais [72]:

- **Homonímia** (*Homonymy*): Consiste na relação entre dois ou mais lexemas (distintos), que possuem significados diferentes, mas que possuem a mesma estrutura fonética-fonológica;
- **Polissemia** (*Polysemy*): Consiste na propriedade de um mesmo lexema (e.g. sinal) representar diferentes significados;
- **Sinonímia** (*Synonymy*): É definida pela propriedade de dois ou mais lexemas distintos representarem o mesmo significado;
- **Marcas Prosódicas e Verbo-Visuais:** Uma questão que talvez traga mais complexidade ao PLN é o fato das LS, assim como as línguas orais, possuírem e utilizarem recursos gestuais e expressões não-manuais para auxiliar o discurso, como ocorre, entre outras situações, para indicar entonação ou emoção. A questão fundamental deste recurso, é o diferenciar quais os aspectos gestuais e os não-manuais são gramaticais (constituintes de sinais) ou quais são de enunciado [42].

Estas características linguísticas, dentre outras, constituem desafios específicos em sistemas de tradução e dicionarização, ou seja, processos que devem considerar e resolver computacionalmente os problemas inerentes aos níveis sintático, morfológico, lexical, semântico e pragmático.

Problemas clássicos de PLN, no contexto das LS, incluem a correção de pronúncia (variações do sinalizador), o reconhecimento de sinais (e.g. análise de probabilidades de características reconhecidas), a síntese de escrita para sinais e a recuperação de informação (e.g. análise semântica de termos pesquisados e análise dos classificadores como anáforas em enunciados [48]) [72].

Estes aspectos intrínsecos às línguas naturais não foram abordados por esta tese e não foram tratados computacionalmente pelo modelo proposto, pois são problemas específicos das LS que devem ser resolvidos posteriormente por meio de processos computacionais de PLN previstos na Arquitetura HCI-SL.

O *framework* proposto nesta tese, bem como o modelo gerado, utiliza os aspectos gestuais, visuais e espaciais (fonética) que constituem os sinais das LS em uma representação computacional formal por meio de regras e de uma estrutura organizacional. Outros níveis e traços gramaticais como a sintaxe, a morfologia, a semântica, entre outros, devem ser analisados e incluídos posteriormente no modelo pela Linguística.

2.2 Fonologia das Línguas de Sinais

A fonologia das LS tem o intuito de investigar como os sinais são estruturalmente organizados [110], ou seja, pesquisar e definir quais são os componentes mínimos que constituem os sinais e estudar padrões de combinação destas sub-unidades, bem como possíveis variações na forma dos sinais [30].

Para Viotti (2008)[29] *"a fonética é a área da linguística que se ocupa da descrição e análise da massa amorfa fônica ou gestual. E a fonologia é a área da linguística que se ocupa da descrição e análise dos significantes de cada língua, ou seja, da porção que cada língua formatou a partir da massa amorfa fônica ou gestual"*. Assim, a fonética das LS é responsável por investigar quais os gestos que o ser humano é capaz de produzir, enquanto a fonologia organiza essas unidades a partir de regras e classes.

Esta tese tem como escopo de pesquisa os sistemas linguísticos que descrevem o nível fonético-fonológico, pois neste nível gramatical das LS são investigadas as unidades mínimas que constituem os sinais e a forma como são estruturalmente organizadas. O estudo deste conjunto de sistemas linguísticos tem o intuito de servir como base estrutural ao modelo computacional proposto, bem como de analisar os parâmetros descritos em cada sistema a fim de compreender e de levantar traços articulatórios distintivos necessários para uma correta representação (visando o maior nível de precisão possível na representação).

Além disso, o modelo utilizado para o tratamento computacional, no contexto desta tese, é restrito quanto a capacidade de representar as sub-unidades que constituem os sinais, pois a partir destas propriedades poderão ser analisadas, na sequência, os aspectos dos demais níveis gramaticais pela Linguística, tais como marcações sintáticas e morfológicas [29] [48] [13] [42].

Como exemplo, pode-se citar o trabalho de Supalla & Newport (1978) [104] que mostra que os verbos denominais se diferenciam de seus substantivos correlatos por meio de sub-unidades de movimento, e o de Klima & Bellugi (1979) [76] exemplifica que sub-unidades podem indicar flexões de tempo ou direção que podem caracterizar aspecto verbal, de concordância verbal e intensificador. Brentari (1998) [13] cita Valli (1990), que exemplifica que parâmetros da fonologia podem ser utilizados para realizar rimas em poesias em LS.

Felipe (2013) [42] faz uma análise de expressões não-manuais como um *"[...] componente verbo-visual gramático-discursivo [...]"* das LS que, quando executadas paralelamente durante o sinal ou a sentença, *"[...] integram-se ao plano fonológico, morfossintático e semântico-discursivo"* da língua.

Por exemplo, o sinal LADRÃO (Figura 2.2.A) é descrito somente por expressões não-manuais (inflar a bochecha) e sem as mãos (marcas fonológicas). Já no sinal MUITO LONGE (Figura Figura 2.2.B) as expressões não-manuais utilizadas na composição do sinal caracterizam um morfema complexo: morfema adjetivo (*"um tipo de movimento da*

cabeça e a direcionalidade do olhar, simultâneos ao sinal manual, marca uma qualidade para o substantivo ou intensificador para um advérbio ou adjetivo") [42].

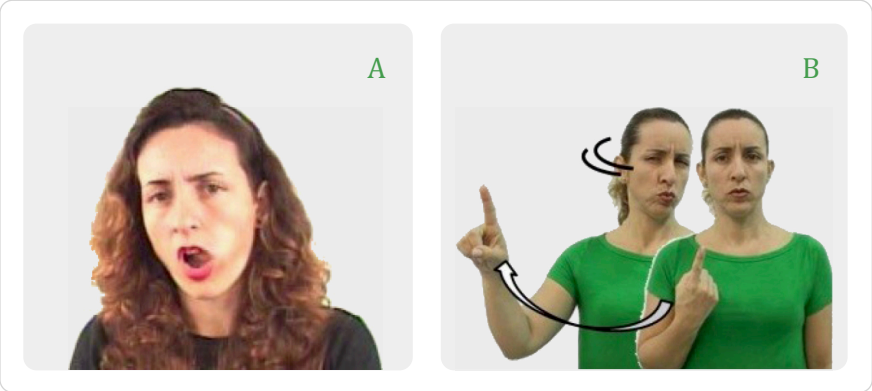


Figura 2.2: A) Sinal LADRÃO; B) Sinal MUITO LONGE
Fonte: Modificado pelo autor (2014) [42]

O nível sintático das LS, bem como as regras de formação de sentenças, também não foi abordado por esta tese. Cabe ressaltar que o encadeamento das representações geradas pelo modelo proposto para a produção de sentenças em LS pode ser realizado, mas ele exige um estudo adicional em relação à transição entre os sinais, característica que pode influenciar nas questões computacionais de reconhecimento e síntese.

Como mencionado anteriormente, embora esta tese se restrinja ao estudo dos sistemas fonéticos-fonológicos, um dos requisitos do modelo computacional projetado é permitir a extensão para a inclusão de outros níveis gramaticais. Esta tarefa poderá ser executada por linguistas e especialistas em LS, formalizando as regras por meio do *framework* proposto nesta tese.

A comunidade de pesquisadores da Linguística das LS tem pesquisado e desenvolvido diversos sistemas fonético-fonológicos para a descrição das sub-unidades que compõem os sinais e a sua estrutura organizacional. Como pode ser visto na próxima seção, estes sistemas apresentam diversos conceitos relacionados entre si, porém cada um deles resolve com mais ênfase um determinado problema (e.g. foco em movimento, ou em expressões não-manuais, entre outros).

Cabe destacar, portanto, que a literatura das LS ainda não elegeu um sistema fonético-fonológico de referência, o que, para o autor desta tese, parece decorrer uma série de fatores, entre os quais: a falta de completude, a dificuldade de leitura e de aprendizado, e a falta de regras.

Neste sentido, com o intuito de atingir os objetivos propostos, a construção da base conceitual para o modelo computacional proposto não deve ser limitada pelo uso de um único sistema fonético-fonológico, mas que ela deva ser desenvolvida por meio da construção do conjunto dos conceitos, dos princípios e das teorias que melhor tratassem cada aspecto gestual-visual que formam os sinais, buscando o maior nível de detalhamento e a

completude de representação[1].

2.2.1 Stokoe (1960)

A afirmação científica e a sistematização dos estudos linguísticos em relação às LS tiveram seu início nos estudos de William Stokoe (1960) [101], que realizou uma pesquisa ampla acerca dos gestos utilizados na comunicação dos surdos que objetivou entender os elementos internos que constituíam as regras de formação dos sinais. Neste sentido, o pesquisador descreveu os primeiros fonemas das LS, *"um número limitado de unidades [...], as quais são combinadas com movimentos sistemáticos, localizados em determinados pontos do corpo, que produzem unidades de sentido"* [51].

O estudo de Stokoe [101] demonstrou que as LS possuem todos os traços linguísticos de uma língua natural, *"seja no léxico, na sintaxe ou na capacidade de gerar infinitas sentenças* [30], ou seja, que os sinais são formados por unidades mínimas que foram descritas no nível fonológico.

Como citado por [29] e [30], Stokoe (1960)[101] empregou o termo "*quirema*"(do grego, '*quiros*', mão) (em substituição a "fonema") para se referir às sub-unidades articulatórias que formam os sinais e o termo "*quirologia*"(em substituição a "fonologia") para denotar o estudo das regras para a combinação destas unidades mínimas. Entretanto, pelo fato de as LS serem definidas como línguas naturais, linguistas que deram continuidade às pesquisas decidiram manter os termos padrão da literatura: fonema e fonologia.

Stokoe (1960) [101] analisou e descreveu a estrutura da Língua de Sinais Americana (ASL - *American Sign Language*) determinando que os sinais eram formados por três parâmetros mínimos e um conjunto finito de combinações [110] [30]:

- a **configuração de mãos** (*handshape - designator*);
- a localização ou **ponto de articulação** (*location - tabula*);
- o **movimento** (*movement - signation*);

O modelo fonético-fonológico desenvolvido por Stokoe (1960) [101] descreveu que os sinais eram formados por uma combinação e pela execução simultânea destas unidades mínimas [47].

Este sistema se caracteriza pela simplicidade estrutural e por um conjunto pequeno de parâmetros. O sistema de Stokoe (1960) [101], bem como toda sua pesquisa, foi o marco que impulsionou diversas pesquisas desta área da linguística.

[1] A completude consiste na capacidade de um modelo computacional em conseguir representar quaisquer sinais de uma LS, assim como quaisquer parâmetros fonéticos.

2.2.2 Battison (1979) e Klima & Bellugi (1979)

A partir da pesquisa realizada por Stokoe (1960) [101] e a proposição de um modelo que descrevia as unidades fonético-fonológicas presentes nos sinais, novos estudos linguísticos foram desenvolvidos com o intuito de explorar os aspectos articulatórios dos sinais.

Assim, Battison (1974, 1978) [10] [11] e Klima & Bellugi (1979) [76] investigaram um corpus da ASL e, como uma extensão ao modelo de Stokoe (1960) [101], descobriram outro traço fonético distintivo: a **orientação da palma da mão (OP)** - *Orientation*.

Durante a pesquisa, com base na descrição dos sinais pelo modelo de Stokoe [101], Battison (1974, 1978) [10] [11] encontrou uma série de sinais considerados "pares mínimos", ou seja, sinais que se diferenciavam entre si apenas por um traço articulatório.

No mesmo sentido, foram levantados diversos sinais que não podiam ser diferenciados entre si quando representados por meio do sistema fonológico de Stokoe [101], mas que diferiam entre si quando era considerada a orientação que a palma da mão assumia durante a execução do sinal. Assim, a OP foi adicionada como parte do conjunto de unidades mínimas necessárias para descrever e distinguir os sinais das LS.

O conjunto das unidades mínimas deste modelo fonético-fonológico das LS pode ser articulado durante o sinal tanto com uma ou com ambas as mãos, definidas na linguística, respectivamente, como **mão dominante** e **mão não-dominante** [30], [71], [10], [11], [96], [74].

Estas foram definições relevantes, pois pode ser um traço distintivo durante a articulação dos sinais. Como exemplo, é possível citar uma investigação na ISL - *Israeli Sign Language* que mostra alguns sinais que não podiam ser distinguidos pelos quatro parâmetros inicialmente definidos, pois eles possuíam as mesmas CM, PA, OP e MOV e a única diferença percebida foi o número de mãos (uma ou ambas) utilizadas na articulação de cada sinal [71] .

Adicionalmente, o modelo proposto por Battison (1978) [11] descreveu duas condições (restrições) em relação ao conjunto de sinais articulados simultaneamente com as duas mãos: Condição de Simetria (*Symmetry Condition*) e Condição de Dominância (*Dominance Condition*).

A **Condição de Simetria** descreve que sinais articulados com ambas as mãos realizando um movimento devem possuir a restrição: *"a configuração de mão deve ser a mesma, a locação e a orientação devem ser as mesmas ou simétricas, e o movimento deve ser simultâneo ou alternado para as duas mãos"* [4].

Já a **Condição de Dominância** descreve que em sinais executados com ambas as mãos que apresentam configurações de mão distintas, a mão dominante funciona como mão ativa (realiza o movimento) e a mão não-dominante serve como mão passiva (como apoio e/ou ponto de articulação para a mão dominante) [4].

Para [30], estas condições restringem a complexidade para a articulação e a compre-

ensão dos sinais, resultando em maior previsibilidade e regras controladas para a formação de sinais. Assim, há regras definidas quanto à formação de sinais que utilizam ambas as mãos para a articulação.

2.2.3 Conjunto Mínimo de Parâmetros

A partir das pesquisas discutidas nas subseções anteriores e com base nos trabalhos de [4], [30], [44], [43], [42], [51], entre outros, podemos definir os parâmetros principais que descrevem a estrutura global de formação dos sinais, bem como traços distintivos principais, descritos por Stokoe (1960) [101], Battison (1974, 1978) [10] [11], e Klima & Bellugi (1979) [76]:

Configuração de Mão (CM): A CM consiste na forma assumida pela mão enquanto um sinal específico é articulado, isto é, na disposição que os dedos assumem na mão dominante ou em ambas as mãos na execução do sinal.

Como mostra [15] a CM pode permanecer a mesma enquanto um sinal é articulado ou pode mudar para outra CM. Assim, é importante destacar que existem sinais que são compostos por mais de uma CM (e.g. a soletração em LS de uma palavra do Português).

Cada LS tem associados conjuntos pré-determinados de CM apresentados por diversos pesquisadores. No contexto desta tese, Brito (1995) [15] apresenta um conjunto de 46 configurações diferentes para a Libras que variam a partir da posição dos dedos, do número de dedos abertos, curvados, flexionados ou fechados, e do contato entre os dedos e a mão. Posteriormente, Felipe (2002) [46] apresentou um conjunto de 73 configurações a partir 8.000 sinais coletados para o Dicionário Digital da Libras. Essas CM são apresentadas na Figura 2.3).

Com o objetivo de desenvolver um modelo computacional que fosse capaz de representar qualquer sinal, foi necessário investigar na literatura quais eram as sub-unidades utilizadas para a formação dessas configurações de mão (ou seja, a estrutura interna das CM) entre as quais: rotação e a flexão de cada dedo. Isto é fundamental para certas aplicações (e.g. processamento de avatares 3D) e para formalizar as regras para a criação de novas CM (assim, as CM de outras LS também podem vir a ser representadas pelo modelo computacional).

Ponto de Articulação (PA): O PA, também tratado como "Localização", consiste do local no corpo ou no espaço onde o sinal é articulado.

Para Quadros & Karnopp (2004) [30] este espaço de articulação compreende todos os pontos dentro do raio de alcance das mãos e, desta maneira, os sinais podem ser articulados tanto em locais no corpo como no espaço. Um conjunto de PA foi apresentado na Tabela 2.1 [101], [15], [54].

O PA no espaço é determinado pelo raio de alcance das mãos nas três dimensões (X, Y e Z) durante a realização de um sinal pelo interlocutor.

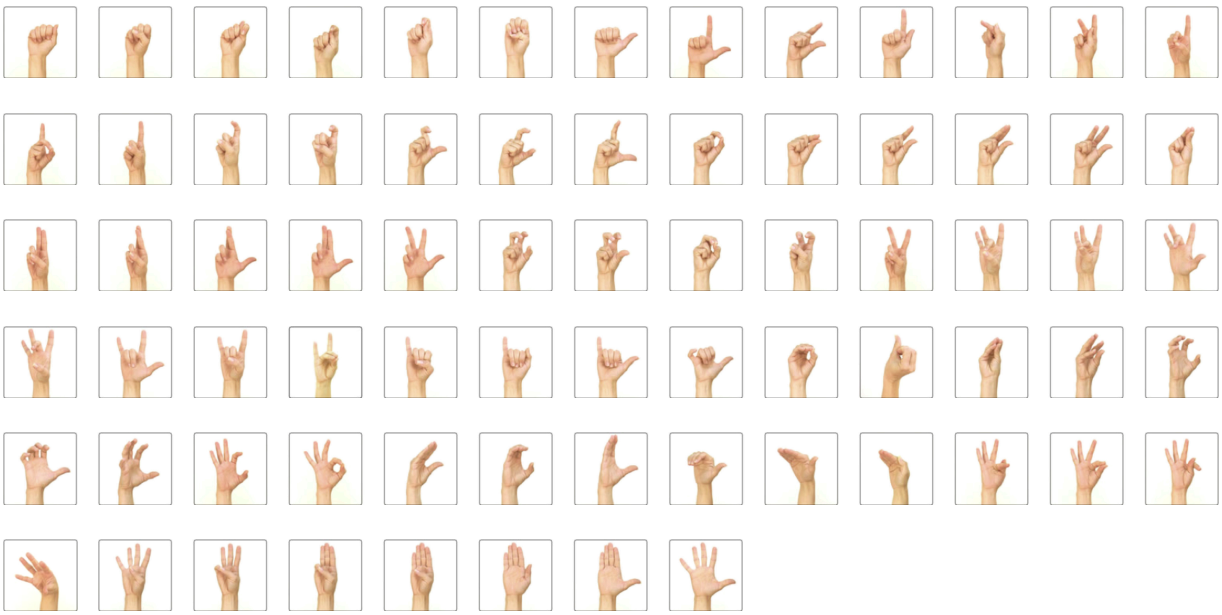


Figura 2.3: 73 Configurações de Mão da Libras
Fonte: Modificado pelo autor (2015) [46]

Adicionalmente, Brito (1995) [15] apresenta uma discussão acerca da precisão dos pontos de articulação no corpo. Por exemplo, ao descrever que um sinal é articulado na testa, poderia ser importante (como traço distintivo) mostrar maior exatidão no local (i.e. considerando o centro como referência, o PA pode estar localizado acima, abaixo, a direita, etc.).

Tabela 2.1: Resumo de um Conjunto de PA da Libras.

Cabeça	**Tronco**	**Mão**
Topo da cabeça	Pescoço	Palma
Testa	Ombro	Costas das mãos
Rosto	Busto	Lado do indicador
Parte superior do rosto	Estômago	Lado do dedo mínimo
Parte inferior do rosto	Cintura	Dedos
Orelha	Braços	Ponta dos dedos
Olhos	Braço	Dedo mínimo
Nariz	Antebraço	Anelar
Boca	Cotovelo	Dedo médio
Bochechas	Pulso	Indicador
Queixo		Polegar

Fonte: Modificado pelo autor (2013) [15]

Alguns autores como Battison (1978) [11] e Sandler (1989) [96] classificaram os PA em "principais"e "sub-espaços". Os PA principais consistem de regiões mais abrangentes tais como cabeça, tronco, mãos e espaço. Já os sub-espaços de localização possuem maior nível de detalhes em relação aos PA principais, entre os quais podemos citar olhos, dedo

indicador e palma.

Vale destacar que "*se um movimento de direção ocorre, este é tipicamente o resultado da especificação de dois sub-espaços, os quais estão associados e ligados a uma localização principal*"[30, p. 59]. Para a autora [30], mesmo que haja um movimento que altere a locação na articulação de um mesmo sinal, esta variação seria apenas entre sub-espaços e, desta forma, o sinal teria apenas uma locação principal.

De maneira análoga à empregada com as CM e visando uma melhor representação computacional, foram investigados modelos que descrevessem com maior nível de detalhamento os PA tanto no espaço quanto no corpo. Critérios como a alteração do PA decorrente de movimentos ou mesmo posições específicas devem ainda ser passíveis de uma melhor descrição.

Orientação da Palma (OP): Este parâmetro descreve a orientação (sentido) da palma da mão durante a articulação de um sinal. Comumente são utilizados apenas seis valores para representá-la: esquerda, direita, frente, trás, para baixo e para cima.

Em uma análise preliminar deste parâmetro, percebe-se que ele está relacionado diretamente com a CM. Neste sentido, nota-se que em certas configurações de mão, somente a OP pode não ser suficiente para distinguir o sinal, pois o sentido dos dedos ou o posicionamento do antebraço são variáveis que podem produzir resultados diferentes utilizando a mesma CM e OP.

Por exemplo, se considerar uma CM em 5 (mão e dedos abertos) e a OP para a esquerda, podem ser produzidos sinais distintos se o antebraço estiver na vertical ou na horizontal. O antebraço não está descrito na CM e na OP, mas cabe uma investigação mais aprofundada em outros modelos para garantir uma correta representação do sinal.

Movimento (MOV): Este parâmetro consiste na ação de trajetória que a(s) mão(s) realiza(m) no espaço ou no corpo. Este é um traço articulatório que apresenta grande complexidade, pois pode ser realizado de diversas formas, para várias direções, dentre outras variações que tem um papel fundamental da representação correta do sinal [76].

Karnopp (1999) [74] cita Klima & Bellugi (1979) [76] na justificativa de que o movimento é um parâmetro complexo, pois envolve uma *"vasta rede de formas e direções, desde os movimentos internos da mão, os movimentos do pulso e os movimentos direcionais no espaço"*. A autora também cita que certas variações no movimento especialmente críticas para a correta interpretação dos sinais. Por exemplo, Supalla & Newport (1978) mostram que variações no parâmetro de movimento podem distinguir verbos entre si [104].

As pesquisas conduzidas por Stokoe (1960) [101], Friedman (1977) [54], Klima & Bellugi (1979) [76], entre outros, definiram uma organização do MOV em sub-classes mais abrangentes, que incluem as características de tipo, direcionalidade, maneira e frequência. Alguns valores de cada sub-classe principal podem ser visualizados na Tabela 2.2.

Wilbur (1987) [112] realiza um estudo mais abrangente em relação ao MOV verificando que os movimentos podiam ser divididos em: "movimentos de trajetória"(*path movement*)

e "movimentos locais"(*local movement*). A questão fundamental é que existe um grande conjunto de sinais que utilizam somente movimentos locais de pulso ou dedos sem que haja uma trajetória no espaço de sinalização. Segundo Quadros & Karnopp (2004) [30], esta diferenciação é importante pois existem sinais que combinam os dois tipos de movimento e sinais que utilizam somente um deles.

Tabela 2.2: Resumo das Classes que formam o Movimento

Tipo	***-Contorno:*** retilíneo, helicoidal, circular, semicircular sinuoso, angular, pontual; ***-Interação:*** alternado, de aproximação, de separação, de inserção, cruzado; ***-Contato:*** de ligação, de agarrar, de deslizamento, de toque, de esfregar, de riscar, de escovar, de pincelar; ***-Torcedura do pulso:*** rotação, com refreamento; ***-Dobramento do pulso:*** para cima, para baixo; ***-Interno das mãos:*** abertura, fechamento, curvamento e dobramento.
Direcionalidade	***-Unidirecional:*** para cima, para baixo, para direita, para esquerda, para dentro, para fora, para o centro, para a lateral inferior esquerda, para a lateral inferior direita, para a lateral superior esquerda, para a lateral superior direita, para especifico ponto referencial; ***-Bidirecional:*** para cima e para baixo, para a esquerda e para direita, para dentro e para fora, para laterais opostas - superior direita e inferior esquerda;
Maneira	***Qualidade, tensão, velocidade*** -contínuo -de retenção -refreado
Frequência	***Repetição:*** simples ou repetido

Fonte: O autor (2011) [4]

Para finalizar esta visão geral, destaca-se que o Modelo Fonético-Fonológico definido nas pesquisas de [101], [102], [76], [10], [11], entre outros ficou conhecido como **"Modelo Baseado em Parâmetros (MBP)"**: um modelo que utilizava simultaneamente os quatro parâmetros principais para a articulação, a produção e a distinção entre os sinais.

2.2.4 Baker (1983)

As Expressões Não-Manuais (ENM) como uma sub-unidade distintiva na articulação dos sinais das LS, começaram a ser investigadas nas pesquisas de Stokoe et al. (1965) [102], que ao analisarem um corpus de discursos em ASL começaram a perceber a importância das expressões faciais na execução de alguns sinais específicos e marcações sintáticas. Por exemplo, para sinais como SIM ou NÃO, os surdos começaram a omitir alguns dos parâmetros (CM, OP, LOC e MOV), mostrando a necessidade de investigação adicional

para analisar se essas ENM deveriam ser consideradas como um traço distintivo para o MBP.

O estudo em relação as ENM para as LS teve um maior destaque a partir da pesquisa de Ekman (1978) [40] que investigou a importância das expressões faciais na comunicação humana. Especificamente para as LS, Baker (1983) [8] investigou e classificou as ENM utilizadas durante a articulação dos sinais em: movimentos da face, olhos, cabeça e tronco. É importante destacar que além de ser definido como um parâmetro do nível fonético-fonológico, as ENM também marcam construções sintáticas (sentenças interrogativas ou exclamativas, concordância, entre outras) e distinguem itens lexicais entre si [42] [30].

Como mostram Herrmann and Steinbach (2013) [65], os articuladores não-manuais têm um papel fundamental em diversos níveis gramaticais, pois expressam uma variedade de funções lexicais, morfossintáticas, prosódicas, semânticas e pragmáticas. Felipe (2013) [42] faz uma análise ampla das marcações não-manuais na Libras e apresenta diversos exemplos do papel das ENM em cada nível gramatical. Os sinais LADRÃO e MUITO LONGE foram apresentados anteriormente como exemplos (Figura 2.2).

Como um dos aspectos articulatórios que constituem os sinais das LS, as ENM podem ter um papel ainda mais importante: em diversas LS podem existir sinais que são formados apenas por ENM. Por exemplo, na Libras, o sinal LADRÃO (Figura 2.2.A) é formado apenas por uma expressão facial [44] [42].

Com base no trabalho Baker (1983) [8], Brito (1995) [15] faz um mapeamento das ENM básicas das LS, que foram classificadas em "rosto", "cabeça", "rosto e cabeça", e "tronco"(Tabela 2.3). Uma maior especificidade na descrição pode ser vista nos trabalhos de [80] e [44].

Como mostrado por [15], é importante destacar que as ENM podem ser articuladas simultaneamente em um mesmo sinal. Por exemplo, um sinal poderia executar, simultaneamente, uma expressão negativa (balançando a cabeça para os lados) e uma interrogativa (franzindo as sobrancelhas, movendo o tronco para frente e inclinando a cabeça para trás). Portanto, as ENM são fundamentais na constituição dos sinais das LS, bem como mais um desafio para a criação de modelos computacionais e para o PLN (principalmente, em relação ao nível de detalhes que o processamento computacional demanda).

Como discutido por Brentari (1998) [13], os modelos linguísticos que descrevem o nível fonético-fonológico podem ser classificados em dois momentos: MBP - Modelos Baseados em Parâmetros (Figura 2.4)[2] e MBS - Modelos Baseados em Segmentos (que usam os parâmetros principais como classes globais, distribuindo suas sub-classes e suas sub-unidades em uma estrutura específica na forma de segmentos, que indicam basicamente

[2]Os diagramas e os termos técnicos extraídos dos modelos correlatos foram documentados nesta tese no idioma Inglês por uma questão de padronização. Quando necessário, por exemplo, em uma definição e na primeira ocorrência no texto, foi adotada a tradução do termo para o Português, por exemplo, *Orientation* (Orientação da Palma).

estaticidade ou movimento).

Os modelos baseados e estendidos de Stokoe [101] eram inseridos nesta premissa de parâmetros, ou seja, os sinais eram constituídos pelas cinco unidades mínimas principais - fonemas, que eram executados simultaneamente.

Tabela 2.3: Algumas Expressões Não-Manuais das LS

Rosto:
Parte Superior
sobrancelhas franzidas
olhos arregalados
lance de olhos
sobrancelhas levantadas
Parte Inferior
bochechas infladas
bochechas contraídas
lábios contraídos projetados e sobrancelhas franzidas
correr da língua contra a parte inferior interna da bochecha
apenas bochecha direita inflada
contração do lábio superior
franzir do nariz
Cabeça:
balanceamento para frente e para trás (sim)
balanceamento para os lados (não)
inclinação para frente
inclinação para o lado
inclinação para trás
Rosto e Cabeça:
cabeça projetada para frente, olhos levemente cerrados, sobrancelhas franzidas
cabeça projetada para trás e olhos arregalados
Tronco:
para frente
para trás
balanceamento alternado dos ombros
balanceamento simultâneo dos ombros
balanceamento de um único ombro

Fonte: Modificado pelo autor (2013) [15], [4]

Entretanto, outros estudos apresentaram novos modelos linguísticos [30], tais como o Modelo *Movement-Hold* [80], o Modelo *Hand Tier* [96], o Modelo *Moraic* [91] e o Modelo *Prosodic* [13], que apresentaram uma nova perspectiva da estrutura fonético-fonológica dos sinais a partir da pesquisa e da inclusão da **sequencialidade**.

Estes modelos utilizam o conceito de segmentos “estáticos”e “dinâmicos”, ou seja, extraem o movimento das sub-unidades globais caracterizando-o como um segmento dinâmico, enquanto aspectos como as CM, os PA e as OR são sub-unidades, na maioria das vezes, presentes no segmento estático (que não produz movimento com trajetória) [52].

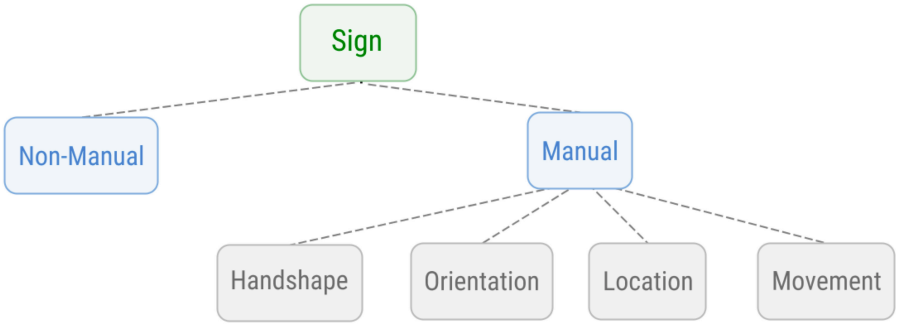


Figura 2.4: Organização Esquemática do MBP
Fonte: Modificado pelo autor (2014) [77]

2.2.5 Modelo *Movement-Hold* (1989)

Liddell & Johnson (1989) [80] mostraram que há dois tipos de sinais nas LS [114]:

- **sinais unitários**, que são definidos pelo aspecto estático das sub-unidades de CM, PA, OP enquanto o sinal é executado (com ou sem movimento);
- **sinais sequenciais**, aqueles em que durante a articulação do sinal há mudança de valor em alguma sub-unidade que segue uma determinada sequência.

Vemos assim que os autores [80] encontraram diversos sinais formados por mais de uma CM e/ou mais de um MOV e/ou mais de um PA, precisamente articulados em uma sequência específica que determinava a forma correta de um certo sinal.

Liddell & Johnson (1989)[80] desenvolveram o (MMS) - Modelo Movimento-Suspensão *(Movement-Hold)*, no qual os sinais são representados por dois segmentos:

- ***suspensão***, caracterizado pela falta de movimento com trajetória e por manter inalterados as suas sub-unidades de CM, LOC e OP durante a sinalização;
- ***movimento***, definido pelo aspecto dinâmico do movimento e de suas sub-unidades (e.g. frequência, tipo, plano, etc), além de ser caracterizado pela alteração de pelo menos um parâmetro estático devido à trajetória feita pelo movimento [114] [4] [29].

Além da sequencialidade, uma característica que destaca o MMS [80] é o alto nível de detalhamento de cada um dos segmentos, bem como o de cada um dos parâmetros

principais e suas sub-unidades. O MMS procurou detalhar, para cada parâmetro (CM, PA, OP, MOV e ENM), quais são e como são organizadas cada sub-classe específica e suas sub-unidades, criando um amplo conjunto de traços distintivos.

Segundo Xavier (2006) [114] e Viotti (2008) [29] o conceito base do MMS consiste em classificar os parâmetros em dois Feixes: Articulatório (FA) e Segmental (FS).

O FS define se o segmento em questão consiste de uma suspensão ou de um movimento (e os detalhes do movimento). Algumas características incluem:

1. **Traços de classe maior:** Determinam se o segmento é uma suspensão ou um movimento. No caso dos movimentos, pode ou não existir trajetória (*path movements, non-path movements* - ou movimentos locais). Se não houver dinamicidade das mãos no espaço ou sobre uma localização no corpo, o segmento é definido como uma suspensão (estado estático).

2. **Contornos de movimento:** Dado um segmento de movimento com trajetória, este traço descreve os contornos que ele pode assumir, tais como o tipo reto (*straight*) e o "circular"(*round*) - que ainda apresenta como sub-divisões os valores em "arco"(LOC inicial e final distintas) e circular (na qual a LOC inicial e final são iguais).

3. **Planos de contorno:** Descreve os planos (plano horizontal, vertical, de superfície, oblíquo, de linha medial) nos quais as mãos estão posicionadas na execução do sinal. Por exemplo, um movimento de deslizar a mão sobre o braço, consiste em um plano de superfície.

4. **Traços de qualidade:** Descreve três sub-unidades: qualidade temporal (maneira que o sinal é articulado em relação ao tempo, por exemplo, prolongado, acelerado e reduzido), qualidade não-temporal (que caracteriza a extensão do movimento como longa ou curta) e a tensão (que indica traços de intensidade).

5. **Movimentos locais:** São movimentos sem trajetória que podem ser articulados pelos dedos ou pulsos. Alguns valores possíveis são movimentos de "tamborilar", "circular"e a "oscilação"da CM, ou da OP, ou da LOC.

Com relação aos parâmetros e sub-unidades classificados no feixe articulatório, estão:

1. **Configuração de mão:** Descreve os sub-segmentos responsáveis por identificar a orientação do antebraço, a disposição dos dedos indicador, médio, anelar e mínimo (abertos, fechados, achatados ou em gancho), além de disponibilizar um parâmetro para descrever o relaxamento dos músculos (para compreender possíveis variações de sinalizador na articulações dos sinais). Este aspecto também descreve detalhadamente as sub-unidades específicas para o polegar, tais como as características de rotação e de contato com os demais dedos na produção de uma CM.

2. **Ponto de contato:** Descreve as LOC onde os sinais são articulados. O diferencial aqui consiste em que o MMS descreve traços para a precisão do local no corpo e para representar qual o lado do corpo no qual o sinal é executado (Figura 2.5). O traço denominado "parte da mão"representa os pontos de contato quando a LOC de um sinal é definida sobre a mão não-dominante. Já os traços de "proximidade"e "relação espacial"descrevem a posição das mãos no espaço, trazendo um aspecto importante para determinar, de forma um pouco mais precisa, os pontos espaciais em relação às dimensões X, Y e Z.

3. **Orientação da mão:** São utilizados dois traços para representar a orientação: a face (que consiste na direção apontada pelos dedos) e a orientação propriamente dita (que descreve a parte da mão que está paralela ao chão).

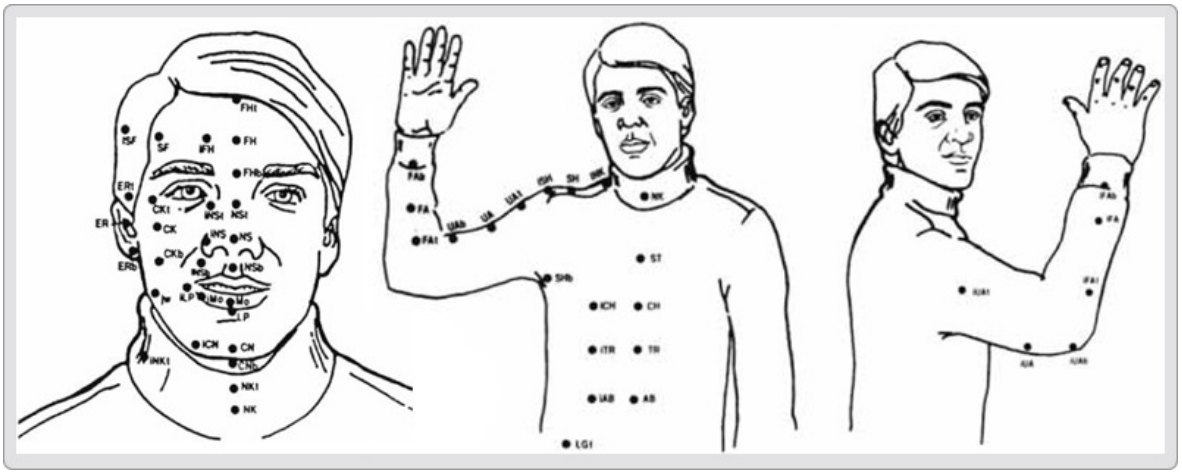


Figura 2.5: Pontos de Contato no Corpo do MMS
Fonte: Liddell and Johnson (1989) [80] [114]

A noção de que os sinais podem ser compostos de estruturas internas sequenciais é importante para a composição de sinais, uma limitação evidente no Modelo de Stokoe [71]. Neste sentido, Supalla & Newport (1978) [104] pesquisaram os traços distintivos em um grande conjunto de pares mínimos que representavam substantivos e verbos, descobrindo que grande parte deles se diferenciavam entre si por aspectos sequenciais de movimentos.

Como explica Johnson & Liddell (2011) [71] esta noção foi importante, pois evidenciou aspectos temporais no movimento, que podem caracterizá-lo de acordo com a "maneira"distinta de sua trajetória no inicio, no meio ou no final da articulação de um certo sinal (e.g. contínuo, pausado, entre outros).

Adicionalmente, é importante destacar que os modelos baseados em segmentos, por trazerem o aspecto de sequencialidade, tornam possível a representação de sinais compostos. Segundo Johnson & Liddell (2011) [71], uma limitação do Modelo de Stokoe era a

representação de sinais compostos, definidos, por Klima & Bellugi (1979) [76], como sinais que possuem um significado, mas que são formados a partir da combinação de outros sinais já existentes no léxico, em uma sequência específica.

Por exemplo, na Libras, o sinal IGREJA (que tem o significado próprio) é composto pela sequência dos sinais CASA e CRUZ (que, isoladamente, pertencem ao léxico tendo cada um deles seu próprio significado) - Figura 2.6.

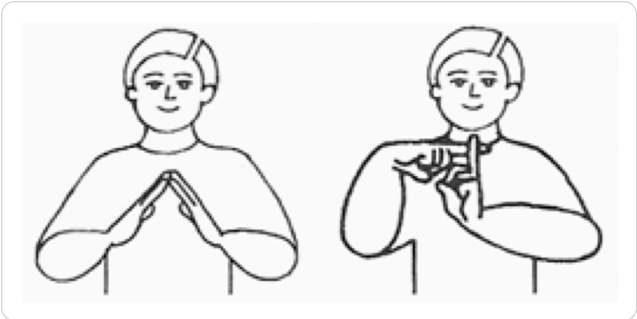


Figura 2.6: Sinal IGREJA: Composição pelo sinal CASA e CRUZ em sequência
Fonte: Capovilla et al. (2009) [17]

2.2.6 Modelo *Hand Tier* (1989)

No mesmo sentido ao MMS [80], o Modelo *Hand Tier* - MHT proposto por Sandler (1989) [96] utiliza o conceito baseado em segmentos para organizar estruturalmente os parâmetros básicos descritos pelo MBP de Stokoe [101].

Diferentemente do MMS, Sandler (1989) [96] propõe um modelo que descreve basicamente dois segmentos (camadas): o primeiro, que descreve as CM e a OP, e o segundo, que representa o MOV e a LOC.

Em sua pesquisa, a autora argumenta que as configurações de mão (que consideram a orientação da palma e dos dedos) possuem recursos específicos e complexos que formam uma camada distinta, separada da LOC e do MOV [13].

Desta forma, o MHT descreve uma estrutura mais detalhada para as CM, no sentido de representar os detalhes de cada dedo em um segmento próprio. Além disso, essa camada separada mantém os conceitos do parâmetro "configuração de mão"e consegue capturar tanto os aspectos sequenciais quanto os simultâneos [96].

Por exemplo, esse aspecto seria interessante para a representação fonético-fonológica de um sinal do tipo soletração (uma sequência de configurações de mão para representar cada uma das letras de uma certa palavra da língua oral-auditiva). Um exemplo da árvore das configurações de mão do MHT é apresentado na Figura 2.7.

O fator diferencial do MHT é que ele define uma organização estrutural baseada em características geométricas, diferentemente do MMS, que trabalha com dois feixes (articulatório e segmental) [77].

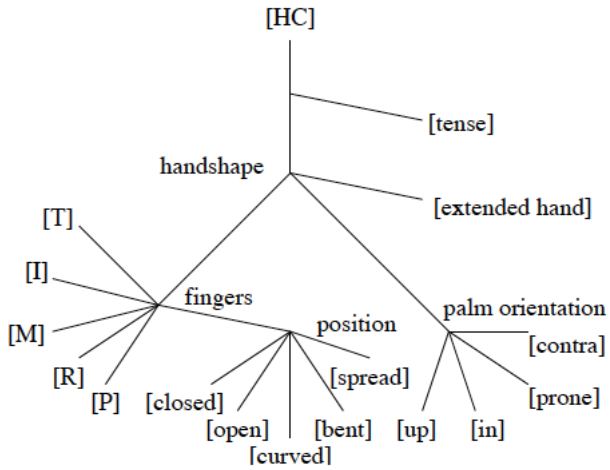


Figura 2.7: Árvore de Representação das CM no MHT
Fonte: Brentari (1998) [13]

Nota-se que, embora a árvore de descrição das mãos descreva os elementos que fazem parte da formação das CM, faltam algumas regras e relações quanto ao uso destas sub-unidades. Como exemplo, pode-se perceber que (no modelo esquemático da Figura 2.7) não há uma relação de contato entre os dedos, fato fundamental para a formação de CM.

No MHT o feixe segmental proposto pelo MMS é mantido, enquanto a propriedade de localização substitui o estado de suspensão (*hold*) do MMS, descrevendo que o movimento de trajetória é a ligação espacial e dinâmica entre dois PA de maneira sequencial.

Como define Sandler (1989) [96], o segmento de localização é caracterizado pelo momento em que a mão-dominante obrigatoriamente atinge uma outra LOC específica durante a articulação de certo sinal. As LOC são definidas no MHT pelo parâmetro *place* (local) (Figura 2.8), com um nível detalhamento menos preciso que o MMS.

Outra diferença em relação ao MMS é que o MHT define a LOC como o segmento estático (o movimento é a consequência / transição entre duas LOC e compartilha as mesmas características da camada de LOC). A árvore da camada de LOC do MHT pode ser vista na Figura 2.8.

Esta é uma forma encontrada para eliminar possíveis redundâncias na representação do MMS, na qual é possível descrever, por exemplo, um mesmo sinal como uma sequência de suspensão-movimento ou suspensão-movimento-suspensão (i.e. sem a formalização do MMS para a definição de uma regra para início e fim de sinais que possuam movimento).

Similarmente ao MMS, Sandler (1989) [96] especifica algumas características para descrever a LOC (Figura 2.8) no MHT: distância (*distance*), elevação (*height*), lateralidade (*laterality*) e contato (*contact*). Essas características são compartilhadas para especificação da LOC e do MOV [77].

- **Lateralidade - Ipsi**: A LOC está no mesmo lado do corpo que a mão dominante (principal);

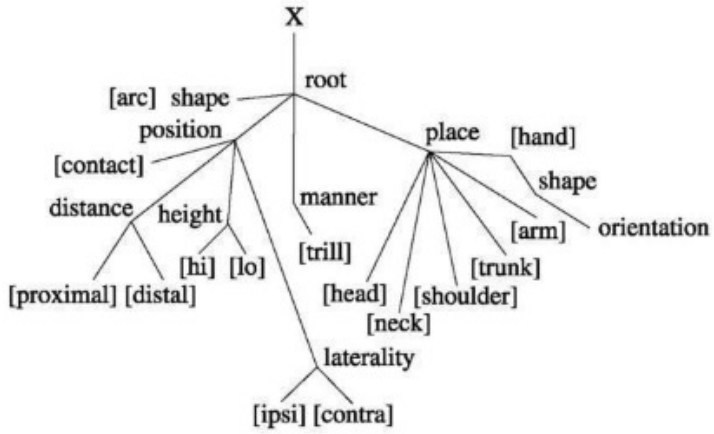


Figura 2.8: Árvore de Representação da LOC no MHT
Fonte: Brentari (1998) [13]

- **Lateralidade - Contra**: A LOC está no lado contrário do corpo que a mão não-dominante (secundária);
- **Elevação - Alta** (*High*): A mão é posicionada acima do meio da LOC especificada (no caso de a LOC ser na mão, indica a proximidade das pontas dos dedos);
- **Elevação - Baixa** (*Low*): A mão é posicionada abaixo do meio da LOC especificada (no caso de a LOC ser na mão, indica a proximidade da base da mão ou pulso);
- **Distância - Proximal**: A mão é posicionada a uma curta distância do corpo (alguns centímetros);
- **Distância - Distal**: A mão é posicionada a uma distância longa do corpo (a distância do braço estendido para a frente);
- **Contato**: O contato que a mão dominante faz com uma LOC.

2.2.7 Modelo Moraico (1990)

A partir de 1990, novos estudos realizados por Perlmutter [91] culminaram no desenvolvimento do *Moraic Model* ou Modelo Moraico (MM). A representação fonético-fonológica dos sinais no MM é baseada em um tipo de estrutura moraica.

Brentari (1998) [13] explica que o MM defende uma forma *"de sílaba que é o domínio de uma série de restrições fonológicas, tais como as mudanças em ambas as configurações de mão nos dedos selecionados (chamado contrastes de configuração de mão) e as mudanças nas configurações de abertura (chamado contornos de configuração de mão)"*.

Na Fonologia Moraica, esta "sílaba curta"é definida como uma **mora**, um termo utilizado em pesquisas fonológicas no sentido de definir "métricas"para relacionar uma unidade

mínima de tempo, com o objetivo de observar as características prosódicas de uma certa língua (e.g. ritmo, entonação, entre outros) [88].

Para Clark et al (2007) [24], uma mora consiste em um "peso"para a sílaba, que pode indicar aspectos de ritmo ou métricas de tempo. Uma característica inovadora do MM consiste em utilizar aspectos relacionados à sonoridade para tentar descrever os padrões de organização das sub-unidades fonológicas dos sinais das LS, por exemplo, movimentos de tamborilar com a mão (vibração da mão) [13].

Embora haja uma diferença conceitual em relação ao MMS e ao MHT, no sentido de substituir as unidades de tempo de segmentos por unidades moraicas, o MM também dispõe de uma estrutura que inclui os segmentos de Movimento e de Localizações para a representação dos sinais.

Os movimentos ainda são definidos por ações das mãos com trajetória, enquanto a Localização é o ponto de articulação da(s) mão(s) sem movimento ou com movimentos locais (non-path movements) [91] [13].

Após fazer uma análise no MM, Brentari (1998) [13] concluiu que o modelo não apresentava uma estrutura específica para características fonológicas, mas uma possível vantagem seria a definição de uma unidade de tempo moraica genérica que poderia ser relacionada tanto com os movimentos como com os pontos de articulação.

Observa-se que essa característica de unidade temporal tem o intuito de fornecer um melhor detalhamento na representação fonética-fonológica dos sinais, como um aspecto adicional ou até mesmo de caráter complementar em relação às propriedades fonológicas e sua organização estrutural descritas tanto pelo MMS quanto pelo MHT, além de evidenciar com clareza aspectos prosódicos.

2.2.8 Modelo Baseado na Fonologia de Dependência (1993)

Os conceitos baseados na Fonologia de Dependência (DP) foram muito aplicados em pesquisas de LS. Algumas características constantemente vistas são a complexidade de aspectos geométricos ou a estrutura formacional e seus aspectos fonológicos [13], e a utilização de uma correlação binária-assimétrica de um conjunto específico de elementos [74].

Karnopp (1999) explana que com a finalidade de desenvolver um modelo que utilizasse características gerais em relação à produção e à modalidade (neste caso as LS), Hulst (1993) fez a proposição de um modelo fonológico para as LS que contemplou essas características em uma estrutura binária, na qual "*uma unidade pode ser o núcleo de sucessivos constituintes inclusivos*"[74].

Esta Fonologia de Dependência oferece uma base para descrever os demais traços articulatórios dos sinais (CM, MOV e LOC), além de determinar que os sinais podem ser feitos com as duas mãos. O modelo de Hulst (1993) [66] também segue a característica

de sequencialidade demonstrada pela pesquisa do MMS.

Adicionalmente, ao especificar um núcleo de configurações de mão, o autor se baseia no MHT em relação à árvore de CM proposta, generalizando aspectos de configurações de dedos como um sub-núcleo, considerando a não mudança desta característica. Neste sentido, o autor descreve uma propriedade chamada ***finger selection - fingsel*** (seleção de dedos). Esta estrutura define quais os dedos estão selecionados e quais não estão na CM [26].

Assim como o MMS, Crasborn et al. (2002) [26] mostram que não basta descrever os dedos selecionados, mas é necessário, também descrever, a configuração dos dedos em relação à abertura, à flexão e à lateralidade.

O modelo de Hulst (1993) [66] utiliza o conceito de "localizações principais"e "sub-localizações". Neste sentido, um movimento com trajetória é representado pela mudança entre os sub-espaços (LOC) [74].

Crasborn et al. (2002) [26] destacam esta estrutura da marcação de início e de fim do movimento, utilizada por diversos trabalhos ([80], [91], [96], [13]) como um "esqueleto"(*LML-skeleton*) que representa a localização inicial (L), as especificidades do movimento intermediário (M) e a localização final (L). Entretanto, o modelo de Hulst (1993) [66] faz uma redução no modelo *LML-skeleton* para uma estrutura bi-posicional: o *XX-skeleton*.

Hulst (1993) [66] define o movimento como uma mudança na localização, ou na orientação ou na configuração de mão, definindo três tipos de movimentos:

1. ***Path Movement*** (movimento de trajetória): É o movimento da mão (articulador) como um todo. Essa trajetória (percurso) consiste na alteração da LOC entre pontos na região do corpo ou no espaço neutro;

2. ***Aperture Change*** (mudança de abertura): É a alteração da CM, ou seja, a especificação de duas CM;

3. ***Orientation Change*** (mudança de orientação): Consiste da alteração da orientação durante o movimento (por exemplo, uma rotação do antebraço).

Tanto a mudança de abertura quanto a de orientação foram definidas como alterações locais (*local changes*), em oposição ao movimento global de trajetória (*global path*). Isto é importante, pois esses eventos dinâmicos podem ocorrer de forma isolada ou simultânea. Por exemplo, um movimento de trajetória pode combinar movimentos locais de abertura e de orientação. Assim, o movimento global é definido como "primário"e o movimento local como "secundário"[26].

Uma exemplificação esquemática deste modelo é apresentada na Figura 2.9. Neste esquema, percebe-se que a LOC principal (*LocPrinc0*), a OR e a abertura das mãos (*Abert*) possuem dois estados (*a, b*). Esses estados indicam as duas posições (início e fim) do movimento no modelo *XX-skeleton*.

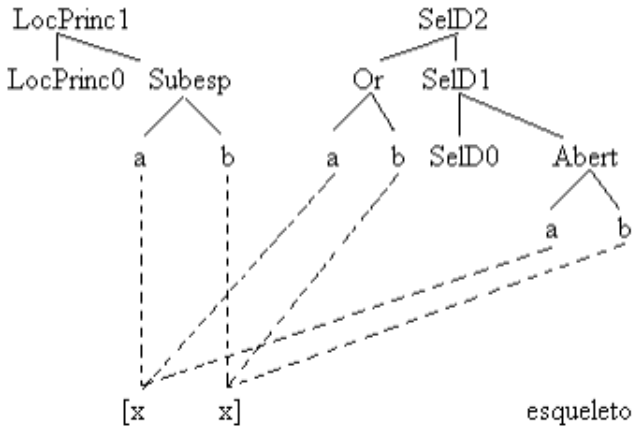


Figura 2.9: Exemplo de Árvore de Representação do Modelo de Hulst (1993)
Fonte: Karnopp (1999) [74]

2.2.9 Modelo da Fonologia Visual (1994)

Uyechi (1994; 1995) [108] e [109] desenvolveu o Modelo da Fonologia Visual (MFV) *(Visual Phonology Model)* que utiliza conceitos da geometria para representar diversas generalizações em relação aos aspectos fonético-fonológicos das LS, em particular fazendo uma modelagem no sentido das capacidades anatômicas dos articuladores [13].

Segundo Brentari (1998), as juntas articulatórias são definidas como um tipo de pontos "pivô"que contêm um sistema de coordenadas de duas dimensões. O espaço de sinalização para os movimentos consiste em um sistema de quatro cubos de três dimensões [13], [108] e [109] (Figura 2.10):

1. ***Hand Prism* (HP)**, utilizado para as mãos;
2. ***Local Signing Space* (LSS)**, utilizado para sinais simples com uma LOC no corpo ou espaço neutro;
3. ***Global Signing Space* (GSS)**, para sinais complexos;
4. ***Discourse Signing Space* (DSS)**, espaço para elementos constituintes dos sinais considerados mais amplos no discurso.

Adicionalmente, o MFV é caracterizado por não definir um segmento apenas para o movimento. No mesmo sentido que o MM ou o MHT, os movimentos são representados *"por 'unidades de transição' decorrentes de 'transformações de corpo rígido'"* [13] dos articuladores fonológicos .

Mesmo com esta mudança conceitual, o MFV possui aspectos de intersecção com os modelos anteriores. Como exemplificado em [13], as unidades de transição no MFV são expressos por alterações de CM, LOC ou OP de forma pragmática. Esta forma é uma

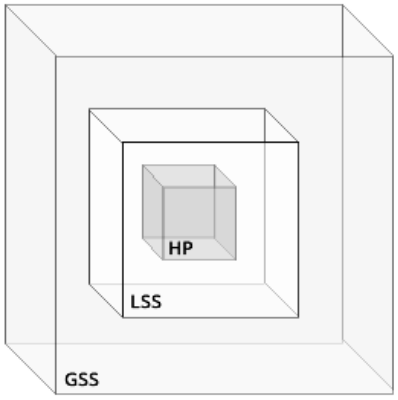


Figura 2.10: Representação do MFV para as dimensões HP, LSS e GSS
Fonte: Modificado pelo autor (2013) [13]

construção similar ao conceito de "peso"definido no MM, pois tem um papel formal distinto na estrutura fonológica, além de ser um traço distintivo importante para caracterizar a complexidade de um sinal.

A OP no MFV não é definida por um conjunto de valores para orientação das mãos, palma ou dedos como no MHT e MMS. O aspecto de orientação constitui uma série de relações entre os planos articulatórios, o que pode indicar uma similaridade no sentido da relação espacial e proximal do MMS (definição de um plano em um tipo de coordenadas de três dimensões) com a combinação de planos do HP, do LSS e do GSS do modelo de Uyechi (1994; 1995) [108] e [109].

2.2.10 Modelo Prosódico (1998)

Após uma análise de todos os modelos fonético-fonológicos, citados anteriormente, em relação às suas bases conceituais, a forma de organização quanto aos parâmetros constituintes dos sinais e a maneira como os modelos representam esses sinais das LS (também aspectos complementares que envolvem demais aspectos gramaticais, Brentari (1998) [13] desenvolveu o Modelo Prosódico (MP) - *Prosodic Model* - que contemplou diversos conceitos dos modelos MHT, MMS, MM e MFV.

Como primeira característica marcante, o MP considera tanto as propriedades inerentes (parâmetros globais definido no MBP, tais como CM, LOC e OP) quanto as propriedades prosódicas (unidades de tempo, pesos, entre outras).

De acordo com Brentari (1998) [13], vários pesquisadores aprimoraram tanto a árvore de configuração de mão (Corina 1990a, 1993; Johnson 1994 *apud* Brentari 1998) quanto os *clusters* de características (sub-unidades fonológicas) (Wilbur 1993; van der Hulst 1993).

Neste sentido, a definição de um tipo de camada formada pelas características inerentes foi necessária, principalmente, porque todos os modelos fonológicos são definidos por especificações dos traços articulatórios e de uma estrutura relacional entre eles.

O MP teve como motivação a necessidade de características geométricas como as mostradas no MHT. Entretanto, o MP possui algumas diferenças conceituais no que tange à sua estrutura, composta por duas classes de nodos principais em sua árvore: uma formada pelas características inerentes **(IF - Inerent Features)** e outra formada pelas características prosódicas (**PF - Prosodic Features**). Esta organização estrutural foi apresentada na Figura 2.11.

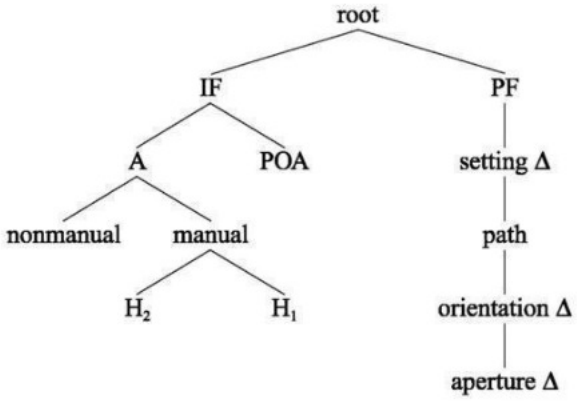


Figura 2.11: Árvore do MP, com seus dois nodos principais: IF e PF
Fonte: Brentari (1998) [13]

Algumas diferenças conceituais, também em relação aos modelos anteriores da literatura, são a estrutura interna para a representação dos aspectos geométricos e o fato de não considerar o movimento como uma unidade segmental, mas, sim, como uma primitiva das unidades prosódicas.

Dentre os conceitos semelhantes do MP estão a noção de planos de articulação, que foi baseada no MFV, e o fato de que o MP utiliza a idéia de pesos para melhorar a representação de propriedades prosódicas (assim como o MM ou o conceito do MFV) [13].

Na árvore do MP (Figura 2.11) é possível ver que o nodo da classe IF é sub-dividido em Articuladores (A) e POntos de Articulação (POA). O nodo que representa os articuladores (A) define os aspectos das camadas não manuais e manuais. As características manuais ainda definem a mão dominante e a mão não-dominante para um determinado sinal. O nodo que representa os traços manuais (Figura 2.12), como citado por [13], agrega um dos traços mais complexos de representação, devido ao nível de detalhes necessários para a formação de uma CM.

Neste sentido, a autora desenvolve um alto nível de detalhamento para as mãos, assim como o MMS, incluindo parâmetros para dedos selecionados ou não (***selected / non-selected fingers***), disposição dos dedos (***extended*** - estendido e ***flexed*** - flexionado), características das juntas (***joints***) e um detalhamento separado para os dedos (polegar e demais dedos em nodos separados, assim como o MMS) - ***fingers*** e ***thumb***.

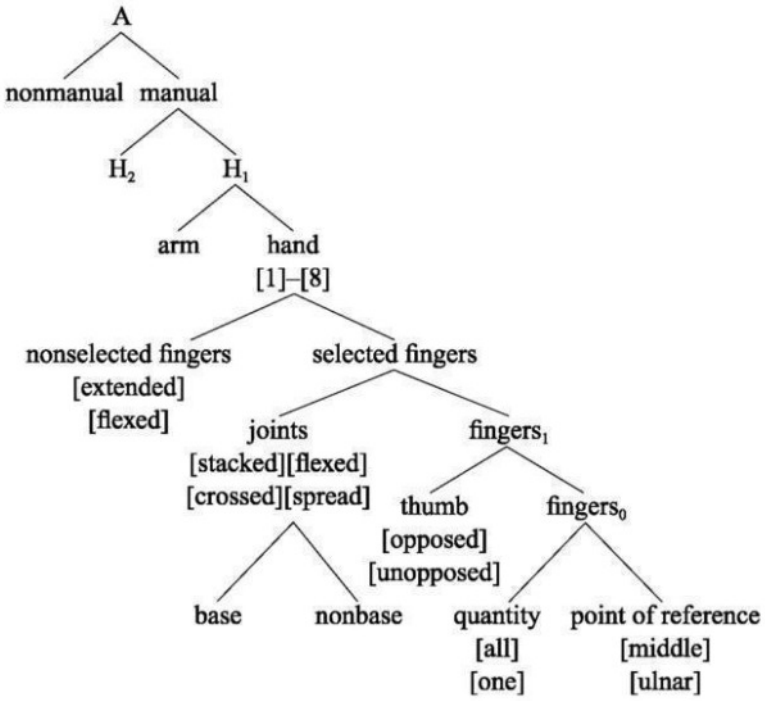


Figura 2.12: Árvore dos Traços Manuais do MP
Fonte: Brentari (1998) [13]

O detalhamento das juntas é uma característica importante, pois torna passível a descrição de configurações de mão onde os dedos estão cruzados (*crossed*), empilhados (*stacked*), flexionados (*flexed*) e separados lateralmente (*spread*). A Figura 2.13 ilustra quatro CM da Libras que necessitam desses parâmetros para sua descrição.

Figura 2.13: Exemplo de detalhamento das juntas para as CM da Libras: 1) separados lateralmente, 2) dedos empilhados, 3) dedos cruzados, 4) dedos flexionados
Fonte: Modificado pelo autor (2014) [13]

O MP também traz como característica importante um parâmetro para a descrição do antebraço (***arm***), que tem um papel importante no posicionamento das mãos no espaço de sinalização. Os POA consistem nos locais onde os sinais são articulados. Em uma linha semelhante ao MMS, o MP descreve um conjunto finito de valores para a relação espacial dos planos de articulação, para os eixos X, Y e Z (Figura 2.14).

Adicionalmente, o MP também especifica (documenta) o posicionamento fundamental de repouso (*fundamental standing position*) e o posicionamento padrão de sinalização (*fundamental signing position*) do interlocutor (Figura 2.14).

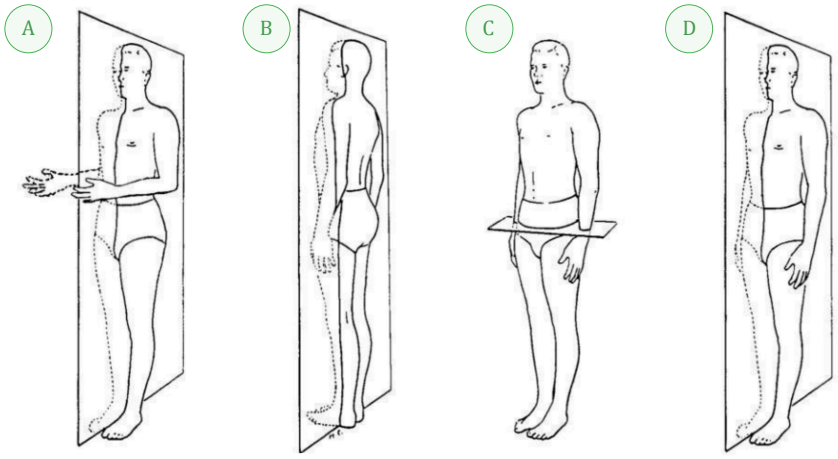


Figura 2.14: Posições Fundamentais: A) de Sinalização e B) de Repouso. Já os planos de articulação dos sinais são representados por B (plano Y), C (plano X) e D (plano Z)
Fonte: Modificado pelo autor (2014) [13]

O MP descreve os locais específicos para o corpo, de maneira semelhante aos modelos anteriores, sendo: cabeça (topo, testa, olhos, nariz, boca, acima da boca, abaixo da boca e queijo), corpo (pescoço, ombro, clavícula, peito - acima, meio e abaixo, barriga), braços (superior, cotovelo frente e trás, antebraço frente e trás e lateral, pulso frente e trás).

O MP também define o nodo PF (Figura 2.15) que contém os elementos para a descrição prosódica, do movimento e da orientação (definida como a relação entre a parte da mão e a locação, podendo haver uma variação e caracterizar um movimento).

Brentari (1998) [13] define 7 tipos de movimentos para o nodo "funcionalidades prosódicas":

1. Movimentos de trajetória (*path movements*): São articulados pelo cotovelo ou ombro, resultando em uma alteração no PA (corpo ou espaço de sinalização);

2. Características de trajetória (*path features*): Especificam a forma do movimento (e.g. arco), a direção ou o braço;

3. Movimentos locais (*local movements*): São articulados pelo pulso ou pelos dedos (nas juntas), resultando na alteração da CM ou da OP, ou um movimento de vibração;

4. Movimentos simples (*simple movements*): Envolvem somente um movimento local ou de trajetória;

5. Movimentos complexos (*complex movements*): Combinam movimentos locais e de trajetória simultaneamente no mesmo sinal;

6. Movimentos lexicais (*lexical movement*): São especificações na representação subjacente para um lexema ou afixo;

7. Movimentos de transição (*transitional movements*): São as transições que ocorrem entre os sinais durante uma frase.

Figura 2.15: Árvore das Características Prosódicas (PF) do MP
Fonte: Brentari (1998) [13]

2.2.11 Modelo de Dependência (2002)

Kooij (2002) [77] propõe um novo modelo para a Língua de Sinais Holandesa (SLN - *Sign Language of the Netherlands*) baseado no modelo de Hulst (1993): o Modelo de Dependência (MD) - *Dependency Model*. Os dados utilizados para as análises fonológicas foram armazenados em banco de dados denominado SignPhon.

O MD sub-divide um sinal em dois nodos principais: **forma de movimento** (*manner of movement*) e **articulação** (*articulation*). Esses dois nodos formam relações estruturais de dependência com os nodos filhos. Uma abstração do MD é apresentado na Figura 2.16.

Assim como os autores anteriores, o MD descreve o movimento como uma transição entre dois sub-espaços e também os aspectos de forma (qualidade) em um nodo específico.

O nodo **forma de movimento** propicia uma característica importante ao modelo: ele é capaz de descrever características de forma tanto para os aspectos manuais quanto para os não-manuais. Os atributos utilizados nesta estrutura são: tensão, repetição, direcionalidade, alternância, circunferência e cruzamento. Por exemplo, para um dado sinal é possível descrever a forma de articulação como **tensão** e, assim, afetar tanto o movimento quanto uma expressão facial, ambos sendo marcados com tensão. O nodo de **articulação** descreve os nodos **manual** (*manual articulation*) e **não-manual** (*non-*

manual articulation). A articulação não-manual compreende as marcações não-manuais presentes nos sinais.

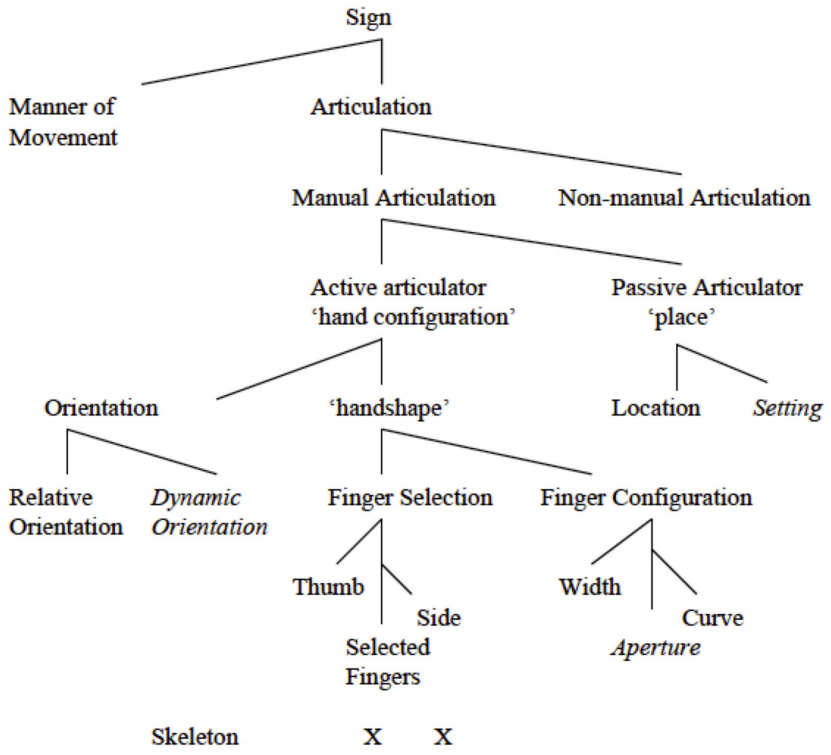


Figura 2.16: Estrutura Macro do Modelo de Dependência
Fonte: Kooij (2002) [77, p. 36]

A Articulação Manual (AM) é o centro do MD e foca principalmente nos componentes das configuração de mãos e seus elementos. A AM é sub-dividido em *articulador ativo* (que descreve toda a configuração de mão) e *passivo* (que descreve a localização e a configuração - *setting* para o espaço de sinalização).

Os elementos do MD se baseiam nas pesquisas anteriormente apresentadas, tais como: Sandler (1989) [96], Hulst (1993) [66], Brentari (1998) [13] e Uyechi (1995) [109].

2.3 Considerações sobre Modelos Fonológicos

Esta revisão dos principais modelos da fonologia das LS teve como objetivos buscar um entendimento claro de quais são os elementos que formam os sinais e da forma como eles estão estruturalmente organizados e podem ser combinados para a produção dos sinais. Neste contexto, uma estrutura conceitual formal foi definida para o modelo computacional com base em uma árvore, que agregou as características gestuais-visuais mais adequadas de cada modelo fonológico revisado.

Além disso, o modelo computacional centrado na HCI-SL deve incluir uma completude de representação (descrever quaisquer sinais), um alto nível de detalhamento (conseguir

diferenciar os sinais), regras computacionais formais para a representação (para minimizar ambiguidade e permitir um processamento adequado), entre outros.

Neste sentido, como apresentado no início deste capítulo, esta fundamentação teórica se baseia em quatro pilares principais: um entendimento sobre as LS, uma síntese das principais características dos modelos da fonologia que foram estudados e, posteriormente, as bases computacionais necessárias.

Percebeu-se neste estudo que cada modelo fonológico apresenta características específicas, mas de forma evidente incluem e adaptam os conceitos dos modelos previamente desenvolvidos.

Seguindo uma ordem cronológica, nota-se uma evolução e complementação dos modelos a partir da inserção de novos parâmetros, do detalhamento de características específicas (e.g. movimento) e de melhorias na organização estrutural baseados em novas teorias (geometria, prosódia, dependência, etc).

Como resultado deste estudo teórico, um modelo conceitual foi construído para representar as principais características analisadas em cada modelo visando a construção do modelo computacional. O conjunto dos conceitos e dos seus relacionamentos foi formalizado na forma de um mapa conceitual, apresentado na Figura 2.17.

Na revisão apresentada, ficou evidente que todos os modelos revisados têm como base principal ou como um segmento/camada os cinco parâmetros principais do MBP derivados dos estudos de [101], [10], [76] e [8]: CM (configurações de mão), OP (orientação da palma), LOC (pontos de articulação), ENM (expressões não-manuais) e MOV (movimento).

A percepção pelos surdos desses cinco parâmetros principais da fonologia das LS durante o discurso, como critério para a distinção entre os sinais, pode ser suficiente, mas como mostram os estudos de [6] e [4], mesmo percebendo esses parâmetros em um sinal, uma pessoa também percebe inconsciente e naturalmente outros detalhes específicos durante a produção do sinal (e.g. pequenas variações na locação ou expressões faciais).

Nesta situação, com o objetivo de criar um modelo computacional que contenha a capacidade de identificar corretamente os sinais, mesmo quando são muito similares, é necessário incluir um alto nível de detalhes. Assim, as características intrínsecas a cada modelo fonológico (Figura 2.17) (parâmetros, conceitos, organização estrutural, entre outros aspectos) foram compilados e adaptados em uma estrutura computacional formal e expressiva.

A organização estrutural proposta no MP [13] e no MD [77] foram úteis para a construção da base para o modelo computacional. O MP apresenta duas características principais: Inerentes (IF) e Prosódicas (PF). As IF possuem dois grupos principais: os articuladores e os pontos de articulação. Este nodo do MP tem uma similaridade com o articulador manual do MD, pois descreve todas as características das CM (dominante e não-dominante).

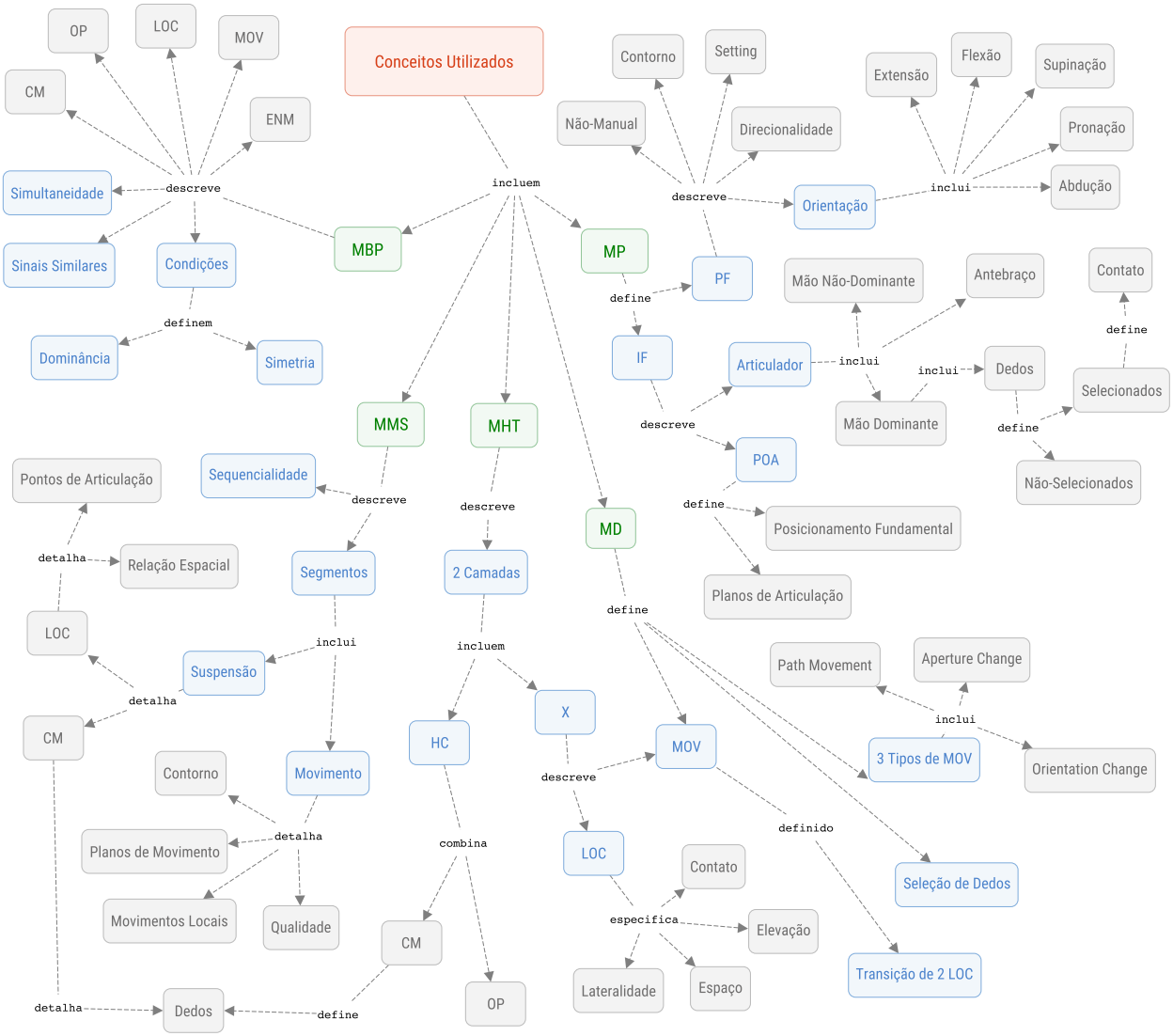


Figura 2.17: Mapa Conceitual com as principais características de cada modelo
Fonte: O autor (2014)

Da mesma forma, a classe principal do MD (*manner of movement*) tem relação com as características prosódicas do MP, no que tange às unidades responsáveis por detalhar o movimento (e.g. forma, tensão, etc.). Assim, foi possível utilizar uma combinação de ambas as estruturas para a construção da base estrutural do modelo computacional.

Cabe ainda ressaltar que o MD descreve os articuladores manuais em duas classes: ativo (CM) e passivo (LOC e o espaço). A combinação de CM, LOC e OP na mesma camada parece ser mais relevante e concisa para descrever o articulador principal. O conceito de articulador passivo do MD também foi útil para a definição de movimentos de trajetória entre a mão principal e a mão passiva.

Como foi visto, esta definição era necessária porque existem sinais onde não há alteração na CM durante o movimento (LOC-MOV-LOC), porém, podem existir sinais que possuam a CM inicial diferente da CM final. Neste caso, para cada LOC (estado inicial

e final) deve ser descrita uma CM. Esta possibilidade indica que a LOC deve estar junto com a CM.

Em relação à forma como os sinais são articulados e constituídos, percebeu-se complexidade do ponto de vista computacional de alguns aspectos das CM, MOV, LOC e ENM. Primeiro, notou-se que devido aos detalhes e à grande quantidade de articuladores na formação de uma CM, poderiam ser criadas CM muito similares ou que contivessem detalhes muito específicos necessários e diferenciadores dos sinais que as utilizam. Este problema pode ser minimizado, posteriormente, de duas maneiras: **a)** padronização por linguistas em cada LS; **b)** validação e análise das CM existentes para remover redundâncias; e **c)** conjuntos fechados de CM como apresentado em [46].

A árvore de CM é bem detalhada no MMS, especificando cada um dos dedos que compõem as CM e um atributo de relaxamento dos músculos para dar mais naturalidade à configuração. O MP, o MHT e o MD também compartilham de um detalhamento por meio de um atributo para os dedos selecionados.

Na literatura, o MOV proporciona uma grande discussão e sua especificação varia em cada modelo. Nesta tese, as características de MOV que consideramos mais adequadas foram combinadas e adaptadas para a formalização computacional.

No MMS, a complexidade do MOV se mostra pelo número de sub-classes distintivas, tais como o tipo, a frequência, a maneira, o plano, entre outras. Esses detalhes são importantes para caracterizar o movimento, principalmente quanto às suas especificidades.

No MHT, o MOV é definido pela trajetória entre dois sub-espaços, compartilhando as características da LOC. Esta proposta de estrutura foi útil porque especifica os pontos de início e fim de um sinal, fato que é importante computacionalmente.

Embora não haja uma padronização para o MOV na literatura, o modelo computacional adotou e adaptou os conceitos de modelos fonológicos distintos para a formalização das regras. Por exemplo, pareceu viável utilizar os conceitos de estados (início e fim) do MHT combinado com o detalhamento do MMS e do MP. Adicionalmente, o MOV também foi complementado com a categorização do MD, que dividiu os movimentos em três tipos (trajetória, mudança de orientação e de abertura).

Além disso, foram evidenciadas pela literatura as condições de **simetria** e de **dominância** propostas por Battison (1974) [10] em relação aos sinais realizados com ambas as mãos durante o movimento. Essas condições foram importantes também para o modelo computacional, pois minimizam o número de combinações e impõem regras para os movimentos.

Entretanto, como apresentado em Antunes (2011) [4], existem sinais na Libras que não respeitam essas condições de simetria e dominância, por exemplo, no sinal JORNAL em que as duas mãos apresentam CM distintas e ambas realizam o movimento. Esse fato pode ter sido marcado por uma variação do intérprete que realizou o sinal ou pode ter sido por um erro ainda não revisado (esta hipótese determinaria um desafio ainda maior

computacionalmente, ao se considerar que essas bases de dados possam incluir vídeos incorretos).

Vimos que as LOC também podem ter uma alta complexidade quando é necessária uma maior precisão do ponto de articulação do sinal. O MMS, como revisado em Kooij (2002) [77], apresenta um alto nível de detalhamento para as LOC. O modelo sub-divide as LOC em cabeça, corpo, mãos e no espaço - detalhando cada ponto e eventuais variações. Esse detalhamento foi necessário e é relevante no modelo computacional.

O MHT define, também, uma boa estrutura para LOC, mas não abrange todas as possibilidades para o espaço de articulação. O modelo trabalha com valores proximal (próximo ao corpo) e distal (mãos longe do corpo), mas não especifica uma distância média para a execução dos sinais.

O aspecto de ENM talvez seja o atributo de maior complexidade computacional, pois além de ser um traço articulatório distintivo na fonologia, pode representar marcas prosódicas e verbo-visuais - o que gera um desafio, principalmente em sistemas de PLN, para distinguir entre os aspectos inerentes ao sinal e as características do sinalizador.

Neste contexto, as ENM são elementos importantes na constituição dos sinais, podendo até mesmo serem o único traço articulatório na representação de um sinal (i.e. sinais compostos somente de ENM). Uma análise aprofundada das ENM na Libras foi realizada por Felipe (2013) [42], mostrando que as marcações não-manuais podem representar ao sinal diversos níveis gramaticais.

Uma premissa importante compreendida nesta revisão de literatura, que foi fundamental para esta tese, é a questão de que as LS e, em particular a Libras, possuem a mesma modalidade gestual-visual e utilizam as mesmas propriedades e traços fonéticos na produção dos sinais [44], [46], [48], [74], [30], [15], [51], [13]. Esta premissa foi fundamental, para que o modelo computacional, que contempla os modelos fonológicos, também possa, por hipótese a ser provada posteriormente, representar sinais de quaisquer LS.

Os modelos apresentados e os estudos em relação a fonologia, além de incluírem de alguma forma os parâmetros principais da fonologia (MBP) em sua estrutura, são derivados das pesquisas na ASL. Como mostra Felipe (2007) [48], os estudos linguísticos sobre a Libras teve início em 1980 baseado na ASL, com pesquisas também do ponto de vista estruturalista e gerativista.

Xavier (2006) [114] mostra que a estrutura proposta no MMS [80] é capaz de descrever os sinais da Libras. Segundo Brentari (2011) [14] outros pesquisadores tem mostrado que essas cinco unidades fonológicas principais e suas combinações formam os sinais de outras LS, tais como a Língua de Sinais Holandesa e a Língua de Sinais de Israel, mostrando que as LS possuem propriedades similares e são de mesma modalidade.

Liddell (2011) [71] explana que em uma perspectiva fonológica os sinais são formados pelas CM, LOC, OP, MOV e ENM, e são traços contrastivos e distintivos nas LS, como exemplificado também na Língua de Sinais de Israel.

Para Meir et al. (2007) [85] o recurso de design universal da linguagem humana torna possível a criação de um vasto vocabulário de formas significativas a partir de um número limitado/finito de unidades fonológicas.

Na pesquisa de Karnopp (1999) [74] é mostrado que na aquisição da LS, bem como na fonologia, a criança adquire uma estrutura de representação fonológica básica, composta por elementos que formam o núcleo em uma abordagem baseada na Fonologia de Dependência. Neste sentido, esta base inicial de representação fonológica é formada pelos parâmetros de CM, LOC e OP, sendo que o MOV é considerado uma consequência da transição entre LOC ou CM.

Karnopp (1999) [74] explica que a LOC e CM *"expressam propriedades nucleares determinadas pela Gramática Universal (GU)"*, ou seja, na aquisição da fonologia pressupõe-se que a articulação de qualquer sinal (mesmo simples) exija a utilização de um núcleo tanto de CM quanto de LOC. Karnopp (1999) [74] diz que *"a literatura tem sido unânime em apontar que a configuração de mão, ponto de articulação, movimento, orientação de mão e expressões não-manuais são os componentes formacionais dos sinais"*.

Outro aspecto importante observado na literatura consiste nas restrições para a formação dos sinais. Segundo Karnopp (1999) [74], devido às restrições físicas e linguísticas somente algumas combinações entre as unidades de CM, LOC, OP e MOV podem ser especificadas para a formação dos sinais, ou seja, restrições do próprio sistema perceptual (visual) e em relação ao sistema de articulação (limitado pela anatomia das mãos e dos demais membros).

As propriedades do sistema de percepção visual também restringem a produção de sinais, pois o receptor tende a fixar seu olhar na região da face do interlocutor, principalmente, devido às ENM [74]. Por isso, existe uma maior facilidade do receptor em reconhecer variações sutis na face do que nas CM, MOV e LOC (Siple, 1978 [74]).

Esse fato também é percebido na pesquisa de Battison (1978) [11] que mostra que sinais são articulados com maior frequência nesta região facial, onde há um número maior de diferentes locações. Além disso, o campo de percepção dos sinais também depende da visão periférica, e desta forma, os sinais tendem a ser localizados mais em um ponto de vista central.

Essas regras e restrições foram adaptadas ao modelo computacional, visando a minimização do número de combinações das sub-unidades fonológicas, e contribuindo, também, para garantir uma estrutura robusta e não-ambígua, que possuísse regras de formação e de representação dos sinais bem definidas.

A questão dos pares mínimos, mostrada inicialmente por Battison (1974) [10], é uma questão fundamental para o tratamento computacional. Primeiramente, Liddell (2011) [71] fala que o conceito de pares mínimos é frequentemente usado em pares de sinais que possuem uma mesma sequência (forma) de fonemas, exceto uma (ou seja, os sinais variam apenas em uma propriedade).

A hipótese de existirem sinais muito similares entre si nas LS determina um desafio significativo para o modelo computacional construído nesta tese: um sistema de indexação e busca não poderia dispor de um recurso de pesquisa exata e, portanto, deve trabalhar no desenvolvimento de um sistema de busca por similaridade, o que implicou em pesquisar formas de construir funções de distância para calcular o grau de similaridade entre uma entrada e os demais sinais presentes em uma base. Adicionalmente, também foram estudados os ruídos e variações de cada pessoa na entrada do sistema de reconhecimento, bem como os traços distintivos muito semelhantes visualmente (i.e. no caso de algumas CM, LOC e ENM).

Na literatura, percebeu-se que a maioria dos modelos desenvolvidos após a pesquisa de Liddell & Johnson (1989) [80] contempla os conceitos de sequencialidade e simultaneidade. Ou seja, é de consenso e demonstrado pelas pesquisas linguísticas que os sinais são formados pela organização simultânea de algumas sub-unidades, mas nos casos de sinais que façam a alteração de um dos seus cinco parâmetros básicos na articulação de um sinal, a especificação da sequência correta é também necessária.

Ao analisar a estrutura dos modelos apresentados juntamente com os conceitos abordados em relação a movimentos e à sequencialidade, uma combinação das estruturas do MMS, do MD e do MP pareceu mais adequada ao modelo computacional, por possuir um alto nível de detalhamento (dispondo de características para descrever cada uma das cinco classes de parâmetros principais, ou seja, descrever em detalhes as CM, LOC e MOV), além de trabalhar no conceito de segmentos de suspensão (segmento estático) e de movimentos (segmento dinâmico). A transição entre duas LOC trouxeram o caráter de início e fim, necessários computacionalmente.

Neste sentido, no modelo computacional precisaram ser modeladas regras nas quais sempre que houvesse movimento em um sinal fosse definido por início-movimento-fim (i.e. apresentando um estado estático, um dinâmico e um estático para representar o fim do sinal).

Esta característica fica mais evidente quando se analisam os demais modelos, como o MHT, que tratam o movimento como uma transição entre duas locações. Ao se considerar que o segmento de suspensão do MMS contempla o parâmetro de locação, então o movimento consiste de um segmento dinâmico entre duas suspensões (MMS). A questão fundamental consistiu, também, em aproveitar as estruturas dos modelos que fornecem um melhor detalhamento ao MOV, para proporcionar ao modelo computacional mais traços distintivos.

Além disso, considerar segmentos estáticos e dinâmicos mostrou-se interessante principalmente para sinais que sejam formados por uma sequência de CM (soletrações), que apenas usem uma pequena variação de CM ou que variem a LOC sem que haja necessariamente uma trajetória (e.g. alterando a CM ou a OP).

O conjunto de características (CM, LOC, MOV, OP e ENM) do MBP ainda é pre-

dominante como o conjunto principal de traços fonológicos utilizados para a formação e distinção dos sinais durante o discurso, pois seria complexo ter que analisar um conjunto muito grande de traços articulatórios [30] [4].

Embora o MMS tenha um grande número de características, cerca de 299 sub-unidades [13], e seu uso na prática das línguas de sinais pareça inviável, computacionalmente, o modelo traz um conjunto adequado de elementos capazes tanto de formar quanto de distinguir os sinais, pois detalha cada um dos parâmetros principais (MBP) em relação a maneira como são formados.

Por exemplo, como as CM pré-definidas apresentam diferenças entre cada LS, era preciso ter uma estrutura capaz de descrever e formar quaisquer configurações. Logo, foi necessário considerar todos os elementos articulatórios que compõem as CM. No mesmo sentido, no parâmetro de LOC a estrutura computacional deveria ser capaz de, além de representar as LOC nas mãos e no corpo, representar os pontos no espaço de sinalização onde os sinais são articulados. Esses elementos são importantes para que um sistema de reconhecimento possa diferenciar um espaço neutro de um ponto específico no espaço (constituinte do sinal), bem como para que um sistema de processamento gráfico possa sintetizar adequadamente um avatar.

Brentari (1998) [13] analisa o MMS e remove diversas redundâncias no modelo, construindo o MP em uma base sólida e sem repetições. No parâmetro de MOV, também pareceu relevante considerar as características para a representação de movimentos locais, pois existem uma série de sinais que não apresentam trajetória no espaço, mas têm variações nas CM nos dedos ou pulsos.

2.4 Sistemas Computacionais para a Representação de Sinais

Esta seção tem o intuito de apresentar os trabalhos relacionados diretamente com esta tese: os sistemas e as modelagens computacionais propostos na literatura para descrever / representar os sinais das LS computacionalmente, bem como seus contextos de aplicação e suas características principais.

Neste sentido, pretendeu-se explorar, no texto, as vantagens e as limitações dos modelos existentes visando explicitar propriedades e questões fundamentais que deveriam ser resolvidas pelo modelo proposto nesta tese. Também, os modelos existentes foram aqui analisados em relação ao seu contexto de aplicação (pois a aplicabilidade do modelo proposto dentro da Arquitetura HCI-SL foi um dos objetivos desta tese).

Para facilitar a compreensão e o escopo de cada trabalho, os modelos computacionais da literatura foram classificados em três tipos principais em relação à abordagem de representação utilizada. Esta classificação , determina três conjuntos de modelos [42]:

1. **Modelos Baseados em Linguagens de Marcação Gestual (LMG):** Esta

classe de modelos é caracterizada por adaptar modelagens ou linguagens de marcação de gestos humanos para representar os sinais das LS.

No geral, são utilizados recursos para representar gestos com o intuito de, em um avatar 3D, deixar mais natural o discurso de um intérprete virtual para a língua oral, ilustrando gesticulações inerentes a uma pessoa durante um discurso.

Não é dado um caráter específico sem um tratamento adequado para o contexto das LS nesta classe de modelos, mas eles devem ser analisados para fins de completude das possibilidades anatômicas de movimento .

2. **Modelos Baseados em Sistemas de Escrita de LS (SELS):** Os modelos desenvolvidos nesta abordagem utilizam como recurso principal, e para a organização da estrutura computacional, os sistemas de escrita existentes para as LS, modelando os símbolos gráficos em estruturas do tipo XML (*EXtensible Markup Language*), entre outras.

   Como visto a seguir, estes sistemas de escrita não contemplam alguns aspectos imprescindíveis para a formação dos sinais (e.g. sequencialidade, parâmetros e detalhes adicionais, ENM, entre outros). Portanto, não determinam uma abordagem robusta em relação ao tratamento computacional das LS.

   Novamente, cabe, mesmo assim, revisar esta classe de trabalhos, para entender como os sistemas de escrita estão sendo representados computacionalmente, visto que uma das aplicações previstas na arquitetura de hipótese era possibilitar a visualização (*output*) dos sinais na forma escrita (gráfica).

3. **Modelos Baseados na Fonologia das LS (FLS):** Consistem em modelagens computacionais que buscam a utilização de modelos fonético-fonológicos (Linguística das LS) para tentar representar os sinais. Esta é uma abordagem preferível às anteriores, pois se baseia em uma visão mais ampla de como os sinais são constituídos, possibilitando a representação dos sinais das LS e, além disso, podendo não ser limitados na capacidade de representar sinais.

   Entretanto, uma análise mais aprofundada precisou ser realizada a fim de verificar se essas modelagens consideravam uma visão ampla desses sistemas linguísticos e se demonstravam propriedades importantes para a representação dos sinais em diversos contextos computacionais.

   Neste sentido, faz-se uma sub-divisão nesta abordagem em dois tipos de modelagens: **a)** as que englobam apenas os cinco parâmetros principais do MBP, e **b)** as que consideram algum modelo fonético-fonológico que aborda conceitos de segmentos e sequencialidade.

2.4.1 LMG

Chung & Lee (2004) [22] propõem o MCML (*Motion Capture Markup Language*), que consiste em uma linguagem de marcação baseada em XML, com o intuito de criar um modelo para dar suporte à tecnologias de captura de movimentos usadas para resolver problemas de animação em tempo real (para permitir que com os dados de movimento capturados possam gerar melhores animações).

Uma desvantagem deste modelo é que sua estrutura é utilizada para representar apenas movimentos (gestos) humanos, ou seja, não foi desenvolvido dentro do escopo das LS, como tão pouco não representa demais traços articulatórios fundamentais para a formação de sinais.

Embora o modelo proposto por Chung & Lee (2004) [22] não contemple especificamente a LS, ele deixa em evidência um aspecto importante para o estudo desenvolvido por esta tese: a utilização de um modelo para representar as características gestuais, visuais e espaciais, neste caso das LS, seria capaz de viabilizar um ambiente de síntese (geração) por meio de animação com maior precisão, mais naturalidade e expressividade.

Em um contexto semelhante, o VHML (*Virtual Human Markup Language*), proposto por Marriott (2001) [82], consiste em um conjunto de sub-sistemas com o objetivo de especificar uma maneira realista, fácil e natural para a interação entre uma pessoa e um avatar (*virtual human*) na web.

O VHML é formado por três camadas de sub-sistemas: 1) uma camada com o EML (*Emotional Human Markup Language*) e o GML (*Gesture Markup Language*) para respectivamente controlar traços de emoção e gestos no discurso humano; 2) uma camada definida pelo SML (*Speech Markup Language*) e o BAML (*Body Animation Markup Language*) responsáveis respectivamente por marcações de fala e para marcações de gestos corporais; e 3) uma camada que define a FAML (*Facial Animation Markup Language*) responsável por mapear os traços de expressões faciais utilizados no discurso.

Com o VHML é possível especificar no discurso escrito de uma língua oral-auditiva os traços prosódicos, as expressões emotivas, as marcações gestuais, a fim de obter uma representação virtual humana mais natural e próxima da realidade durante um discurso. O VHML, segundo [22], consiste em uma linguagem projetada para simular ambientes tridimensionais na web e por meio do suporte do H-ANIM, permite especificar a estrutura para a modelagem tridimensional de um avatar.

Assim, mesmo que o VHML, assim como o MCML, não seja um modelo de representação específico para LS, vale destacar que sua característica de especificar as configurações e as disposições do esqueleto, das juntas, dos movimentos geométricos, das expressões corporais e faciais, entre outros, indica que o uso de um modelo computacional robusto com foco particular nas LS e que contemple todos os seus atributos para distinção dos sinais pode impulsionar o desenvolvimento de sistemas de síntese por meio

de Avatares 3D que sejam mais naturais e expressivos para os usuários finais.

Por exemplo, no EML são apresentados um atributo de **intensidade** para a expressão de emoção e de **duração** para descrever o tempo de duração da emoção. Este parâmetro de intensidade pode ser relacionado, por exemplo, com os aspectos de qualidade dos modelos fonéticos revisados, como no caso da sub-unidade de tensão. Adicionalmente o EML descreve algumas expressões emotivas padrão: surpresa, felicidade, raiva, expressão padrão, neutro, cansado, confuso, entre outras.

No VHML, também são descritas algumas expressões faciais específicas (no FAML): olhos para os lados ou para cima, balanceamento da cabeça, olhar para baixo e para os lados, entre outras.

O MURML (*Multimodal Utterance Representation Markup Language*), Kranstedt et al. (2002) [79], foi desenvolvido para um contexto semelhante ao VHML, ou seja, consiste em uma linguagem de marcação baseada em XML, focada nas inter-relações que existem em diversas modalidades de conversação face-a-face, no que diz respeito a aspectos verbais e não-verbais, bem como à síntese automática de gestos e fala.

Ao analisar a revisão de literatura sobre as modalidades linguísticas e no tratamento de LS, o autor [79] considera que o objetivo do modelo proposto é incluir sistemas de notações de LS para que possa representar os sinais das LS, e não os gestos.

O problema é que o MURML, inicialmente, não faz uma distinção clara entre aspectos gestuais e as expressões não-manuais das LS com as características de paralíngua intrínsecas às línguas orais. Ou seja, o modelo tem como foco principal dar mais expressividade na geração de agentes virtuais (avatares) para interação / síntese da fala, e para os conceitos gestuais se baseia em um sistema de transcrição das LS: o HamNoSys [94].

Kranstedt et al. (2002) [79] explanam que *"para representar um gesto que se destina a transmitir uma determinada intenção comunicativa, basta especificar as características espaço-temporais essenciais que compõem a morfologia ao longo de sua fase significativa"*. Neste sentido, o MURML contempla esses recursos a partir de elementos como localização, forma das mãos, orientação, planos de movimento, entre outros, pertencentes ao sistema HamNoSys [94].

Nos exemplos apresentados no trabalho [79], as representações em XML incluem simultaneamente uma especificação textual da língua oral, bem como a especificação dos gestos (com base no sistema HamNoSys [94]). Essa especificação de gestos não tem o intuito de representar sinais das LS, mas tem uma característica interessante no sentido de fazer a marcação no modelo tanto do enunciado da língua oral, quanto de um sinal, mostrando-se um conceito que pode ser significativo se abordado em processos de PLN que envolvam aprendizado baseado em exemplos.

Na mesma linha, Kopp et al. (2006) [78] propõem o BML *Behavior Markup Language*, uma linguagem para marcação em XML de comportamentos humanos durante o discurso, tais como questões relativas a simultaneidade, repetições, emoções, movimentos, entre

outras.

A BML também tem como objetivo modelar aspectos gestuais e faciais, a fim de produzir agentes virtuais 3D com maior expressividade e naturalidade, para simular melhores ambientes de comunicação virtual.

Segundo del Puy Carretero et al. (2005) [32], as principais características necessárias à construção de um mecanismo de animação (síntese de gestos) são a animação facial, a animação corporal, a produção texto-fala e a representação de emoção, além de características de diálogo (essenciais para a integração do avatar em dispositivos de telecomunicação como TV Digital, entre outros).

Neste sentido, os autores [32] realizam uma análise comparativa das principais linguagens de marcação com foco em animação de avatares ou agentes virtuais 3D, tais como VHML, MURML, AML (*Avatar Markup Language*), CML (*Character Markup Language*), entre outras. O nível de detalhes inerentes à representação utilizada é proporcional à capacidade do avatar em apresentar uma comunicação mais natural, sendo importante criar uma animação facial natural, bem como uma expressividade corporal realista.

Novamente, destaca-se que essas linguagens são constituídas de marcações para gestos e possibilidades de expressões faciais e do movimento humano, não tratando especificamente as LS. Por exemplo, na maioria desses modelos, parâmetros como configurações das mãos e expressões não-manuais são pré-definidos na estrutura (e.g. não é possível descrever em sub-unidades uma expressão facial de alegria). Adicionalmente, nota-se que nestes modelos não há a especificação de pontos de articulação, uma vez que eles não representam parâmetros das LS.

No entanto, de maneira análoga à empregada com os sistemas anteriores, notou-se que os conceitos utilizados poderiam servir de base para o modelo computacional proposto, no intuito de garantir a completude dos movimentos e expressões dentro das regras definidas para a produção dos sinais. Percebe-se que, se existir um recurso de avatar 3D automático dentro dessas possibilidades, será possível mapear o modelo computacional das LS neste agente virtual para que este conjunto passe a ser um sistema que sintetize os sinais das LS.

Considera-se que os modelos que utilizam linguagens de marcação de gestos poderiam incorporar dois atributos principais para representar as LS: o primeiro, a especificação de um conjunto de traços distintivos que formam os sinais das LS (não seria suficiente cobrir todas as possibilidades de movimento, mas era necessário, também, ter um conjunto de traços articulatórios mínimo capaz de diferenciar os sinais representados computacionalmente, a fim de poder ser aplicado em questões de indexação, busca, entre outras, ou seja, não poderia ser considerado de maneira isolada para a geração de avatares 3D); e o segundo, a especificação de um conjunto de regras bem definido e expressivo que combine essas sub-unidades distintivas para a produção de sinais.

2.4.2 SELS

Adicionalmente, existem uma série de modelos desenvolvidos para representar computacionalmente os sinais com base em sistemas de escrita das línguas de sinais. Como é de conhecimento, qualquer forma escrita de uma língua perde informações importantes acerca do discurso, como expressões faciais, gestos que indicam intensidade, marcações de ritmo, entre outros.

Por outro lado, como mostrado nos trabalhos de Antunes (2011) [4] e Antunes et al. (2011) [6], o nível de detalhamento inerente às LS torna necessário o uso de uma estrutura que contenha todos os traços articulatórios capazes de expressá-la. Devido à diferença de modalidade em relação às línguas orais, as LS precisam de todos os recursos gestuais e espaciais em uma representação computacional. Por exemplo, como citado anteriormente, existem sinais em que a ENM é o único traço que define o sinal, portanto, sua consideração é imprescindível para representar adequadamente os sinais.

Como os sistemas de escrita das LS tem o objetivo de proporcionar uma maneira prática, simples e intuitiva de registrar o conhecimento, uma forma pela qual o usuário consiga utilizar de todo o potencial inerente à forma escrita das LS, na maioria das vezes eles não contemplam uma série de características próprias deste tipo de língua.

Neste sentido, o HamNoSys *(Hamburg Sign Language Notation System)* [94] consiste em um sistema alfabético de símbolos gráficos para a transcrição fonética dos sinais de forma linear, possuindo aproximadamente 200 símbolos para a transcrição das CM, OP, LOC na cabeça e no tronco e os movimentos por meio de representações icônicas que facilitam o reconhecimento e o entendimento pelo usuário.

O sistema foi criado para a aplicação em qualquer LS, o uso da iconicidade para facilitar a lembrança e o aprendizado, facilitar a integração com o computador e permitir extensão quando necessário. Uma característica importante do HamNoSys [64] é a forma de descrição da orientação da palma. O conjunto de valores considera o braço na vertical e o braço na horizontal, dispondo de nove possibilidades de configuração. Esta característica pode ser utilizada para melhorar a descrição da OP no modelo computacional (Figura 2.18).

Percebe-se que uma limitação inicial do sistema era a falta de uma maneira de descrever a simultaneidade das sub-unidades articulatórias distintas que formam um certo sinal, pois eles são representados de forma sequencial e linear. Além disso, o sistema parece não ser capaz de representar marcações das LS tais como expressões não-manuais (faciais) e detalhes em cada parâmetro tal como as CM (especificação dos dedos), e a LOC (detalhes de precisão).

Baseado no HamNoSys, Elliott et al. (2001) [75] propõem o SiGML *(Signing Gesture Markup Language)*, um sistema de transcrição computacional representado por XML (um formato simples e flexível para a troca de dados estruturados e semi-estruturados), que

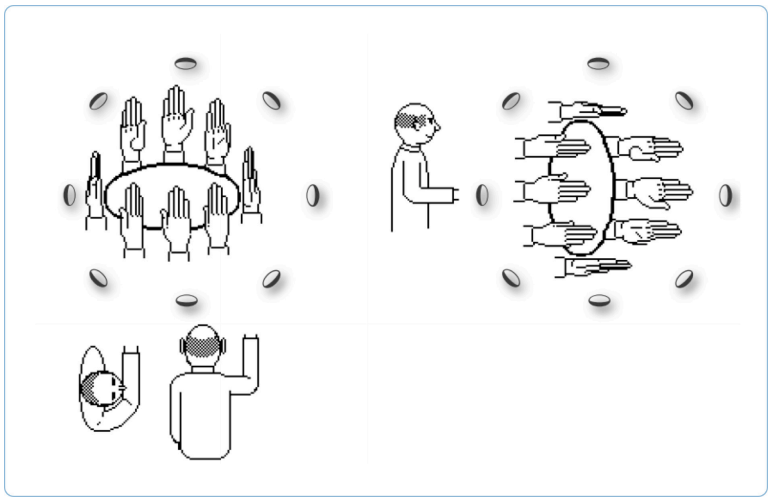


Figura 2.18: Documentação da OP no HamNoSys
Fonte: Hanke (2004) [64, p. 2]

permite representar uma sequência de sinais das LS com a finalidade de poder executar ou gerar estes sinais em tempo real por meio de um agente virtual (avatar).

Glauert et al. (2004) [57] utilizam o SiGML para a construção de um agente virtual sinalizador ou avatar 3D para a execução de sinais da BSL (*British Sign Language*). Segundo os autores, um modelo de representação como o SiGML também permite que dados de entrada, além de forma manual, possam ser coletados por meio de tecnologias de captura de movimento, obtendo, assim, um conjunto de valores mais precisos.

Cabe ressaltar que o uso de certas tecnologias de captura pode restringir a naturalidade do usuário durante a execução dos sinais, consequentemente, reduzindo a precisão.

O modelo, segundo Jemni & Elghoul (2008) [70], traz uma dificuldade no momento de representar ou executar o sinal, devido ao contraste existente entre a linearidade deste sistema de transcrição e a não-linearidade da LS. Um exemplo do SiGML é apresentado na Tabela 2.4.

O SiGML é basicamente composto de duas estruturas: manual e não-manual. A primeira é responsável por descrever os aspectos manuais, bem como de quantas e quais as mãos que executam o sinal e quais os movimentos realizados (quando há); a segunda camada é responsável pelos movimentos e expressões feitas pela face, ombros e cabeça [57].

Adicionalmente, o usuário que irá descrever o sinal deve conhecer e entender os *" símbolos e seu equivalente na língua de sinais. Na verdade, o usuário deve descobrir os parâmetros de formação de sinais (configuração de mão , orientação , localização , movimento e expressão facial) e representá-los com o uso desta transcrição"* [70]. Portanto, o SiGML não é um sistema que proporcione completude em relação às características fonológicas, robustez e expressividade na representação, à facilidade de descrição ou recuperação e atributos para a precisão exigida computacionalmente.

Tabela 2.4: Exemplo de Representação de um Sinal em SiGML

```
<sigml>
  <hns_sign gloss="DGS_going-to">
    <hamnosys_manual>
      <hamsymmpar/><hamfinger2/>
      <hamthumboutmod/><hamextfingeruo/>
      <hampalml/><hamparbegin/>
      <hammoveo/><hamarcu/>
      <hamreplace/><hamextfingerdo/><hamparend/>
    </hamnosys_manual>
  </hns_sign>
</sigml>
```

Fonte: Jemni and Elghoul (2008) [70, p.3]

O *SignWriting* - Sutton (1981) [105] - consiste de um sistema de notação gráfica para a escrita das LS. Seus elementos, que representam os traços articulatórios que formam os sinais, permitem que quaisquer LS sejam representadas visualmente na modalidade escrita [4].

Para Capovilla & Raphael (2001) [16], o *SignWriting* tem o intuito de ser, além de uma notação científica, um sistema simples e prático de escrita para as LS, proporcionando aos surdos uma forma de comunicação e de registro escrito do conhecimento.

O *SignWriting* [105] faz parte de um sistema mais amplo para a representação de movimentos, o *Sutton Movement Writing & Shorthand*, que tem a capacidade de representar sistematicamente qualquer movimento (e.g. dança, mímicas, esportes, fisioterapia, entre outros) [16].

Segundo Stumpf (2005), o *SignWriting* [105] oferece um sistema robusto com a capacidade de registrar graficamente quaisquer LS, *"funcionando como um sistema alfabético, no qual as unidades gráficas correspondem às unidades que formam os sinais"* [4].

Com base no *SignWriting* [105], Costa (2000) [25] desenvolve o SWML (*SignWriting Markup Language*), uma linguagem de marcação em XML que foi desenvolvida com o objetivo de ser utilizado em sistemas computacionais relacionados às LS, com um caráter específico para o processamento do SignWriting, neste sentido, proporcionando a capacidade da troca de textos em diversos sistemas utilizando a mesma forma escrita [53]. Um exemplo de sinal representado em *SignWriting* e em SWML pode ser visto na Tabela 2.5.

Como o *SignWriting* consiste em uma notação gráfica para a escrita das LS, certos

> *"parâmetros importantes como velocidade, frequência, marcações de sequencialidade, entre outros, não são disponibilizados, pois, na maioria das vezes, são percebidos de forma natural e inconsciente pelos usuários das línguas de sinais. No entanto, para um tratamento computacional robusto pode ser necessário mais detalhes da descrição dos parâmetros fonológicos dos sinais*

(e.g. velocidade e frequência podem ser indicativos de intensidade)" (Antunes, 2011 [4]).

Tabela 2.5: Representação em *SignWriting* e em SWML respectivamente

	`<signbox>`
	`  <symb x="46" y="37" x-flop="0" y-flop="0" color="0,0,0">`
	`  <category>04</category>`
	`  <group>02</group>`
	`  <symbnum>001</symbnum>`
	`  <variation>01</variation>`
	`  <fill>01</fill>`
	`  <rotation>04</rotation>`
	`  </symb>`
	`  ...`
	`</signbox>`

Fonte: Modificado pelo autor [53, p.4]

Nesta modelagem em XML também podemos notar a falta de usabilidade e de legibilidade. Como a linguagem é específica para SignWriting, ela especifica o posicionamento dos elementos por meio de coordenadas e as categorias com números não documentados (e.g. o que significa "rotação 4"?). Para um modelo computacional conseguir ser aplicado em diversos cenários e utilizado por qualquer pessoa, deve-se documentar cada um dos valores, além de permitir uma fácil leitura pelo seu usuário.

Destaca-se que esta seção teve o objetivo de apresentar modelagens computacionais desenvolvidas a partir de sistemas que representam os sinais de forma escrita, tal como o Sign Writing e o SWML. Existem outros sistemas de escrita de sinais, mas não foram considerados nesta revisão literária devido à falta de uma versão modelada em uma linguagem computacional.

2.4.3 FLS

Como discutido na revisão sobre os modelos fonológicos, deve-se considerar um maior nível de detalhes articulatórios para representar os sinais computacionalmente, de maneira a garantir a correta descrição e a distinção entre os sinais. Além disso, é importante considerar outros aspectos tais como os atributos de sequencialidade dos sinais, os sinais compostos e as soletrações.

Felice et al. (2007) [41] propõem o e-LIS, uma linguagem para a representação de sinais para a LIS (Língua Italiana de Sinais), com o objetivo específico de auxiliar na construção de um dicionário eletrônico. Este modelo consiste de uma ontologia baseada nos parâmetros de CM, LOC, MOV e OP definidos pelo MBP fazendo uma relação desses elementos e categorizando-os por sub-classes.

Por exemplo, as CM são sub-divididas em classes de acordo com o número de dedos selecionados (e.g. um sub-grupo somente de CM com apenas 2 dedos selecionados). O objetivo dessas sub-classes dentro da ontologia proposta é criar um melhor filtro nas tarefas de busca que o usuário pode fazer na interface do dicionário.

Como cita [36], este modelo define 56 possíveis configurações de mão (não apresentando regras para descrição/criação de novas CM, além de não implementar uma estrutura de representação para as ENM. Desta maneira, o modelo limita o sistema na representação de sinais que não contenham ENM.

No trabalho de Fusco & Brega (2009) [55] é apresentado um sistema para o gerenciamento de vocabulários para a Libras: o X-Libras. O sistema consiste em um ambiente informacional que utiliza XML para a representação dos metadados das propriedades articulatórias que constituem cada sinal, com o intuito de fornecer uma arquitetura padronizada para consulta de sinais em Libras.

Como analisado por [36], o modelo proposto se restringe às especificações fonéticas apresentadas por Brito (1995) [15], desconsiderando, entre outros parâmetros, todos os tipos de movimentos (detalhamento maior) e não possibilitando a utilização de CM em que haja um cruzamento dos dedos.

Novamente, esse modelo de representação tem como contexto a representação de avatares 3D para síntese de sinais quando consultados no vocabulário. Portanto, não tem associados estudos adicionais de como este modelo poderia ser utilizado como suporte para a solução de outros sub-problemas computacionais das LS, tal como os previstos na arquitetura de hipótese.

O FleXLIBRAS [36] [35] tem o objetivo de descrever uma linguagem formal e expressiva para descrever os sinais da Libras a partir da especificação de cada um dos cinco parâmetros principais (CM, OP, LOC, MOV e ENM) para que um certo sinal representado possa ser gerado por um avatar humanóide 3D.

Segundo os autores, a linguagem proposta, baseada em uma estrutura em XML, é flexível e permite que novos parâmetros sejam especificados no sistema se necessário, como exige o caráter dinâmico da Libras. Nesse trabalho, é considerada uma abordagem baseada no MBP estudado por Ferreira Brito (1995), pois considera que os sinais são compostos pelos cinco parâmetros básicos da fonologia. O modelo [36] considera 60 CM definidas na Linguística da Libras e demais elementos descritos no estudo de Ferreira Brito (1995) e parece não definir a estrutura de composição das CM, ou seja, a estrutura das mãos, dos dedos e das suas possibilidades de organização.

Para avaliar o modelo proposto, os autores realizaram um estudo de caso por meio da construção colaborativa de uma base de vocabulário para Libras. Entende-se que o FleXLIBRAS foi desenvolvido para um contexto bem específico de uso, ou seja, uma linguagem intermediária para a representação dos sinais em um "dicionário"para permitir que um avatar 3D possa gerá-los posteriormente. Portanto, esse modelo também não

contempla os usos adicionais previstos da Arquitetura HCI-SL.

Os autores [36] definem que na linguagem proposta cada sinal consiste de *"um conjunto de movimentos, onde cada movimento possui uma configuração inicial e final das mãos, braços e face, um tipo de trajetória [...] , uma direção [...], além de flags para indicar quais mãos são usadas"* [35].

Desta maneira, o sinal é entendido como um conjunto de movimentos que possuem uma CM inicial e final. Certamente, esta é uma grande restrição à representação computacional, pois, como mostrado anteriormente, existem diversos sinais que não utilizam o movimento, ou mesmo sinais que são constituídos apenas de ENM. Também, nota-se a falta de uma maior especificação para locações no espaço de articulação, bem como uma estrutura que permita descrever tanto a simultaneidade como a sequencialidade para o segmento de ENM.

Em relação à inserção de novos parâmetros no modelo, o FleXLIBRAS define uma estrutura que utiliza uma biblioteca de poses, que consistem na configuração de descritores específicos de uma linguagem de marcação para avatares 3D, sendo necessário especificar posicionamento, rotação, contato em cada osso etc.

Esta característica no FleXLIBRAS, fundamental para um modelo computacional, traz consigo a dificuldade de os usuários precisarem conhecer, de forma textual e aparentemente sem referência gráfica, quais os parâmetros necessários e como eles podem modelar o avatar. Por exemplo, qual a configuração de mão 17? É preciso especificar este modo gráfico / visual para que o modelo disponha da propriedade de usabilidade e, neste sentido, possa ser fácil e corretamente utilizado.

Esse modelo [36] também parece não especificar regras de produção de sinais bem definidas e, desta maneira, podem ser representados sinais de forma incorreta ou até mesmo de forma ambígua. Por exemplo, como o modelo define que um sinal possui um CM inicial e final, parece não ser possível a descrição de uma soletração (várias CM em sequência e sem movimento).

Ye et al. (2009) [115] apresentam o CSLML (*Chinese Sign Language Markup Language*), um sistema desenvolvido para a síntese da CSL (*Chinese Sign Language*) que, além das características da fonologia, inclui uma estrutura para aspectos relacionados à prosódia. O CSLML [115], Figura 2.19, é composto por dois níveis:

1. **camada funcional**: responsável por fornecer informações abstratas em relação ao conteúdo dos sinais e aos aspectos prosódicos, facilitando a descrição textual dos sinais com o intuito de ser aplicado em sistemas de síntese automática;

2. **camada fonética**: que representa os traços articulatórios baseados na fonética e na fonologia que objetiva interpretar o significado da camada funcional.

Segundo os autores [115], as características prosódicas presentes na articulação dos sinais constituem um fator importante para o realismo da síntese das LS. Neste sentido,

Ye et al. (2009) [115] consideram que a informação contida na representação das LS pode ser classificada em: conteúdo e prosódia.

A informação de conteúdo define uma sequência de atributos não-verbais para construir o discurso (de acordo com a estrutura gramatical, tais como fonologia, morfologia, sintaxe etc), sendo um componente base e invariante para a expressão das LS.

As informações de prosódia se referem a componentes variantes que incluem características pessoais e emoções (fatores externos), além de variações nas unidades fonológicas que compõem os sinais (fatores internos). Em LS, ambas as informações podem conter traços manuais e não-manuais.

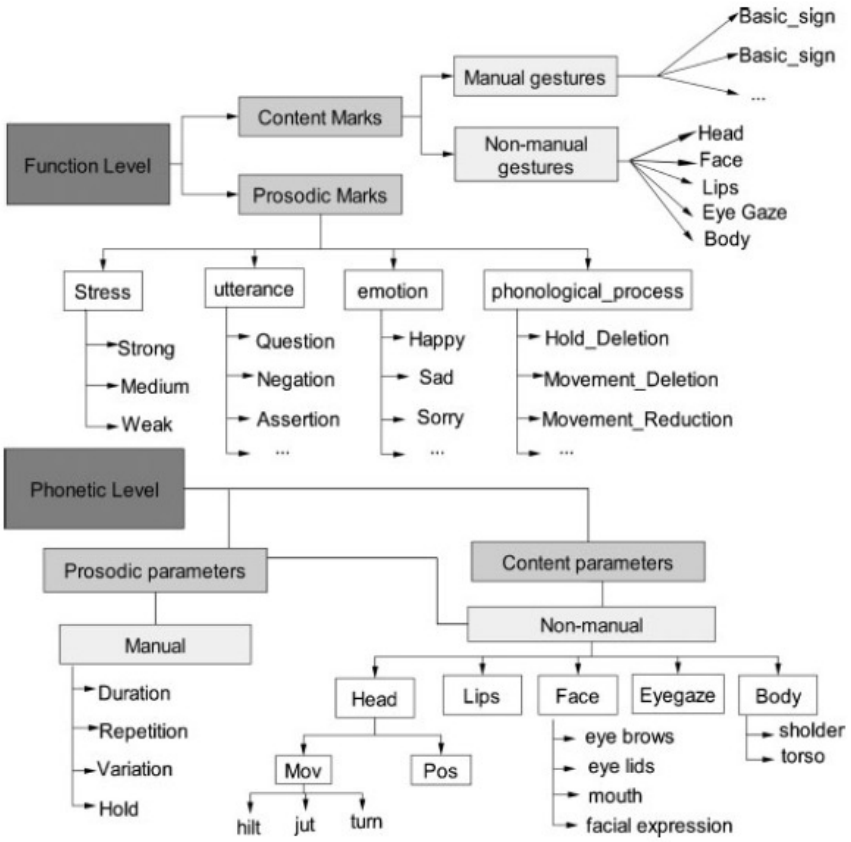


Figura 2.19: Representação do CSLML nos níveis funcional e fonético
Fonte: Ye et al. (2009) [115]

A camada funcional do CSLML, que trata dos aspectos prosódicos (e.g. representação de variações interpessoais), tem o intuito de gerar mais expressividade e naturalidade para a síntese de sinais das LS, agregando traços não apenas importantes linguisticamente, mas, também, como papel fundamental na comunicação [115].

Percebe-se que o CSLML tem uma relação direta com o MP (Brentari, 1998 [13]), por considerar uma camada para os parâmetros e outra para os aspectos de prosódia. Vale destacar que, embora o modelo contenha uma camada para descrever as marcações

prosódicas (e.g. aspectos semânticos, emoções, entre outros), muitos traços gramaticais podem ser representados pela combinação de fonemas das LS (e.g. expressões não-manuais [42]). Por exemplo, um traço fonético de tensão pode indicar estresse no discurso, assim como um parâmetro de frequência de movimento pode indicar intensidade naquele sinal durante o discurso.

A questão principal deste modelo [115] é que sua camada prosódica oferece a possibilidade de descrever esses traços quando houver, não sendo necessária uma análise linguística posterior, assim como há uma clara separação entre o nível fonético e o nível de informações abstratas. Adicionalmente, também é importante ressaltar que essas características são importantes para a análise do discurso, pois para sinais isolados não teriam impacto.

Esta separação do CSLML em camadas fonética e funcional evidencia uma característica importante para o modelo proposto nesta tese em relação à extensibilidade. Neste sentido, foi possível utilizar esta idéia do CSLML e propor um framework para a incorporação de novos níveis gramaticais, tais como morfológico, sintático, semântico, entre outros.

O segundo tipo de modelos computacionais baseados na FLS fazem uma abordagem um pouco mais robusta e com maior nível de detalhes na representação dos sinais. Ou seja, são modelos desenvolvidos com base em um ou mais modelos da fonologia posteriores ao estudo inicial baseado em parâmetros, neste sentido, baseados em modelos que consideram uma melhor classificação de cada sub-unidade (e.g. especificação de toda a CM), os conceitos de sequencialidade de sinais (demonstrada por Liddell & Johnson [1989] [80]), aspectos de ENM e demais recursos necessários computacionalmente.

Amaral (2012) [34] aponta algumas características importantes para um sistema de transcrição computacional, tais como: simultaneidade e sequencialidade dos sinais, detalhamento das ENM, organização estrutural única e não-ambígua, detalhamento do MOV, entre outras. Inicialmente, a pesquisa de [34] faz uma revisão geral em relação aos sistemas desenvolvidos por linguistas para representar a escrita das LS, analisando as notações existentes como SignWriting, HamNoSys, entre outras.

Esta abordagem baseada em sistemas de escrita apresenta limitações quanto ao nível de detalhamento para a representação dos sinais, pois não dispõe de todos os elementos e características fonéticas das LS. Posteriormente, a autora faz uma análise do MMS [80] baseada na pesquisa de Xavier (2006) [114] para a Libras.

Amaral (2012) [34] desenvolve um "sistema de transcrição"para a Libras específico para a aplicação em um sistema de agentes virtuais 3D, que utiliza o XML como linguagem de marcação para a representação dos sinais no sistema proposto. O sistema de transcrição de [34] foi baseado na estrutura do MMS de Liddell & Johnson (1989) [80] dividido em segmentos estáticos e dinâmicos (i.e. suspensões e movimentos).

Na estrutura inicial do modelo de Amaral (2012) [34], um problema é a obrigatoriedade de representar a mão-dominante (tratada como mão direita na representação em UML - *Unified Modeling Language*). Como vimos, existem uma série de sinais que são formados somente de ENM, ou seja, não utilizam as mãos e nem executam movimentos.

No parâmetro de CM, [34] detalha sub-unidades para a descrição de cada dedo no que concerne às juntas (sua rotação e extensão lateral). Entretanto, uma limitação percebida é que não há atributos específicos para descrever o contato que um dedo pode realizar com outro na formação de uma configuração de mão.

A pesquisadora [34] faz uma análise para a representação dos pontos de articulação no espaço de sinalização. O sistema trabalha com coordenadas tridimensionais para os ombros e para o antebraço, uma abordagem interessante, mas não fica claro se estas coordenadas são dadas em graus ou em outro sistema métrico.

Neste sentido, o sistema permite inúmeras possibilidades de valores, o que geram inúmeras possibilidades de descrição de cada sinal. Esta falta de padronização nas relações do espaço de articulação ([13] e [80]) podem trazer dificuldades para as buscas e a indexação dos sinais.

O sistema de [34] apresenta um bom nível de descrição das ENM. Porém, embora as expressões faciais pré-definidas sejam um consenso na literatura [40] é importante descrever a forma como tais expressões são organizadas visando entender e dar mais expressividade à síntese, por exemplo, aplicando traços de variação para dar um caráter mais natural.

Além disso, como apresentado em Felipe (2013) [42], as ENM podem caracterizar diferentes aspectos gramaticais: fonológicos, morfológicos, tipos de frases, entre outros. Assim, tornou-se imprescindível detalhar as sub-unidades que compõem as ENM.

Em relação ao parâmetro de movimento, o sistema de transcrição de [34] carece de um maior nível de detalhe. Por exemplo, faltam características tais como o plano em que o movimento ocorre, tipos e contornos de movimento.

A proposta para movimentos com trajetória incluem a definição de uma lista de locações no espaço (pontos) que indicam o caminho percorrido pela mão entre duas suspensões (locações). Entretanto, falta um melhor detalhamento desses pontos e, principalmente, da maneira como são representados e de quais são os movimentos de trajetória comuns estudados na fonologia das LS.

Uma limitação do sistema de transcrição de [34] é seu caráter específico para avatares 3D, no sentido de buscar resolver um problema muito pontual, mas não indicar ou ampliar o estudo como suporte à solução de outros problemas no âmago do PLN, do reconhecimento, entre outros. Por exemplo, o sistema inclui um elemento denominado *scripts* que tem o intuito de chamar funções de animação externas, ou seja, dando ao modelo um caráter específico de sistema, não proporcionando uma forma genérica de representação.

Sobre as propriedades de usabilidade (facilidade do sistema ser entendido e aplicado posteriormente em outros trabalhos) o sistema de transcrição não apresenta documentação adequada de suas regras. Como o sistema é desenvolvido especificamente para avatares 3D, valores numéricos que representam coordenadas são frequentemente encontrados. Isto gera uma dificuldade, pois é necessário ter uma referência para esses valores para entender o que eles significam. Por exemplo, na Tabela 2.6, vemos a descrição de uma configuração de mão da Libras na qual todos os dedos e as juntas são especificados por valores numéricos.

Tabela 2.6: Exemplo de Descrição no Sistema de Transcrição

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <index proximal="-63.0" medial ="0" distal="-96.8" inclined="-2.5"/>
  <middle proximal="-61.9" medial ="0" distal="-96.6" inclined="2.2"/>
  <ring proximal="-59.8" medial ="0" distal="-94.5" inclined="5.5"/>
  <little proximal="-64.9" medial ="0" distal="-89.7" inclined="14.1"/>
  <thumb proximal="-22.8" distal="-29.3" metacarpal="-7.3" inclined="-7.5"/>
</configuration>
```

Fonte: Amaral (2012) [34]

No trabalho, a autora [34] realiza um teste de inteligibilidade de seu sistema, selecionando alguns sinais da Libras e apresentando-os para surdos fluentes para verificar o grau de entendimento que o sistema proporciona. Neste sentido, uma limitação do teste foi não ter explorado a questão de sinais similares, para verificar a capacidade do sistema de transcrição, bem como do avatar, em distinguir estes sinais.

Adicionalmente, a pesquisa [34] apresenta uma proposta em relação à representação de enunciados em Libras, ou seja, como fazer a ligação entre vários sinais na produção de sentenças e como alterar o sinal para a flexões de gênero e de número.

Como discutido anteriormente, aspectos da fonologia podem ser analisados para extrair informações gramaticais. Por exemplo, o parâmetro de frequência de movimentos podem indicar intensidade, enquanto uma expressão não-manual com sobrancelhas levantadas pode indicar um aspecto exclamativo. Portanto, é necessário um estudo mais amplo quanto à representação das questões gramaticais pelos traços fonético-fonológicos, por exemplo, fazendo um estudo exploratório em modelos como o MP [13].

Antunes (2011) [4] [6] e [5] realiza um estudo exploratório em relação a alguns modelos da fonologia das LS e propõe uma modelagem (baseada em XML) para representação computacional dos sinais das LS.

Neste estudo, o autor [4] explora a organização e a estrutura dos MBP e do MMS, adaptando este conhecimento como a base estrutural da árvore do modelo proposto. Além disso, impulsionado pela pesquisa de Xavier (2006) [114] que faz um estudo da aplicabi-

lidade do MMS na Libras, Antunes (2011) [4] [6] e [5] utiliza o conceito de estados de suspensões e movimentos do MMS, principalmente devido ao seu nível de detalhamento.

Uma característica importante no modelo proposto por [4] [6] e [5] consiste na definição das ENM como um segmento separado, com o intuito de representar essas expressões de modo sequencial ou simultâneo quando utilizadas junto com segmentos de suspensões e de movimentos, ou como único segmento para a descrição de um sinal (sinais formados somente de ENM).

Em relação ao parâmetro de CM, Antunes (2011) [4] [6] e [5] propõe uma estrutura com alto nível de detalhamento, especificando as configurações de cada dedo, bem como a representação visual de como os valores se comportam na formação da CM. Adicionalmente, o autor descreve atributos para o contato que os dedos podem fazer com o polegar e em qual região específica ocorre o contato. Neste sentido, há uma maior flexibilidade na formação de novas CM, por exemplo, para outras LS. Este caso pode ser visto na Figura 2.20, para o contato do polegar pelas pontas, almofadas, almofada na unha e unha na almofada (respectivamente, Figura 2.20.2).

Figura 2.20: 1) Especificação do contato e 2) seus valores
Fonte: O autor (2011) [4]

No estudo realizado por Antunes et al. (2011) com representantes de uma comunidade local de surdos, ficou evidente a importância dos aspectos articulatórios para a distinção entre os sinais, bem como para a sua correta representação.

Na pesquisa, foi solicitado ao grupo que informasse sinais para uma certa CM selecionada. Em diversos momentos, os participantes discutiram sobre qual seria a CM correta utilizada para representar o sinal, além das variações interpessoais. Neste sentido, o estudo mostrou que a investigação sobre a representação isolada dos sinais é um grande desafio. Adicionalmente, o estudo também foi interessante pois os membros da comunidade de surdos levantaram diversos sinais semelhantes, o que testou efetivamente o modelo no quesito precisão (distinção dos sinais, mesmo muito parecidos).

Na questão de LOC, Antunes (2011) [4] descreve uma estrutura para as mãos, tronco e cabeça. Para os pontos de articulação no espaço de sinalização, o autor trabalha com uma abordagem tridimensional baseada no MMS e no MHT, informando em três parâmetros um conjunto de valores específicos para determinar a localização das mãos no espaço:

- para simular o eixo X são utilizadas três posições entre o ombro, o peito e o eixo central do corpo;
- para o eixo Z são utilizadas quatro posições (distâncias) da mão em relação ao corpo (proximal, distal, medial e estendido);
- para o eixo Y são utilizados os pontos de articulação no corpo como referência.

Esta é uma abordagem interessante, pois limita-se o número de pontos no espaço. Entretanto, ainda é necessário um estudo mais específico para verificar como representar possíveis variações interpessoais nas descrições.

O parâmetro de MOV também é detalhado por Antunes (2011) [4], no sentido de descrever aspectos de tipo (forma do movimento), qualidade (velocidade, tempo, tensão etc), direcionalidade, plano e frequência dos movimentos. No estudo exploratório, Antunes (2011) [4] analisou um conjunto de sinais do dicionário de [18] [17] e descreveu no modelo, de maneira a verificar possíveis parâmetros não abordados pelo modelo e analisar os sinais de um ponto de vista computacional.

Por exemplo, ao descrever o sinal da Libras para MENINA ficou evidente que, mesmo representando um movimento do tipo reto com contato e localização na bochecha, era necessário descrever um atributo que indicasse qual parte da mão dominante realizava o contato com a locação (Tabela 2.7). Adicionalmente, este sinal ilustra a importância de um alto nível de detalhamento interno de cada sub-unidade: os sinais MENINA e MULHER diferem somente pelo aspecto de extensão do movimento (qualidade), um com extensão curta e outro com extensão longa.

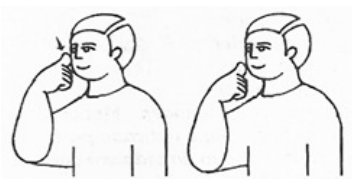
O estudo exploratório realizado por Antunes (2011) [4], bem como o modelo proposto, foram utilizados como a base para esta tese, ou seja, pretendeu-se dar continuidade à construção de conhecimento iniciada nesse trabalho.

Uma das limitações do modelo de [4], decorrente da falta de um estudo mais aprofundado em relação aos modelos linguísticos e computacionais era a impossibilidade de garantir ao modelo a propriedade de completude, para que quaisquer sinais pudessem ser representados na estrutura.

Adicionalmente, visando o desenvolvimento da Arquitetura HCI-SL, bem como a solução de seus sub-problemas computacionais, tornou-se necessário o estudo de propriedades que este modelo deveria ter: uma estrutura formal, com regras de produção bem definidas, que contivesse característica de unicidade, indexabilidade, buscabilidade, usabilidade, robustez e expressividade, dentre outras.

Assim, esta tese buscou realizar um estudo exploratório, com o intuito de verificar quais propriedades formais este modelo computacional deveria incluir e explorar a forma como este modelo poderia dar um suporte efetivo na solução de grande parte dos problemas da HCI-SL.

Tabela 2.7: Sinal MENINA em XML pelo Modelo de Antunes (2011)



Fonte: [17, p. 1485]

```
<sinal nome="menina">
  <suspensao>
    <mao-dominante>
      <configuracao-mao>2</configuracao-mao>
      <locacao ladocorpo="mão dominante">
        <cabeca-loc contato="polegar (ponta)">bochecha</cabeca-loc>
      </locacao>
      <orientacao mao="vertical">
        <palma>para esquerda</palma>
      </orientacao>
    </mao-dominante>
  </suspensao>
  <movimento>
    <mao-dominante>
      <tipo>
        <contorno>reto</contorno>
        <contato-mov>esfregar</contato-mov>
      </tipo>
      <direcionalidade sentido="queixo">
        <unidirecional>para esquerda e para baixo (diagonal)</unidirecional>
      </direcionalidade>
      <qualidade>
        <extensao>curta</extensao>
      </qualidade>
    </mao-dominante>
  </movimento>
</sinal>
```

Fonte: O autor (2011) [4]

2.5 Considerações sobre os Sistemas Computacionais

Uma limitação da maioria dos trabalhos apresentados consiste no nível de especificidade do problema que os modelos resolvem, ou seja, muitos modelos apresentados são construídos para resolver problemas muito específicos (por exemplo, a geração de sinais via avatar 3D). Nesta situação eles não fazem uma indicação ou um estudo mais aprofundado em relação a como o modelo pode contribuir para a solução de outros problemas relacionados ao tratamento computacional das LS (problema global), por exemplo, o reconhecimento automático de sinais (*input*).

Neste contexto, embora os trabalhos apresentados possam auxiliar na solução questões específicas, a solução do problema global, o tratamento computacional de uma LS, ainda carece de soluções na literatura.

Como discutido neste capítulo, existe uma série de sinais nas LS que são muito semelhantes entre si, além dos sinais denominados "pares mínimos"(sinais que se diferenciam

entre si apenas por um parâmetro principal do MBP). Assim, se considerarmos no contexto da Arquitetura HCI-SL um sistema de Reconhecimento Automático de Sinais (RAS), é evidente que este recurso apresentará ruídos na entrada, bem como irá capturar variações intrínsecas aos usuários ao invés de distinguir os sinais similares.

Neste sentido, uma função para a recuperação de sinais (necessária em diversos cenários da arquitetura) deveria trabalhar com um conceito de busca por similaridade, tenso isto exigido um estudo e a indicação de uma métrica para o cálculo da distância entre um sinal de entrada e os sinais armazenados na base de descrições no modelo computacional, bem como uma análise das formas de indexação para permitir um melhor desempenho da função de busca. Esta característica não foi abordada em nenhum dos modelos computacionais revisados. Uma contribuição desta tese foi realizar este estudo para que as representações de sinais a partir do modelo pudessem ser armazenadas, indexadas e recuperadas de forma eficiente.

Os modelos apresentados também não fazem um estudo detalhado sobre uma forma canônica para a representação dos sinais. Esta propriedade de unicidade é fundamental para questões de busca e para a aplicação do modelo computacional em processos de PLN e sistemas de RAS. Portanto, um ponto também abordado por esta tese foi a formalização do modelo.

Os modelos baseados em SELS, mesmo representando uma notação da forma escrita das LS, apresentam algumas limitações pela falta de aspectos articulatórios importantes na constituição dos sinais (e.g. aspectos de frequência, intensidade, classificadores, sequencialidade, entre outros) além de exigirem que o usuário conheça todos os símbolos gráficos do sistema e traduza esses símbolos em códigos textuais / numéricos.

Como vimos no exemplo da Tabela 2.5, para representar os símbolos gráficos do SignWriting no modelo em XML são utilizados números. Assim, conclui-se que este tipo de representação computacional não cobre todas as necessidades dentro da Arquitetura HCI-SL, como as questões de precisão necessárias em um sistema de reconhecimento, e um maior nível de detalhes que podem ser fundamentais para a síntese automática de sinais mais natural e próxima da realidade, e com subsídios para a construção dos processos de PLN necessários.

Em relação aos modelos de marcação de gestos como o VHML e o MCML, mesmo que eles não tenham sido propostos para um contexto de processamento computacional das LS, foi importante analisar suas estruturas comparativamente aos modelos da fonologia das LS, pois isto permitiu levantar possíveis características ou propriedades importantes para o desenvolvimento do modelo computacional proposto nesta tese.

Essa análise também teve como justificativa a questão da completude de representação, no sentido de poder representar quaisquer sinais de forma correta, dentro das possibilidades da fonologia, mas permitindo ao modelo dispor de uma capacidade para que novos parâmetros ou valores possam ser inseridos caso haja a necessidade.

Dentre outras limitações, podemos citar, de maneira geral, que os trabalhos não apresentam uma boa usabilidade tanto para o usuário final quanto para a comunidade científica em relação à aplicação do modelo. Como apresentado, as LS são de modalidade gestual-visual e, portanto, ao se especificar um modelo computacional que represente os seus sinais, espera-se que o modelo possa ser aplicado e utilizado de maneira fácil em novas pesquisas.

Assim, se o modelo descreve apenas de forma textual quais são os parâmetros e o que eles representam, fica evidente a falta de uma documentação ou referência visual/gráfica que documente em detalhes o que cada parâmetro e cada valor significam em relação aos gestos e expressões produzidas pelo interlocutor.

Por exemplo, quando um modelo considera um movimento do tipo retilíneo no espaço neutro, podem ser determinadas algumas questões: qual o ponto exato no espaço de sinalização? Qual o início e o final deste movimento no espaço? Onde especificar possíveis variações do sinalizador? Essas questões, dentre outras, ilustram a necessidade de promover maior usabilidade ao modelo computacional, proporcionando um claro entendimento do que ele representa visando um tratamento computacional dos sinais mais eficiente e preciso.

Como mostrado no contexto desta tese, a interação em qualquer sistema para a comunidade de surdos deve ser mediada pelas LS. Como contra exemplo, nos testes realizados no FleXLIBRAS os surdos que testaram o ambiente proposto encontraram dificuldades pelo uso exacerbado de descrições textuais na interface, sugerindo aos pesquisadores mais recursos gráficos e vídeos explicativos. Uma alternativa, neste caso, seria a capacidade de o usuário final poder manipular diretamente o avatar de modo simples para a descrição dos sinais.

Muitos dos modelos apresentados impõem, além das restrições linguísticas, restrições computacionais no sentido de limitar a representação dos sinais em estruturas baseadas na linguagem XML e, muitas vezes, não definindo regras claras para a representação dos sinais.

Assim, o processo de pesquisa registrado nesta tese pretendeu desenvolver um modelo formal que permitisse a geração de cada representação de sinal em diversos formatos de saída: seja XML, texto ou outro formato necessário aos cenários de uso da Arquitetura HCI-SL.

Nos modelos computacionais baseados nas LS, percebeu-se como uma restrição o fato de eles serem baseados apenas nos MBP (Stokoe, 1960) ou no MMS (Liddell & Johnson), ou nos sistemas de escrita das LS como SignWriting ou HamNoSys. Ou seja, estes trabalhos não realizaram um estudo mais aprofundado a fim de verificar quais modelos linguísticos poderiam trazer mais benefícios computacionalmente, bem como de analisar a forma em que cada modelo descreve seus traços articulatórios.

Por exemplo, vimos que no MMS o movimento possui um alto nível de detalhamento,

enquanto o MP apresenta características complementares que poderiam auxiliar posteriormente os processos de PLN na identificação de aspectos gramaticais.

Para finalizar, percebeu-se que os trabalhos correlatos atendem a problemas muito pontuais, desconsiderando um caráter mais amplo para apoiar a solução do problema global (tratamento computacional dos sinais), assim como apresentam limitações em relação à formalização e à definição de regras de produção, à completude quanto aos aspectos articulatórios da fonologia, à usabilidade dos modelos para facilitar sua aplicação, à capacidade de extensão para explorar a solução de novos problemas, às restrições tecnológicas, dentre outras.

Adicionalmente, devido ao fato de os modelos computacionais existentes não abordarem questões mais amplas com o intuito de dar suporte à solução do problema global, esta tese mostrou-se relevante por fazer este estudo exploratório, assim como por levantar um conjunto de propriedades intrínsecas a um modelo computacional capaz de auxiliar o desenvolvimento da Arquitetura HCI-SL.

Portanto, buscou-se desenvolver e apresentar o CORE-SL, um modelo computacional que tem o objetivo de resolver as limitações apresentadas em outros modelos da literatura, sendo um candidato para fornecer um suporte na solução dos problemas em relação ao processamento computacional das LS. A justificativa deste desenvolvimento reside na necessidade de um estudo relacionado ao conjunto de propriedades formais necessário para possibilitar ao modelo auxiliar em diversos contextos da Arquitetura HCI-SL com a proposição de hipóteses de pesquisa, de *frameworks* e de metodologias capazes de dar suporte à área de Ciência da Computação na construção desta interação baseada em LS.

2.6 Gramáticas e Linguagens Formais

Como discutido nas seções anteriores, um problema comum dos modelos correlatos é a falta de especificação de regras formais para a representação canônica dos sinais. Esses modelos consistem de estruturas em formato de árvore que representam os parâmetros fonéticos organizados hierarquicamente de acordo com as classes utilizadas (MBP, MMS, MHT, etc).

Entretanto, esses modelos não especificam as regras ou os métodos de como estas árvores devem ser criadas, possibilitando a descrição de árvores ambíguas, representações incompletas pela falta de algum parâmetro que deveria ser obrigatório, ordenação incorreta das sub-unidades (podendo perder o significado de sinais que envolvem simultaneidade e sequencialidade), entre outros.

Para resolver este problema, um modelo computacional para a representação de sinais deveria ser especificado em uma linguagem ou modelo formal (uma estrutura ou uma metalinguagem desenvolvida pela Teoria da Computação para a representação de conceitos formais e de linguagens de programação).

Estes formalismos consistem em meta-linguagens ou modelos que possibilitam a especificação formal da gramática de uma linguagem, sendo capaz de descrever por meio de regras seus elementos intrínsecos, sua estrutura organizacional, suas propriedades, os relacionamentos entre seus elementos, entre outros. A partir deste conjunto de regras, o conteúdo produzido por meio desta gramática pode ser validado, assim como podem ser produzidos conteúdos por meio desta gramática.

Assim, uma meta-linguagem sintática consiste de uma notação para definir a "sintaxe"de uma linguagem por meio de regras formais. Cada regra (símbolo não-terminal - *non-terminal*) descreve uma característica desta linguagem, que pode ser formada por uma sequência de símbolos. Os valores que estas regras instanciam são chamados de símbolos terminais (*terminal symbols*) [1].

Destaca-se, para esta seção, que o termo "gramática"refere-se ao conjunto de regras formais computacionais para a representação sintática de uma linguagem e não à gramática das LS ou das línguas naturais em geral.

Chomsky (1956) [103] classifica as gramáticas em quatro tipos: **tipo-3** (gramáticas regulares), **tipo-2** (gramáticas livres de contexto), **tipo-1** (gramática sensível ao contexto e **tipo-0** (gramáticas irrestritas). Esta classificação relaciona restrições quanto à formalização das regras de produção da gramática [87].

Por exemplo, uma gramática é classificada como tipo-0 se não há restrições na forma das produções, ou seja, *strings* arbitrárias são permitidas tanto no lado esquerdo, quanto no lado direito das regras.

Nesta tese foi adotada uma gramática do tipo-2 (gramática livre de contexto) para a especificação formal do modelo proposto, por ser mais utilizada para a formalização de linguagens de programação e de modelos computacionais, além de ser bem utilizada em sistemas de PLN para a descrição, para a validação de sintaxe (*parsings*) e para a síntese (geração) de uma língua [103] [87].

A Gramática Livre de Contexto (GLC) é formada por um símbolo inicial, um conjunto de variáveis (símbolos não-terminais), de palavras (símbolos terminais) e de regras de produção que possuem a forma $\beta \rightarrow \alpha$, onde:

- α consiste de uma sequência arbitrária de símbolos terminais ou não-terminais;
- β consiste de um símbolo não-terminal singular;
- dada qualquer ocorrência de β durante a fase de *parsing*, este símbolo não-terminal pode ser substituído por um símbolo α independente do contexto.

O *parsing* (análise sintática) de uma GLC consiste em processar iterativamente uma *string* de entrada, agrupando palavras (*tokens*) em uma árvore sintática de acordo com as regras de produção da gramática [87].

Assim, um *parser* (analisador sintático) tem o papel de identificar as entradas válidas e as inválidas, podendo apontar os *tokens* incorretos nesta entrada. Esta característica é comumente percebida em corretores ortográficos e em compiladores de linguagens de programação [103] e [87].

O *parser* tem o papel de gerar uma árvore de derivação (*parsing tree*), que consiste da correspondência de uma sequência de *tokens*, extraídos da sentença de entrada, com os elementos da gramática a partir da raiz (símbolo inicial).

Assim, uma árvore de derivação é formada por: a) **raiz** que é o símbolo inicial da gramática, b) **vértices interiores** que consistem das variáveis (não terminais), c) **folhas** que correspondem aos símbolos terminais.

É importante destacar que se um vértice interior consiste do símbolo não-terminal X, então seus filhos são definidos como $X_1, X_2, ...X_n$, logo, $X \rightarrow X_1X_2...X_n$ é uma produção.

A partir de uma GLC uma árvore de derivação pode ser construída de diversas maneiras. No entanto, é comum utilizar alguns padrões para a derivação das sentenças visando operações mais sistemáticas no *parser*.

Neste sentido, uma *derivação mais à esquerda* consiste em uma sequência de aplicações de derivação direta realizando a substituição do símbolo não-terminal sempre mais à esquerda na regra de produção. De maneira análoga, uma *derivação mais à direita* consiste na substituição do símbolo não-terminal (variável) sempre mais à direita.

Uma sentença produzida por uma GLC só pode ter uma árvore de derivação (sintática). Neste caso, uma gramática é ambígua se existe pelo menos uma sentença, produzida por essa gramática, que possui mais de uma árvore de derivação (mais à esquerda ou mais à direita) [103].

Destaca-se que não existe uma solução algorítmica para detectar a ambiguidade em uma gramática, sendo este problema considerado indecidível computacionalmente [103] e [87]. Da mesma forma, não existe um algoritmo para remover a ambiguidade de uma GLC. Para isto, deve-se identificar a fonte da ambiguidade e reescrever a regra.

Para minimizar a ambiguidade podem-se valer de algumas técnicas ou heurísticas, tais como: verificar se alguma regra de produção mistura recursão à direita e à esquerda, transformar a gramática em uma forma normal (e.g. Forma Normal de Chomsky [103]) e realizar a simplificação das regras (e.g. eliminar símbolos terminais nulos).

No contexto de Programação de Computadores tornou-se padrão definir a gramática formal de uma linguagem a partir de 1960, com a formalização da linguagem Algol por meio da notação BNF (*Backus-Naur Form*) [103]:

- ::= representa a relação “definido como”;
- | representa uma escolha (alternativa);
- $< \beta >$ representa uma regra e o símbolo não-terminal β;

- símbolos que não estão dentro dos sinais <> são considerados terminais.

Por exemplo, a Tabela 2.8 apresenta um exemplo simples de representação de um número do tipo inteiro por meio da notação BNF.

Tabela 2.8: Exemplo de regras em BNF para números do tipo inteiro

```
<numero> ::= <digito> | <numero> <digito>
<digito>  ::= [0..9]
```

Fonte: O autor (2014)

Esta notação tem sido muito utilizada para a definição formal de linguagens e de sintaxe para uma GLC. Alguns problemas ou limitações consistem no conflito causado quando a gramática utiliza meta-símbolos para a descrição (< > | e ::=) e no grande conjunto de regras gerado para a representação de repetição [1].

Todavia, diversas notações foram criadas com base na BNF, incorporando tanto vantagens (e.g. códigos para a redução do número de regras) quanto desvantagens (e.g. notações confusas e a inserção de ambiguidade na representação).

A BNF Estendida (*Extended BNF* - EBNF), Wirth (1977) [113], possibilita a definição formal de uma sintaxe para diversos tipos de especificações, não somente linguagens de programação [1]. Como apresentado em [1], uma meta-linguagem de formalização padrão, tal como a EBNF, deve ser:

1. **concisa**: deve possibilitar que uma linguagem seja definida rapidamente e facilmente compreendida;
2. **precisa**: deve descrever regras não-ambíguas;
3. **formal**: as regras devem poder ser analisadas ou processadas computacionalmente;
4. **natural**: a notação e o formato devem ser relativamente simples de aprender e compreender, mesmo para não especialistas em linguagem;
5. **genérica**: a notação deve ser adequada para muitos fins, incluindo a descrição de muitas línguas diferentes;
6. **simples**: a notação deve evitar usar caracteres especiais não disponíveis no teclado;
7. **auto-descrição**: deve ser possível descrever a própria linguagem com ela mesma;
8. **linear**: deve ser expressa com uma única sequência de caracteres.

A EBNF (ISO/IEC 14977) [1] define um padrão de meta-linguagem sintática baseado no BNF. O formalismo também inclui as extensões mais utilizadas e os recursos adicionais que muitas vezes são necessários em uma definição formal.

Uma das principais vantagens da EBNF consiste da possibilidade de representar repetições de regras (símbolos não-terminais) sem a necessidade de definir regras recursivas (e.g. onde os símbolos não-terminais podem ser consecutivamente substituídos por eles mesmos ou símbolos terminais). Isso proporciona uma facilidade de compreensão da gramática, bem como sua implementação em *parsers*.

O EBNF define: que **símbolos terminais** são representados entre aspas duplas ou simples (e.g. "x"ou 'x'), a **definição explícita de um número de itens** (e.g. letters ::= 5 * caractere), os **casos excepcionais**, comentários, que os elementos **não-terminais** são representados por uma palavra simples ou entre parêntesis e a **extensibilidade** na qual o usuário pode estender a meta-linguagem. A EBNF é resumida na Tabela 2.9.

Adicionalmente, algumas extensões são utilizadas como convenção na EBNF para facilitar as formalizações. Esses padrões foram inspirados na sintaxe das expressões regulares:

- ***** (*Kleene Star*): representa 0 ou N repetições;
- **+** (*Kleene Cross*): representa 1 ou N repetições;
- **?** (*Optional*): significa a ocorrência de 0 ou 1 (opcional);
- **(...)**: uso de parêntesis para agrupamento;

Tabela 2.9: Síntese dos operadores do EBNF

EBNF	**Operador**	**Definição**
Palavra sem Aspas		Símbolo Não-Terminal
"..."		Símbolo Terminal
'...'		Símbolo Terminal
(...)		Parêntesis
[...]		Símbolos Opcionais
{...}		Símbolos Repetidos 0 ou N vezes
{...}-		Símbolos Repetidos 1 ou N vezes
=	infixo	Símbolo de Definição
;	sufixo	Símbolo de Terminação de Regra
|	infixo	Símbolo de Escolha
,	infixo	Símbolo de Concatenação
−	infixo	Símbolo de Exceção
(*...*)		Símbolo para Comentários

Fonte: Modificado pelo autor (2014) [1]

Como a EBNF é um formalismo bem utilizado na computação e apresenta regras simples, esta tese adotou este padrão para a formalização do modelo computacional.

Além disso, para uma visualização mais simples da formalização foi gerado para cada regra um diagrama de sintaxe (*Railroad Diagram*). Basicamente o diagrama é construído após um *parsing* na gramática EBNF. Um exemplo é apresentado na Figura 2.21.

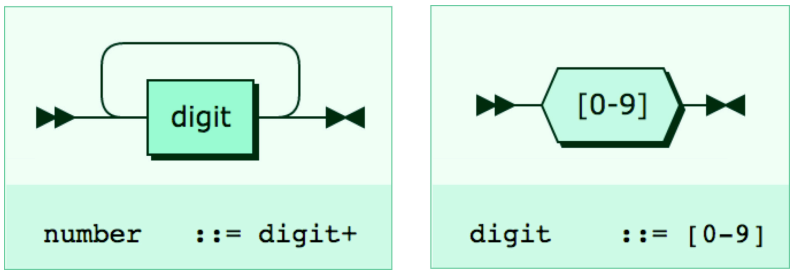


Figura 2.21: Diagrama de Sintaxe para as Regras do Número Inteiro
Fonte: O autor (2014)

2.7 Considerações

A partir da revisão dos sistemas fonético-fonológicos das LS, foi possível a construção de um mapa conceitual (Apêndice A) com as principais características e com as regras de cada modelo visando um maior nível de detalhamento para o processamento computacional. Assim, buscou-se sempre escolher os conceitos que descrevessem com maior nível de granularidade cada parâmetro fonético.

Por exemplo, foi selecionado a estrutura de configuração de mão do MMS pelo fato deste modelo detalhar cada dedo, a disposição de cada junta e possíveis contatos entre os dedos. Assim, foi possível construir um modelo computacional que possibilitou a inclusão de novas configurações de mão, por meio da descrição de cada um dos dedos.

Para a resolução de possíveis conflitos e das redundâncias utilizou-se a seguinte estratégia: primeiro, relacionou-se no mapa conceitual (Apêndice A) os conceitos correlatos para auxiliar na etapa de formalização; e segundo, na hipótese de haver conflito de representação de um conceito em um nó filho, o parâmetro escolhido foi a mesma estrutura do nó pai.

Adicionalmente, foram especificadas as regras formais para a produção dos sinais de maneira que cada conceito fonético pudesse ser representado somente de uma maneira com o modelo. Caso haja a necessidade, posteriormente, de representar um conceito de outra maneira, a regra deverá ser alterada e não duplicada.

Por meio da revisão das modelagens computacionais correlatas foi possível utilizar as limitações de cada sistema na forma de requisitos para o desenvolvimento do modelo proposto nesta tese, bem como o uso das vantagens de cada modelo, quando necessário, para a complementação do CORE-SL.

Para finalizar, realizou-se um estudo em relação às meta-linguagens utilizadas para

a representação de linguagens de programação e de modelos formais. A seção analisou algumas meta-linguagens tais como a BNF, a EBNF e os Diagramas de Sintaxe.

A EBNF e os Diagramas de Sintaxe apresentam uma facilidade de uso e de entendimento, mostrando-se mais indicados para a utilização na formalização do CORE-SL. Além disso, a EBNF permite diversas simplificações que reduzem o número de regras e de redundância para representar certas definições, contribuindo para o objetivo do CORE-SL quanto à capacidade de aplicação nos diferentes contextos da Arquitetura HCI-SL.

Os próximos capítulos apresentam o desenvolvimento do CORE-SL com base na fundamentação teórica discutida aqui, apresentando os processos utilizados e as propriedades formais relacionadas ao uso e à qualidade do modelo proposto. Um estudo sobre o problema de busca não-exata é apresentado, assim como alguns cenários de aplicação do CORE-SL na Arquitetura HCI-SL como abordagem de desenvolvimento.

CAPÍTULO 3

CORE-SL: MODELO COMPUTACIONAL PROPOSTO

Este capítulo tem como objetivo apresentar o *framework* proposto e utilizado para a construção do CORE-SL (***CO****mputational Model for* ***RE****presentation of* ***S****ign* ***L****anguage*) - Modelo Computacional para a Representação de Sinais de Línguas de Sinais.

Este *framework* descreve as metodologias e os processos utilizados para a construção do CORE-SL, bem como o contexto, a relação dos conceitos estudados na fundamentação teórica, a arquitetura proposta para o modelo, os níveis estruturais e seus componentes, e as propriedades formais centradas na aplicação do modelo na arquitetura HCI-SL.

3.1 Posicionamento Conceitual

Para o claro entendimento do escopo e do nível de abrangência do CORE-SL em relação à Computação, torna-se necessária a definição de alguns conceitos sobre os tipos de sistemas de notação relacionados às LS.

Sistema de Escrita de Sinais

Consiste em um sistema alfabético ou pictográfico utilizado por surdos para registrar na forma escrita um evento linguístico, com o intuito de comunicar ou de recuperar essa informação escrita posteriormente [67].

Um exemplo de sistema de escrita para as LS é o SignWriting [105], um sistema icônico que traduz em símbolos gráficos as sub-unidades fonéticas utilizadas na composição e na articulação dos sinais das LS.

Este tipo de sistema não é adequado para a representação computacional dos sinais, pois não agrega todos os elementos articulatórios e as sub-unidades fonéticas necessárias computacionalmente para um alto nível de detalhes.

Por exemplo, o SignWriting não possui símbolos específicos para representar os pontos de articulação, alguns aspectos tais como a velocidade e a frequência do movimento, a sequencialidade, dentre outros.

Sistema de Transcrição

É utilizado para anotar de forma rápida e precisa amostras da LS em diversos formatos com o intuito de facilitar os estudos linguísticos ou para documentar exemplos [67].

Este tipo de sistema utiliza *tokens* (palavras-chave) ou símbolos gráficos para descrever fragmentos da LS (e.g. anotar em um vídeo as CM utilizadas). Como exemplos tem-se o HamNoSys (gráfico) [64] e o Sistema de Notação em Glosa (textual) [43] [44].

Esta classe de sistemas de notação também não é adequada para a representação computacional de sinais, pois muitos sistemas não incluem um alto nível de detalhes na descrição (e.g. não são detalhados os dedos das CM), não possuem uma estrutura organizacional e de regras de descrição bem definidas.

Sistema de Codificação

Consiste de uma modelagem que tem o propósito de registrar e de anotar amostras da LS para serem processadas e analisadas computacionalmente, permitindo que esses dados possam ser classificados, contabilizados e recuperados.

Em geral, este tipo de sistema utiliza palavras-chave que representam sub-unidades gramaticais para registrar os sinais em tabelas ou em banco de dados, utilizando estas estruturas para os estudos linguísticos.

Um sistema de codificação é uma ferramenta que permite aos linguistas investigar sobre vários aspectos da linguagem. Este sistema deve ser simples de utilizar por seus usuários e deve ser simples de armazenar e de recuperar os dados [67].

Ou seja, um sistema de transcrição enfatiza a facilidade e a velocidade de uso na perspectiva do escritor, enquanto o sistema de codificação enfatiza a facilidade e a velocidade de uso na perspectiva do leitor [67].

Um exemplo de sistema de codificação é o SignTyp [67], uma evolução do SignPhon [77]. O SignTyp consiste de um banco de 12 mil sinais estruturados em um sistema de codificação que possui a mesma estrutura de árvores fonológicas, com elementos representados de maneira tabular [67].

Embora este tipo de sistema seja de uso computacional (como modelagem para o registro em banco de dados), ele não contempla uma série de requisitos necessários para o uso na arquitetura HCI-SL.

Esta classe de sistema não formaliza o modelo em uma meta-linguagem computacional (e.g. necessária para implementação computacional de *parsings*), não define um conjunto de regras de representação, não apresenta estratégias de busca de sinais não-exata e não disponibiliza *frameworks* e metodologias específicas para apoiar a construção de artefatos computacionais.

Modelo Computacional para a Representação de Sinais

Consiste de um modelo capaz de representar computacionalmente quaisquer sinais das LS por meio de um formalismo: uma meta-linguagem que define regras formais.

Este tipo de modelo agrega uma abrangência de uso computacional maior que os sistemas de escrita, de transcrição ou de codificação, pois trata de questões fundamentais tais como: a corretude na representação (por meio de regras), a eficiência de armazenamento e de recuperação, a busca por similaridade (como encontrar sinais similares para uma entrada não-exata informada pelo usuário), o desenvolvimento de *frameworks* que tratam de questões específicas para a construção aplicações em LS), dentre outras.

Portanto, no contexto desta tese, os modelos computacionais são sistemas formais para a representação de sinais para o uso computacional. As aplicações desenvolvidas com base neste tipo de modelo podem possibilitar a inclusão dos demais tipos de sistemas de notação das LS.

O CORE-SL é inserido neste contexto: um modelo computacional de representação. A Figura 3.1 apresenta um diagrama que relaciona o nível de abrangência dos tipos de modelos definidos em relação às questões computacionais da arquitetura HCI-SL.

Além disso, a Figura 3.1 também representa a relação de inclusão destas notações (e.g. a partir do CORE-SL deve ser possível gerar como saída um sistema de codificação, ou de transcrição ou de escrita).

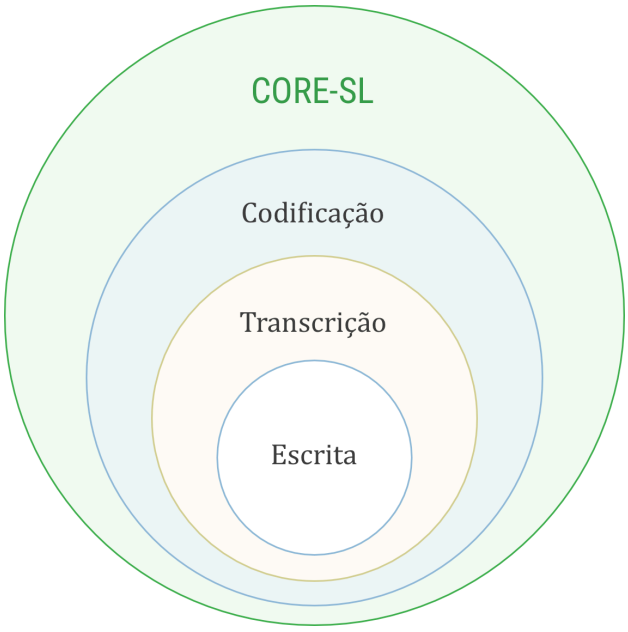


Figura 3.1: Grau de abrangência das notações em relação às questões computacionais da Arquitetura HCI-SL. Também é ilustrada a relação entre as notações
Fonte: O autor (2014)

3.2 *Framework* Proposto

O *framework* proposto e utilizado para a construção do CORE-SL é composto por seis componentes: contextual, conceitual, formal, físico, externo e de aplicação (Figura 3.2).

Esses seis componentes foram incluídos com base na IHC (contexto e uso) [98] [93] [33], na Teoria da Computação [103] [87] [52] [117] [19] e na Modelagem de Dados (conceitos, formalismos e nível físico) [107] [59] [81] [72] [1] e na Integração e Interoperabilidade de Serviços e Sistemas (nível externo e de aplicação) [56].

Cada componente consiste de um nível de abstração responsável por descrever os conceitos, os métodos, as propriedades formais e os recursos necessários a um modelo computacional para a representação de sinais que considere os requisitos funcionais e de qualidade necessários para a solução do problema.

A seguir a definição de cada componente:

1. **Contextual**: consiste da descrição do contexto de aplicação e de uso no qual o modelo gerado pelo *framework* pode ser utilizado. Para este entendimento é necessário conhecer os usuários, suas necessidades, suas principais atividades e o conhecimento sobre a tecnologia considerando um ambiente de uso real. Quando considerado este contexto de uso é possível melhorar a interação humano-computador e desenvolver aplicações mais úteis [33].

2. **Conceitual**: descreve toda a base conceitual do modelo, que inclui a estrutura organizacional e a metodologia utilizada para o desenvolvimento do modelo. Este nível é responsável por definir, em uma árvore conceitual, a relação dos elementos e seus valores capazes de representar computacionalmente um sinal.

3. **Formal**: com base no nível conceitual, concerne a este nível descrever o processo para a criação das regras formais em uma meta-linguagem, bem como analisar e apresentar como essas regras devem ser construídas. A combinação dos níveis contextuais, conceitual e formal define a forma que o CORE-SL pode ser estendido para outros níveis gramaticais das LS.

4. **Físico**: consiste na discussão e na descrição dos aspectos internos ao modelo, tais como o armazenamento, a indexação de sinais e o desempenho (*performance*). Este nível deve apresentar as alternativas para o armazenamento de sinais visando apoiar a aplicação de estratégias de indexação e de busca eficientes para os níveis externo e de aplicação.

5. **Externo**: discute sobre as propriedades externas que o CORE-SL deve incluir para permitir sua aplicação em diversos contextos visando auxiliar na resolução de problemas relacionados. Este nível também é responsável por apresentar um padrão para a documentação do modelo, com o intuito de facilitar o seu uso e seu aprendizado em qualquer contexto de uso na Arquitetura HCI-SL.

6. **Aplicação**: apresenta um conjunto de *frameworks* e de métodos que utiliza o CORE-SL como abordagem para auxiliar a resolução de problemas específicos na

arquitetura HCI-SL. Este nível inclui um estudo mais amplo e específico, que é apresentado no Capítulo 6.

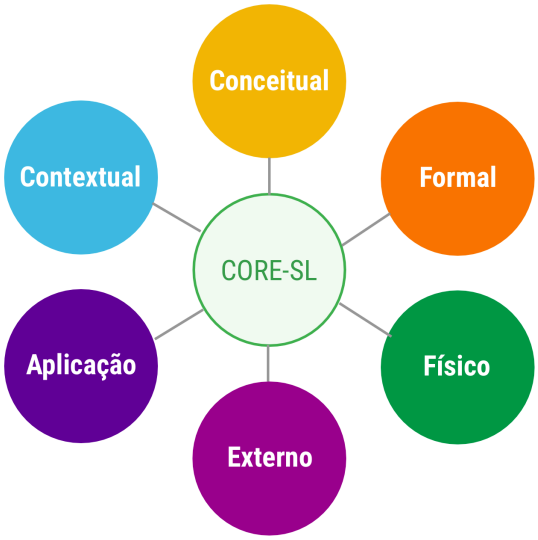


Figura 3.2: Componentes do *Framework* para o CORE-SL
Fonte: O autor (2015)

A seguir apresenta-se o desenvolvimento do CORE-SL seguindo o *framework* proposto. Destaca-se que cada nível apresenta e discute as propriedades formais internas ou externas ao CORE-SL, dentre as quais, algumas têm relação com mais de um nível do *framework*.

Por exemplo, a propriedade de **consistência** é capaz de agregar características dos níveis físico (dados consistentes), formal (coerência das regras seguindo corretamente a meta-linguagem e o nível conceitual) e externo (apresentação consistente para o uso).

3.3 Nível Contextual

O CORE-SL está inserido no contexto da Arquitetura HCI-SL e tem a finalidade de apoiar a resolução do problema de representação computacional formal e eficiente dos sinais das LS para possibilitar o desenvolvimento de aplicações e de serviços mais adequados às necessidades do usuário final.

3.3.1 Escopo

Portanto, o CORE-SL faz um entendimento do **perfil do usuário**: comunidades surdas que utilizam as LS para a interação, a comunicação e a produção de conhecimento. O *design* de uma interação através das LS é uma das necessidades deste perfil de usuário [5] [56] [61] [4].

A Arquitetura HCI-SL descreve um conjunto das principais **atividades** que os surdos necessitam baseado em pesquisas empíricas e etnográficas [56]. Essas atividades são apresentadas e modeladas na forma de aplicações e de serviços [56]. Por exemplo, a utilização de um Ambiente Virtual de Aprendizagem (AVA) pelos surdos é necessária quando considerado a dispersão geográfica das comunidades de surdos e uma necessidade de educação à distância [106] [51].

O desenvolvimento da maioria destas aplicações para o usuário final depende de ferramentas adicionais da arquitetura, apresentadas nas camadas interna e de serviços. Estas camadas descrevem as ferramentas computacionais (e.g. avatares, banco de dados, visão computacional, etc) e os serviços (e.g. dicionários, tesauros, ambientes de comunicação, etc) que podem ser combinados para a construção de soluções ao usuário final.

Como exemplificação, podemos considerar que para a construção de um AVA para os surdos são necessárias algumas ferramentas como um dicionário, um avatar 3D, uma ferramenta de comunicação em vídeo, entre outras.

O CORE-SL tem o papel de funcionar como um núcleo computacional na arquitetura atuando como um protocolo comum entre os serviços para as questões de representação, de armazenamento, de indexação e de busca. Este é um requisito necessário aos serviços da arquitetura HCI-SL e consistiu do problema computacional estudado nesta tese.

A Figura 3.3 apresenta uma abstração da inclusão do CORE-SL na Arquitetura HCI-SL. O CORE-SL atua diretamente como base computacional para as operações relacionadas à representação de sinais (e.g. inserção, leitura, alteração, etc) nas camadas interna e de serviços, além de atuar como abordagem conceitual dentro destas camadas[1].

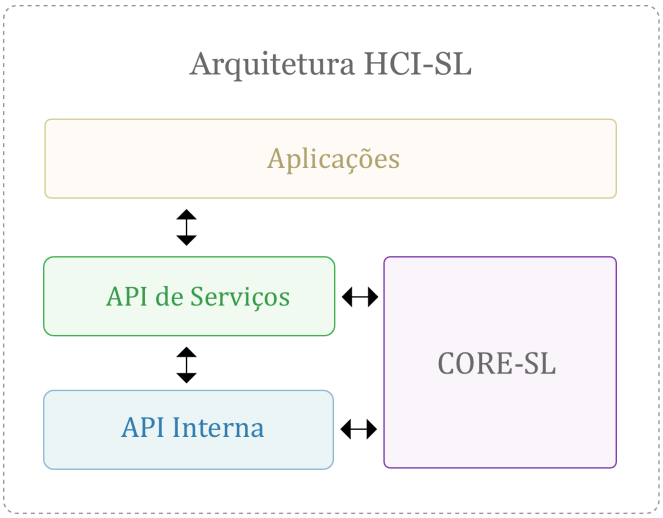


Figura 3.3: Inclusão do CORE-SL na Arquitetura HCI-SL
Fonte: O autor (2015)

[1]Um detalhamento destas abordagens conceituais é apresentado no Capítulo 6.

3.3.2 Arquitetura do CORE-SL

A arquitetura do CORE-SL é composta por quatro camadas principais: interna, conceitual, externa e de uso (Figura 3.4). Esta organização arquitetural é inspirada em um esquema de três camadas (*three-schema architecture*) proposto para sistemas de banco de dados pela ANSI/SPARC [107].

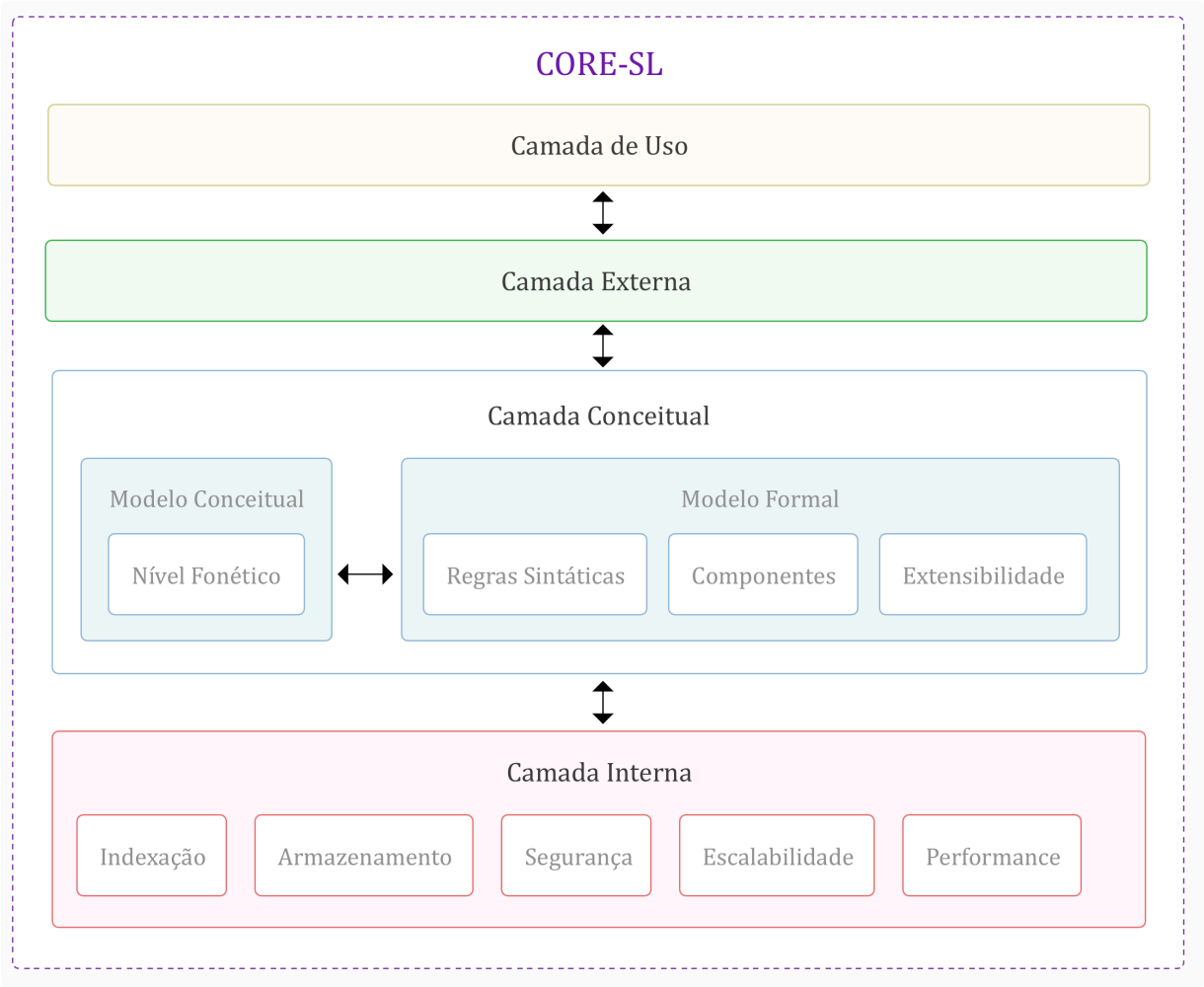


Figura 3.4: Camadas da Arquitetura do CORE-SL
Fonte: O autor (2015)

A camada interna deve tratar das questões relacionadas ao armazenamento e ao acesso dos sinais e seus dados correlatos. Nesta camada da arquitetura devem ser discutidas algumas questões, tais como: a indexação (como agrupar os dados para permitir uma busca eficiente), a escalabilidade (como escalar esta estrutura de armazenamento em relação ao *hardware* disponível e como manter o mesmo desempenho), segurança (como manter o histórico de cada sinal), *performance* (discutir estratégias eficientes para as operações computacionais básicas), dentre outras.

A camada conceitual tem o papel de relacionar os modelos conceitual e formal para descrever a estrutura e as regras de representação dos sinais na camada interna. Esta

camada fornece uma definição clara dos conceitos e a relação entre eles, permitindo várias formas de implementação no nível interno.

A camada externa consiste da apresentação dos dados (sinais) na forma de uma *API* para uso nas aplicações da arquitetura HCI-SL. Eventualmente, esta camada deve descrever as operações básicas da *API* (e.g. inserção), bem como os formatos de saída (*output*) dos dados (e.g. XML, JSON e texto).

Concerne à camada de uso uma aplicação específica do CORE-SL para a arquitetura HCI-SL: um conjunto de *frameworks* que utilizam o modelo como abordagem metodológica para auxiliar no processo de desenvolvimento das ferramentas.

3.4 Nível Conceitual

Nesta seção são apresentadas a metodologia utilizada e uma visão macro dos principais pontos do modelo conceitual do CORE-SL.

O nível conceitual tem um papel fundamental em toda a arquitetura do modelo, pois deve definir os elementos e seus relacionamentos para a representação dos sinais em um alto nível de detalhamento. Esses elementos são abstraídos a partir da revisão dos modelos linguísticos e das modelagens computacionais (revisados na fundamentação teórica).

O resultado desta etapa do *framework* consiste de um modelo conceitual em formato de árvore que relaciona todas as sub-unidades fonéticas revisadas e seus valores. O objetivo deste modelo é apresentar uma visão ampla da árvore de elementos que compõem os sinais e formalizar os termos técnicos para auxiliar o desenvolvimento do nível formal.

3.4.1 Metodologia Utilizada

A Figura 3.5 apresenta as etapas metodológicas utilizadas para a elaboração do modelo conceitual. O processo foi definido de maneira que possa ser aplicado novamente para a inclusão de novos conceitos, bem como para a extensão do modelo para outros níveis gramaticais.

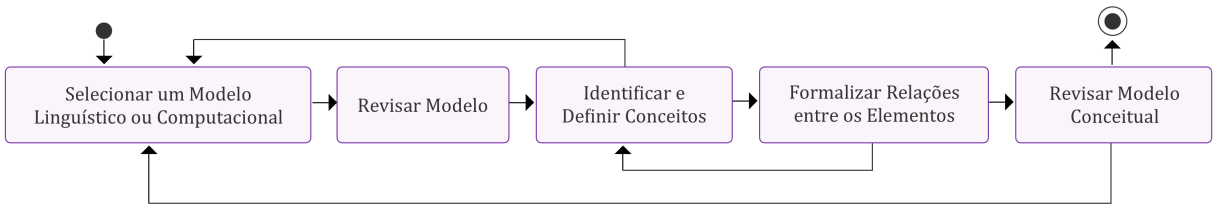


Figura 3.5: Processo para a Construção do Modelo Conceitual
Fonte: O autor (2014)

A primeira etapa consiste em selecionar um modelo (linguístico ou computacional) e, então, fazer a revisão deste modelo. Esta revisão consiste em compreender a organização

estrutural dos elementos e quais as características principais do modelo selecionado. Em seguida os conceitos principais do modelo escolhido são identificados e formalizados no modelo conceitual.

Na etapa de formalização dos conceitos e de suas relações é necessário analisar os elementos relacionados ou similares aos modelos já revisados. Por exemplo, nos modelos fonológicos revisados, o MMS e o MHT definem uma sub-estrutura similar para especificação da configuração dos dedos da CM.

Após identificar e formalizar os elementos do modelo escolhido, deve-se revisar a árvore conceitual para corrigir possíveis redundâncias ou ambiguidades. Por exemplo, como discutido no Capítulo 2, o conceito de MOV é representado de diversas maneiras nos modelos fonológicos.

Para evitar estas redundâncias foi utilizada a seguinte estratégia: definiram-se cinco classes principais de conceitos (CM, OP, LOC, MOV e ENM) baseado no MBP e para cada classe escolheu-se, em cada modelo revisado, o conceito mais abrangente e com características mais adequadas para a adaptação na representação computacional.

Essas características incluem: um alto nível de detalhamento do elemento, marcações de estado de início e fim, restrições para a articulação dos sinais, a organização estrutural e a forma que são incluídos outros níveis linguísticos.

3.4.2 Base Conceitual Inicial

O desenvolvimento desta tese é uma continuação à pesquisa de mestrado de Antunes (2011) [4], que desenvolveu uma modelagem em XML baseada no MBP e no MMS para a Libras. Nesta pesquisa foi apresentado um estudo em relação à representação computacional dos sinais e uma discussão inicial sobre o problema da similaridade. Destaca-se, também, que o trabalho de Antunes (2011) [4] está inserido no contexto desta tese.

Antunes (2011) [4] formaliza as sub-unidades fonéticas e seus respectivos valores na forma de um mapa conceitual e, posteriormente, modela cada descrição dos sinais em arquivos no formato XML seguindo os conceitos definidos no mapa. Uma visão macro do modelo conceitual proposto por Antunes (2011) [4] é apresentado na Figura 3.6.

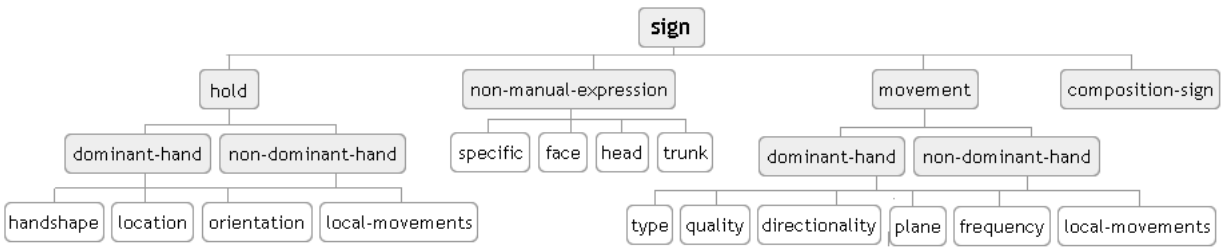


Figura 3.6: Visão geral da estrutura do Modelo de Antunes (2011)
Fonte: Antunes (2011) [4]

O Mapa Conceitual (MC) consiste de uma ferramenta para a representação de conhecimento organizado na forma de uma árvore ou uma "rede". O MC inclui os conceitos por meio de palavras-chave e os relacionamentos são representados por uma linha que conecta dois conceitos [90] e [38].

Adicionalmente, o MC também pode incluir relacionamentos entre os conceitos de domínios diferentes. Neste contexto, o MC mostra-se uma ferramenta adequada para a formalização da estrutura conceitual do CORE-SL.

Por exemplo, para incluir novos elementos ou níveis gramaticais ao CORE-SL é necessário apenas aplicar o processo metodológico (Figura 3.5) e formalizar os conceitos e seus relacionamentos.

Neste caso, se considerarmos a inclusão de um nível morfológico, o MC permite relacionar os seus conceitos intrínsecos e também fazer relações com outros níveis, tal como o fonético. Isto torna o MC uma ferramenta adequada para permitir a extensibilidade do CORE-SL.

3.4.3 Modelo Conceitual

Ao aplicar o *framework*, revisou-se os modelos fonológicos das LS e as modelagens computacionais. Como próxima etapa do *framework*, identificou-se os elementos e as estruturas dos modelos fonológicos com maior grau de granularidade, com o intuito de incluir estes conceitos no CORE-SL para possibilitar um alto nível de detalhamento nas descrições.

No final do Capítulo 2, a figura 2.17 (Apêndice A) apresentou uma visão geral dos conceitos identificados e selecionados com a revisão dos modelos fonológicos, durante o uso do *framework*.

Em seguida, todas as sub-unidades e as estruturas fonéticas foram formalizadas no MC (Apêndice B). Cabe ressaltar, que nesta etapa do *framework* precisou-se analisar o relacionamento de cada sub-unidade com o MC e com a base conceitual existente, com o intuito de evitar redundâncias.

Primeiramente, inspirado pela organização estrutural da modelagem do CSLML [115], adaptou-se a estrutura proposta por Antunes (2011) [4] para inserir como raiz do CORE-SL o conceito de Componente Fonético (*Phonetic Component*) (Figura 3.7).

Isto permite ao CORE-SL trabalhar com um padrão de *design* baseado em componentes, no qual cada componente ou módulo tem o papel de formalizar os elementos e os relacionamentos de um nível gramatical de uma LS. Ou seja, este padrão modular possibilita que o CORE-SL seja ampliado aos demais níveis da gramática tais como o morfológico, o sintático, dentre outros.

Cada sinal (*sign*) pode ser classificado em simples (*simple*) ou sequencial (*sequential*). O primeiro define um sinal que não possui movimentos de trajetória. O segundo define

um sinal que possui algum movimento e requer a descrição da sequencialidade.

Os sinais do tipo *sequential* podem ser articulados com uma mão ou com as duas mãos. No segundo caso, o CORE-SL inclui as condições de Simetria e de Dominância para auxiliar na consistência da formalização computacional e das representações geradas.

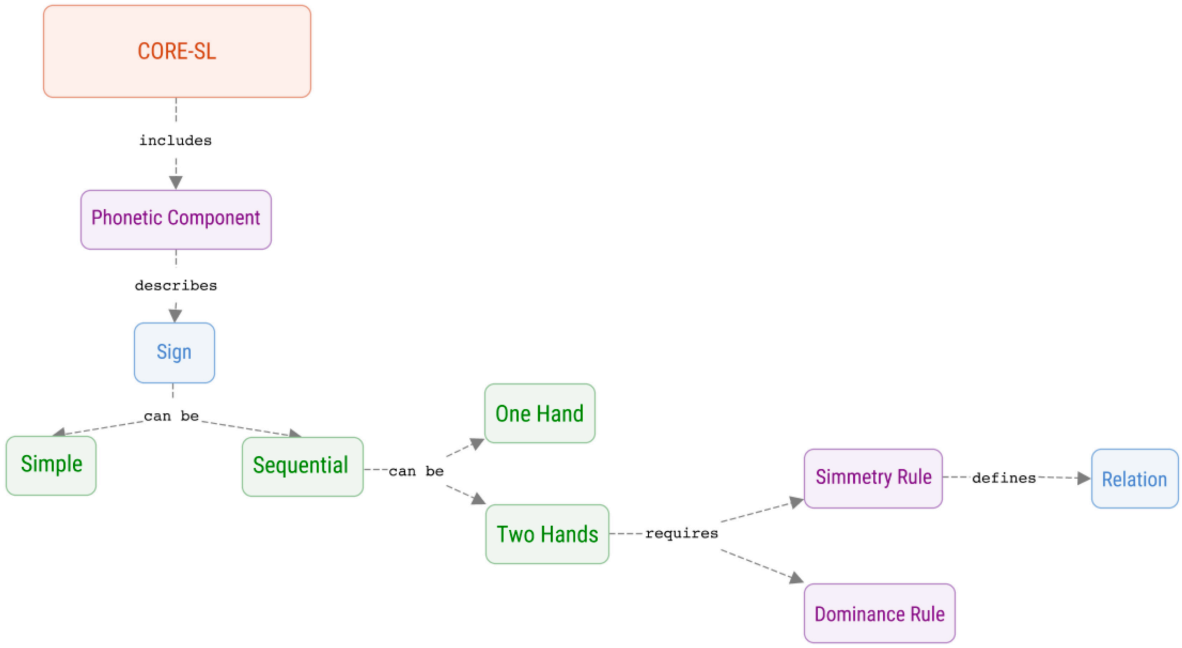


Figura 3.7: Visão macro do CORE-SL e do Componente Fonético
Fonte: O autor (2015)

Os sinais do tipo *sequential* são representados por meio de segmentos. Esses segmentos têm o intuito de possibilitar a representação da simultaneidade (os elementos descritos em cada segmento) e da sequencialidade no nível formal.

Cada segmento é formado por dois elementos: suspensão (*hold*) e movimento (*movement*), baseados na estrutura do MMS (Figura 3.8).

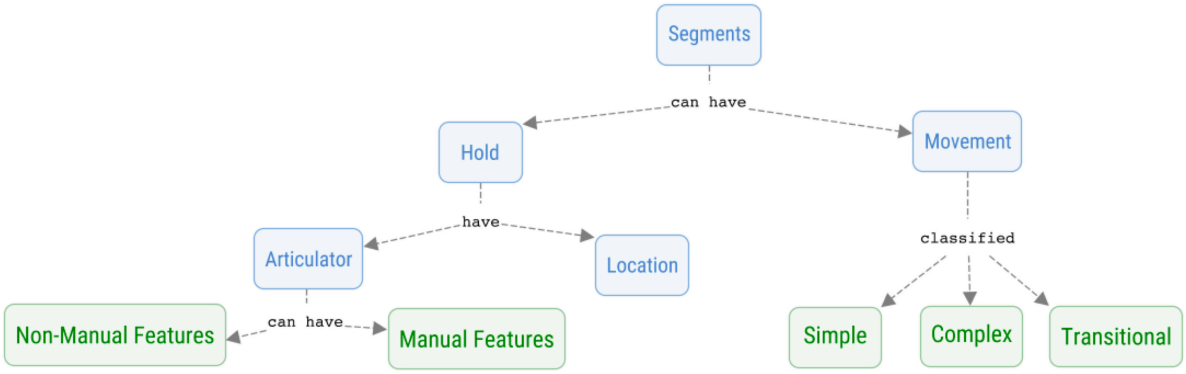


Figura 3.8: Componentes Internos *Hold* e *Movement* do CORE-SL
Fonte: O autor (2015)

Esta sub-divisão em um segmento para representar os articuladores manuais e não-manuais, e em um segmento para os aspectos de qualidade do movimento são utilizadas em outros modelos de maneira similar, tais como o MD, MHT e o MP.

Hold descreve os articuladores (*articulator*) manuais (*manual*) e não-manuais (*non-manual*), bem como uma estrutura para a descrição dos pontos de articulação (*location*). Esta organização foi considerada mais adequada computacionalmente, pois possibilita ao modelo formal definir as regras para os estados estáticos e para os dinâmicos.

O CORE-SL segue o conceito do MOV como uma transição entre duas LOC - tal como apresentado pelos modelos MHT, MP e MD. Porém, o CORE-SL adota a estrutura proposta pelo MP descrevendo o *hold* (os articuladores e a LOC) e o *movement* (detalhamento dos aspectos de qualidade do movimento).

Assim, o nível formal deve especificar as regras para a sequencialidade na forma de segmentos, onde *hold* e *movement* são executados simultaneamente em cada segmento e *movement* é opcional (i.e. não há movimento caso o segmento esteja definindo um estado final do sinal ou de uma sequência).

O CORE-SL também inclui a classificação de MOV proposta pelo MP: *simple* (simples) que são sinais formados por um MOV local ou um de trajetória, *complex* (complexo) que são sinais que usam MOV local e de trajetória simultaneamente, e *transitional* (transição) que corresponde aos MOV naturais de transição entre dois sinais.

A estrutura das características manuais foi definida a partir de uma combinação dos elementos do MMS, MHT e MP. Como apresentado na Figura 3.9, o CORE-SL descreve os parâmetros para as mãos dominante e para a não-dominante.

Como o CORE-SL inclui a condição de simetria [10], quando um sinal é articulado com ambas as mãos, a mão dominante pode descrever o componente *Relation NDH* que consiste em um detalhamento do seu posicionamento em relação à mão não-dominante (e.g. dedos cruzados, acima, lado a lado, etc.) [4].

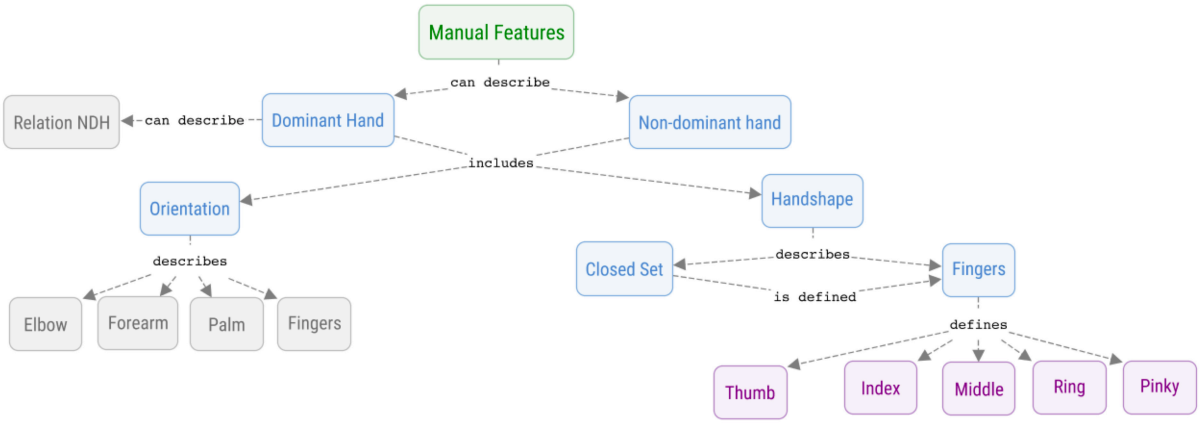


Figura 3.9: Estrutura Macro das Características Manuais do CORE-SL
Fonte: O autor (2015)

A representação das mãos é feita pela CM (*handshape*) e pela OR (*orientation*). A OR descreve a orientação de quatro sub-unidades principais: a palma (*palm*), a direção dos dedos (*fingers*), o posicionamento do antebraço (*forearm*) na vertical ou na horizontal, e o posicionamento do cotovelo (*elbow*).

Por exemplo, a OR pode detalhar o antebraço na horizontal dobrado, palma da mão para à esquerda, os dedos apontando para frente e o cotovelo em uma posição neutra. Uma variação em um desses quatro parâmetros pode mudar o significado do sinal. Este detalhamento é necessário, por exemplo, para o posicionamento correto dos braços em um sinal executado por um avatar 3D.

A CM é composta de dois elementos principais: *closed set* que consiste de um conjunto específico de CM de uma LS (e.g. as 73 configurações da Libras [46]), e *fingers* que descreve a disposição e as configurações de cada dedo (polegar - *thumb*, indicador - *index*, médio - *middle*, anelar - *ring* e mínimo - *pinky*).

Esta especificação dos dedos no CORE-SL foi baseada nos MP e MD. Por meio desta estrutura é possível derivar quaisquer configurações de mão e, portanto, esta sub-estrutura é responsável por definir cada CM do conjunto específico. Assim, o nível formal deve especificar um conjunto fechado de CM por meio da estrutura dos dedos.

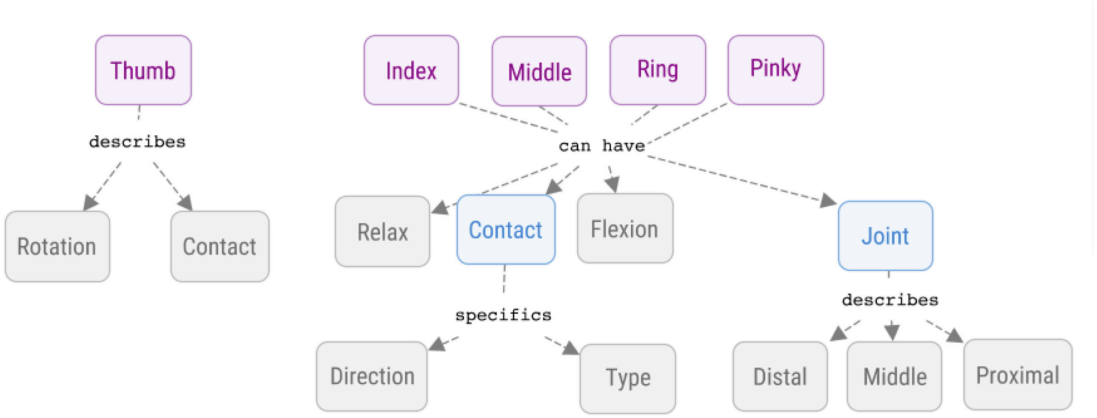


Figura 3.10: Detalhamento da CM por meio da estrutura dos dedos
Fonte: O autor (2015)

Os dedos indicador, médio, anelar e mínimo compartilham da mesma estrutura de elementos:

- *relax* que consiste em um atributo para indicar ou não o relaxamento dos músculos do dedos em questão;
- *flexion* que descreve o tipo de flexão de um dedo (e.g. aberto, fechado e curvado);
- *contact* que tem o papel de descrever se um dedo possui contato com os outros, o tipo de contato (e.g. cruzado) e a direção (e.g. polegar).

Adicionalmente, também foi incluído na descrição das configurações dos dedos um conceito de juntas (*joints*), que tem o papel de representar as três juntas de cada dedo: proximal, medial e distal. Esse nível de detalhamento pode ser necessário na fase de modelagem de avatares 3D.

Já o polegar (*thumb*) possui uma representação própria: a rotação (*rotation*) que indica se o dedo está paralelo ou adjacente aos demais, e o (*contact*) que descreve a forma de contato da ponta do polegar com os demais dedos (e.g. unha na almofada do indicador).

Os pontos de articulação (LOC) são representados pelo articulador *location* e são organizados conforme a Figura 3.11.

A LOC é dividida em quatro classes principais: *head* que representa todos os locais de articulação na cabeça, *body* que descreve as localizações no tronco, no pescoço, nos braços e nas pernas[2], *hand* que descreve os pontos específicos da mão passiva, e *space* que relaciona as posições no espaço de sinalização.

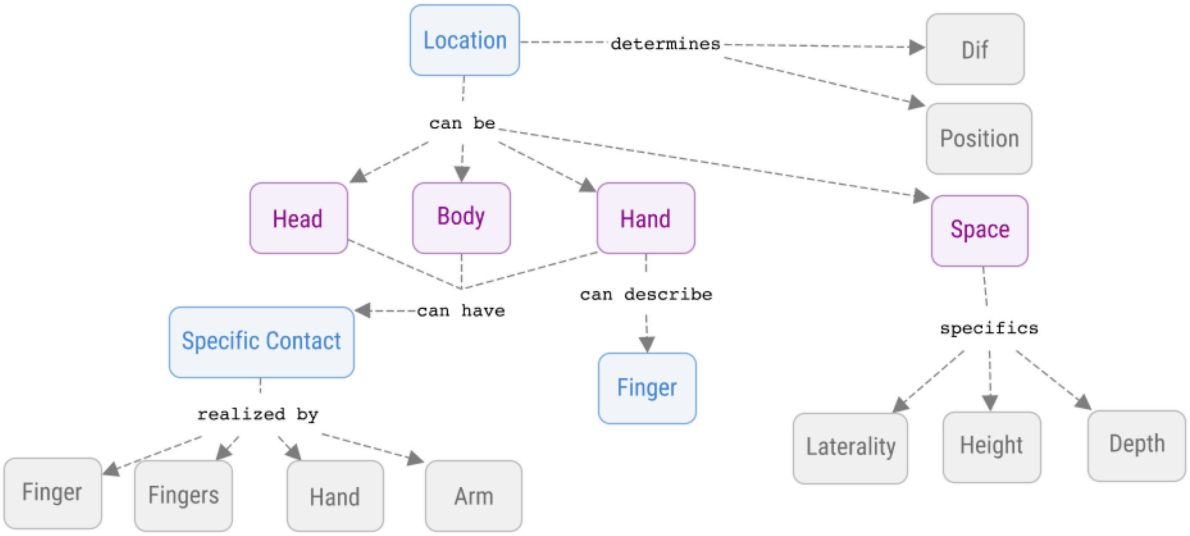


Figura 3.11: Estrutura para a Representação dos Pontos de Articulação
Fonte: O autor (2015)

A LOC ainda inclui um detalhamento em relação ao lado do corpo no qual o sinal é articulado (*position*) e um critério de elevação (*dif*), descrita pelo MMS e o MHT, responsável por descrever pequenas variações na LOC (e.g. um pouco acima ou um pouco abaixo do nariz). Estes dois elementos têm o intuito de aumentar a precisão de descrição.

Os pontos de articulação na cabeça, no corpo e nas mãos contém uma sub-estrutura denominada *specific contact*. Esta estrutura tem o papel de descrever a parte específica da mão dominante (CM) que realiza o contato com a LOC. Por exemplo, no sinal BANHEIRO da Libras (Figura 3.12) a ponta dos dedos da CM é que realiza contato com o braço (LOC).

[2]Os modelos não tem considerado as pernas como LOC, mas na Libras, por exemplo, existe o sinal SAIA que é representado com a LOC nas pernas.

Figura 3.12: Sinal BANHEIRO da Libras
Fonte: Antunes (2011) [4]

O contato específico (*specific contact*) é classificado em: dedos (*fingers*), dedo específico (*fingers*), mão (*hand*) e braço (*arm*). Cada um desses elementos possui seus próprios valores para a representação.

A representação da LOC no espaço (*space*) foi definida a partir do MMS, MHT e do MP. Esta sub-estrutura descreve as coordenadas X por meio da lateralidade (*laterality*), Y por meio da altura (*height*) e a Z por meio da profundidade (*depth*).

Esta convenção permite uma descrição detalhada, mas sem o uso de valores numéricos para o posicionamento das mãos no espaço. Por exemplo, a altura (*height*) descreve o posicionamento por meio da referência das LOC na cabeça e no tronco.

O segmento de movimento (*movement*) possui diferentes definições na literatura, como apresentado no Capítulo 2. Para o MOV, o CORE-SL utiliza uma organização híbrida baseada nas propostas do MHT, MD e MP que classifica os movimentos em locais (*local*) e de trajetória (*path*).

O segmento de MOV funciona como um traço separado dos articuladores, pois tem o intuito (assim como no MP) de descrever as características específicas do movimento.

Os movimentos de trajetória (*path*) são caracterizados pelo descolamento entre uma LOC inicial e uma final. Este deslocamento pode ser caracterizado pelo tipo (*type*), pela direcionalidade (*directionality*), pelo plano (*plane*) e pela maneira (*manner*) - Figura 3.13.

O tipo (*type*) descreve o MOV de trajetória em relação à forma *shape* (e.g. um MOV reto), à interação - *interaction* (e.g. um MOV alternado), e ao contato - *contact* (e.g. um MOV com toque na LOC).

A direcionalidade (*directionality*) descreve as direções do movimento e se sub-divide em dois tipos: *directional* que descreve somente uma direção ou *bi-directional* que descreve mais de uma direção (e.g. para frente e para trás).

A sub-estrutura denominada *manner* representa os traços de qualidade do MOV, tais como a velocidade (*velocity*), a repetição (*repetition*), a extensão (*extension*) e a tensão (tension). Como apresentado na etapa de revisão (Capítulo 2), a qualidade do MOV pode ter um papel no entendimento e na corretude de um sinal que está sendo articulado.

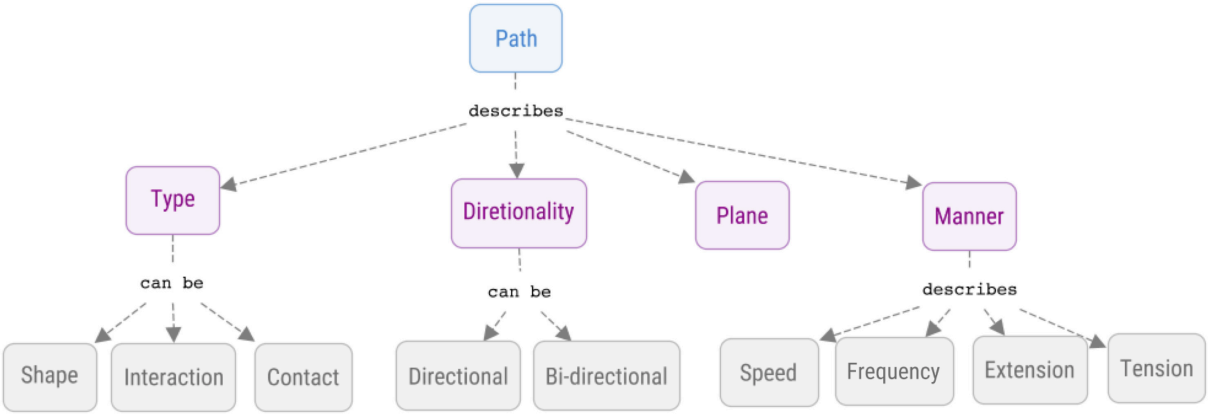


Figura 3.13: Estrutura Macro de Representação do Movimento de Trajetória
Fonte: O autor (2015)

Neste sentido, esses traços de qualidade podem melhorar a distinção dos sinais na entrada (*input*) e possibilitar uma reprodução (*output*) mais precisa e correta (e.g. por meio de um avatar 3D).

O CORE-SL também descreve os movimentos locais (baseado no MMS, MD e MP), que consistem de variações na CM ou na OR através da mão, dos dedos, do pulso ou do antebraço. Assim como os MOV de trajetória, os MOV locais também possuem elementos que descrevem a qualidade (e.g. tensão, velocidade e frequência).

Por exemplo, o sinal BOMBA DE ASMA da Libras (Figura 3.14) não possui movimento de trajetória, mas possui um movimento local no dedo indicador por meio da abertura e do fechamento (flexão) por duas vezes (frequência).

Figura 3.14: Exemplo do Sinal BOMBA DE ASMA da Libras com Movimento Local
Fonte: Antunes (2011) [4]

Cabe ressaltar que os movimentos de trajetória e locais podem ocorrer simultaneamente ou isoladamente na articulação de um sinal, de acordo com a classificação do MOV apresentado pelo MP.

As Expressões não-manuais (ENM) foram definidas com base nos estudos de [4], [42], [15], [40] e [8]. O CORE-SL classifica as ENM em três tipos: afetivas (*emotional*), sintáticas (*syntatic*) e fonéticas (*phonetic*). Uma abstração da estrutura das ENM é apresentada na Figura 3.15.

As ENM afetivas possibilitam ao interlocutor expressar sentimentos tais como alegria, tristeza, angústia, dúvida, surpresa, desgosto, ironia, dentre outras [42] por meio de expressões da cabeça, dos olhos ou da face.

Já as ENM sintáticas consistem de marcações verbo-visuais que "*fazem parte da arquitetura dos níveis fonológico, morfológico, sintático-semântico e discursivo em uma determinada língua*" [42].

Esses dois tipos de ENM podem ser representadas por meio de combinações dos traços da estrutura fonética [40], que possui três sub-estruturas: maneira (*manner*), configuração (*settings*) e posicionamento (*position*).

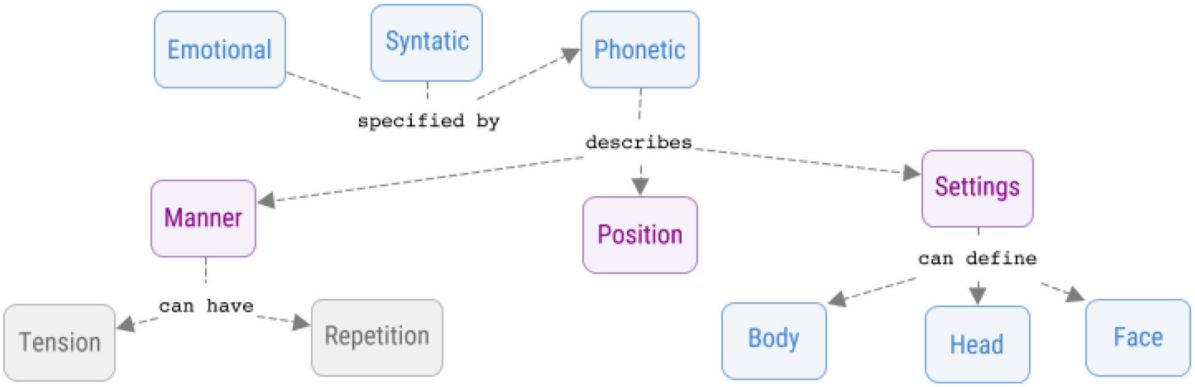


Figura 3.15: Estrutura Macro das Expressões Não-Manuais do CORE-SL
Fonte: O autor (2015)

A sub-estrutura de maneira agrega as características de repetição e de tensão, esta última especificada segundo os níveis de tensão apresentados na pesquisa de Ekman (1978) [40]: neutra, leve, média, severa e máxima.

O elemento (*position*) inclui as sub-unidades necessárias para a representação do posicionamento da cabeça ou do corpo do interlocutor na articulação do sinal. Esta sub-estrutura não descreve os movimentos de cabeça e de tronco, mas sim um posicionamento estático durante um sinal.

Como explana Felipe (2013) [42], *"a postura e movimento do corpo pode alterar o sentido do enunciado sinalizado, podendo estabelecer temporalidade, força ilocucionária para aproximação ou distanciamento dos interlocutores, hierarquia social, entre outras intencionalidades gramático-discursivas já apreendidas para a Libras"*.

Por exemplo, a cabeça pode ser posicionada para frente ou para trás (profundidade), pode ser deslocada lateralmente ou pode ser rotacionada (e.g. olhar para o lado). A característica dessa sub-unidade é a sua estaticidade na articulação do sinal.

Outro cenário de uso pode ser o posicionamento do tronco do interlocutor durante a articulação do sinal (e.g. direcionado para o receptor). Esta característica pode ser utilizada para configuração de perspectiva de visualização em um avatar 3D.

A título de exemplo podemos citar o sinal ALEMANHA da Libras, no qual a cabeça é rotacionada (i.e. olhando para o lado da mão dominante). Então, o sinal é articulado na testa com um movimento local de toque do dedo polegar.

Figura 3.16: Sinal ALEMANHA da Libras, com a cabeça rotacionada
Fonte: Antunes (2011) [4]

Por último, o CORE-SL descreve as configurações (*setting*) para os movimentos do corpo (tronco e ombros), da cabeça e da face (parte superior e parte inferior).

A face consiste das micro expressões realizadas pela parte superior do rosto (testa, sobrancelhas e olhos) e pela parte inferior (nariz, bochechas, boca, lábios e língua).

Diferentemente do posicionamento, esta sub-estrutura descreve os movimentos simultâneos ou sequenciais que ocorrem durante a articulação do sinal.

Por exemplo, no sinal AVIÃO da Libras (Figura 3.17) o interlocutor utiliza simultaneamente uma marcação não-manual de tronco e de cabeça (movimento diagonal para o lado da mão dominante) e uma expressão com a boca (vibrando os lábios fechados). Neste sinal, as ENM são executadas simultaneamente com as marcações manuais.

Figura 3.17: Sinal AVIÃO da Libras
Fonte: Antunes (2011) [4]

3.4.4 Propriedades

Completude

Definição 1. Dado um dicionário dos sinais $S = \{s_1, s_2, ..., s_n\}$ de uma língua de sinais L e um modelo computacional M que possui um conjunto de elementos $E = \{e_1, e_2, ..., e_n\}$

para representar sinais de L, M é completo somente se consegue representar qualquer sinal de S através do conjunto E.

De acordo com Yi (2006) [116], a modelagem computacional do corpo humano e a simulação de seus movimentos consistem em descrever a relação entre as juntas das mãos, os braços e o tronco, além de todas as expressões não-manuais.

Conjectura-se que o CORE-SL seja completo devido ao alto nível de detalhamento dos cinco parâmetros principais do MBP, por agregar as marcações gestuais-visuais e incluir as restrições (e.g. simetria, simultaneidade, dominância, etc) descritas nos principais modelos fonológicos estudados.

Como o CORE-SL foi composto pelas características mais abrangentes destes modelos fonéticos no que tange as possibilidades de movimento e das configurações, especificando valores para as juntas dos dedos, o pulso, o antebraço, o ombro, a cabeça, o tronco e as expressões faciais, existe uma alta probabilidade do CORE-SL ser completo.

Durante a modelagem, buscou-se especificar para cada sub-estrutura um nível máximo de granularidade, por exemplo, detalhando as configurações possíveis para as juntas dos dedos. Esta granularidade indica uma baixa probabilidade de inclusão de novas sub-estruturas e, consequentemente, de novas regras no formalismo.

Adicionalmente, o CORE-SL faz a inclusão de alguns dos parâmetros e das sub-estruturas específicas das modelagens computacionais para a marcação de gestos ou para a descrição de sinais, tais como o CSLML [115], o HamNoSys [64] e o SIGML [57]. Esses modelos consistem de estruturas genéricas para a representação de movimentos e de expressões faciais do ser humano.

Destaca-se que a completude de um modelo computacional de representação é limitada pelas possibilidades anatômicas da biomecânica do movimento humano [63]. Por exemplo, considere por absurdo um sinal em que a cabeça rotacione 360 graus.

Caso haja a necessidade de incluir um novo valor para a descrição de um determinado sinal, basta incluir este valor no conjunto de símbolos terminais do formalismo, mantendo a propriedade de completude ao modelo proposto.

Por exemplo, considere que o CORE-SL não disponha de um valor para descrever o "piscar dos olhos". Este caso não invalida a completude, pois este valor pode ser incluído no alfabeto do formalismo na regra específica de descrição das marcações não-manuais dos olhos.

A hipótese da completude pode ser verificada por meio de uma prova por exaustão, na qual checam-se todos os sinais de uma LS para tentar encontrar um exemplo para provar que o CORE-SL não é completo. Uma vez que o vocabulário de uma língua é dinâmico e não-finito, este cenário de teste demanda tempo e um alto custo, sendo inviável para esta tese.

Extensibilidade

Definição 2. Considere um modelo computacional M para a representação dos sinais de uma língua de sinais L por meio do nível fonético. A extensibilidade consiste na capacidade de M em incluir novos níveis gramaticais de L.

O CORE-SL foi projetado com base em componentes, ou seja, cada nível gramatical de uma LS pode ser especificado por meio de um módulo que pode utilizar ou estender os outros níveis gramaticais já modelados.

Além da inclusão de novos componentes, o CORE-SL também possibilita que um novo módulo reutilize um nível anterior na sua especificação. Para exemplificação, podemos citar o nível morfológico que possui diversos elementos formados pela combinação de sub-unidades do nível fonético.

Como instância temos que as ENM (componente fonético) podem caracterizar um morfema complexo para marcas adjetivas, adverbiais, pronominal, interjeições, de tempo verbal e de grau em adjetivos [42].

Felipe (2013) [42] apresenta como exemplo um morfema adverbial do tipo intensificador: para o sinal TRABALHAR da Libras (Figura 3.18), o uso de uma ENM específica (e.g. o franzir das sobrancelhas ou as bochechas infladas e com arregalar dos olhos) articulada simultaneamente a um adjetivo, um substantivo ou um verbo marca um intensificador.

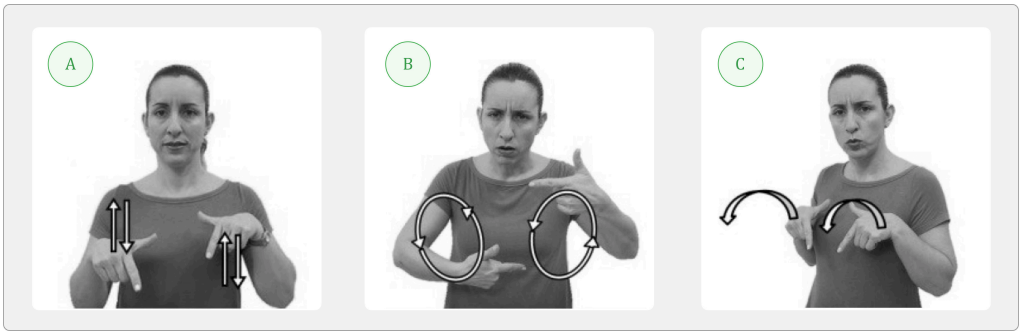


Figura 3.18: Sinais: A) Trabalhar, B) Muito Trabalhar e C) Continuamente Trabalhar
Fonte: Modificado pelo autor (2015) [42]

Outro exemplo para demonstrar que a combinação de sub-unidades fonéticas pode ser estendida para a representação de elementos de outros níveis gramaticais é o caso da derivação de sinais por composição. Neste caso, um sinal de uma LS pode ser representado pela composição de sinais existentes no vocabulário. Este é o caso do sinal IGREJA, que é composto pelos sinais CASA e CRUZ articulados em sequência (Figura 3.19).

Figura 3.19: Sinal IGREJA composto pelos sinais CASA e CRUZ
Fonte: Felipe (2002) [45]

Portanto, espera-se que o CORE-SL seja estendido aos outros níveis gramaticais das LS, para especificar os níveis morfológico, sintático, semântico-discursivo e o pragmático. A título de exemplo, uma abstração conceitual desta organização de componentes é mostrada na Figura 3.20.

Embora a extensibilidade esteja relacionada com os níveis gramaticais, entende-se que pode haver uma aplicação desta propriedade em outras áreas, tal como a Computação (e.g. um nível específico para visão computacional).

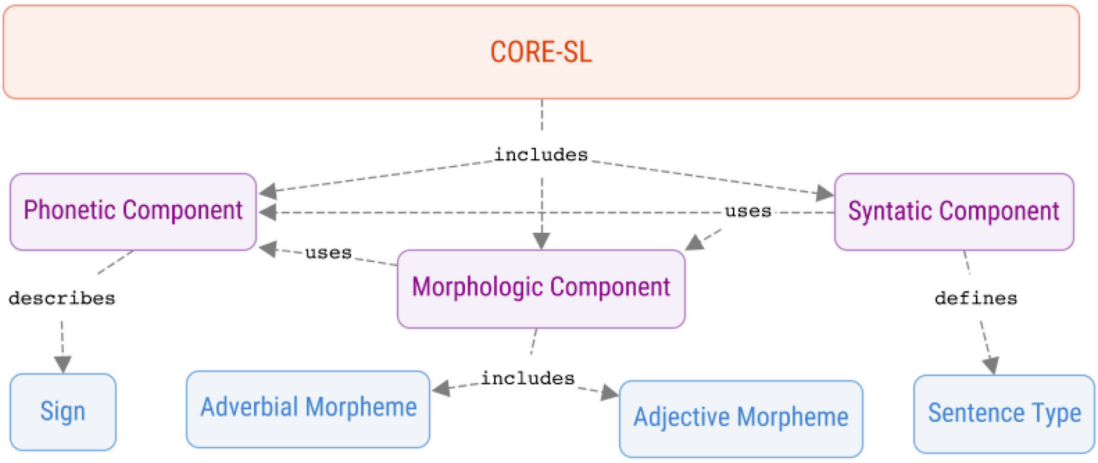


Figura 3.20: Exemplo do CORE-SL para os Níveis Fonético, Morfológico e Sintático
Fonte: O autor (2015)

3.4.5 Considerações

Destaca-se que diversos conceitos e restrições da Linguística foram incluídas no modelo conceitual, tais como a classificação de sinais com uma ou duas mãos [66], a sequencialidade [80], as restrições de simetria e de dominância e a simultaneidade das sub-unidades em cada segmento que compõe um sinal.

Embora o CORE-SL descreva somente o componente fonético, a propriedade de extensibilidade possibilita que especialistas das LS descrevam outros níveis gramaticais.

A próxima etapa do *framework* consiste da descrição do nível formal, que tem o papel de formalizar as regras para a representação dos sinais com base no modelo conceitual. Neste nível são definidas as regras para os conceitos de simultaneidade e de sequencialidade, que não são totalmente claras no modelo conceitual.

3.5 Nível Formal

Esta seção apresenta um estudo relacionado ao formalismo adotado e à definição das regras de produção necessárias para a representação computacional dos sinais com base no modelo conceitual do CORE-SL.

O Nível Formal também discute sobre a meta-linguagem escolhida e a metodologia utilizada para a formalização (fundamental se houver a necessidade, posteriormente, da inclusão de novos parâmetros ou regras).

Adicionalmente são discutidas algumas propriedades fundamentais à formalização do CORE-SL, que impactam diretamente na corretude de representação dos sinais e na análise sintática (*parsing*) do modelo.

3.5.1 Metodologia

A Figura 3.21 apresenta as etapas metodológicas utilizadas para a definição das regras formais do CORE-SL. A documentação deste processo é importante para permitir sua aplicação caso novas regras ou valores sejam inseridos no modelo.

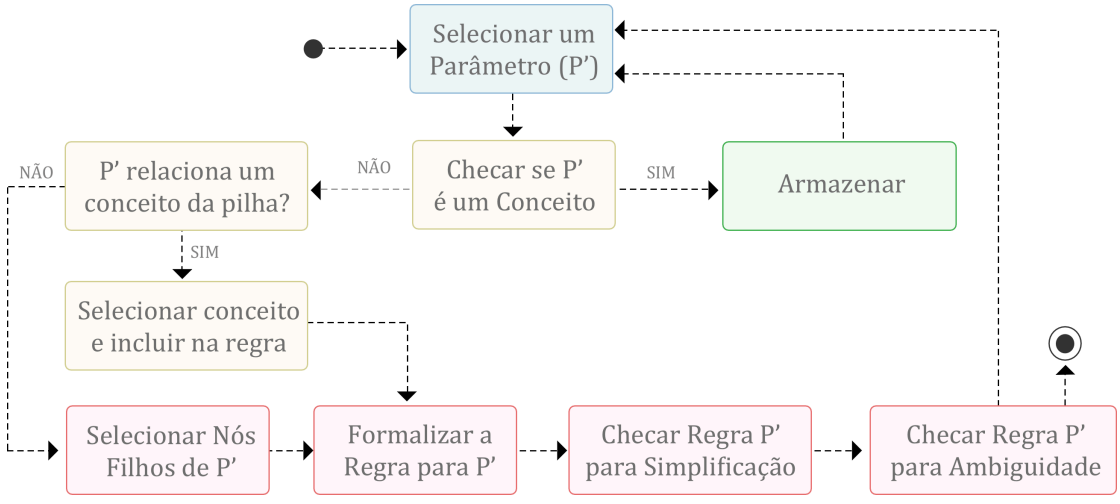


Figura 3.21: Processo para Formalização do CORE-SL
Fonte: O autor (2015)

A primeira etapa consiste em selecionar um parâmetro P' no modelo conceitual. Então, deve ser verificado se P' é um conceito, ou seja, se consiste de uma restrição ou de uma particularidade que deve ser incluída em uma regra específica.

Por exemplo, no início da árvore conceitual existe um elemento denominado *symmetry rule* (regra de simetria). Neste caso, o elemento não consiste de um parâmetro, logo, não consiste de um símbolo não-terminal responsável por definir uma regra de produção. Se P' for um conceito, então ele deve ser armazenado em uma estrutura de dados para aguardar a definição da regra que o inclua.

Se P' não é um conceito, então deve-se checar se este parâmetro relaciona um conceito armazenado na estrutura de dados. Se esta verificação é verdadeira, este conceito deve ser selecionado e incluído na regra de produção para P'.

A definição de uma regra de produção para P' é realizada em quatro passos:

1. Selecionar os nós filhos de P' na árvore conceitual. Esses filhos podem ser símbolos terminais (i.e. valores que instanciam um parâmetro) ou símbolos não-terminais (i.e. novos parâmetros);

2. Então, deve-se formalizar uma regra para P' considerando seus filhos e seguindo as restrições da gramática escolhida;

3. Deve-se checar a regra definida quanto à simplificação (eliminação de símbolos vazios, não utilizados, etc.);

4. Verificar se a regra definida está na forma normal para tentar evitar a ambiguidade.

O *framework* deve ser repetido até que todos os parâmetros, nós interiores e as folhas (símbolos terminais) forem selecionados e suas regras forem definidas.

3.5.2 Formalismo

O CORE-SL é formalizado na forma de uma Gramática Livre de Contexto - GLC (Definição 3 e 4), por meio da meta-linguagem EBNF. Para uma visualização mais clara das regras também são utilizados diagramas de sintaxe (*Railroad Diagrams*).

Definição 3. Uma gramática G é definida por uma quadrupla $G = (V, \Sigma, P, S)$, tal que:

- $\boldsymbol{V}$ consiste em um conjunto finito de variáveis (não-terminais);
- Σ consiste de um conjunto finito de símbolos terminais ou valores (alfabeto);
- $\boldsymbol{P}$ consiste do conjunto de regras de produção;
- $\boldsymbol{S}$ consiste de um elemento distintivo em V denominado de símbolo inicial.

Definição 4. (GLC) Uma gramática $G = (V, \Sigma, P, S)$ é livre de contexto se toda regra de produção possui a forma $R \rightarrow \alpha$ tal que $R \in V$ e α é um elemento do conjunto $V \cup \Sigma$.

Ao aplicar o processo (Figura 3.21), temos o símbolo inicial denominado de `core_sl` que descreve os componentes do modelo, tais como o fonético (`phonetic_component`) e o morfológico (`morphologic_component`). Esta regra inicial é importante pois caso novos níveis linguísticos sejam inseridos na gramática, então seu símbolo não-terminal deve ser inserido no final da regra `core_sl` e sem repetição.

O `phonetic_component` possui um identificador único (`identifier`) para descrever cada sinal (`sign`). Isso possibilita as referências entre as descrições (e.g. no caso de um sinal utilizar uma expressão não-manual específica já descrita pelo CORE-SL).

O `sign` é definido como simples (possui somente uma `hold`), ou sequencial (possui um ou mais segmentos - `segment`), ou pode ser uma soletração (que consiste em uma sequência de `hold`).

Um `segment` é responsável por definir a sequencialidade de sinais com movimento. Para tanto, o `segment` obrigatoriamente descreve uma `hold` (estado inicial), o movimento e, por último, uma nova `hold` (estado final).

A Tabela 3.1 apresenta um exemplo das regras descritas acima para os parâmetros principais da raiz do CORE-SL.

Tabela 3.1: Regras de produção para os símbolos iniciais do CORE-SL na EBNF

```
core_sl = phonetic_component, (morphologic_component?);
phonetic_component = identifier, sign;

sign = (‘simple’,hold | ‘sequential’, (segment+), hold | ‘spelling’,hold+);
segment = hold, movement;
```

Fonte: O autor (2015)

Para exemplificar, as regras apresentadas na Tabela 3.1 podem ser representadas, da mesma maneira, por meio do diagrama de sintaxe (Figura 3.22)[3]. Esses diagramas possibilitam uma visualização gráfica das regras, agregando ao CORE-SL maiores potenciais de comunicabilidade e de usabilidade.

Nos diagramas de sintaxe da Figura 3.22 pode-se perceber as relações de repetição, de escolha, de opção e de concatenação. Por exemplo, no `phonetic_component` percebe-se a concatenação de `identifier` e `sign`. Essas características também auxiliam na representação dos aspectos das SL, tais como a simultaneidade, a sequencialidade e a segmentabilidade (conforme discutido na sub-seção de propriedades).

[3]Existem diversos pacotes de software para a geração automática de diagramas de sintaxe a partir de uma gramática BNF ou EBNF. Para esta tese, utilizou-se o *RailRoad Diagram Generator* (http://bottlecaps.de/rr/ui).

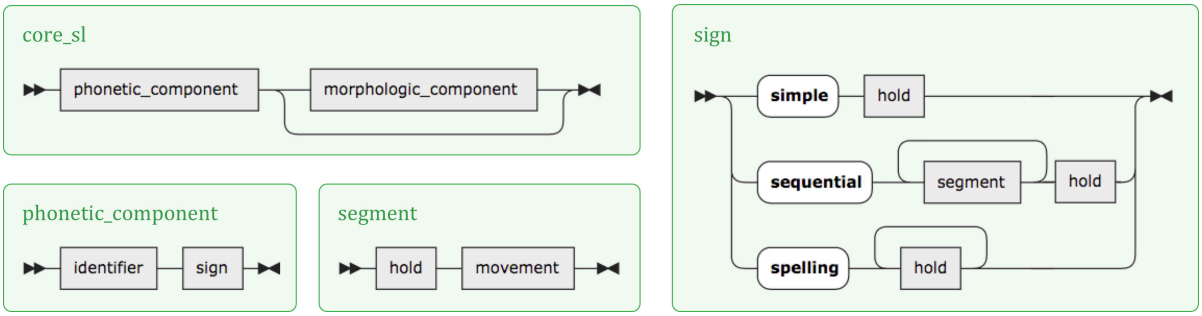


Figura 3.22: Diagrama de Sintaxe para as regras de produção iniciais do CORE-SL
Fonte: O autor (2015)

A Tabela 3.2 apresenta as regras de produção para a definição do articulador manual (`articulator`). Quando este parâmetro foi selecionado na árvore conceitual por meio do processo, verificou-se a necessidade de definir três conceitos que estavam armazenados na estrutura de dados: *one hand*, *two hands* e *symmetry rule*.

Como especificado no modelo conceitual, um sinal pode ser articulado com uma ou com as duas mãos. O primeiro caso, consiste apenas em especificar a mão dominante. No segundo caso, é necessário definir a mão dominante (`dominant_hand`) e a não-dominante (`non_dominant_hand`), bem como a relação entre elas (`relation_non_dominant`).

O parâmetro `non_dominant_hand` descreve a regra de simetria (`symmetry_rule`) como de valor verdadeiro ou falso. Caso seja verdadeiro, significa que as mãos utilizam as mesmas CM e LOC com uma relação simétrica, definida pelo parâmetro `symmetry_relation`.

Tabela 3.2: Regras de produção para o articulador manual do CORE-SL

```
manual = 'one hand', dominant_hand |
         'two hands', relation_non_dominant?,
         dominant_hand, non_dominant_hand;

relation_non_dominant = 'up' | 'down' | 'front' | 'back' | ...  ;
non_dominant_hand = symmetry_rule | dominant_hand;
symmetry_rule = 'true', symmetry_relation | 'false', 'passive';
symmetry_relation = 'symmetry' | 'crossed_fingers' | 'palm_palm' |...;
```

Fonte: O autor (2015)

Por outro lado, se o parâmetro `symmetry_rule` for falso, então significa que a mão não-dominante representa uma configuração de mão passiva (i.e. que é representada como um ponto de articulação para a mão dominante - regra de dominância)[4].

[4]Destaca-se a necessidade de um estudo linguístico em relação às regras de simetria e de dominância no vocabulário da LS que o CORE-SL estiver representando, pois em certos casos podem existir sinais no

Adicionalmente, o conceito de simetria também precisou ser definido para a regra de movimento, tanto para os movimentos locais quanto para os de trajetória.

Neste caso, a simetria consiste em descrever, para os sinais articulados com as duas mãos, que respeitam a condição de simetria, a relação simétrica das mãos durante o movimento: simétrica, alternada e simultânea.

Como exemplos podemos considerar os sinais DESENVOLVER (Figura 3.23.A) e ATENÇÃO (Figura 3.23.B) da Libras. No sinal DESENVOLVER a CM, a OR e a LOC são simétricas e o movimento é articulado de maneira alternada (primeiro a mão dominante e depois a mão não-dominante, repetidamente - respeitando o parâmetro de frequência). Já no sinal ATENÇÃO, o movimento das mãos ocorre de maneira simétrica.

Figura 3.23: A) Relação de Alternância e B) Relação de Simetria entre as Mãos
Fonte: Modificado pelo autor (2015) [4]

O conceito de simetria foi definido na forma de um parâmetro adicional à regra de produção dos movimentos local e de trajetória: `symmetry_movement` (Tabela 3.3).

Tabela 3.3: Algumas regras de produção do parâmetro de movimento do CORE-SL

```
movement = simple | complex;
simple = local | path;
complex = local, path;
local = hand?, symmetric_movement?, direction?, frequency?, speed?,
        tension?, mov_hand?, mov_wrist?, mov_forearm?, mov_fingers?;

path = hand?, symmetric_movement?, type, plane, directionality, manner;
```

Fonte: O autor (2015)

Na regra de movimento (`movement`), também podemos notar a capacidade de o formalismo representar movimentos simples (`simple`) e complexos (`complex`), tornando possível

vocabulário que não foram criados com base nestas regras. Antunes (2011) [4] apresenta alguns exemplos, tais como JORNAL e POLVO. O autor argumenta que estes sinais também podem ter sido articulados incorretamente pelo intérprete na gravação.

que um movimento seja representado somente com um movimento local ou com de trajetória (simples), ou combinar os dois tipos de movimento simultaneamente (complexo).

Complementarmente, podemos notar nas regras para os movimentos local e de trajetória um parâmetro opcional denominado `hand`. Por definição (baseada na Linguística), sinais produzidos com as duas mãos devem respeitar a condição de simetria para o movimento de trajetória (movimentos das duas mãos devem ser simétricos, alternados ou simultâneos).

Também como um padrão, se o movimento de trajetória é realizado somente por uma mão, então quem articula este movimento é a mão dominante. É importante notar que o modelo permite uma descrição simétrica das configurações das mãos (`hold`), principalmente para minimizar o tamanho da representação, mas o movimento pode ser realizado apenas com uma das mãos.

Por exemplo, no sinal ABAIXO da Libras (Figura 3.24) [45] as mãos possuem simetria (CM, OR e LOC) e a mão dominante possui uma relação abaixo da mão não-dominante. Porém, o movimento não é simétrico, pois é articulado somente pela mão dominante (um movimento do tipo reto, para baixo).

Figura 3.24: Sinal ABAIXO da Libras
Fonte: Felipe (2002) [45]

No exemplo do sinal ABAIXO a condição de simetria do movimento de trajetória continua sendo respeitada pelo CORE-SL. Este exemplo apenas deixa clara à diferença entre a simetria das configurações de mão (para minimizar o número de descrições) e a simetria de movimentos de trajetória (que requerem a Condição de Simetria de [10]).

Com o intuito de minimizar a redundância nas regras, observou-se a existência de parâmetros que compartilhavam de uma mesma definição e, assim, simplificou-se esta definição em uma regra própria. Por exemplo, os dedos indicador, médio, anelar e mínimo possuem as mesmas definições, logo, foi definida a regra de produção `finger` (Tabela 3.4).

Com a aplicação do processo para os parâmetros dos dedos (Tabela 3.4), destaca-se a capacidade da meta-linguagem escolhida em representar uma regra de escolha.

Por exemplo, na regra de produção para `finger`, é necessário definir uma flexão geral (`finger_flexion`) para o dedo em questão (e.g. aberto ou curvado). Caso contrário,

torna-se necessário descrever cada uma das juntas dos dedos (proximal, medial e distal) por meio da regra `joint`.

Tabela 3.4: Regras de produção para os dedos no CORE-SL

```
finger = relax, finger_contact, (finger_flexion | joint);

relax = 'true' | 'false';
finger_flexion = 'opened' | 'closed' | 'flexed' | 'curved'
               | 'flexed (proximal joint)' | 'flexed (middle joint)'
               | 'flexed (distal joint)';

finger_contact = contact_type, contact_direction;
contact_type = 'stacked' | 'spread' | 'lateral' | 'crossed';
contact_direction = 'thumb' | 'pinky' | 'neutral';
```

Fonte: O autor (2015)

Uma questão importante que a metodologia e o modelo conceitual proporcionaram ao formalismo foi a definição de conjuntos específicos de valores (`closed set`) para alguns parâmetros, tais como as configurações de mão e as expressões não manuais.

Na Libras, por exemplo, existem 73 configurações de mão já definidas por meio das configurações dos dedos. Como cada língua possui seu conjunto de CM, o CORE-SL possibilita tanto a descrição de cada CM, quanto a definição de um conjunto fechado.

Esta capacidade é importante, pois facilita a descrição de novos sinais por meio da gramática e auxilia na redução do tamanho do arquivo para o armazenamento de cada descrição. Cada regra de um conjunto fechado (símbolo terminal) consiste de de uma sentença produzida e validada pela gramática.

Para exemplificação, pode-se considerar a definição da configuração de mão em "cinco" da Libras (mão e dedos abertos). A Tabela 3.5 apresenta uma descrição textual desta CM de exemplo denominada `handshape_1`. Para facilitar a leitura e o entendimento, o autor desta tese adotou uma representação no formato JSON[5] ao invés de uma string padrão. A formalização completa do CORE-SL é documentada no Apêndice C.

A seguir é apresentada uma discussão com relação às propriedades internas do CORE-SL no Nível Formal. Esta discussão é importante, pois estas propriedades impactam diretamente na qualidade do modelo formal, bem como no desenvolvimento do *parser* e na estratégia de busca por sinais.

[5]JavaScript Object Notation: uma notação formal em árvore para a representação de dados, utilizado comumente como saída em uma API.

Tabela 3.5: Exemplo da CM em Cinco da Libras pelo CORE-SL

```
handshape_1 = [{
    "thumb": {
          "thumb_rotation": "lateral",
          "thumb_flexion": "opened"
    },
    "index": {
      "relax": "false",
      "finger_contact": {
        "contact_type": "spread",
        "contact_direction": "thumb"
      },
      "finger_flexion": "opened"
    },
    ...
}]
```

Fonte: O autor (2015)

3.5.3 Propriedades

Concisão

Definição 5. Um modelo de representação M é conciso se sua meta-linguagem de descrição possibilitar que as regras de produção sejam definidas rapidamente e facilmente compreendidas.

A concisão relaciona a qualidade da meta-linguagem e o contexto computacional da gramática que ela descreve. Neste sentido, a meta-linguagem deve ser apropriada para definir as regras de produção do domínio em questão.

No caso do CORE-SL, a concisão é determinada pela simplicidade da notação EBNF para a representação dos conceitos de concatenação, de repetição (recursão), de escolha e de opção. Devido às suas extensões e à sua estrutura linear, as regras de produção do CORE-SL foram especificadas de maneira rápida seguindo a árvore conceitual.

Para as variáveis e os valores representados na gramática, foram utilizados os termos técnicos da literatura das LS que são de fácil entendimento (e.g. *non-manual expression*). Quando um termo técnico não era de fácil entendimento (e.g. ulnar) ele foi substituído por uma palavra simples que representava o seu conceito (e.g. *pinky side*) [66].

Adicionalmente, os diagramas de sintaxe (*railroad diagrams*) permitem um claro entendimento das regras formais, bem como seu fluxo de concatenação, de escolha e de repetição, possibilitando que um público menos técnico (na Ciência da Computação e na

Linguística) compreenda facilmente a formalização do CORE-SL.

Formal

Definição 6. Um modelo de representação M é formal se as regras de produção da sua gramática podem ser analisadas e processadas computacionalmente.

Esta propriedade é validada pela própria literatura, uma vez que o CORE-SL é formalizado por meio da notação EBNF, que é utilizada para a representação de gramáticas livres de contexto para linguagens de programação e modelos computacionais [87]. Portanto, o CORE-SL mostra-se passível de implementação computacional para o processamento de suas regras e para a análise sintática (*parsing*).

Unicidade

Definição 7. (Ambiguidade) Uma gramática livre de contexto G é ambígua se existe uma palavra P, produzida por G, que possui mais de uma árvore de derivação (à esquerda ou à direita).

Definição 8. (Unicidade) Um modelo computacional M possui unicidade de representação, se todo sinal produzido pelas regras da gramática de M possuir somente uma única forma de representação (forma canônica).

O problema da ambiguidade impacta diretamente na qualidade da implementação de *parsers* e, consequentemente, pode dificultar o desenvolvimento de estratégias eficientes de busca de sinais.

Computacionalmente, não existe um algoritmo que decida se uma gramática é ambígua. Da mesma maneira, a remoção da ambiguidade em uma gramática não é possível por um algoritmo [87], [103]. Alguns *parsers* utilizam estratégias de semi-decisão ou baseadas em heurísticas para resolver tipos específicos de ambiguidade [87].

Por exemplo, se a ambiguidade consiste de um problema de ordenação ou de precedência, a implementação de um *parser* pode encontrar uma função para transformar uma sentença em sua forma canônica. Da mesma forma, algumas restrições podem ser inseridas na gramática para minimizar a ambiguidade, tais como operadores de precedência e minimização de recursão.

A questão da unicidade também pode ser relacionada com a maneira que um humano interpreta um sinal e o traduz como uma descrição do modelo formal. Ou seja, se o problema for semântico é provável que um analisador sintático e/ou a gramática não consigam resolvê-lo. Para minimizar o risco de ambiguidade em uma gramática, bem como melhorar seu desempenho durante o processamento, podem ser utilizadas algumas técnicas durante a definição das regras de produção, tal como a simplificação.

Definição 9. (Simplificação de Gramática) Simplificar uma gramática G consiste em realizar manipulações em G, sem afetar as produções geradas, visando tornar as regras mais simples e em um conjunto numericamente menor. As operações de simplificação são:

- Eliminação de produções vazias na forma $A = \epsilon$, se ϵ não pertence à gramática;
- Eliminação de produções unitárias na forma $A = B$, onde B apenas possibilita a substituição desta variável por um símbolo terminal ou não-terminal. Este tipo de regra não agrega qualquer informação durante a análise sintática.
- Eliminação de símbolos inúteis: um símbolo não-terminal A é inútil, se não for possível derivar pelo menos uma sentença formada somente por símbolos terminais a partir de A.
- Eliminação de símbolos inacessíveis: um símbolo A (terminal ou não-terminal) é inacessível, se não for possível derivar uma sentença pela qual o analisador sintático registre uma ocorrência de A.

O processo metodológico proposto (Figura 3.21) inclui uma etapa para verificar se cada regra de produção definida está na forma simplificada. Desta forma, buscou-se remover as produções vazias, os símbolos inúteis e as produções unitárias.

Para as produções unitárias, foi definida a seguinte estratégia: se o parâmetro candidato à unitário representa um conceito importante no modelo conceitual, então este símbolo não-terminal é removido e transformado em um terminal (*string*) para a marcação deste conceito. Como exemplo, os conceitos *one hand* e *two hands* foram transformados em terminais na regra para o articulador `manual`, apresentado na Tabela 3.2.

Em relação à eliminação de símbolos terminais inacessíveis, argumenta-se que o CORE-SL, por englobar os símbolos terminais descritos pelos modelos linguísticos fonéticos, tais símbolos devem ser utilizados em pelo menos uma produção de algum sinal da LS. Logo, considera-se que os símbolos terminais do CORE-SL são acessíveis.

Outra questão que impacta a unicidade de representação, consiste na ambiguidade semântica inerente às LS. Devido as características intrínsecas às línguas naturais (e.g. polissemia, homonímia, entre outras), podem existir sinais em uma LS que apresentem estruturas fonéticas similares ou iguais para sinais com significados (semântica) distintos. Isto gera um problema, por exemplo, para um cenário de reconhecimento automático de sinais via câmera, pois há o desafio de distinguir pequenas variações dos parâmetros, variações dos interlocutores, ruídos do ambiente, a distância em que o discurso está sendo realizado, entre outros.

Esta é uma questão que exige um esforço adicional da Linguística e da Computação, pois esta entrada de dados de maneira natural (via câmera) depende de funcionalidades de

inteligência e de percepção para tentar distinguir sub-unidades fonéticas muito similares ou mesmo variações interpessoais dos usuários.

Embora haja este desafio quanto à ambiguidade semântica, o CORE-SL possibilita descrever e distinguir os sinais de uma LS. Assim, o problema de ambiguidade inerente às LS não foi tratado pelo CORE-SL e exige um estudo específico, por exemplo, relacionado ao léxico de uma LS (para levantar possibilidades de ambiguidade entre sinais) e em relação à articulação dos sinais pelos interlocutores (para buscar compreender aspectos de percepção e variações naturais de cada usuário).

Segmentabilidade

Definição 10. Um modelo computacional M é segmentável se possibilita a divisão de um sinal S em fragmentos menores que podem ser realizados simultaneamente ou sequencialmente.

O CORE-SL possibilita esta divisão de um sinal de uma LS em segmentos, uma vez que suas regras de produção são baseadas no modelo conceitual: uma árvore que sub-divide os elementos que compõem os sinais em fragmentos menores denominados símbolos não-terminais e terminais.

Embasado pela propriedade de *extensibilidade*, se considerarmos um segmento como a descrição das características de um nível gramatical de uma LS, então pode-se inferir que o CORE-SL também permite esta segmentabilidade, uma vez que possibilita a inclusão de novos níveis gramaticais de maneira independente.

Simultaneidade

Definição 11. Um modelo computacional M possibilita a simultaneidade na representação de um sinal S, se as sub-unidades de S articuladas de forma simultânea podem ser descritas em M em um mesmo segmento, respeitando sua característica síncrona.

A simultaneidade é uma das características fundamentais na articulação dos sinais, uma vez que descreve quais sub-unidades são articuladas ao mesmo tempo pelo interlocutor. O CORE-SL classifica os sinais em dois tipos: unitários e sequenciais. Os sinais unitários são representados por sub-unidades do parâmetro `hold` de forma simultânea. Já os sinais sequenciais, são divididos em segmentos (`segment`) e cada segmento descreve simultaneamente suas sub-unidades fonéticas. Um exemplo é apresentado na Figura 3.25.

Sequencialidade

Definição 12. Um modelo computacional M possibilita a sequencialidade na representação de um sinal S, se M consegue especificar a sequência correta dos segmentos de

S quando ele possui mais de uma sub-unidade de *handshape*, ou *orientation* ou *location*.

Para representar a sequencialidade de sinais com movimento (i.e. que pode alterar a CM, a OR ou a LOC do sinal), o CORE-SL definiu um nó interior denominado `segment`. Assim, um sinal sequencial pode ser representado no CORE-SL por meio de uma série de segmentos ordenados.

A Figura 3.25 apresenta um exemplo do um sinal ACABAR da Libras, que possui tanto a característica de simultaneidade quanto de sequencialidade. A Tabela 3.6 mostra uma abstração da descrição do sinal ACABAR por meio do CORE-SL para ilustrar a sequencialidade e a simultaneidade.

Figura 3.25: A) Sinal ACABAR da Libras - Simultâneo e Sequencial
Fonte: Felipe (2002) [45]

O sinal ACABAR é dividido em três segmentos: 1) é especificada a suspensão (`hold` - CM, a LOC e OR) para as duas mãos e o movimento que a mão dominante executa; 2) é especificada uma nova suspensão (CM, a LOC e a OR), como estado final do movimento anterior, mas marcando o início do próximo segmento, e o movimento simétrico das mãos; 3) o estado de suspensão final do sinal.

No sinal ACABAR, percebe-se a simultaneidade dos parâmetros de suspensão e de movimento (e seus nós interiores), bem como a sequencialidade marcada por três segmentos realizados em uma sequência específica.

Corretude

Definição 13. Um modelo M possibilita uma representação computacional correta de cada sinal S, se a derivação de S é feita respeitando as regras do formalismo de M.

A corretude nesta tese é relacionada com a qualidade sintática, ou seja, todo sinal representado por meio do CORE-SL deve estar validado sintaticamente a partir das regras de produção do modelo.

Considere um certo sinal S como a entrada em um sistema X da Arquitetura HCI-SL que utiliza o CORE-SL. Se S não apresenta uma descrição correta, X deve ser capaz de analisar sintaticamente S e identificar que S está incorreto em relação à gramática do CORE-SL. Logo, a implementação deste *parser* é fundamental para garantir a corretude

Tabela 3.6: Exemplo no CORE-SL para o sinal ACABAR da Libras

```
[{  "core_sl": {
        "phonetic_component": {
          "identifier": "acabar",
          "sign": {
            "type": "sequential",
            "segment[1]": {
              "hold":{
                "manual" : {
                  "type": "two hands",
                  "dominant_hand": {...},
                  "non_dominant_hand":{...}
                }
              },
              "movement": {...}
            },
            "segment[2]": {...},
            "hold":{...}
          }
        }
    }
}]
```

Fonte: O autor (2015)

de cada representação para alimentar as bases de sinais e, assim, permitir a execução correta dos diversos serviços e aplicações na Arquitetura HCI-SL.

A implementação deste *parser* pode também retornar informações adicionais após a análise sintática, tais como: a identificação do fragmento incorreto no sinal de entrada (assim como compiladores para as linguagens de programação) e as sugestões para a correção de erros.

Adicionalmente, este *parser* pode funcionar como uma API e possibilitar que outras implementações que requerem a entrada e a validação de sinais no CORE-SL possam reutilizar as suas funções. Um processo conceitual do funcionamento deste parser é apresentado na Figura 3.26.

No processo de *parsing*, Figura 3.26, pode-se notar a característica de reuso, uma vez que cada sistema da Arquitetura HCI-SL que necessite do *parser* pode utilizar sua API. A API recebe como entrada um ou mais sinais descritos no CORE-SL, então envia para o processo de validação.

Primeiramente, o sistema deve processar a entrada em *tokens* e verificar se eles existem no vocabulário do formalismo (símbolos terminais e não terminais). Caso esta validação falhe, gera-se uma saída e esta resposta é retornada por meio da API para o sistema.

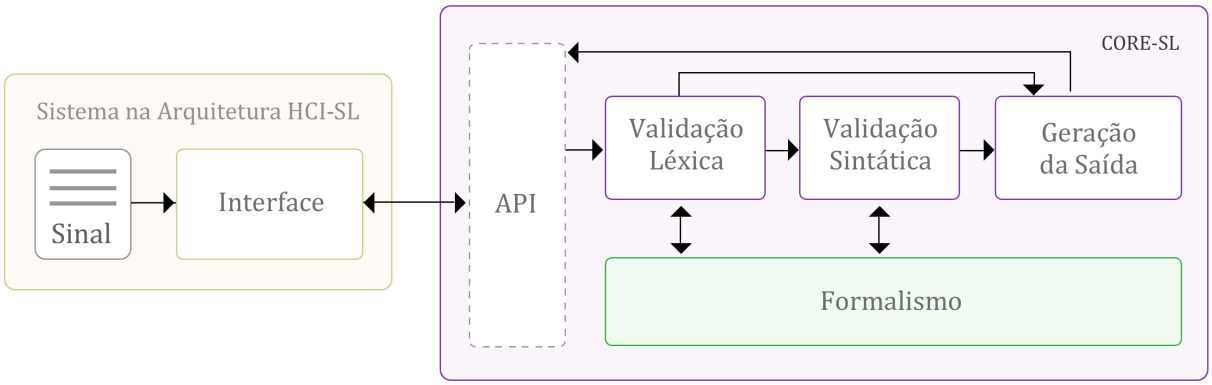


Figura 3.26: Processo conceitual de funcionamento de um *parser* para o CORE-SL
Fonte: O autor (2015)

Se a validação léxica estiver correta, a sequência de *tokens* é analisada sintaticamente por meio da construção de uma árvore de derivação. Se esta construção falhar, o *parser* gera uma resposta e retorna ao sistema principal. Esta resposta pode ser um arquivo no formato texto, JSON ou XML que contém as informações de validação.

A título de exemplo de um sistema, podemos citar uma interface para a descrição ou a busca de sinais no CORE-SL para um vocabulário de uma LS, por meio da manipulação de um Avatar 3D. Neste caso, as possibilidades de manipulação no Avatar 3D devem ser validadas e/ou limitadas por meio das regras formais.

3.6 Nível Interno - Físico

Esta seção, que descreve o Nível Interno, tem o papel de discutir os detalhes técnicos relacionados ao armazenamento físico dos sinais representados pelo CORE-SL. Algumas questões específicas do nível interno incluem a indexação, o desempenho, a segurança e a escalabilidade.

As técnicas utilizadas no nível interno podem impactar diretamente no desenvolvimento das estratégias de indexação e de busca de sinais. Neste caso, devem se utilizadas técnicas computacionais eficientes e que possibilitem flexibilidade para o desenvolvimento de aplicações.

3.6.1 Armazenamento

Embora o nível conceitual do CORE-SL apresente uma estrutura com parâmetros, valores e relacionamentos entre os conceitos, um armazenamento na forma de um banco de dados relacional pode não ser indicado.

Primeiramente, seria necessária a implementação de diversas entidades e relacionamentos, quebrando uma *string* (formato do sinal no CORE-SL) em atributos e valores

para o modelo de banco de dados. Essa operação resultaria em uma etapa de *parsing* adicional a cada nova inserção no banco de dados.

Outra razão, é que o sistema de banco de dados implementa suas próprias regras e algoritmos para as operações sobre os dados, bem como estruturas para a indexação (e.g. chaves primárias, chaves estrangeiras e índices).

Como a representação e a busca de sinais consiste de um problema computacional bem específico (busca por similaridade), uma vez que a entrada de um termo para a busca não é exata, o banco de dados relacional pode limitar o desenvolvimento de estratégias eficientes para solucionar o problema em questão.

Por exemplo, seria necessário adaptar os mecanismos internos do sistema de banco de dados relacional para incorporar uma métrica para o cálculo de distância, utilizada para computar a similaridade entre os sinais.

A partir das discussões apresentadas nos níveis conceitual e formal, nota-se que cada sinal representado pelo CORE-SL consiste, em sua forma original, de uma *string* com a concatenação de *tokens* que estão formalizados no vocabulário do modelo.

Logo, isto indica que cada sinal necessita de poucas informações para o seu armazenamento. Neste sentido, pode-se considerar armazenar cada sinal representado pelo CORE-SL, por exemplo, em um arquivo de texto. Esta é uma estrutura simples que possibilita a implementação e a utilização de algoritmos eficientes para a indexação, o controle de histórico e a escalabilidade.

Além dos banco de dados relacionais, existem outros tipos de bancos dados que utilizam abordagens diferentes em relação aos dados. Por exemplo, existem banco de dados baseado em memória, em chave-valor, em documento, em grafos e em coluna.

Um sistema de banco de dados orientado a documento (BDOD) possui suas operações de armazenamento e de recuperação centradas em dados semi-estruturados na forma de um documento. Ou seja, ao invés de particionar os dados em entidades e em relacionamentos, o BDOD entende que todo o dado é um documento.

Assim, o BDOD possui uma estrutura interna específica para a ordenação dos documentos e a extração de metadados que possibilitam um maior nível de otimização para as operações sobre os dados.

Adicionalmente, algumas vantagens do BDOD consistem da facilidade de escalabilidade horizontal (distribuída), da indexação automática do conteúdo dos documentos (e.g. pode-se adicionar categorias e utilizar o motor de busca do BDOD para retornar documentos baseados neste conteúdo) e dos tipos de dados (e.g. coleções, hierarquia de documentos, etc).

Ao invés de utilizar um sistema de arquivos bruto, pode-se considerar um BDOD para o armazenamento de cada sinal do CORE-SL. Neste caso, o BDOD funcionaria da mesma maneira que um arquivo bruto, permitindo a flexibilidade no formato do documento de armazenamento e na indexação.

Uma grande vantagem, em relação à implementação, é que o BDOD fornece uma API completa para as operações básicas de armazenamento e de leitura, além de resolver questões específicas como a segurança de acesso aos dados. Exemplos de bancos de dados orientado a documento são: Apache CouchDB[6], MongoDB[7] e OrientDB[8].

Para o armazenamento dos sinais, o CORE-SL considera como base a seguinte estrutura do documento[9]

- ***id***: um identificar único do arquivo;
- ***name***: o nome do sinal em forma de texto;
- ***sign_string***: corresponde à *string* de um sinal representado pelo CORE-SL;
- ***sign_json***: armazena um sinal já computado em um formato de saída (e.g. JSON), com o intuito de facilitar a operação de leitura externa (consumo da API);
- ***tags***: que pode armazenar meta-informações para auxiliar a busca por esses dados (e.g. categorias de sinais);
- ***version***: que consiste de uma estrutura para controle de versão (hitórico) dos sinais;

Armazenabilidade

Definição 14. Um conjunto C com uma quantidade Q de sinais descritos pelo CORE-SL é armazenável, se o produto entre Q e o tamanho máximo T, que um único sinal pode ocupar fisicamente, estiver dentro de um limite possível de armazenamento físico.

Para exemplificar esta propriedade, podemos considerar para o conjunto C um vocabulário de sinais da Libras com cerca de 10 mil sinais[10].

Devido ao fato de o conjunto C não estar completamente representado pelo CORE-SL, não é possível determinar o tamanho máximo T que um sinal ocupa. Ou seja, T é determinado pela maior *string* gerada pelo CORE-SL para um sinal de C.

Para fins de exemplificação, escolheu-se um sinal do sub-conjunto [4] utilizado pelo autor no desenvolvimento desta tese, como um pior caso hipotético.

O sinal AFOGAR da Libras (Figura 3.27 [45]) foi considerado um bom exemplo para calcular a capacidade de armazenamento, visto que é um sinal sequencial, executado com as duas mãos e que utiliza expressões não-manuais (passando por praticamente todas as

[6]http://couchdb.apache.org
[7]http://www.mongodb.org
[8]http://orientdb.com
[9]Na implementação, podem ser incluídos mais atributos caso sejam necessários.
[10]O Dicionário da Língua Brasileira de Sinais (Felipe, 2002)[45] conta com um conjunto de 8 mil sinais. No Dicionário Deit-Libras (2009) [18] [17] são representados aproximadamente 10 mil sinais.

características principais representadas pelo CORE-SL).O sinal foi representado em JSON (Apêndice E) e ocupa 12 kilobytes de memória física.

Considerando $T = 12Kb$ e $Q = 10000$, o produto entre eles resulta em 120.000 kilobytes, que corresponde à 120 megabytes. Este tamanho é completamente viável para o armazenamento físico, um armazenamento pequeno quando considerado um hardware de computadores comuns.

Figura 3.27: Sinal AFOGAR da Libras
Fonte: Modificado pelo autor (2015) [45]

3.6.2 Indexação

No contexto desta tese, a indexação consiste de uma organização ou agrupamento dos dados em uma estrutura que possibilite operações de busca de maneira fácil e rápida.

Basicamente o conceito de indexação pode ser dividido em dois tópicos: estratégia conceitual e a implementação.

Uma estratégia conceitual consiste de uma organização teórica dos dados (e.g. clusters), enquanto a implementação resolve a estratégia conceitual por meio de estruturas de dados (e.g. listas, árvores, matrizes, etc.).

O nível físico relaciona as questões de implementação da estratégia de indexação, e tem um papel fundamental no desempenho para as operações de leitura e escrita.

O CORE-SL propõe uma estrutura conceitual baseada em clusters, tal como discutido no Capítulo 5. Para a implementação de uma indexação por clusters podem ser utilizadas diversas estruturas de dados ou mesmo um BDOD.

Por exemplo, cada cluster poderia ser definido por um "documento"no BDOD, que possui como conteúdo uma identificação do cluster e da sua camada (para um cluster hierárquico) e uma coleção dos sinais que fazem parte do cluster em questão, onde cada sinal aponta para o cluster da próxima camada que o contém (Tabela 3.7).

3.6.3 Escalabilidade

A medida que o conjunto de dados e/ou o acesso aos dados crescem, mais *hardware* é requerido para possibilitar a inserção de novos dados, bem como manter um nível de eficiência na recuperação da informação. Existem dois tipos de escalabilidade: vertical e horizontal.

Tabela 3.7: Exemplo de organização de clusters baseado em documentos

```
[{
    "cluster": "1",
    "layer": "A",
    "items": [
      {
        "sign": "_id150",
        "cluster": "3",
        "layer": "B"
      },
      {
        "sign": "_id1140",
        "cluster": "7",
        "layer": "B"
      }
    ]
}]
```

Fonte: O autor (2015)

A vertical consiste em aumentar o *hardware* do computador que está armazenando o conjunto dos dados, em nosso caso, os sinais. Já o horizontal, consiste em distribuir e/ou particionar os dados em diversos computadores (que podem ter uma capacidade de *hardware* menor). Isto permite manter o desempenho do sistema, pois se há a necessidade de aumentar a capacidade, basta adicionar um novo nodo à estrutura.

Além da escalabilidade horizontal, uma estratégia de armazenamento temporário em memória (*cache*) pode ser utilizada para aumentar o desempenho na leitura dos dados.

Por exemplo, no CORE-SL poderia ser utilizado um sistema de *cache* para armazenamento dos clusters e dos conjuntos de sinais, visto que este armazenamento é possível e considerado pequeno (propriedade de armazenabilidade).

Os sistemas que implementam *cache* geralmente trabalham com uma abordagem de chave-valor (*key-value*), armazenando uma coleção de pares chave-valor em uma estrutura de dados na forma de um dicionário na memória temporária (RAM).

Devido à alta velocidade de acesso à memória temporária, os sistemas de *cache* são muito utilizados como uma alternativa para melhorar o desempenho no acesso a grandes conjuntos de dados. Alguns sistemas populares são o Redis[11] e o Memcached[12].

[11] http://redis.io
[12] http://memcached.org

3.6.4 Segurança e Disponibilidade

Para manter a consistência dos dados é importante utilizar uma estratégia de histórico ou controle de versão sobre o conjunto de sinais representamos pelo CORE-SL. Devido às variações na articulação dos sinais entre cada interlocutor, pode ser comum a alteração das descrições até que um determinado padrão seja definido.

Neste caso, torna-se fundamental o uso de um controle de histórico que permita a recuperação de edições anteriores de um sinal. Esta tarefa também depende do tipo de implementação que será utilizada para a arquitetura do CORE-SL.

Por exemplo, se utilizado um sistema de arquivos bruto, podem-se valer de ferramentas de controle de versão, tais como SVN ou GIT. Já para BDOD existem bibliotecas específicas para versionamento de cada dado (documento).

A utilização de um BDOD para a implementação da arquitetura do CORE-SL pode facilitar a resolução de questões como a segurança (por meio de módulos para o controle de acesso e de permissões ao conjunto de dados) e a disponibilidade.

Além da escalabilidade horizontal (particionamento do conjunto de dados), uma técnica que pode ser utilizada é a replicação de dados. Por exemplo, um BDOD pode ter seus dados replicados em diversos servidores diferentes, garantindo uma alta disponibilidade de acesso.

3.7 Conclusões

Este capítulo apresentou os *frameworks* e os processos metodológicos utilizados para a construção do CORE-SL. Ao longo do capítulo foram discutidos os níveis contextual, conceitual, formal e físico do CORE-SL, possibilitando uma visão ampla de sua arquitetura e explorando alternativas para sua implementação.

O próximo capítulo discute sobre os níveis externo e de aplicação, bem como as propriedades externas do CORE-SL que visam possibilitar seu uso nos diversos contextos da Arquitetura HCI-SL.

CAPÍTULO 4

CORE-SL: PROPRIEDADES DO NÍVEL EXTERNO

Após um estudo sobre os níveis contextual, conceitual, formal e interno utilizados para a construção do CORE-SL, este capítulo apresenta um estudo em relação ao nível externo do modelo computacional proposto.

O nível externo tem o papel de discutir as condições de uso para que o CORE-SL seja passível de aplicação nos contextos computacionais da Arquitetura HCI-SL.

Estas condições correspondem às propriedades que o CORE-SL precisa considerar para possibilitar seu uso no desenvolvimento de ferramentas na arquitetura. Este uso consiste em utilizar a sua estrutura técnica na forma de uma API ou em aplicá-lo como uma abordagem metodológica ou conceitual nos processos de desenvolvimento.

Desta maneira, o nível externo analisou os contextos representados na Arquitetura HCI-SL [56], com o intuito de abstrair e de descrever as propriedades externas do CORE-SL, tais como: reprodução, recuperação, usabilidade, dentre outras.

4.1 Metodologia

Para este estudo, utilizou-se o *framework* apresentado na Figura 4.1.

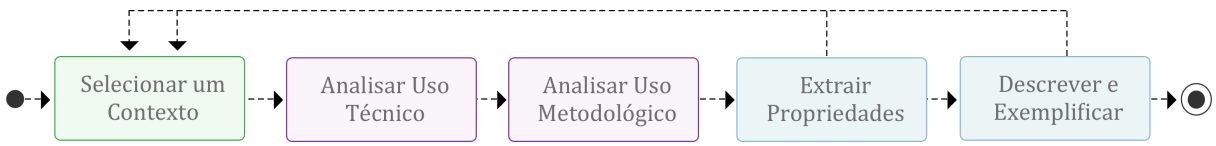


Figura 4.1: Processo Metodológico para o Nível Externo
Fonte: O autor (2015)

O objetivo deste processo metodológico consiste em possibilitar sua aplicação futura de acordo com a demanda de novos contextos de uso na Arquitetura HCI-SL. Este processo tem o intuito de selecionar um contexto na Arquitetura HCI-SL (e.g. ferramenta para o reconhecimento automático de sinais) e analisar a possibilidade da utilização técnica do CORE-SL (e.g. para a busca de sinais), bem como a viabilidade de uso metodológico (e.g. para a construção de uma base de sinais representativa para teste).

Destaca-se que um estudo específico sobre a capacidade de uso metodológico e técnico do CORE-SL é apresentado no Capítulo 6.

A próxima etapa do processo consiste em extrair as propriedades desejadas para possibilitar o uso do CORE-SL no contexto analisado. Em seguida, deve-se descrever e apresentar exemplos (quando possível) para estas propriedades.

Com a aplicação deste processo para explorar os contextos de uso do modelo na Arquitetura HCI-SL, foram abstraídas as seguintes propriedades: universalidade, recuperabilidade, reproduzibilidade, legibilidade, usabilidade, disponibilidade, precisão e relevância.

4.2 Universalidade

Definição 15. Um modelo computacional utilizado para a representação de sinais é universal, se a sua estrutura possibilita a descrição dos sinais de qualquer LS.

A prova desta propriedade mostra-se inviável, uma vez que para esta verificação é necessário garantir que o modelo seja testado em todas as línguas de sinais quanto à descrição dos sinais do seu vocabulário. Ou seja, consiste de um teste exaustivo e inviável.

O CORE-SL foi desenvolvido baseado no contexto das LS, utilizando de um estudo amplo em relação aos principais modelos da Linguística que tratam sobre os componentes que formam os sinais.

Especificamente, a construção do CORE-SL abordou o nível fonético dos modelos linguísticos, pois descrevem a estrutura organizacional das sub-unidades que compõem os sinais de uma LS.

Dentre estas sub-unidades fonéticas estão as configurações de mão e dos dedos, as expressões não-manuais, os pontos de articulação no corpo e no espaço e os movimentos das mãos. A combinação destas sub-unidades de modo simultâneo e/ou sequencial produzem cada um dos sinais de uma LS.

Hipótese 1. Percebe-se que as sub-unidades fonéticas representadas pelos modelos linguísticos utilizam os componentes gestuais-visuais intrínsecos à biomecânica do movimento [63] e ao sistema muscular humano. Neste contexto, a produção dos sinais das LS é limitada pelas possibilidades anatômicas inerentes ao sistema músculo-esquelético [73].

Hipótese 2. Como o CORE-SL foi desenvolvido com base nos modelos fonéticos, ou seja, utiliza as sub-unidades fonéticas como parâmetros no modelo formal, entende-se que o CORE-SL cobre as possibilidades fonéticas para a produção dos sinais.

Hipótese 3. Com base na argumentação das hipóteses 1 e 2, considere um universo U de LS que compartilham o mesmo conjunto de traços fonéticos para a produção de sinais. Conjectura-se que o CORE-SL é passível de aplicação universal para a representação de sinais de qualquer LS de U.

Exemplo 1. A título de ilustração desta capacidade de representação computacional universal do CORE-SL, considerando as hipóteses 1, 2 e 3, apresenta-se a descrição do sinal LIKE[1] da Língua de Sinais da Nova Zelândia (NZSL) (Tabela 4.1) e do sinal CLOCK[2]

[1]Imagens do Sinal LIKE retirado de http://nzsl.vuw.ac.nz/signs/1447
[2]Imagens do Sinal CLOCK retirado de http://www.spreadthesign.com/gb/8131

da Língua de Sinais Russa (RSL) (Tabela 4.2). O CORE-SL conseguiu descrever completamente os dois sinais. As descrições completas dos dois sinais podem ser vistas no Apêndice F e G.

Tabela 4.1: Exemplo de representação do sinal LIKE da NZSL

```
... "hold": {
          "manual": {
           "type": "one hand",
           "dominant_hand": {
             "handshape": {...},
             "orientation": {...},
             "location": {
                "body":"chest"
             }
           }
         }
       },
       "movement": {
         "simple":{
           "local":{
             "frequency": "3",
             "speed":"normal",
             "tension":"touch",
             "mov_wrist":{
               "wrist_twist": "twist"
             }
           }
         }
       } ...
```

Fonte: O autor (2015)

O sinal LIKE (Tabela 4.1) consiste de um sinal simples, executado com apenas uma mão. A mão dominante articula o sinal no peito (localização) com um movimento local de flexão (dobradura) do pulso. Este movimento é repetido por três vezes.

Já o sinal CLOCK (Tabela 4.2) consiste de um sinal que utiliza as duas mãos para a articulação. A mão dominante trabalha como mão ativa, enquanto a mão não-dominante serve como ponto de articulação para o sinal (no pulso). A mão dominante então executa

um movimento local com o antebraço de balanceamento, retirando a mão do pulso e realizando um toque em seguida, por duas vezes.

Tabela 4.2: Exemplo de representação do sinal CLOCK da RSL

```
... "dominant_hand": {
        "handshape": {...},
        "orientation": {...},
        "location": {
           "position": "dominant hand",
           "hand": "wrist"
        }
    },
    "non_dominant_hand": {
        "handshape": {...},
        "orientation": {...},
        "location": {
           "position": "dominant hand",
           "space":{
              "laterality": "parallel to midline",
              "height": "chest",
              "depth": "medial"
           }
        }
    }
    ...
```

Fonte: O autor (2015)

A discussão sobre esta questão da universalidade de um modelo computacional é fundamental. Se o modelo proposto pode ser aplicado em qualquer LS, os artefatos computacionais desenvolvidos para solucionar sub-problemas específicos relacionados à HCI-SL podem funcionar em mais de uma LS.

Neste sentido, a contribuição para o desenvolvimento das ferramentas computacionais que são a base para o funcionamento da Arquitetura HCI-SL pode ser otimizado de maneira colaborativa e pelo reuso dos recursos construídos com base no CORE-SL.

Exemplo 2. Suponhamos que um sistema de reconhecimento automático está desenvolvido e utiliza como abordagem um reconhecimento baseado em sub-unidades. Se este sistema reconhece os parâmetros e os valores do CORE-SL, que tendem a ser universais, então este sistema pode ser ajustado para reconhecer sinais de qualquer LS.

4.3 Recuperabilidade

Definição 16. Dado um conjunto de sinais C representados por um modelo computacional M, a recuperabilidade consiste na capacidade de M possibilitar que um sinal de C seja recuperável por meio de um mecanismo de busca.

Como discutido no Capítulo 3, o nível físico do CORE-SL é responsável por armazenar as descrições em um formato baseado em documentos e por utilizar algoritmos e estruturas de dados eficientes para as operações computacionais básicas. Este armazenamento desempenha um papel fundamental na capacidade de busca.

A busca de um termo (T) pode ser classificada em dois tipos: exata e não-exata. A Figura 4.2 ilustra a diferença entre as buscas exata e não-exata.

Dado T com a entrada, a busca exata consiste em encontrar exatamente T em um banco de dados. Geralmente, a busca exata funciona fazendo a recuperação do índice de T no banco de dados e retornando o conteúdo de T ao cliente. Esta estratégia funciona bem para palavras do Português, por exemplo, a busca do significado da palavra "recuperação".

Na busca não-exata, T não consiste de um índice exato para a recuperação do conteúdo em um banco de dados. Neste caso, o mecanismo de busca deve encontrar os objetos similares à T em um banco de dados. Esta busca por similaridade é feita pela computação de uma função de distância que calcula o grau de semelhança entre T e cada um dos objetos da base. Ao final, o mecanismo de busca retorna uma lista com os termos candidatos.

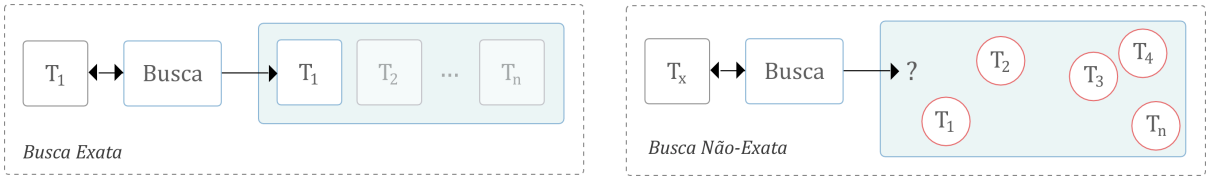


Figura 4.2: Diferença entre a busca exata e a busca não-exata
Fonte: O autor (2015)

No contexto desta tese, um termo T é definido como um sinal de uma LS representado por meio do CORE-SL em seu nível interno.

A busca exata, no contexto da Arquitetura HCI-SL, será utilizada para aplicações bem específicas. Os cenários de uso se limitam a correspondência direta de um sinal por meio de seu índice. Por exemplo, para mostrar o símbolo do SignWriting de um sinal X, um sistema deve simplesmente recuperar X por meio do seu índice já conhecido.

Para este uso, o CORE-SL permite a recuperação exata de um termo por meio do seu índice. Esta é uma tarefa trivial para um sistema de armazenamento, tal como um sistema de banco de dados.

Já a busca não-exata exerce um papel fundamental na Arquitetura HCI-SL e um estudo específico sobre ela é apresentado no Capítulo 5. Considere o exemplo 3 para a ilustração da importância da busca não-exata na arquitetura.

Exemplo 3. Considere um sistema para o reconhecimento automático de sinais via câmera de vídeo, para possibilitar uma entrada de sinais em um mecanismo de busca por meio da língua natural. Este sistema captura o vídeo de um sinal de entrada, processa e gera uma representação no CORE-SL. Como os algoritmos utilizados para o reconhecimento não garantem 100% de precisão, a representação gerada não é precisa. Logo, para realizar a busca na base, deve-se utilizar o conceito de similaridade.

Além disso, podemos citar as variações possíveis que um usuário pode fazer ao descrever um sinal para a busca. Isto também causa ruídos para determinar corretamente o termo de busca.

4.4 Reproduzibilidade

Definição 17. Um modelo computacional (M) para a representação (R) de sinais é reproduzível se possibilita a reprodução de sinais de maneira bidirecional e inteligível: **A)** M deve conseguir sintetizar um sinal a partir de R e este sinal deve ser bem compreendido e **B**) deve ser possível representar um sinal por meio de M e esta descrição ser correta.

A prova desta propriedade para o caso B pode ser obtida por meio de testes exaustivos em um sistema de reconhecimento automático de sinais, que pode processar sinais em vídeo, que estão representados pelo CORE-SL, e gerar uma descrição por meio do modelo.

Assim, a capacidade de reprodução (reconhecimento) pode ser verificada por meio da similaridade entre a descrição gerada e a original na base de dados.

No mesmo sentido, a capacidade de reprodução para a síntese (caso A) pode ser verificada com testes em um Avatar 3D que gera sinais por meio das representações do CORE-SL que estão na base de dados.

Os sinais produzidos podem ser comparados com os vídeos originais da base de dados quanto à similaridade. Se este sinal gerado for entendido pelo usuário final, então o modelo é reproduzível em relação à síntese.

A viabilidade destes testes depende dos artefatos computacionais correspondentes (Sistema de Reconhecimento e Avatar 3D) para possibilitar a avaliação do CORE-SL em condições reais de uso na Arquitetura HCI-SL.

Uma maneira de simular estes testes pode ser através de usuários que conheçam o CORE-SL (podem ser treinados). Neste caso, pode-se aplicar o mesmo framework macro

especificado para o teste de reproduzibilidade, formalizado na Figura 4.3.

O *framework* funciona com a escolha de um sub-conjunto de sinais representados por meio do CORE-SL e a escolha de qual teste deve ser realizado: reconhecimento ou síntese.

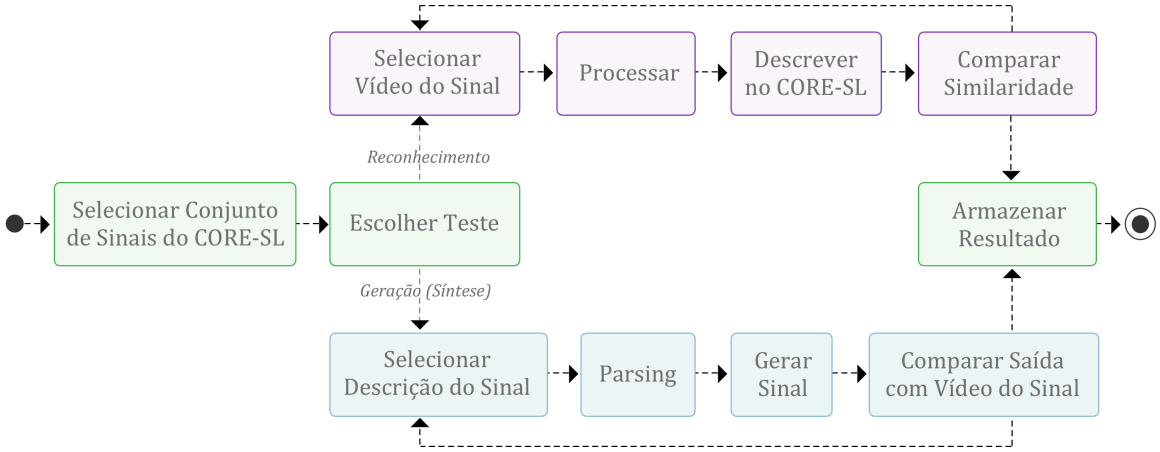


Figura 4.3: *Framework* para os testes de reprodução no CORE-SL
Fonte: O autor (2015)

Para o teste de reconhecimento, um vídeo do sinal é selecionado e processado pelo sistema (considerando que o sistema já está treinado). Em seguida, os parâmetros extraídos no processamento são transformados em uma descrição no CORE-SL.

Então, esta descrição deve ser comparada à descrição original do conjunto de sinais quanto à similaridade. Os resultados são armazenados para a verificação posterior.

Se o teste escolhido for síntese, uma descrição de um sinal deve ser selecionada no conjunto. Esta descrição é processada em um *parser* que extrai os parâmetros da descrição e gera um sinal baseado nestes dados.

Este sinal gerado (por um humano ou pelo Avatar 3D) deve ser comparado com o vídeo original do conjunto de entrada. Se houver a correspondência entre os sinais, a síntese possui reproduzibilidade.

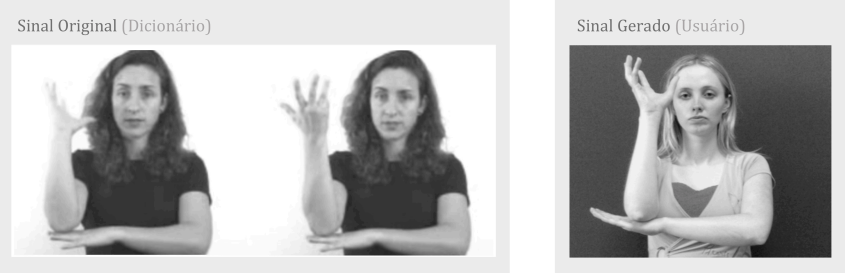
Exemplo 4. Para o caso do reconhecimento podemos tomar como base o sinal ÁRVORE da Libras. Com a aplicação do *framework*, um usuário, que aprendeu previamente o modelo de descrição, realizou a leitura e a interpretação da representação do sinal. Em seguida, este usuário articulou o sinal da maneira interpretada por ele.

Como apresentado na Tabela 4.3, o usuário conseguiu uma similaridade razoável em relação ao sinal original documentado em vídeo [45]. Notou-se apenas uma pequena variação em ambas as configurações de mão e uma variação na localização do cotovelo na mão (no sinal original a localização é nos dedos) que não comprometeram o entendimento do sinal.

Ressalta-se que este é apenas um exemplo de como o *framework* pode ser utilizado para simular o teste de reprodução. Um conjunto de testes em condições reais de uso é

fundamental para avaliar o modelo e seu uso na arquitetura. A descrição completa do sinal ÁRVORE encontra-se no Apêndice H.

Tabela 4.3: Exemplo de descrição do sinal ÁRVORE




```
...      "segment[1]": {
             "hold": {
               "manual": {
                 "type": "two hands",
                 "dominant_hand": {
                   "handshape": {...},
                   "orientation": {...},
                   "location": {
                     "position": "dominant hand",
                     "hand": "fingers",
                     "specific_contact": "elbow"
                   }
                 },
                 "non_dominant_hand": {
                   "handshape": {...},
                   "orientation": {...},
                   "location": {...}
                 }
               }
             },
             "movement": {
               "simple":{
                 "local":{
                   "frequency": "1",
                   "direction": "counterclockwise",
                   "mov_forearm": "rotation"
                 }
               }
             }
           },
           "hold": {...}
...
```

Fonte: O autor (2015), Imagem: [45]

4.5 Legibilidade

Definição 18. Um modelo computacional para a representação de sinais é legível se disponibiliza formatos de visualização de fácil leitura para seus usuários.

A legibilidade é um problema que impacta diretamente a utilização de um modelo nos contextos computacionais e de uso exigidos na Arquitetura HCI-SL. Como vimos na revisão de literatura, a maioria dos trabalhos relacionados descrevem seus modelos por meio de códigos numéricos muito específicos que não permitem uma facilidade de leitura.

Além disso, percebeu-se que muitos modelos na literatura não são documentados quanto ao uso e às suas propriedades. Logo, a falta desta documentação pode prejudicar ou não possibilitar a utilização do modelo por terceiros, uma vez que estes não conseguem aprender ou interpretar o modelo.

A legibilidade em relação ao CORE-SL pode ser percebida em diversos níveis do modelo. Primeiramente, o nível conceitual descreve os parâmetros por meio de um mapa conceitual, com o intuito de possibilitar uma facilidade de entendimento das sub-unidades de representação e seus relacionamentos.

Outro fator que possibilita uma facilidade de leitura ao mapa conceitual foi a adaptação da estratégia de termos técnicos apresentada por Hulst (1993) [66]. A estratégia consistiu em utilizar os termos técnicos dos modelos linguísticos que são facilmente compreendidos (e.g. movimento). Quando um termo técnico não possibilitou um fácil entendimento (e.g. radial) foi substituído por um termo mais simples (e.g. thumb side).

O nível formal do CORE-SL utiliza uma meta-linguagem para a formalização do modelo conceitual, que possibilita a legibilidade no contexto computacional. Ou seja, o nível formal agrega uma notação comum para a implementação, por exemplo, de *parsers* e de uma API. Este formalismo, embora voltado para o contexto técnico, também possibilita uma facilidade de leitura de suas regras por usuários humanos.

O nível interno do CORE-SL, embora seja de contexto computacional, também apresenta legibilidade na estrutura sugerida para o armazenamento de cada sinal. Neste nível, discutiu-se sobre o uso de uma estrutura baseada em documento que inclui um formato de chave-valor para a representação do conteúdo dos sinais. Por exemplo, sugeriu-se a chave *name* para representar o nome do sinal no conteúdo de um documento.

A legibilidade exerce um fator fundamental no nível externo, pois tem o objetivo de facilitar o entendimento do CORE-SL pelos usuários (técnicos ou não) que pretendam utilizá-lo em um contexto da Arquitetura HCI-SL. Nesta perspectiva, o CORE-SL deve proporcionar uma legibilidade em duas classes:

API: Deve-se fornecer uma documentação detalhada e legível sobre a utilização e a integração das funções e dos parâmetros do CORE-SL, após a sua implementação, para permitir sua correta aplicação no desenvolvimento de novos sistemas no contexto da Arquitetura HCI-SL. Alguns dados relevantes na documentação incluem:

- Documentar as rotas de chamada remota (i.e. como as funções devem ser chamadas na API pelos serviços que a integram);
- Uma descrição sobre cada recurso (i.e o que cada função retorna quando invocada);

- A lista dos parâmetros requeridos ou opcionais em cada chamada, bem como o seu tipo de dado;

- Descrever os campos e o formato em que os dados de resposta são apresentados. O formato de saída da API impacta diretamente na legibilidade (computacional ou humana) do resultado.

  Desta maneira, deve-se utilizar formatos que sejam legíveis tanto para o usuário quanto para o computador, tais como JSON e o XML.

  Na Tabela 4.4 é apresentado um exemplo da descrição de um sinal retornado pelo CORE-SL: primeiro por meio do Texto Puro e depois por JSON. O formato JSON pode possibilitar uma maior facilidade de leitura.

- Também deve-se fornecer um exemplo para a saída, para facilitar a verificação de resposta na integração da API, ou seja, verificar nos testes de integração se a API está retornando os dados corretamente com base na resposta de exemplo.

Tabela 4.4: Exemplo de Saída em Texto Puro e em JSON, respectivamente

orientationor_palmdown or_fingersright or_forearmtypehorizontalvalue0º dominant-hand side-deor_elbowtypedepthlocal0º (dominant-hand side)locationpositiondominant handspacelateralityparallel to shoulderheightstomachdepthmedial	`"orientation": {`
	`"or_palm":"down",`
	`"or_fingers":"right",`
	`"or_forearm": {`
	`"horizontal": "0° dominant-hand side"`
	`},`
	`"or_elbow":{`
	`"depth": "0° (dominant-hand side)"`
	`}`
	`},`
	`"location": {`
	`"position": "dominant hand",`
	`"space":{`
	`"laterality": "parallel to shoulder",`
	`"height": "stomach",`
	`"depth": "medial"`
	`}`
	`}`

Fonte: O autor (2015)

Documentação: Além da API, a estrutura do CORE-SL também deve ser documentada de maneira adequada.

Esta documentação deve descrever cada um dos parâmetros do modelo e seus respectivos valores, além de uma explanação sobre o conceito tratado (por exemplo, explicar que o parâmetro `movement` representa os movimentos das mãos com trajetória no espaço de articulação).

A documentação do CORE-SL também está relacionada à propriedade de usabilidade, a qual é discutida na próxima seção.

4.6 Usabilidade

Definição 19. A usabilidade consiste na capacidade de um modelo computacional de representação de sinais proporcionar uma facilidade de aprendizado de seus parâmetros e de sua estrutura para os seus usuários e uma facilidade de uso para sua aplicação no contexto computacional.

Segundo McCleary & Viotti (2007) [84], os sistemas disponíveis na literatura para a representação dos sinais têm apresentado soluções limitadas ou muitas vezes de uso complexo. Um bom sistema de representação deve conseguir um balanceamento da sua complexidade estrutural (alto nível de detalhamento) com a disponibilização de mecanismos que "*possam ilustrar as generalizações de forma transparente para leitores não especialmente treinados* [84, p.2]".

A propriedade de usabilidade do CORE-SL tem o intuito de resolver este problema. Embora o CORE-SL englobe um alto nível de detalhamento, exigido para o contexto computacional da Arquitetura HCI-SL, uma solução para a facilidade de aprendizado e de uso do modelo deve ser explorada.

O desenvolvimento de uma documentação adequada aos usuários do CORE-SL deve incluir uma explicação de cada parâmetro do modelo, uma descrição textual sobre cada conceito, um exemplo de uso e os valores que podem instanciar cada parâmetro.

Como trata-se da documentação das características gestuais-visuais das LS, os valores que instanciam cada parâmetro do CORE-SL não devem ser representados somente de maneira textual.

Neste sentido, a documentação do CORE-SL pode utilizar uma estratégia de representação visual (gráfica) para os parâmetros e seus valores. Esta representação pode ser classificada em dois tipos: **estática** e **dinâmica**.

A representação visual estática relaciona os parâmetros do CORE-SL que não requerem movimento, tais como a configuração de mão e os pontos de articulação no corpo. Para esta representação podem-se utilizar:

- **fotografias**: que permitem um alto grau de realismo na documentação. Este realismo pode permitir a representação dos parâmetros gestuais-visuais de maneira mais precisa, um requisito para os parâmetros que agregam muitos detalhes, tal como as expressões faciais. Para exemplificação, pode-se considerar as figuras 4.4, 4.5 e 4.6;

- **avatar 3D**: uma alternativa às fotografias pode ser a utilização de um avatar 3D para a modelagem de cada um dos parâmetros do CORE-SL. Este recurso pode não fornecer um alto grau de realismo, mas pode ser mais flexível na documentação do que as fotografias, uma vez que permite fácil atualização (na hipótese de um parâmetro não estar bem representado);

- **desenho 2D**: adicionalmente, pode-se utilizar um recurso de desenhos vetoriais para representar os traços gestuais-visuais do CORE-SL. Na literatura, podemos citar o Dicionário Enciclopédico Ilustrado Trilingue da Língua de Sinais Brasileira (Libras) [18] [17], que utiliza desenhos vetoriais para a documentação de sinais. As figuras 4.7 e 4.8 ilustram esta representação para o CORE-SL.

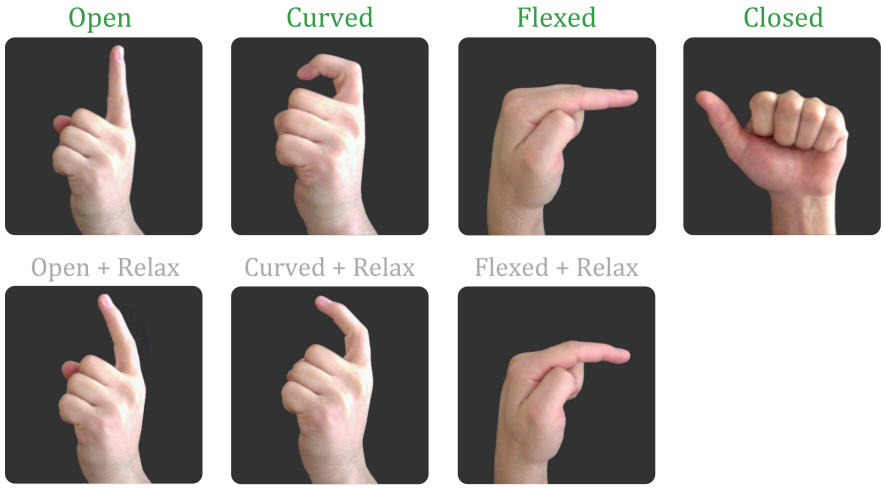


Figura 4.4: Exemplo de documentação da flexão dos dedos
Fonte: O autor (2015)

A Figura 4.4 apresenta a representação visual dos valores correspondentes ao parâmetro `finger_flexion` do CORE-SL. Este parâmetro descreve a disposição (flexão) dos dedos indicador, médio, anelar e mínimo. Para exemplificação, representamos somente o dedo indicador, uma vez que os demais dedos apresentam a mesma configuração.

Como mostrado na Figura 4.4, os dedos podem estar abertos (*opened*), fechados (*closed*), curvados (*curved*) e flexionados (*flexed*).

Abaixo de cada valor é apresentada a sua configuração na hipótese de o atributo `relax` (relaxamento dos músculos) estiver descrito como verdadeiro. Por exemplo, pode-se ver o dedo aberto (*open*) com relaxamento dos músculos (*open + relax*).

Um exemplo da documentação visual da configuração do dedo polegar (*thumb*) é apresentado por meio da Figura 4.5.

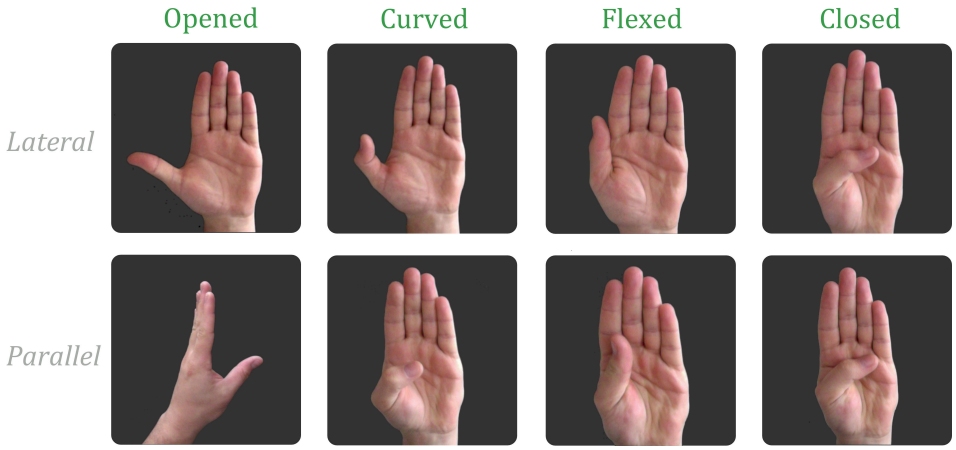


Figura 4.5: Exemplo de documentação para a rotação e disposição do polegar
Fonte: O autor (2015)

A configuração do dedo polegar é feita por meio da rotação (`thumb_rotation`) que pode ser lateral ou paralela (*parallel*) e pela disposição do polegar em cada rotação (Figura 4.5): aberto (*opened*), fechado (*closed*), curvado (*curved*) e flexionado (*flexed*).

Esta representação visual por meio de fotografias pode proporcionar ao usuário do CORE-SL um claro entendimento da definição de cada parâmetro e de seus valores.

Com este nível de realismo, é possível até representar os mínimos detalhes das sub-unidades que formam certos parâmetros no CORE-SL, tal como a relação de contato (`specific_contact`) do polegar com os demais dedos.

Na Figura 4.6, podemos ver as quatro maneiras de contato do polegar com um outro dedo: pelas pontas (*fingertip*), pelas digitais (*digital*), unha do polegar na digital do outro dedo (*nail in digital*) e a digital do outro dedo na unha do polegal (*digital in nail*).

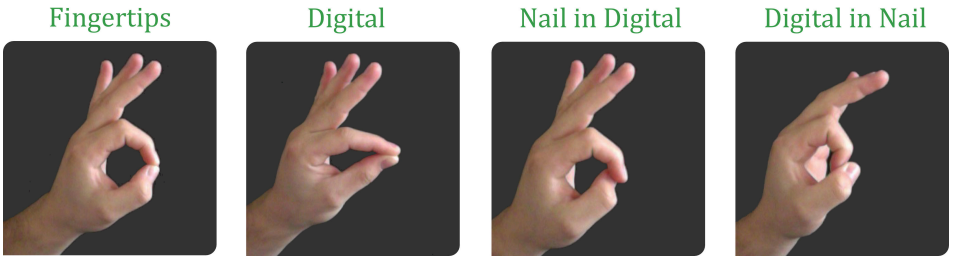


Figura 4.6: Exemplo de documentação para os tipos de contato do polegar
Fonte: O autor (2015)

Em relação ao uso de desenhos vetoriais, eles podem ser bem utilizados dependendo da forma como são desenvolvidos. Ou seja, para representar corretamente os conceitos, estes desenhos vetoriais devem manter ao máximo as proporções de um corpo humano real e incluir detalhes para permitir certo grau de realismo.

Além disso, os desenhos vetoriais podem ser muito úteis para a representação de parâmetros gestuais em relação ao espaço de sinalização, tais como movimentos de trajetória e pontos de articulação no espaço. Por exemplo, a Figura 4.7 ilustra como o CORE-SL descreve a localização no espaço de sinalização, mapeando as coordenadas X, Y e Z, respectivamente, para os parâmetros de lateralidade (*lateraly*), de altura (*height*) e de profundidade (*depth*).

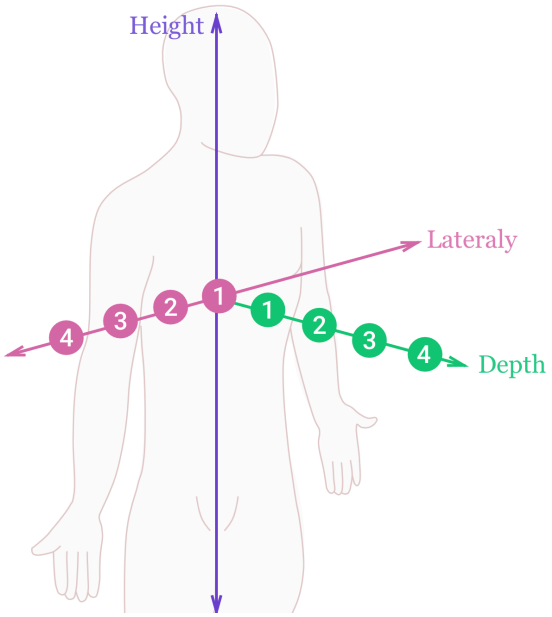


Figura 4.7: Exemplo de documentação para a localização das mãos no espaço
Fonte: O autor (2015)

Para cada um dos parâmetros, ao invés de utilizar representações numéricas como coordenadas, o CORE-SL utiliza valores textuais relacionados às localizações no corpo e à flexão do braço. Por exemplo, para a lateralidade temos quatro opções:

1. paralelo à linha medial (*parallel to midline*): a mão é localizada no espaço referente à linha medial (central) do corpo do interlocutor;

2. paralelo ao peito (*parallel to chest*): a mão é posicionada no espaço à frete do peito;

3. paralelo ao ombro (*parallel to shoulder*): a mão é localizada no espaço paralelo à linha do ombro;

4. lateral no espaço (*distal in space sideways*): a mão é localizada no espaço em relação à lateral do corpo.

De forma análoga, o parâmetro de profundidade (*depth*) das mãos no espaço pode se configurado por quatro valores:

1. **proximal**: ponto próximo ao corpo;
2. **medial** (*middle*): distância média, flexionando levemente o braço para frente;
3. **distal**: ponto longe do corpo, com o braço quase estendido;
4. **flexionado** (*flexed*): as mãos são posicionadas na distância máxima possibilitada pela flexão do braço.

A Figura 4.8 apresenta um exemplo de desenho vetorial para a documentação de alguns pontos de articulação na cabeça. Pode-se notar que, embora a representação seja por um desenho vetorial, é possível entender facilmente os pontos na cabeça onde os sinais são articulados.

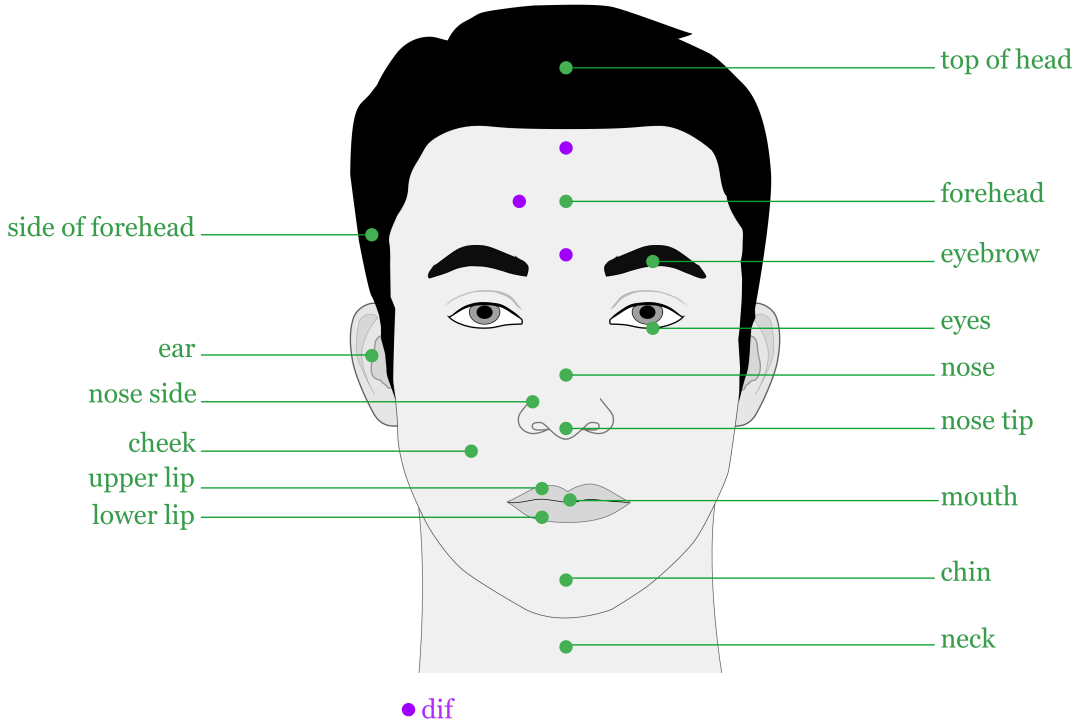


Figura 4.8: Exemplo de documentação para os pontos de articulação na cabeça
Fonte: O autor (2015)

Nos pontos de articulação da cabeça, Figura 4.8, também é ilustrado o atributo `dif`, que consiste em quatro variações possíveis de um ponto de articulação específico. Como exemplo, a localização na testa (*forehead*) pode ter uma pequena variação para cima, ou para à direita, ou para à esquerda ou para baixo.

Para a representação visual dinâmica que consiste em documentar movimentos das mãos, dos dedos, do corpo e da cabeça pode-se considerar as seguintes alternativas:

- **vídeos**: a representação de movimentos pode ser ilustrada por um grupo de sinais de exemplo que utilizam cada um dos movimentos descritos pelo CORE-SL. Uma limitação desta alternativa, é que a documentação precisa ser exclusivamente digital;

- **GIF**[3] **animados**: os movimentos podem ser representados por animações gráficas em um formato de imagem dinâmica. Novamente, esta abordagem exige o uso de uma documentação digital;

- **fotografias**: este é um recurso muito utilizado na literatura da LS para a documentação do movimento dos sinais. Pode-se utilizar uma fotografia dos estados (segmentos) do movimento de um determinado sinal, e então utilizar setas para indicar a direção, o contorno e a tensão do movimento.

A Figura 4.9 apresenta um exemplo da representação visual de um movimento local de dedos do tipo tamborilar (*vibration*). O movimento foi dividido em três segmentos e nos primeiros dois estados são utilizadas setas para representar o sentido em que cada dedo deve realizar o movimento.

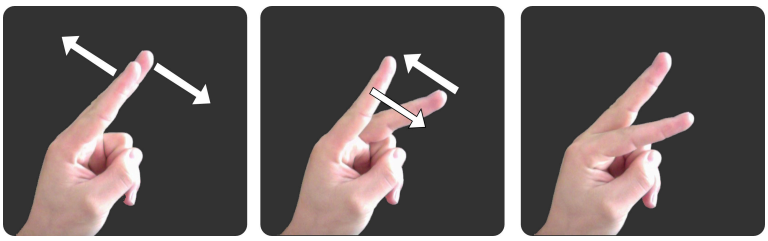


Figura 4.9: Exemplo de documentação um movimento local de dedos
Fonte: O autor (2015)

Podemos notar que a representação visual proporciona uma facilidade de aprendizado e de reprodução do CORE-SL, uma vez que o usuário entende como configurar as mãos, os pontos de articulação, entre outros parâmetros, que em conjunto produzem os sinais.

A partir da usabilidade, o CORE-SL pode ser implementado e interpretado de uma forma mais fácil para os contextos de uso da Arquitetura HCI-SL.

4.7 Acessibilidade

Definição 20. A acessibilidade, no contexto da Arquitetura HCI-SL, refere-se à facilidade de acesso aos dados e às funções de um modelo computacional para a representação de sinais, assim como à facilidade de integração em outros sistemas.

Uma abordagem arquitetural que possibilita a facilidade de acesso e de integração entre sistemas é um padrão de API. Uma API permite definir uma interface única para o acesso remoto aos conjuntos das funções, dos dados e dos objetos que compõem o CORE-SL.

Para exemplificação, o conjunto de funções do CORE-SL inclui as operações básicas (inserção, leitura, atualização e remoção - CRUD[4]), as funções específicas para a busca de sinais e um analisador sintático.

[3]GIF - *Graphics Interchange Format* consiste de um formato de imagem que permite animação.
[4]Operações computacionais básicas CRUD - *Create*, *Read*, *Update* e *Delete*

Esta abordagem baseada em API pode facilitar as questões de integração, pois funciona muito bem em Arquiteturas Orientadas à Serviço (SOA[5]), tal como a Arquitetura HCI-SL.

Neste contexto, cada serviço que necessite do CORE-SL pode realizar chamadas remotas à API por meio de um protocolo de comunicação definido na implementação. Então, a API recebe cada requisição, processa e retorna uma resposta geralmente em um formato de saída, tais como o JSON e o XML.

A seguir, são discutidos alguns pontos específicos que podem impactar diretamente nesta capacidade de acessibilidade no CORE-SL.

4.7.1 Disponibilidade

Da mesma forma que o nível interno, uma API para o CORE-SL deve possuir um alto grau de disponibilidade, ou seja, que a API mantenha o seu funcionamento no maior tempo possível. Esta característica torna-se necessária para que os serviços da Arquitetura HCI-SL, que dependem do CORE-SL, possam manter o seu correto funcionamento.

Uma das maneiras de possibilitar uma alta disponibilidade de uma API é garantir que a estrutura não tenha um único ponto de falha, ou seja, que o funcionamento da API não dependa somente de um computador (servidor).

Para tanto, pode-se valer de uma técnica de escalabilidade horizontal, tal como a redundância. Este padrão, consiste em replicar o serviço da API em mais de um servidor (chamado nodo). Desta maneira, na hipótese de um nodo falhar, as requisições dos clientes da API do CORE-SL podem ser redirecionadas para os nodos ativos.

Assim, uma boa implementação de um padrão de alta disponibilidade deve: a) possibilitar a redundância, b) identificar falhas e c) redirecionar requisições se um nodo falhar.

4.7.2 Desempenho

Além dos algoritmos e das estruturas de dados eficientes que o CORE-SL deve contemplar no nível físico, uma API deve possibilitar um bom desempenho durante a sua utilização pelos serviços da arquitetura.

O desempenho tem relação com a quantidade máxima de requisições que o serviço da API consegue processar e retornar um resultado para cada cliente em um tempo razoável.

Novamente, considere como um exemplo o serviço do CORE-SL operando em um único servidor. Este servidor possui um limite para novas conexões de clientes e para a quantidade de requisições que consegue processar paralelamente. No momento em que este servidor sofre uma sobrecarga, o tempo de acesso é aumentando, além da geração de possíveis erros aos clientes.

[5]SOA - *Service-Oriented Architecture*

Assim como a disponibilidade, uma maneira de minimizar este problema é utilizar uma escalabilidade horizontal do serviço, ou seja, distribuindo a API em mais de um servidor.

Os nodos desta estrutura devem ser gerenciados por um balanceador de carga, que tem o papel de controlar as requisições de entrada e distribuí-las entre os nodos ativos.

Este padrão de escalabilidade permite controlar o limite de carga que cada nodo pode processar. Quando a estrutura esta sobrecarregada, mais nodos podem ser criados dinamicamente, mantendo o desempenho do sistema.

4.7.3 Controle de Acesso

A implementação da API do CORE-SL também deve considerar uma estratégia para a autenticação, com o intuito de manter a segurança dos dados e um controle de acesso ao nível interno do modelo.

Para o processo de autenticação podem ser utilizadas estratégias baseadas em Chave de API (*Token-based Authentication*) que consiste de um *token* criptografado único que permite o acesso direto à API pelo serviço requerente e em autenticação básica (*Basic Authentication*) que consiste em criptografar uma chave de acesso por meio de um nome de usuário e de senha.

O controle de acesso é fundamental para manter a segurança dos dados, bem como para o controle de versão (histórico) das descrições dos sinais.

4.7.4 Documentação

Para os serviços conseguirem utilizar e integrar o CORE-SL corretamente é fundamental fornecer uma boa documentação da API.

Assim como apresentado na propriedade de legibilidade, esta documentação deve disponibilizar o conteúdo de forma organizada (e.g. categorizada por funcionalidades), fornecer explicações detalhadas sobre cada função, descrever os atributos requeridos e opcionais para a composição de cada requisição, estratégias de filtros, os formatos de saída (resposta), dentre outras informações relevantes.

Uma proposta para uma organização arquitetural para a implementação da API do CORE-SL que considere a disponibilidade, o desempenho e o controle de acesso é esquematizado na Figura 4.10. Este exemplo foca apenas na ilustração da escalabilidade horizontal da API, não descrevendo a escalabilidade do nível interno (Discutida no capítulo 3).

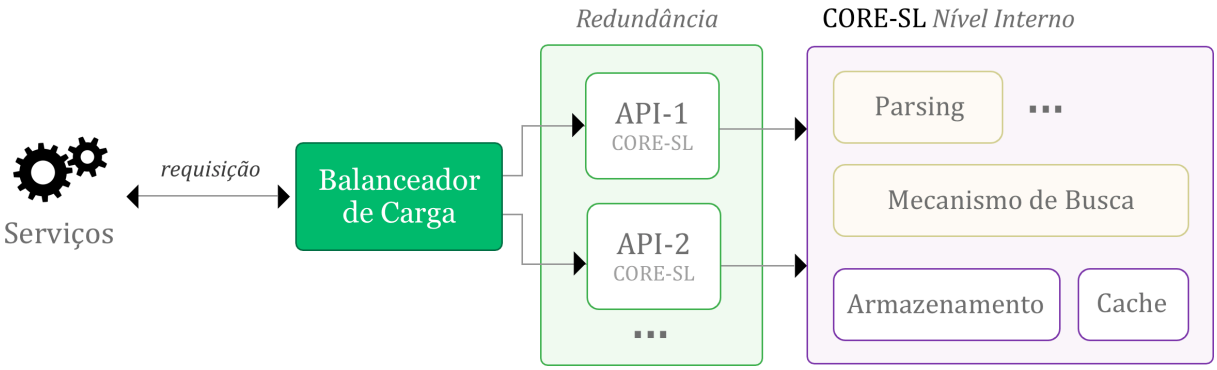


Figura 4.10: Proposta de Arquitetura para a Implementação do CORE-SL
Fonte: O autor (2015)

4.8 Precisão

Definição 21. A propriedade de precisão consiste na capacidade de um modelo computacional representar os sinais de uma LS de maneira precisa. Esta precisão é determinada pelas propriedades de completude, de usabilidade, de armazenabilidade, de recuperabilidade, de acessibilidade e de legibilidade.

Baseando-se na argumentação de cada uma das propriedades e dos exemplos apresentados, conjectura-se que o CORE-SL possui a capacidade de representar os sinais de maneira precisa.

Os exemplos discutidos durante o desenvolvimento desta tese mostram evidências de que o CORE-SL consegue representar diversos sinais com precisão, sejam sinais simples, com movimentos ou expressões faciais complexas, ou muito similares.

Uma representação precisa está relacionada, também, com a propriedade de completude do CORE-SL, que deve possibilitar ao modelo descrever quaisquer sinais e conseguir diferenciá-los.

Como o CORE-SL agrega um alto nível de detalhamento das sub-unidades fonéticas das LS (e.g. faz um detalhamento das juntas em cada dedo, que consiste no grau máximo de granularidade para uma configuração de mão), presumi-se que o modelo consegue representar os sinais com precisão.

Exemplo 5. Considere os sinais ABSORVENTE e CHEQUE da Libras (Figura 4.11). O sinal ABSORVENTE se distingue do sinal CHEQUE somente por possuir um movimento local de flexão de pulso, no final da articulação do sinal. Esta diferença pode ser representada no CORE-SL por meio dos parâmetros: `symmetric_movement` para simetria das mãos e `mov_wrist` para descrição da flexão do pulso.

Exemplo 6. Considere os sinais TERÇA-FEIRA e SOLDADO da Libras (Figura 4.11). O sinal TERÇA-FEIRA se distingue do sinal SOLDADO somente pela configuração da mão dominante, especificamente a disposição dedos.

Em TERÇA-FEIRA os dedos estão espalhados lado-a-lado, enquanto em SOLDADO os dedos estão unidos lateralmente. Esta diferença pode ser representada no CORE-SL por meio dos parâmetros: `finger_contact` que representa o contato entre os dedos e `finger_flexion` para a disposição dos dedos.

Exemplo 7. Considere o sinal METRÔ da Libras (Figura 4.11). Este sinal é articulado com as duas mãos, sem contato entre as mãos e somente a mão dominante realiza o movimento.

A particularidade deste sinal consiste no posicionamento da mão não-dominante acima da mão dominante. O CORE-SL consegue descrever esta relação entre as mãos por meio do parâmetro `relation_non_dominant`.

Figura 4.11: Sinais ABSORVENTE, CHEQUE, TERÇA-FEIRA, SOLDADO e METRÔ
Fonte: Antunes (2011) [4]

4.9 Relevância

Definição 22. A propriedade de relevância consiste da capacidade de um modelo computacional para a representação de sinais possibilitar sua aplicação como uma abordagem metodológica para auxiliar o desenvolvimento de serviços e de aplicações nos contextos da Arquitetura HCI-SL.

A aplicabilidade do CORE-SL como abordagem metodológica ou técnica nos processos de desenvolvimento é uma das maiores contribuições desta tese.

O Capítulo 6 apresenta um estudo específico para apresentar os cenários de aplicação do CORE-SL para subsidiar o desenvolvimento de serviços e de aplicações, exemplificando a relevância do CORE-SL neste contexto.

4.10 Considerações

Este capítulo discutiu sobre as propriedades externas do CORE-SL relacionadas à qualidade e às condições necessárias para que o modelo possa ser utilizado nos contextos computacionais da Arquitetura HCI-SL.

O conjunto de propriedades discutido neste capítulo, tem o intuito de formalizar critérios que ao serem contemplados por um modelo computacional de representação de sinais, assim como o CORE-SL, podem possibilitar a sua utilização como um suporte na resolução de diversos problemas envolvendo o tratamento computacional das LS.

Adicionalmente, as propriedades discutidas no nível externo do CORE-SL podem ser estendidas e formatadas na forma de um *framework* para a avaliação de outros modelos computacionais. Neste sentido, outros modelos podem ser testados e comparados com o CORE-SL, objetivando a complementação e a melhoria dos modelos disponíveis.

A partir do desenvolvimento dos serviços da Arquitetura HCI-SL e da aplicação do CORE-SL, novas condições de uso podem ser exigidas, tornando-se necessário aplicar o processo novamente para estudar novas propriedades, estratégias de implementação e os possíveis impactos na arquitetura.

No próximo capítulo é apresentado um estudo sobre o problema de indexação e de busca de sinais no contexto da Arquitetura HCI-SL, relacionando as propriedades do nível interno e do externo, tais como a acessibilidade, a recuperabilidade e a armazenabilidade.

CAPÍTULO 5

INDEXAÇÃO E BUSCA DE SINAIS

Este capítulo apresenta um estudo em relação ao problema da indexação e da busca de sinais por meio do CORE-SL. Devido a importância destes tópicos na Arquitetura HCI-SL dedicou-se um capítulo específico para esta discussão. O problema foi abordado do ponto de vista computacional e do usuário. Algumas questões abordadas são:

- A apresentação do problema geral da busca de sinais na Arquitetura HCI-SL;
- Proposição de uma estratégia de indexação para os sinais;
- Discussão sobre as métricas de similaridade;
- Um estudo relacionado às estratégias que podem melhorar a experiência do usuário final, que venha a utilizar um mecanismo de busca de sinais em um dos serviços da Arquitetura HCI-SL.

5.1 Problema

O problema da busca de sinais foi introduzido no capítulo 4, por meio da propriedade de **recuperabilidade**. Basicamente, a busca de sinais pode ser definida da seguinte maneira, no contexto da Arquitetura HCI-SL:

Definição 23. Considere $\boldsymbol{S}$ como um termo de pesquisa e $\boldsymbol{X}$ como uma base de sinais, ambos descritos por meio do CORE-SL. A busca de sinais consiste em utilizar estratégias algorítmicas para encontrar o conteúdo de $\boldsymbol{S}$ em $\boldsymbol{X}$ de forma rápida e que minimize o consumo de recursos computacionais. Se encontrar $\boldsymbol{S}$, o mecanismo de busca retorna o conteúdo para o cliente (usuário ou serviço).

Pode-se abordar o problema da busca do ponto de vista computacional (e.g. estratégias para o desenvolvimento de uma busca eficiente) e em relação ao usuário final (e.g. técnicas para possibilitar uma facilidade de uso do mecanismo de busca).

Como vimos no capítulo 4, a busca de sinais pode ser **exata** e **não-exata**. A busca **exata** consiste em encontrar um sinal na base de dados por meio da utilização do índice único de um sinal. Por exemplo, ao definir um termo de pesquisa como "*computação*", o mecanismo de busca deve procurar exatamente este índice no banco de dados e retornar o conteúdo correspondente.

Este tipo de busca é trivial para um sistema de banco de dados e somente pode ser utilizado, no contexto da Arquitetura HCI-SL, quando se conhece previamente o índice de um sinal. Caso contrário, não é possível buscar exatamente um sinal.

Devido à falta de uma ferramenta que possibilite a entrada de um termo de pesquisa (um sinal) em língua natural (e.g. por meio do SignWriting ou de um Sistema de Reconhecimento via Câmera), alguns dicionários na literatura tem utilizado uma técnica de pesquisa por meio de alguns parâmetros fonéticos.

No Dicionario Digital da Língua Brasileira de Sinais o usuário pode selecionar uma configuração de mão para tentar encontrar um sinal. No Dicionário da Língua de Sinais da Nova Zelândia (NZSL)[31] o usuário pode escolher a configuração de mão e um ponto de articulação.

Para exemplificar o funcionamento, considere a intenção de buscar um sinal o qual não conhecemos o índice exato e não conhecemos o significado. Este sinal de entrada é visto na Figura 5.1. Para realizar a busca podemos selecionar no Dicionário da NZSL [31] esta configuração de mão e esta localização no espaço neutro.

Figura 5.1: Exemplo de Termo de Busca para o Dicionário da NZSL
Fonte: NZSL Online Dictionary (2015) [31]

Ao realizar a busca deste sinal (Figura 5.1), além de um alto tempo de processamento, o dicionário retornou 348 sinais como possibilidades em relação aos parâmetros selecionados. O sinal, que consistiu a intenção de busca, foi encontrado na página 37 dos resultados.

Este exemplo mostra a inviabilidade desta estratégia, uma vez que não apresenta resultados totalmente precisos e dificilmente retorna um conjunto pequeno de sinais bem similares à intenção de busca.

Além disso, mesmo esta entrada por parâmetros pode ser não-exata, uma vez que o usuário pode escolher incorretamente os parâmetros, devido à alguma variação no entendimento do sinal.

Por exemplo, o sinal correto representado no dicionário, poderia ser descrito com uma configuração de mão com os dedos espalhados, diferentemente da intenção de busca em que os dedos estão juntos. Neste caso, o sistema não iria encontrar o sinal pesquisado, embora ele seja muito similar ao sinal armazenado na base de dados.

Neste sentido, um mecanismo de busca de sinais eficiente deve retornar uma lista de resultados mais precisa em relação à intenção do usuário, bem como possibilitar uma interação mais natural para a entrada do termo de pesquisa.

No conceito de busca não-exata ou busca por similaridade, o termo de pesquisa é interpretado pelo sistema que utiliza uma função para calcular a similaridade entre o sinal de entrada e os sinais da base de dados. Então, uma lista de sinais candidatos à solução é construída, ordenada pelo grau de semelhança e retornada ao cliente final.

Como exemplo, ao buscar um sinal ***X***, representado pela Figura 5.2.A, espera-se que o sistema calcule e encontre rapidamente uma lista de sinais candidatos similar à ***X***, por exemplo, a lista representada pela Figura 5.2.B, composta pelos sinais APRENDER, LARANJA e SÁBADO da Libras.

Figura 5.2: Exemplo de uma lista de resultados por um mecanismo de similaridade. A) consiste da intenção de busca, e B) consiste da lista de sinais candidatos
Fonte: A) Felipe (2002) [45]; B) Antunes (2011) [4]

Por meio deste mecanismo de similaridade podem ser construídas diversas ferramentas para auxiliar a busca, tais como recursos para a sugestão de correção (*spelling correction*), mecanismos de "auto-completar"para tentar prever o sinal a ser pesquisado e a apresentação de um histórico de pesquisa.

5.1.1 Contexto

Na Arquitetura HCI-SL, a busca por similaridade exerce um papel fundamental nos serviços e nas aplicações que requerem a pequisa de sinais por meio do CORE-SL.

Alguns contextos, já mencionados anteriormente, incluem um sistema de reconhecimento de sinais via câmera, o reconhecimento de caracteres do SignWriting e uma interface baseada em inteligência artificial para auxiliar a descrição dos sinais no CORE-SL.

Estas ferramentas têm o objetivo de interpretar e de processar a entrada, maximizando o conjunto de parâmetros e de valores utilizados para descrever o sinal pesquisado.

Como ilustrado pela Figura 5.2, a precisão no cálculo de similaridade entre o sinal de entrada e os sinais da base de dados depende do grau de detalhamento que o modelo de representação possibilita para os sinais.

Anteriormente discutido pelas propriedades de completude e de unicidade, o CORE-SL agrega um alto nível de detalhamento, um fator que viabiliza a distinção entre os sinais representados e que pode auxiliar no ajuste de precisão em uma função de similaridade.

É importante destacar que a qualidade de um sistema de busca por similaridade, no contexto da Arquitetura HCI-SL, é influenciada também pelas propriedades do CORE-SL dos níveis formal (unicidade e completude) e interno (armazenamento, indexação e a disponibilidade). Neste sentido, as estratégias apresentadas a seguir consideram intrinsecamente estas propriedades.

5.1.2 Cenário de Complexidade

Uma estratégia eficiente para a indexação dos sinais e para o cálculo de similaridade é fundamental para um mecanismo de busca de sinais na Arquitetura HCI-SL.

A cada busca realizada no sistema, a métrica de similaridade deve computar o índice de similaridade entre a entrada e todos os sinais da base de dados. Logo, se a indexação dos sinais e a métrica não forem adequadas, o sistema pode demandar um alto tempo para o processamento dos resultados.

Exemplo 8. Considere uma base de dados com **10.000 sinais**[1] representados pelo CORE-SL, armazenados sem uma estratégia específica de indexação. Se considerarmos que uma função de similaridade gasta aproximadamente **0.2 segundos** para cada comparação entre um sinal de entrada e todos os sinais da base de dados, a lista de sinais candidatos (resultado) é determinada em aproximadamente **30 minutos**.

Este exemplo, que desconsidera os fatores de infra-estrutura, ilustra a necessidade de desenvolvimento de uma estratégia de indexação eficiente para a base de sinais do CORE-SL, que permita minimizar o número de computações necessárias para o cálculo de similaridade e, consequentemente, reduzir o tempo de processamento e de resposta.

5.2 Busca por Similaridade *(Similarity Search)*

Segundo Chavez et al. (2001) [19] *"o problema de procurar os elementos de um conjunto que estão perto de um determinado elemento de consulta sob algum critério de similaridade tem um grande número de aplicações em muitos ramos da ciência da computação, desde reconhecimento de padrões para texto e recuperação de informação multimídia"*.

Este conceito de similaridade ou proximidade é determinado por meio de uma função (métrica) que calcula a distância entre pares de objetos de um conjunto. Esta métrica deve satisfazer as seguintes propriedades relacionadas ao conceito de distância [118]:

[1] Como apresentado anteriormente, o Deit-Libras inclui aproximadamente 10.000 sinais no vocabulário do dicionário [18] e [17].

- **não-negatividade**: a distância entre dois objetos deve ser sempre positiva;
- **identidade**: a distância de um objeto para ele mesmo deve ser zero;
- **simetria**: a distância de um objeto x para y deve ser a mesma que y para x;
- **desigualdade triangular**: a distância entre x e y deve ser sempre menor que a distância de x e y intermediada pelo ponto z.

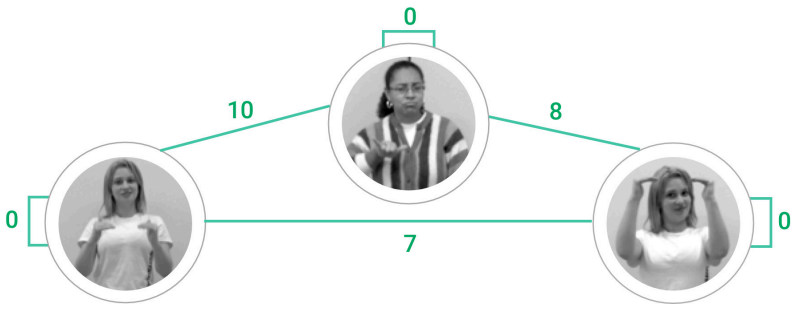


Figura 5.3: Exemplo das propriedades de uma função de similaridade
Fonte: O autor (2015)

A Figura 5.3 exemplifica, no contexto do CORE-SL, estas propriedades de uma função de distância. Na imagem, cada aresta possui um valor numérico que representa uma distância calculada entre dois sinais por uma função. Formalmente, podemos considerar as seguintes definições:

Definição 24. Um Espaço Métrico M consiste de um par $M = (D, d)$, onde D é o conjunto de dados de um domínio e d consiste de uma função para o cálculo de distância.

Definição 25. Uma métrica ou função de distância definida por $d : D \times D \rightarrow \Re$ é utilizada para calcular a similaridade entre pares de objetos em D e deve satisfazer as propriedades:

- $d(x, y) \geq 0$ para todo $x, y \in D$ (não-negatividade);
- $d(x, y) = 0$ se e somente se $x = y$ (identidade);
- $d(x, y) = d(y, x)$ para todo $x, y \in D$ (simetria);
- $d(x, y) \leq d(x, z) + d(z, y)$ para todo $x, y, z \in D$ (desigualdade triangular).

Como explana Sarmento (2010) [119], a descrição da similaridade entre dois objetos de um domínio (conjunto de todos os objetos) a partir de uma função de distância possibilita que sejam construídas estruturas de dados que podem armazenar e indexar dados complexos, que quando pesquisados pela métrica conseguem retornar o resultado eficientemente.

Na literatura [19] existem diversas métricas para os cálculos de distância, principalmente para conjuntos de dados distribuídos em um espaço euclidiano, ou seja, a posição de cada um dos objetos é mapeada por meio de coordenadas numéricas de um plano cartesiano.

Devido à especificidade do problema de busca para o contexto do CORE-SL e da Arquitetura-HCI, o conjunto de objetos (sinais representados por meio do modelo) não estão distribuídos por meio de coordenadas. Neste sentido, torna-se necessário um estudo de métricas que possam ser adaptadas para o problema tratado por esta tese.

5.2.1 Indexação de Sinais

Definição 26. No contexto da Arquitetura HCI-SL, a indexação consiste na organização de um conjunto C de representações de sinais pelo CORE-SL, por meio do agrupamento dos sinais de C em uma estrutura de dados que permita minimizar o número de computações necessárias para o cálculo de similaridade realizado por um mecanismo de busca.

Conforme ilustrado pelo cenário de complexidade, uma estratégia de indexação tem um papel fundamental em relação à eficiência de um mecanismo de busca.

Para Novak (2008)[89], o objetivo fundamental de qualquer estrutura para a indexação de dados consiste em particionar o conjunto de dados em segmentos, de modo que nem todos os segmentos necessitem ser pesquisados no momento da busca.

Neste sentido, deve-se desenvolver uma forma de indexar o conjunto não-finito dos sinais de uma LS representados por meio do CORE-SL de maneira que minimize o número de computações de distância e, consequentemente, o tempo de processamento.

É importante destacar que o problema da indexação de sinais pode ser tratado de duas maneiras: primeiro, pela definição de uma estratégia conceitual para o particionamento do conjunto de sinais e, segundo, pela possibilidade de utilização de uma estrutura de dados para a implementação desta estratégia conceitual. Esta implementação pode ser adaptada da literatura ou mesmo ser desenvolvida uma abordagem específica ao problema.

Na literatura de Busca por Similaridade [19] existem diversas estratégias de indexação para problemas específicos, tais como o processamento de imagens, o processamento de linguagem natural, classificação de texto, entre outros. No geral, as estratégias consistem em particionar os dados em sub-conjuntos de objetos similares, denominados ***clusters***.

Uma das estratégias algorítmicas mais utilizadas para a *clusterização* é o K-mens e as suas variações. O algoritmo tem o intuito de classificar os dados em *clusters* a partir da definição randômica de K objetos como centros (*centroids*). A partir da definição dos centros, o algoritmo computa a distância de cada objeto com os K-centros, definindo K-clusters para o conjunto. A cada iteração os centros podem ser atualizados quanto a sua posição para melhorar o particionamento.

Este algoritmo pode ser muito utilizado para processar pesquisas na forma: retornar os X objetos mais próximos de um objeto Y. Entretanto, uma limitação das estratégias relacionadas ao *K-means* é a necessidade de o conjunto de dados estar representado em um espaço de coordenadas. Como as representações do CORE-SL não são representadas com coordenadas numéricas, estas estratégias não podem ser utilizadas.

Como estamos trabalhando com o conceito de proximidade entre os sinais representados pelo CORE-SL, é ideal que um determinado sinal esteja localizado somente em *cluster*. Desta maneira, pode-se trabalhar com estratégias para *clusters* hierárquicos ou baseados em densidade [21].

Em relação à precisão no resultado de uma busca por similaridade deve-se considerar a hipótese de *clustering* [21]:

Hipótese 4 (*Clustering Hypothesis*)**.** Os objetos de um mesmo *cluster* se comportam de maneira semelhante em relação à relevância para as necessidades de informação.

Esta hipótese, utilizada na área de Recuperação da Informação (IR - *Information Retrieval*), define que se um objeto de um determinado *cluster* é relevante para o resultado da busca, então os demais objetos deste *cluster* também podem ser relevantes, uma vez que compartilham de propriedades similares.

Esta hipótese é importante, pois auxilia na definição da estratégia de indexação, uma vez que pode minimizar os cálculos necessários em certos *clusters*.

5.2.1.1 Estratégia Proposta

A partir de uma análise do problema da busca de sinais na Arquitetura HCI-SL e da sua complexidade, foi proposta uma estrutura de indexação baseada em *clusters*. A estratégia foi proposta com base em dois conceitos presentes na literatura [117]: *cluster* hierárquico e representante central (*centroid*).

Considerando que ainda não existem *clusters* no sistema, a estratégia consiste em particionar um conjunto de sinais descritos por meio do CORE-SL, onde cada sinal deve fazer parte de um único *cluster*. Em um cenário ideal, cada *cluster* deve ter um número limite L de sinais que o compõe.

Cada *cluster* de sinais do CORE-SL deve possuir um sinal denominado de **representantivo central**. Este sinal representativo tem o papel de atuar como o "centro"de um *cluster*, no qual todos os seus demais sinais são similares a este sinal central.

Este sinal representantivo tem o objetivo de funcionar como um resumo ou uma visão geral em relação a todos os sinais descritos no *cluster*. Esta característica é presente, uma vez que todos os sinais pertencentes a um mesmo *cluster* são similares.

Esta estratégia possibilita ao mecanismo de busca aplicar a métrica para a verificação de similaridade somente no sinal representativo central de cada *cluster*. Desta forma, se

o conjunto de sinais está indexado em N *clusters*, o mecanismo de busca irá executar somente N computações de similaridade.

Exemplo 9. Considere uma base de dados com **10.000 sinais** representados pelo CORE-SL, indexados em N clusters com a estratégia de sinal representativo central. Se considerarmos que uma função de similaridade gasta aproximadamente **0.2 segundos** para cada comparação entre um sinal de entrada e todos os sinais representativos centrais dos N *clusters*, a lista de sinais candidatos (resultado) pode ser determinada em aproximadamente **3 minutos**, considerando $N = 1000$ *clusters*.

Este exemplo mostra que aplicando-se a estratégia de indexação proposta, pode-se obter um desempenho de busca muito superior em relação à busca ingênua, que compara o termo de entrada com todos os sinais do banco de dados.

Considerando um conjunto $\boldsymbol{S}$ de sinais ainda não indexado e um coeficiente α, um algoritmo para a criação de *clusters* baseado em sinais representativos centrais consiste da seguinte estrutura:

Algorithm 1 Sinal Representativo Central (SRC)

```
1: α ← input()                                  ▷ O coeficiente limitante é informado
2: clusters ← [ ]                               ▷ clusters é inicializado e vazio
3: S ← [S_1, S_2, ...S_n]                       ▷ S recebe o conjunto de sinais
4: novo ← true
5:
6: while S ≠ [ ] do                             ▷ Repita até analisar todos os sinais de S
7:     escolha S_i ∈ S                          ▷ Selecione uma descrição de um sinal em S
8:     S ← S − S_i                              ▷ Remover S_i de S
9:
10:    if !novo then                            ▷ Se não é a primeira execução
11:        for all central in clusters do       ▷ Para todos os sinais centrais dos clusters
12:            calcule a distância d(central, S_i)
13:
14:        if a menor distância calculada < α then
15:            inserir S_i no cluster correspondente
16:        else                                 ▷ Senão, não é suficientemente similar
17:            novo ← true                      ▷ Então inserir um novo cluster
18:
19:    if novo then
20:        criar um novo cluster
21:        definir S_i como representativo central
22:        novo ← false
23:
24: return clusters                             ▷ return the solution
```

Este peseudo-código é apenas uma exemplificação de como a estratégia de indexação pode ser implementada. Esta abordagem é eficiente computacionalemente, mas uma possível limitação do algoritmo é que se os *clusters* não forem limitados por um número máximo de sinais, os *clusters* podem ficar desbalanceados. Neste caso, os primeiros *clusters* do pré-processamento tendem a ter mais sinais do que os *clusters* criados posteriormente.

Este problema poderia impactar na relevância dos resultados da busca, pela hipótese de apresentar muitos sinais de resposta. Uma solução para minimizar este problema é incluir no algoritmo uma constante ***N*** que corresponde ao número máximo de sinais que é permitdo em um *cluster*.

Adicionalmente, pode-se executar um processamento em *background* para re-processar a indexação visando balancear os *clusters* criados pelo algoritmo.

Embora a estratégia apresentada proporcione uma melhora significativa na indexação dos sinais, a busca ainda não é totalmente eficiente nesta estrutura de *cluster*. Isto se deve à possibilidade da criação de muitos clusters, o que ainda demanda um número alto de verificações de similaridade.

Uma estratégia para minimizar este problema, consiste em adicionar à estrutura de *clusters* baseado em representantes centrais, um processo de camadas para a formação de uma organização hierárquica de *clusters*.

Considere um número máximo de sinais (N) em cada *cluster* e um conjunto de sinais agrupados em *clusters* pela estratégia de representante central. Uma organização em camadas hierárquicas consiste em aplicar uma versão do algoritmo de construção dos *clusters* apenas no sub-conjunto de sinais representativos centrais.

Ou seja, se considerarmos um cenário ideal no qual o Algoritmo 5.2.1.1 tenha gerado 1000 *clusters* com 10 sinais em cada um, os 1000 representantes centrais poderiam ser processados em uma nova camada de *clusters*. Neste sentido, poderiam ser gerados 100 *clusters* com 10 sinais em cada um em uma segunda camada, onde cada *cluster* também teria seu representante central. Este processamento pode ser repetido até que haja somente um *cluster* principal com 10 sinais.

Esta estrutura hierárquica por camadas é ilustrada pela Figura 5.4. Neste cenário, um mecanismo de busca pode iniciar a pesquisa pelo *cluster* da Camada D. Este *cluster* não possui um representante central, pois o mecanismo de busca deve comparar o sinal pesquisado com todos os sinais deste *cluster* a fim de encontrar o mais similar. Ao encontrar um sinal similar, este consiste de um representante central de um *cluster* da Camada C.

O algoritmo então localiza o *cluster* na Camada C e faz novamente o cálculo de similaridade entre todos os seus sinais. Ao escolher o mais similar, novamente, este sinal consiste de um representante central de um *cluster* da Camada B. Então, o processo segue-se até que um *cluster* da última camada seja encontrado. Este pode ser a lista de sinais candidatos (resultado).

Exemplo 10. Considere uma base de dados com **10.000 sinais** representados pelo CORE-SL, indexados em clusters com a estratégia de sinal representativo central e organizados em quatro camadas hierárquicas. Se considerarmos que uma função de similaridade gasta aproximadamente **0.2 segundos** para cada comparação entre um sinal de entrada e

todos os sinais representativos centrais dos *clusters*, a lista de sinais candidatos (resultado) pode ser determinada em aproximadamente **6 segundos**.

Para calcular este tempo, o algoritmo precisa apenas fazer no máximo K x Q cálculos de similaridade, onde K é o número máximo de sinais que um *cluster* pode conter e Q é a quantidade de camadas. Como cada *cluster* possui um representante central que relaciona diretamente a um *cluster* da próxima camada, o algoritmo possibilita encontrar o resultado do processamento de maneira rápida e otimizada.

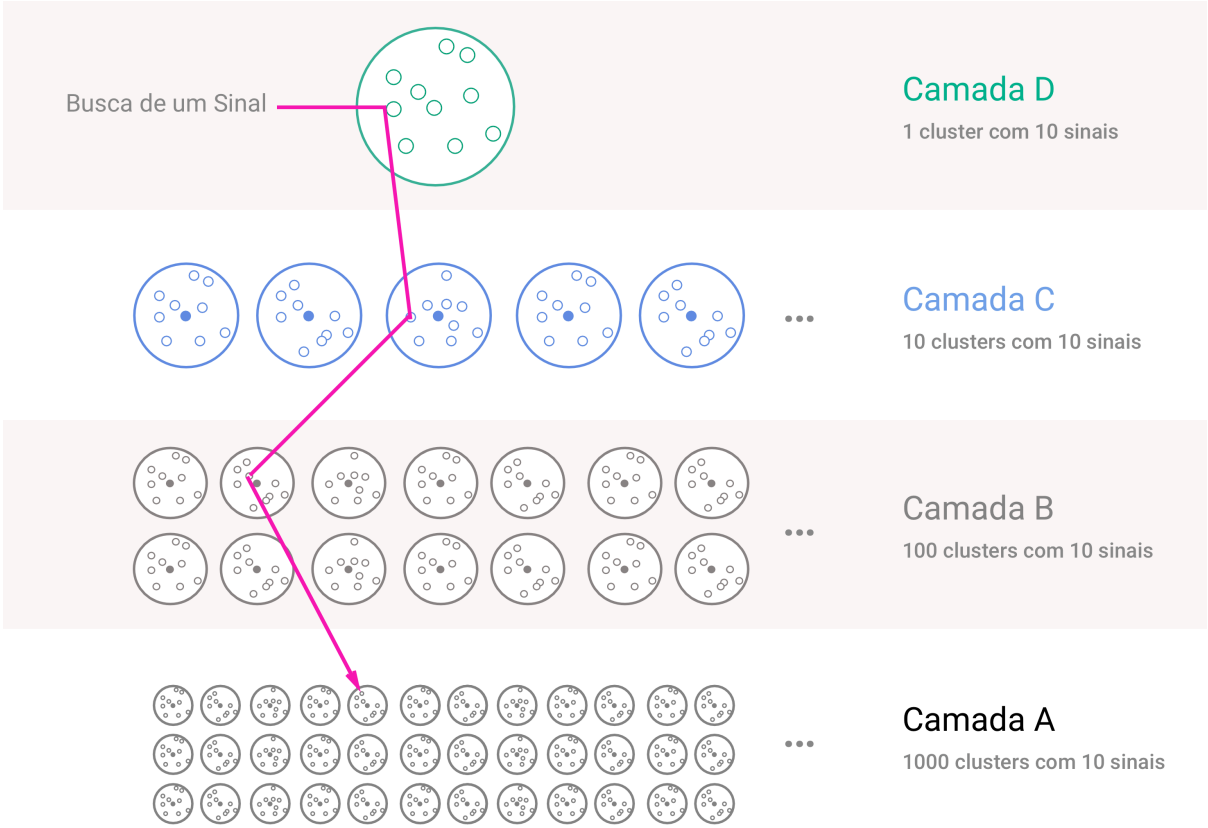


Figura 5.4: Exemplo de particionamento em clusters hierárquicos pela estratégia proposta
Fonte: O autor (2015)

Desempenho e Implementação

Zezula et al. (2006) [117] apresenta algumas características que uma estrutura de indexação deve conter para possibilitar um bom desempenho:

- **Dinamicidade**: a capacidade de permitir inserções e remoções na estrutuda de dados, minimizando custos;
- **Armazenamento**: a estrutura deve permitir grandes coleções de dados;

- **Otimização**: a estrutura de dados deve ser otimizada para utilizar o menor número de recursos de CPU e de I/O.

Como discutido no nível interno do CORE-SL, o acesso aos dados pode ser otimizado por meio de serviços de cache em memória secundária, bem como a escalabilidade horizontal para distribuir o banco de sinais, possibilitando alta performance e disponibilidade de acesso.

A questão de dinamicidade é completamente viável na estrutura proposta, uma vez que podem-se utilizar algoritmos em *background* para realizar estas operações de inserção e remoção, não prejudicando o desempenho de uso do cliente final.

A partir da implementação desta estratégia de indexação dos dados e a definição de uma métrica de similaridade adequada ao problema, pode-se utilizar o cálculo proposto por [19] para avaliar o tempo de processamento de uma busca $\boldsymbol{T}$ = *cálculos de distância* x *complexidade da função de distância + tempo extra de CPU e I/0*. O objetivo é buscar minimizar T ao máximo.

A estratégia de indexação baseada em *clusters* possibilita diversas maneiras de implementação, desde o uso de árvores métricas ao uso de listas [19]. Como discutido e exemplificado no nível interno do CORE-SL, pode-se até mesmo utilizar uma estrutura em JSON em um banco de dados baseado em documento para representar cada *cluster* e seus relacionamentos com as outras camadas hierárquicas.

Na literatura, uma implementação que pode ser adequada à estratégia de indexação proposta consiste no uso de Listas Recursivas de Clusters - RLC (*Recursive Lists of Clusters*) [119] e [118].

A estrutura da RLC se baseia no conceito de *clusters* e no uso de um objeto que representa o centro do agrupamento. Adicionalmente, cada *cluster* na RLC inclui um valor de raio (na estratégia proposta podemos entender como o coeficiente que limita a similaridade dos clusters), o tamanho do agrupamento e os seus elementos [119] e [118].

A RLC também permite fazer a ligação entre duas listas, sendo um recurso fundamental para representar os relacionamentos entre as camadas hierárquicas propostas para a indexação do CORE-SL [118].

Adicionalmente, Sarmento (2010) [119] faz um estudo sobre a implementação da RLC utilizando memória secundária. Os resultados deste estudo indicaram que a RLC possibilita buscas por similaridade e operações de inserção de dados de maneira eficiente.

5.2.1.2 Indexação por Características

Como apresentado no nível interno do CORE-SL, se a estrutura de dados para o armazenamento das descrições for baseada em um padrão de documento, pode-se utilizar metadados para servir como índices para determinados tipos de busca.

Em um dicionário de LS é muito comum a categorização dos sinais de acordo com níveis gramaticais. Por exemplo, um sinal pode ser classificado como substantivo, adjetivo, verbo, entre outros. Desta forma, uma indexação por características pode auxiliar em buscas específicas, tal como selecionar grupos de sinais baseado em uma categoria.

No contexto da Arquitetura HCI-SL este tipo de busca pode ser útil para selecionar conjuntos de sinais, por exemplo, para o desenvolvimento de jogos educacionais para o ensino de conceitos fundamentais. Alguns exemplos incluem grupos de sinais sobre higiene, saúde e segurança.

A implementação interna deste recurso pode ser baseada em listas de índices, onde dado uma categoria (chave) são mapeados todos os sinais (índices) que descrevem esta categoria.

5.2.2 Métrica de Similaridade

Uma métrica de similaridade para o CORE-SL deve permitir calcular a distância entre dois sinais, por meio da comparação de seus parâmetros com um mínimo de tempo e de recurso computacional gasto.

Além disso, uma função para calcular a distância entre sinais deve se adequar à estrutura de indexação utilizada. Neste sentido, o autor desta tese buscou na literatura algumas alternativas para métricas que pudessem ser adaptadas ao problema de similaridade de sinais em uma estrutura de indexação baseada em cluster.

Todavia, diversas métricas existentes na literatura, como mostrado anteriormente, são específicas para o uso em um sistema de indexação baseado em coordenadas (espaço euclidiano), não sendo passíveis de aplicação na estrutura do CORE-SL.

Uma das métricas que pode ser adaptada para o contexto do CORE-SL é a Distância de Edição (*ED - Edit Distance*), pois consiste de uma função que calcula o número mínimo de operações de edição para transformar uma sequência de caracteres em outra. Desta forma, a ED calcula a similaridade pelo menor número de operações de inserção, de substituição e de remoção para transformar uma sequência A em B [117].

Por se tratar de uma função de distância que realiza o cálculo sobre sequências de caracteres, existe uma possibilidade de a ED poder ser adaptada para o CORE-SL, que também utiliza um conjunto de caracteres (*string* do modelo formal) para representar os sinais.

As operações da ED são definidas como [117]:

- **inserção**: $ins(x, i, c)$ consiste em inserir o símbolo $\boldsymbol{c}$ na string $\boldsymbol{x}$ na posição $\boldsymbol{i}$;
- **remoção**: $del(x, i)$ consiste em remover o símbolo na posição $\boldsymbol{i}$ da *string* $\boldsymbol{x}$;
- **substituição**: $replace(x, i, c)$ consiste na alteração do símbolo localizado na posição $\boldsymbol{i}$ da *string* $\boldsymbol{x}$ para o símbolo $\boldsymbol{c}$.

Uma questão importante em relação a essas três operações definidas pela ED, é que elas podem ter custos diferentes. Neste caso, deve-se balancear a métrica por meio de pesos.

Para a adaptação da ED para o contexto do CORE-SL é necessária a definição das operações possíveis para transformar um sinal A em um sinal B, além da definição de possíveis pesos quando necessário. Esta tarefa exige um alto nível de conhecimento linguístico e sobre a similaridade entre os sinais de uma LS. Portanto, o autor desta tese apenas levantou a hipótese de a ED poder ser adaptada para o contexto de pesquisa. Este é um trabalho específico, que demanda testes exaustivos e análises linguísticas complexas.

5.2.3 Precisão e Acurácia

Além da estratégia de indexação e de busca dos sinais, o CORE-SL deve possibilitar uma maneira de calcular a precisão e a relevância dos resultados da busca.

Estes cálculos podem auxiliar na verificação do desempenho e para a melhoria do mecanismo de busca, bem como nas estratégias de indexação e na métrica de similaridade.

Na área de *IR - Information Retrieval* a precisão e a acurácia podem ser calculadas por meio do conjunto de objetos retornados pelo mecanismo de busca e do sub-conjunto destes objetos que são relevantes em relação a um critério [21].

A Precisão (P), em relação aos sinais do CORE-SL, consiste na fração do conjunto de sinais recuperados que são relevantes para (Equação 5.1).

$$P = \frac{sinais\,relevantes\ retornados}{sinais\ retornados} \tag{5.1}$$

Já a Acurácia (A) - *Recall* - consiste na relação do conjunto de sinais relevantes que são retornados (Equação 5.2).

$$A = \frac{sinais\ relevantes\ retornados}{sinais\ relevantes} \tag{5.2}$$

No contexto do CORE-SL, deseja-se um balanceamento entre a precisão e a acurácia. Ou seja, os resultados retornados por um mecanismo de busca devem ser precisos, entretanto devem maximizar a acurácia, ou seja, a relevância de cada sinal retornado em relação ao sinal pesquisado.

5.3 HCIR: *Human-Computer Information Retrieval*

Além das estratégias técnicas para a indexação e a busca de sinais na Arquitetura HCI-SL, o CORE-SL deve estudar e discutir estratégias que possibilitem ao usuário final, em um cenário de busca, ferramentas para melhorar a sua experiência durante o processo de recuperação da informação.

A área de HCIR - *Human-Computer Information Retrieval* ou Recuperação de Informação Humano-Computador engloba os conceitos técnicos da IR sobre a perspectiva de IHC para possibilitar melhores recursos para a recuperação da informação. O conceito principal consiste em utilizar a inteligência humana para melhorar os processos de pesquisa [81] e [59].

A HCIR também desenvolve técnicas para facilitar a navegação e a entrada de termos de pesquisa em uma interface de busca, tais como mecanismos de reformulação do termo de pesquisa para melhorar os resultados, sugestões de correção, busca por categorias e *feedbacks* de relevância.

5.3.1 Arquitetura do Sistema

A Arquitetura de Busca do CORE-SL é apresentada na Figura 5.5. A arquitetura contempla as estruturas técnicas relacionadas à indexação e à métrica de similaridade, bem como os recursos de HCIR voltados ao usuário final, que podem ser disponibilizados nos sistemas que envolvam busca de sinais.

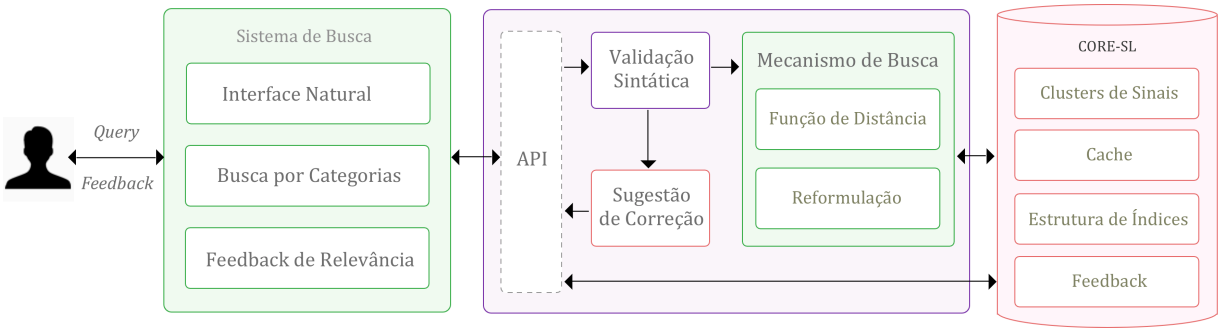


Figura 5.5: Abstração da arquitetura proposta para o sistema de busca do CORE-SL
Fonte: O autor (2015)

Na arquitetura o usuário pode pesquisar um sinal por meio da entrada por linguagem natural, via SignWriting ou Visão Computacional, além de poder fazer a busca por índices de categorias. A interface do sistema da Arquitetura HCI-SL que está usando o mecanismo de busca do CORE-SL reconhece a entrada do usuário e envia para a API de busca.

O termo de pesquisa é então processado e transformado em uma descrição no CORE-SL, que é validada sintaticamente de acordo com as regras do modelo formal. Se esta validação falha, pode-se customizar a resposta para o usuário por meio de um recurso de sugestão de correção. Ou seja, se houver a possibilidade, o sistema mostra ao usuário onde está o erro e uma sugestão de correção em relação ao vocabulário.

Se o termo é validado, então o mecanismo de busca tem o papel de comunicar-se com o CORE-SL no nível físico e fazer os cálculos de similaridade para encontrar o conjunto de sinais candidatos à resposta. Caso a resposta não seja encontrada pela métrica de

similaridade, um mecanismo de reformulação pode tentar refazer a descrição do termo de busca, baseando-se nas operações de transformação definidas pela função de distância.

No nível físico, a base de sinais representada por meio do CORE-SL é indexada utilizando a estratégia de *clusters*, além de possuir uma estrutura de dados para o armazenamento dos índices de categoria.

5.3.2 Mecanismo de Reformulação Automática

A reformulção automática do termo de pesquisa (*Automatic Query Reformulation*) consiste de um recurso interno de um mecanismo de busca, que tem o objetivo de executar transformações ou extensões no termo de entrada para possibilitar melhores resultados na busca.

Por exemplo, na arquitetura de busca (Figura 5.5), um recurso de reformulação pode ser utilizado quando o mecanismo de busca não consegue encontrar resultados por meio da métrica de similaridade. Neste caso, o mecanismo tenta realizar transformações ou extensões no termo de pesquisa para tentar buscar novamente na base de sinais.

Esta reformulação pode ser feita por meio de inferências baseadas nas operações de transformação definidas na métrica de similaridade, além do uso dos dados estatísticos provenientes da avaliação da precisão e da relevância pelos usuários.

5.3.3 Sugestão de Correção

A sugestão de correção (*Spelling Sugestion*) consiste de um recurso interno do mecanismo de busca, que pode ser utilizado no contexto de descrições de entrada com problemas de sintaxe (forma).

Neste sentido, um recurso de correção pode utilizar transformações no termo de entrada baseado na gramática formal do CORE-SL e retornar ao usuário uma sugestão do termo de busca que ele está tentando descrever.

Este recurso é empregado constantemente em sistemas de busca, que processam o termo e retornam uma mensagem de *feedback* ao usuário perguntando: *"você quis dizer* ***X****"*? Assim, mesmo com uma entrada não-exata e sem precisão, o sistema pode fornecer *feedback* ao usuário e auxiliá-lo na busca.

5.3.4 Feedback de Relevância

O *Feedback* de Relevância consiste em possibilitar no mecanismo de busca uma forma para o usuário avaliar a relevância dos resultados da pesquisa. Esta avaliação é calculada internamente por meio da precisão e da acurácia.

Mesmo com a possibilidade de executar testes internos utilizando as métricas de precisão e de acurácia, a interação com o usuário final é fundamental para a melhoria do

sistema e, consequentemente, da interação.

Na arquitetura do mecanismo de busca (Figura 5.5), quando o usuário faz a classificação da relevância do resultado, a interface envia os dados para a API da busca, que salva em uma estrutura de dados estas estatísticas de feedback, que podem ser analisadas posteriormente para a melhoria da métrica de similaridade e do sistema de busca.

Quando o mecanismo de busca retorna os resultados, a interface do sistema pode solicitar aos usuários que selecionem quais os sinais do conjunto retornado é similiar ao termo de busca de entrada (precisão) ou quais são os sinais que correspondem à sua intenção de pesquisa (acurácia - relevância).

5.4 Considerações

Este capítulo abordou sobre o problema de indexação e da busca de sinais por similaridade no contexto da Arquitetura HCI-SL, apresentando uma visão geral sobre o problema do ponto de vista técnico e do ponto de vista conceitual para o usuário final.

Em relação ao ponto de vista técnico, o capítulo apresentou um cenário de complexidade para ilustrar o problema da busca e discutiu uma estratégia de indexação de sinais baseado em *clusters*. A abordagem proposta englobou os conceitos de sinal representativo central para cada um dos *clusters* e um conceito de agrupamento hierárquico.

A estrutura algorítmica mostrou indícios de que a estratégia proposta pode ser aplicada em um sistema de busca em um contexto real de uso na Arquitetura HCI-SL, com o objetivo de resolver o problema de busca por similaridade em relação à indexação.

Ao longo do capítulo, mostrou-se a melhoria do cenário de complexidade que passou de uma busca ingênua com um grande tempo de processamento, para uma estratégia de busca eficiente realizada em um tempo muito pequeno e com baixo consumo de recursos computacionais. Além disso, mostrou-se também uma estrutura de dados que pode ser utilizada para a implementação da estratégia de indexação, a RLC.

Adicionalmente, o capítulo discutiu sobre a Distância de Edição, uma métrica de similaridade utilizada para calcular a proximidade entre *strings*. Conforme discutido, esta métrica pode se adequar ao contexto do CORE-SL, desde que sejam definidas as relações de recorrência e as operações de transformação específicas para os sinais representados no modelo computacional.

Estas operações de transformação, por exemplo, de inserção, de substituição e de exclusão não foram definidas pelo autor desta tese, uma vez que considerou-se que estas operações demandam um grau de complexidade que a Linguística pode auxiliar posteriormente.

Para finalizar, o capítulo abordou o conceito de HCIR que agrega os recursos técnicos da IR e as técnicas de IHC para o desenvolvimento de recursos de pesquisa que propiciem uma boa experiência de uso para os usuários finais. Neste sentido, foram discutidas

algumas técnicas que podem ser implementadas no mecanismo de busca do CORE-SL, que consideram as necessidades do usuário final durante a busca e, consequentemente, podem potencializar a facilidade de uso destes recursos.

Na próxima seção são apresentados alguns cenários de aplicabilidade do CORE-SL para auxiliar o desenvolvimento de ferramentas da Arquitetura HCI-SL.

CAPÍTULO 6

ESTUDO DA APLICABILIDADE DO CORE-SL

Este capítulo apresenta um estudo exploratório em relação à utilização do CORE-SL em cenários específicos da Arquitetura HCI-SL. Especificamente, o capítulo descreve algumas hipóteses relacionadas à aplicação do CORE-SL para auxiliar o desenvolvimento de serviços e de aplicações computacionais na arquitetura.

Este estudo da aplicação do CORE-SL consiste em apresentar abordagens conceituais e ténicas, na forma de *frameworks*, que tem o intuito de fornecer estratégias para a melhoria dos processos de desenvolvimento relacionados à arquitetura, visando a resolução do problema macro da tese: o tratamento computacional das LS.

O resultado deste estudo exploratório constituiu uma contribuição essencial desta tese para a resolução dos problemas computacionais da arquitetura. Desta maneira, esta pesquisa buscou disseminar todo o conhecimento científico desenvolvido para a comunidade de Ciência da Computação, disponibilizando recursos que auxiliem na resolução do problema tecnológico e que, posteriormente, corroborem a resolução do problema de inclusão sofrido pelas comunidades de surdos.

6.1 Arquitetura HCI-SL

Como discutido anteriormente, esta tese se insere no contexto de uma Arquitetura Computacional Baseada na Interação Humano-Computador em Língua de Sinais (HCI-SL). O objetivo desta arquitetura, é proporcionar e desenvolver um ambiente integrado, bem como hipóteses, estratégias metodológicas e serviços, capazes de resolver o problema do tratamento computacional das LS e, consequentemente, possa auxiliar na eliminação da barreira social de acesso à informação e ao conhecimento sofrida pelas comunidades de surdos [56].

Como apresentado por Garcia et al. (2013) [56], a Arquitetura HCI-SL (mostrada em uma visão macro na Figura 6.1) consiste de uma estrutura específica para o tratamento computacional das LS e é composta basicamente de três camadas:

1. **API Interna**: camada interna de API responsável por disponibilizar os serviços computacionais básicos e fundamentais para que aplicações para o usuário final possam ser desenvolvidas.

   Nesta camada, devem ser incluídos os sistemas de reconhecimento automático de sinais, o aprocessamento de SignWriting, a síntese automática de agentes virtuais 3D, processos de PLN, os serviços relacionados ao CORE-SL, dentre outros. O acesso

a estes serviços é realizado por meio de requisições a um framework computacional, baseado nesta abordagem de IHC;

2. **API de Serviços**: segunda camada responsável por representar recursos computacionais que utilizam os serviços da camada interna de API em sua constituição.

   Esta camada de serviços, tem o objetivo de funcionar de duas maneiras, a primeira, disponibilizando sistemas para o usuário final (por exemplo, dicionários, tradutores, vocabulários controlados, ambientes virtuais de comunicação, entre outros) e, a segunda, consiste na capacidade destes sistemas funcionarem também como serviços na forma de API.

   Neste sentido, qualquer aplicação que necessitar pesquisar o significado de um sinal ou mesmo traduzir um conteúdo poderia apenas utilizar o serviço da camada interna.

3. **Aplicações**: esta é a camada na qual é desenvolvida uma série de aplicativos para o usuário final, tais como Ambientes de Educação a Distância, Jogos, Sistemas para Letramento, dentre outros.

   Conforme discutido anteriormente, esta Arquitetura utiliza uma abordagem baseada na IHC para a resolução do problema de pesquisa e, neste sentido, os trabalhos e as pesquisas baseados nesta arquitetura fazem um claro entendimento das necessidades comunicacionais e linguísticas dos surdos.

O problema abordado por esta arquitetura [56], especificada com a participação continuada de uma comunidade de surdos, foi que a falta de ferramentas computacionais e de sistemas de informação que auxiliem de fato o acesso ao conhecimento por estes indivíduos se deve à ausência de alguns recursos computacionais específicos, necessários para promover uma interação em LS.

Como citado anteriormente, um dos objetivos do CORE-SL consiste em investigar como a sua aplicação pode auxiliar a construção desses recursos computacionais, bem como pode impulsionar o desenvolvimento de aplicações que realmente disponibilizem uma IHC em LS e, assim, promover o acesso à informação e à inclusão dos surdos na sociedade.

No esquema mostrado na Figura 6.1 são apresentadas as relações específicas entre os serviços da Arquitetura HCI-SL, bem como a necessidade do CORE-SL como um núcleo de software necessário para o suporte efetivo na construção de sistemas ao usuário final.

Neste contexto, o CORE-SL tem um papel central na arquitetura, tornando-se responsável por todo o intercâmbio das informações relacionadas à representação computacional dos sinais, bem como as funções computacionais básicas inerentes ao modelo formal, tais como a inserção, a busca por similaridade, a atualização, a indexação, dentre outros.

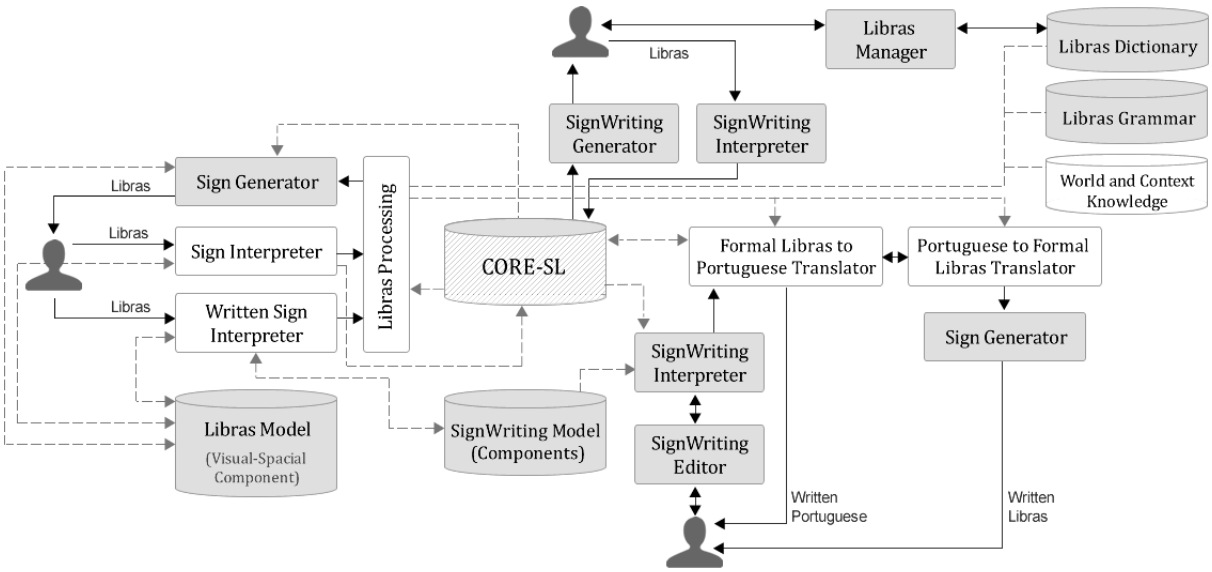


Figura 6.1: Aplicação do CORE-SL nos processos internos da Arquitetura HCI-SL
Fonte: García et al. (2013) [56]

Neste fluxo de funcionamento, o CORE-SL fica evidenciado como um componente arquitetural central, responsável por controlar as informações e as funções computaicionais relacionadas à representação dos sinais. Desta maneira, os aplicativos que requerem funções como a busca, o armazenamento, a indexação e a reprodução utilizam necessariamente o CORE-SL.

Dentre o conjunto de contextos da Arquitetura HCI-SL, as próximas seções apresentam um estudo específico em relação à aplicabilidade do CORE-SL nos problemas de reconhecimento de sinais e da construção de uma base de sinais contextual.

Este estudo mostra como o CORE-SL pode ser utilizado como abordagem metodológica ou técnica para a produção de um banco de sinais baseado em um contexto real de uso, com o objetivo de apoiar o desenvolvimento de ferramentas na Arquitetura que requeiram desta base de sinais.

Em relação ao problema de reconhecimento, apresenta-se um *framework* conceitual para apoiar o desenvolvimento de um sistema de reconhecimento automático de sinais, que pode permitir a construção de ferramentas para a interação em LS.

6.2 Framework para o Reconhecimento Automático de Sinais

O problema do Reconhecimento Automático de Sinais (RAS) das LS consiste em um dos sub-problemas mais complexos da Arquitetura HCI-SL. Devido ao caráter gestual-visual das LS, a entrada de sinais em qualquer sistema por meio das LS, em geral, deve ser feita de forma natural.

Desta forma, podem ser desenvolvidos mecanismos computacionais que capturem os

sinais como dados de entrada, executem o processamento e retornem uma representação computacional no CORE-SL como saída.

Segundo Antunes et al. (2011) [5], as pesquisas para o RAS começam a surgir do final da década de 80. Desde então, foram realizados diversos estudos na tentativa do reconhecimento de sinais que, mesmo tendo apresentado resultados quanto às abordagens matemáticas, algorítmicas e computacionais, ainda não apresentaram artefatos tecnológicos que proporcionem uma real interação em LS [5].

Antunes et al. (2011) [5] apresenta a discussão sobre uma série de trabalhos relacionados ao RAS, elencando os cinco problemas principais, discutidos à seguir.

6.2.1 Abordagem Conceitual

Muitos trabalhos apresentam uma inadequação em relação ao objeto de pesquisa, ou seja, objetivam dar suporte aos surdos no acesso à informação, mas não levam em consideração os surdos no processo de desenvolvimento, bem como a sua cultura, as suas necessidades e um entendimento correto sobre as LS.

Para ilustrar esta inadequação, Antunes et al. (2011) [5] citam:

- **o tratamento inadequado das LS**: estudos de caso simples de sinais que representam a soletração do alfabeto da língua oral; conjuntos de sinais pequenos, isolados, aleatórios e não representativos; e a consideração de um sinal como a unidade mínima de significado;

- **problemas metodológicos**: na seleção da amostra de sinais, na abordagem utilizada para o reconhecimento e nos materiais e métodos utilizados;

- **problemas de tecnologia**: uso de tecnologias específicas sem considerar o usuário e um contexto real de uso neste processo. Alguns problemas incluem a interação por *hardwares* específicos tais como *luvas de dados*, que restringem a naturalidade do usuário;

- **ambientes controlados**: uso de ambientes controlados que desconsideram as situações reais de uso;

O desconhecimento das reais necessidades dos surdos, bem como da sua língua, levam à resultados que não promovem quaisquer facilitadores no acesso à informação, replicando constantemente resultados puramente algorítmicos (e.g. gráficos e taxas de acerto).

Esta abordagem gera "sistemas"que são autilizados apenas em laboratório na forma de ambientes controlados, não visando a adequação ou a construção de serviços que possam ser usados na prática pelos surdos por meio da HCI-SL.

A tecnologia deve ser utilizada como um recurso poderoso para dar suporte ao desenvolvimento de ferramentas efetivas que considerem às necessidades do usuário (o surdo),

fazendo um entendimento correto da LS com o intuito de desenvolver uma interação baseada em LS [4].

6.2.2 Abordagem Metodológica

As abordagens de reconhecimento incluem dois tipos principais, a **baseada em palavras** e a **baseada em fonemas**.

A primeira, consiste em reconhecer um conjunto de características visuais dos sinais e mapear este vetor diretamente a uma palavra da língua oral em questão, sendo necessário treinar o sistema para cada um dos sinais da base.

A segunda abordagem, consiste em utilizar as propriedades fonéticas que compõe os sinais para segmentá-los em sub-unidades. No contexto de IHC, esta abordagem baseada na nos parâmetros fonéticos é mais adequada, pois possibilita a geração de conjuntos de treinamento mais adequados e uma maior capacidade no reconhecimento de sinais que não foram treinados no sistema [5].

Todavia, a maioria dos trabalhos não define ou não utiliza um modelo computacional para representar os sinais, bem como a estrutura de suas sub-unidades. Em muitos casos são realizados estudos com base somente am algumas CM, alguns fonemas, ou sub-unidades estáticas.

Neste sentido, o CORE-SL pode auxiliar a resolver este problema. Este *framework* apresenta algumas estratégias em relação ao uso do CORE-SL como abordagem técnica e metodologógica para a resolução do problema de reconhecimento.

6.2.3 Abordagem para a Geração de Bases de Teste

Como mostrado por [5], um problema também relacionado ao objeto de pesquisa consiste da inadequação das bases de sinais utilizadas para o treinamento e o teste dos algoritmos de reconhecimento.

Na prática, não existe uma metodologia para selecionar adequadamente um conjunto de sinais, o que causa a desconsideração de aspectos fundamentais para a LS e que podem impactar na qualidade e precisão do sistema.

Alguns problemas encontrados incluem a utilização de bases de imagens estáticas, sinais com apenas algumas configurações de mão, conjuntos aleatórios e pequenos de sinais (dezenas de sinais), e conjuntos maiores selecionados de maneira isolada e aleatória sem nenhum critério.

Além desses conjuntos não representarem toda a LS, não são selecionados os casos de sinais similares que tem um papel importante na avaliação da acurácia do sistema RAS em relação a este problema específico.

6.2.4 Abordagem Tecnológica

Adicionalmente o trabalho de [5] ainda ressalta as restrições das pesquisas em relação a tecnologias muito específicas utilizadas no desenvolvimento, que além de não proporcionarem ou facilitarem a construção de um ambiente para o usuário final, limitam a captura de dados ou restringem a capacidade/totalidade das LS.

Portanto, no *framework* desenvolvido, são consideradas tecnologias de visão computacional baseadas em câmeras de vídeo convencionais aos usuários.

6.2.5 Framework Proposto Baseado no CORE-SL

A abordagem inicialmente tem seu escopo definido dentro da Arquitetura HCI-SL. Neste sentido, a abordagem utilizada no processo de RAS é baseada no CORE-SL. Primeiramente podemos exemplificar um dos cenários em que o modelo pode ser aplicado em toda a interação do usuário com o sistema.

O exemplo apresentado na Figura 6.2, consiste no cenário de uma pessoa surda que deseja pesquisar sobre um determinado sinal em uma aplicação ou serviço de dicionário.

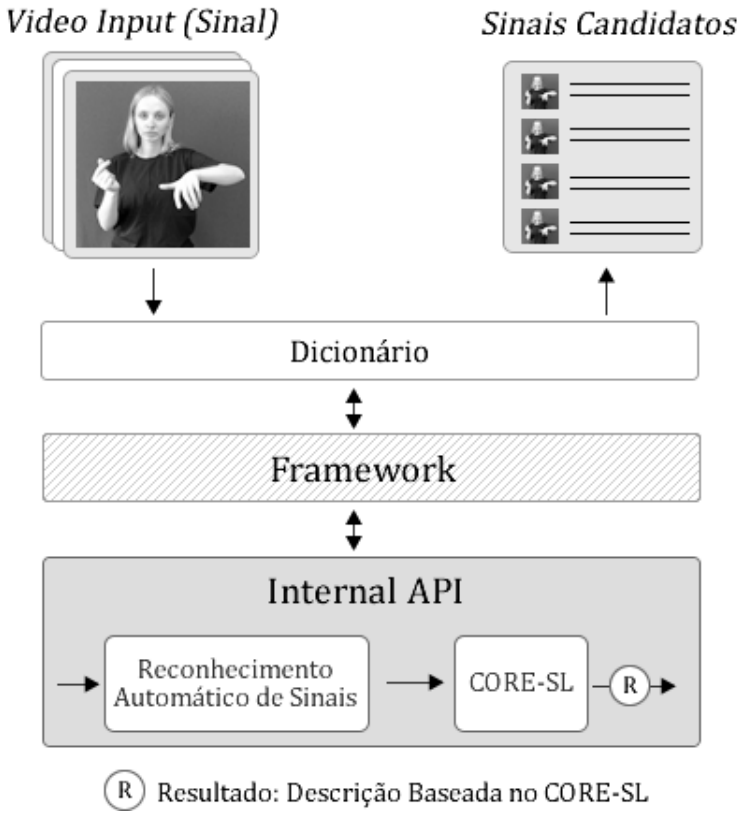


Figura 6.2: Exemplo de Aplicação do CORE-SL para uma Abordagem de RAS
Fonte: Antunes et al. (2011) [5]

O usuário entra com o sinal por meio da câmera e a aplicação envia esta entrada em vídeo para o *framework*, que solicita para o serviço de RAS o reconhecimento do sinal.

O sistema então processa esta entrada, faz a extração das características fonéticas e gera a representação destas sub-unidades no CORE-SL. A partir desta representação (as unidades que compõe aquele sinal) - retornada pelo *framework* - a aplicação pode pesquisar uma lista de sinais próximos (que contém aquela descrição fonética), retornando esta lista como resultado ao usuário.

Como mostrado na Figura 6.3, o CORE-SL está em uma camada acima ao processo de reconhecimento de sinais, uma vez que tem o objetivo de servir como *framework* conceitual para apoiar a abordagem técnica e a conceitual.

Internamente, o RAS é baseado em duas abordagens principais: **baseada em fonemas** e no **conjunto mínimo-máximo de sinais**. Neste esquema é possível observar todo o processo computacional envolvido, desde à abordagem, à construção da base de sinais e suas descrições e até a camada técnica de visão computacional.

Baseado no CORE-SL, a abordagem consiste no uso das sub-unidades fonéticas modeladas no nível formal que representam as características gestuais-visuais a serem reconhecidas pelo sistema.

Neste sentido, a abordagem disponibiliza um conjunto finito e pequeno de possibilidades que, ao serem treinadas no sistema, possibilitam o reconhecimento de quaisquer sinais, mesmo os que não foram utilizados na base de treinamento e de teste. Esta propriedade é possibilitada, pelo fato de o reconhecimento ser baseado nos parâmetros e não nos sinais como um todo.

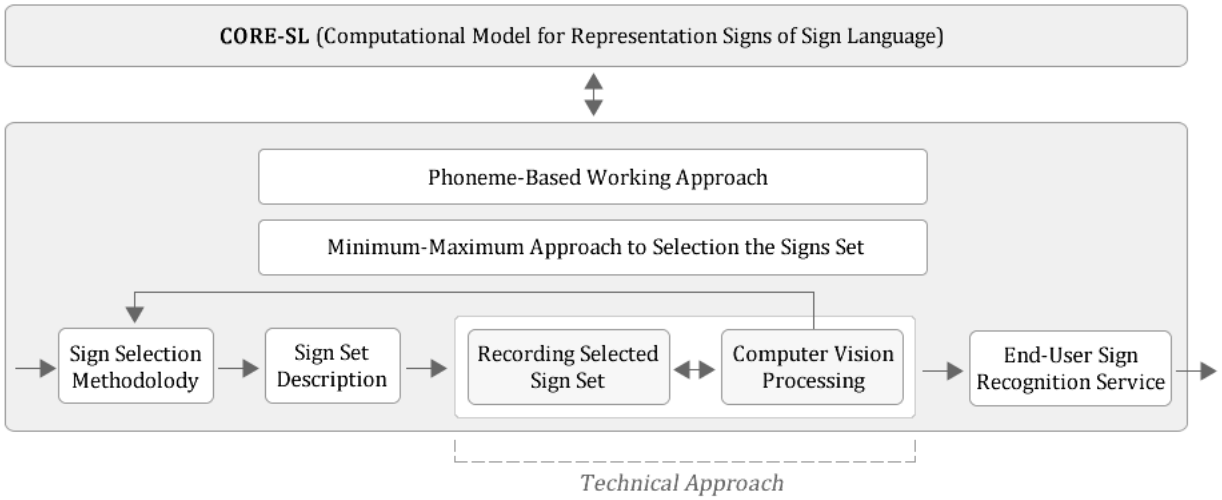


Figura 6.3: Framework proposto para o suporte a construção de um serviço final de RAS
Fonte: O autor (2011)

É importante destacar que se a propriedade de universalidade do CORE-SL for considerada nesta questão, as soluções técnicas desenvolvidas com base neste *framework* podem ser aplicadas a quaisquer LS, uma contribuição fundamental para a solução do problema computacional.

O primeiro passo do *framework* proposto consiste em utilizar a metodologia de colaboração para a seleção de sinais desenvolvida por Antunes (2011) [4] e [6].

A metodologia consiste de reuniões colaborativas presenciais ou *online* com membros da comunidade de surdos fluentes em LS, com o objetivo de levantar um conjunto de sinais representativos dentro da comunidade linguística (Figura 6.4).

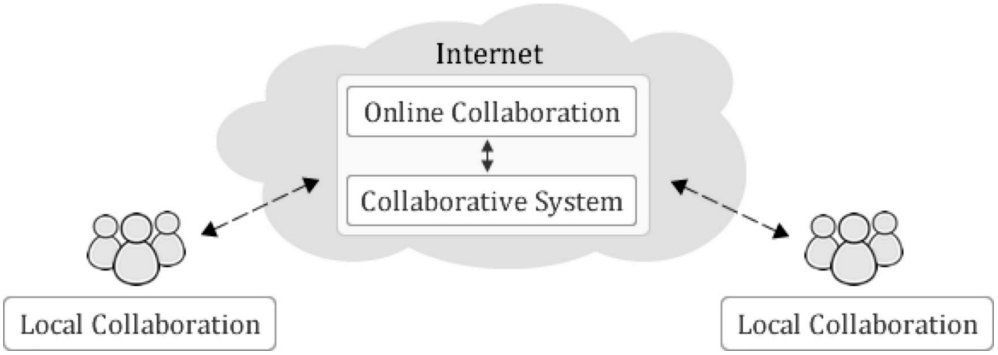


Figura 6.4: Representação dos tipos de colaboração para selecionar os sinais
Fonte: Guimarães et al. (2011) [61]

A seleção dos sinais é dada por meio dos parâmetros do CORE-SL, e alguns passos utilizados são:

A) Colaboração Local (CL)

1. Escolher uma sub-unidade (parâmetro) do CORE-SL;
2. Selecionar um valor para este parâmetro;
3. Solicitar aos participantes sinais que contenham o valor do parâmetro em questão;
4. Discutir cada sinal do sub-conjunto levantado pelos usuários;
5. Definir a forma correta do sinal (em relação às possíveis variações);
6. Gravar um vídeo de cada sinal e armazenar em um sistema;
7. Descrever cada sinal por meio do CORE-SL;
8. Repetir os passos 2 a 7 até todos os valores do parâmetro escolhido serem cobertos;

B) Colaboração Online (CO)

1. Organizar grupos para a colaboração online;
2. Escolher uma sub-unidade (parâmetro) do CORE-SL colaborativamente;
3. Selecionar um valor para a sub-unidade;

4. Pesquisar no sistema online um sinal que instancia o parâmetro escolhido;
5. Para cada sinal, discutir os resultados discutidos em grupos locais;
6. Definir um consenso sobre a representação dos sinais;
7. Registrar o consenso no sistema (vídeo e descrição);
8. Repetir os passos 2 a 7 até todos os valores do parâmetro escolhido serem discutidos;
9. Repetir os passos 1 a 8;

Uma questão fundamental nas discussões dos parâmetros, consiste em registrar em vídeo cada sinal uma vez que os interlocutores possuem variações na articulação dos sinais e, desta maneira, a representação no CORE-SL pode conter diferenças em relação ao vídeo do sinal.

E etapa de descrição do conjunto de sinais selecionados pode ser realizada por meio de um sistema que possibilite a descrição dos sinais corretamente de acordo com o nível formal do CORE-SL. Por exemplo, pode-se utilizar o sistema desenvolvido por Antunes (2011) [4], ilustrado pela Figura 6.5.

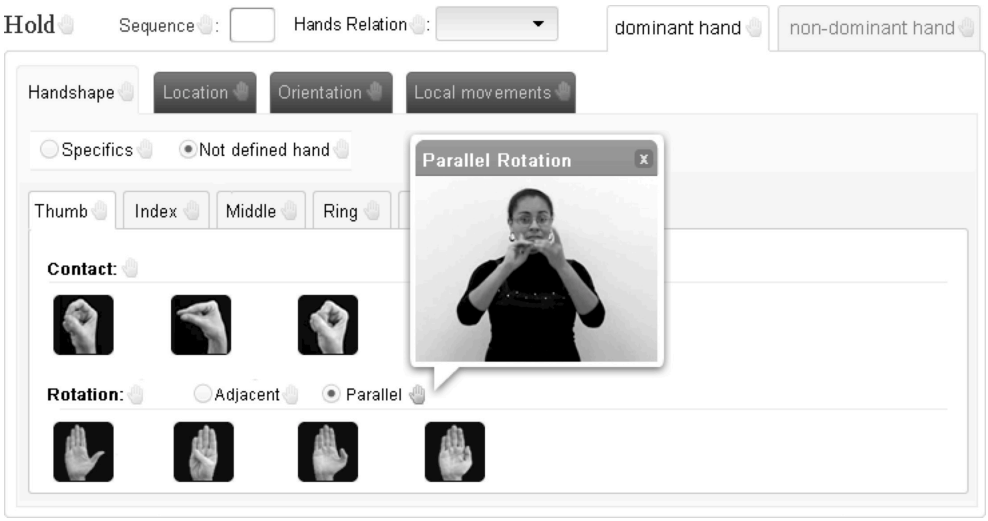


Figura 6.5: Exemplo de Sistema para a Representação de Sinais no CORE-SL
Fonte: Antunes (2011) [4]

Cabe ressaltar que, se já existir uma base de sinais (e.g. um dicionário) pode ser pulada a etapa para seleção dos sinais e, desta maneira, apenas fazer a descrição dos sinais da base no CORE-SL. .

Após a utilização das abordagens metodológicas e técnicas em relação a aplicação do CORE-SL, podem ser utilizadas as abordagens técnicas tradicionais da área de Visão Computacional, ou seja, a definição e o uso de um protocolo técnico (de escolha própria, de acordo com o contexto) para capturar estes sinais para o treinamento e para os testes do algoritmo de reconhecimento.

6.2.6 Critério Adequado para Construção de Bases de Dados

Dentro do contexto do conceito inicial do *framework* que utiliza o CORE-SL, deve-se explorar também uma metodologia para construção de bases de sinais e de descrições para o treinamento e os testes de algoritmos de reconhecimento, que considere um conjunto representativo de sinais da LS em questão, utilizando como abordagem de representação o CORE-SL.

Na próxima seção é apresentado um *framework* colaborativo baseado no contexto dos surdos para a construção desta base de sinais e de descrições.

6.2.7 Método de Treinamento e Teste Interativo

Ao analisar o *framework* para o treinamento e o teste do sistema de reconhecimento, percebe-se a relevância de se obter mais dados a partir da interação com os usuários reais. Uma maneira de treinar o sistema para condições reais de uso, poderia ser por meio do *design* de um sistema para o treinamento iterativo com o usuário final.

Neste sentido, poderia ser implementado o sistema de RAS e, usuários de uma LS poderiam executar algumas tarefas de pesquisa para simular condições de uso reais.

Por exemplo, ao articular um sinal e o sistema retornar os resultados de uma busca, a interface poderia perguntar ao usuário se o sinal retornado está correto (descrição no CORE-SL). Se não estiver, o usuário pode informar onde está o erro na descrição, auxiliando na melhoria da precisão do sistema.

Este tipo de teste foi apresentado pela técnica de *Feedback* de Relevância do Capítulo 5 relacionado à estratégia de busca e de indexação dos sinais.

No mesmo sentido, o sistema poderia apresentar ao usuário como resposta um conjunto de sinais similares e solicitar ao usuário para que ele escolha o sinal correto relacionado à sua intenção de busca.

Esta interação do usuário com o sistema poderia trazer mais dados de variabilidade entre os usuários, bem como determinar margens de erro para o sistema de reconhecimento, além da grande contribuição que seria um sistema aplicado em um contexto real de uso.

6.3 *Framework* Colaborativo Baseado em Contexto para a Construção de uma Base de Sinais por Usuários Reais

Conforme discutido no *Framework* Conceitual para o Reconhecimento de Sinais, uma base de sinais possui grande importância para o treinamento e os testes dos algoritmos. Além disso, uma base de sinais representada por um modelo computacional, tal como o CORE-SL, pode apoiar diversos estudos que envolvam a descrição dos sinais.

Uma limitação das bases de sinais da literatura é a falta de contextos de uso reais, a falta de uma estrutura para a representação (descrição) computacional de cada sinal e a falta de uma estratégia para a escolha de sinais representativos no vocabulário de uma LS. O *framework* apresentado nesta seção tem o intuito de facilitar a resolução destas três questões.

6.3.1 Contexto, Colaboração e *Framework*

O contexto consiste em compreender os usuários, suas necessidades, suas principais atividades e seu conhecimento sobre a tecnologia em ambientes de uso real. O contexto envolve uma situação, um ambiente adequado e uma atividade realizada por um grupo de pessoas.

Quando considerado o contexto de uso, é possível melhorar os recursos de interação [33]. Entretanto especificar corretamente um contexto não é uma tarefa simples, pois engloba diversas características do ambiente social, distinção de objetivos, tecnologias, etc [3].

Um contexto pode ser classificado nas categorias de **atividade**, de **identidade**, de **localização** e de ***time***. Essas categorias podem auxiliar na descrição do contexto quanto às tarefas (atividade), ao ambiente (localização) e às pessoas envolvidas (identidade) [2].

A colaboração consiste de trabalhar junto com a intenção de compartilhar objetivos e contribuir para solucionar um problema. A colaboração em atividades locais envolve processos tais como comunicação, negociação, compartilhamento e coordenação [9].

Um framework consiste de um esquema conceitual ou um modelo de domínio específico que descreve as suas situações, as suas propriedades e os seus relacionamentos. Assim, um framework tem o intuito de compartilhar ideias, definir domínios e descrever o contexto para representar os métodos e os processos para o desenvolvimento de sistemas [27] e [99].

6.3.2 Base de Sinais

Como citado anteriormente, as abordagens conceituais e técnicas utilizadas para a construção de bases de sinais na literatura possuem diversas limitações, dentre as quais estão sumarizadas na Tabela 6.1.

A Purdue RVL-SLLL [83] classifica os dados da ASL por configurações de mão e movimentos, sinais e sentenças. O primeiro critério de classificação (CM e MOV) é importante porque fornece uma abordagem mais adequada no sentido de reconhecer as sub-unidades (abordagem baseada em fonemas), antes de conseguir processar os sinais isolados ou as sentenças. A base consiste de 2576 vídeos de 39 primitivas de movimentos e 62 CM.

A base de dados RWTH-BOSTON-400 é formada por 843 sentenças, diversos interlocutores e sub-conjuntos para treinamento, desenvolvimento e o teste de aplicações [37].

Tabela 6.1: Problemas comuns em bases de sinais

Categoria	**Problemas**
Falta da IHC	**a)** não inclui um real contexto de uso; **b)** ambientes muito controlados; **c)** uso de câmeras especiais, tal como sensores de profundidade; **d)** uso de sensores específicos, tal como luvas de dados;
Abordagem da LS	**a)** sinais selecionados randomicamente sem critérios; **b)** modelos com poucas sub-unidades; **c)** repetição de sub-unidades; **d)** sinais sem similaridade;

Fonte: Antunes et al. (2011) [5]

Esta base de sinais também trabalha com ambientes controlados e uma abordagem baseada em palavra inteira. Para estes autores, ainda não está claro a melhor maneira de reconhecer os parâmetros articulatórios dos sinais.

O projeto BSL [97] utiliza uma metodologia relacionada com a sócio-linguística e com corpus linguístico. O projeto inclui interlocutores nativos da língua de sinais, e a metodologia consiste nos usuários contarem histórias pessoais, utilizando um software para a anotação dos vídeos.

O Dicta-Sign [39] é um projeto que agrega a captura de uma base de sinais, com foco específico para sistemas de ASLR, de Animação e de Tradução. Um protótipo de um sistema de reconhecimento automático utiliza um sendor de profundidade para reconhecer os movimentos. Os sinais são descritos fonéticamente, mas o modelo utilizado possui um baixo nível de detalhamento das propriedades fonéticas.

Outras bases de sinais encontradas descrevem amostras de dados que contém vídeos de conversação, sinais isolados, sinais de configurações de mão, soletração manual, captura com sensores muito específicos, foco específico da linguística, entre outros. Melnyk et al. (2014) [86] apresenta uma revisão de literatura de bases de sinais existentes com uma perspectiva relacionada a sistemas ASLR.

Um dos principais problemas destas bases de sinais é a desconsideração do contexto de uso ou por se limitarem a ambientes controlados que restringem os movimentos e a naturalidade do interlocutor na articulação dos sinais. Outro problema comum é em relação à abordagem conceitual: a) uso da abordagem palavra-inteira e b) sinais selecionados de maneira randômica.

Neste caso, uma base de sinais com um grande número de sinais pode não cobrir todas as sub-unidades fonéticas necessárias. Adicionalmente, o uso de conjuntos sem similaridade no treinmento e no teste de uma aplicação, fornece a baixa relevância nos resultados de sistemas de busca e, consequentemente, uma experiência de uso ruim.

6.3.3 Framework Proposto

O framework proposto (Figura 6.6) descreve: o **framework computational** (abordagem, armazenamento, compartilhamento, etc.) que auxilia todos os estágios do processo, tal como o **contexto de uso e a atividade**, a **abordagem colaborativa**, a **base de sinais** e o **melhoramento contínuo**. Cada módulo do *framework* auxilia os demais módulos, forcenendo uma abordagem integrada para a construção de uma base de sinais.

Nas próximas sub-seções os módulos do *framework* são validados, apresentando alguns resultados da aplicação do *framework* em um contexto real de uso.

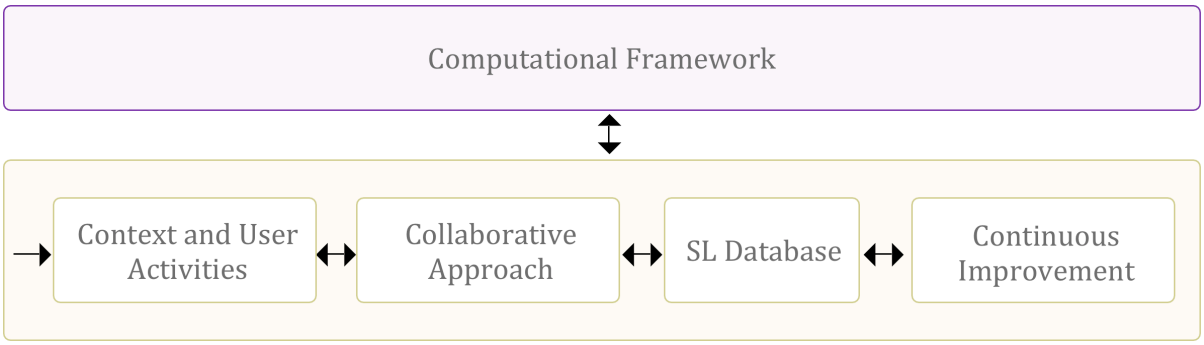


Figura 6.6: Framework Colaborativo Baseado em Contexto
Fonte: O autor (2015)

6.3.3.1 Contexto e Atividades

Este módulo destina-se a descrever o contexto específico de uma comunidade de surdos. A descrição deve especificar: **a)** os perfis, **b)** o ambiente, **c)** a atividade desenvolvida neste contexto, e **d)** o tempo. No estudo de caso utilizado para validar este módulo, o contexto foi descrito da seguinte maneira:

Perfil. Os usuários consistem de estudantes surdos do curso de Graduação em Letras-Lilbras. Este perfil de usuário foi escolhido devido à facilidade de acesso a eles dentro da universidade. Entretanto, qualquer perfil de usuário pode ser escolhido desde que esteja devidamente relacionado com a atividade, o ambiente e o tempo.

Ambiente. Nós executamos as atividades dentro da sala de aula, um local comum utilizado diariamente pelos estudantes. É importante mencionar que devido à característica visual das LS, as salas de aulas são configuradas de maneira que permitam otimizar a conversação entre todos os membros.

Atividade. A discussão e a aplicação dos conceitos aprendidos na sala de aula, neste caso, a Fonologia das LS. Outras atividades também podem ser escolhidas. Durante as atividadaes (discurso), os sinais isolados devem ser capturados e armazenados em um sistema pelo coordenador da atividade.

Tempo. As reuniões foram realizadas na forma de atividades complementares, alguns dias após o aprendizado dos conceitos em sala de aula. Isto proporcionou alguns benefícios como a contabilização das atividades complementares como atividades formativas e possibilitou aos surdos treinar e discutir os conceitos aprendidos em sala.

6.3.3.2 Abordagem Colaborativa

A estratégia **colaborativa** pode ser local ou *online* baseada em um sistema. A questão principal consiste em desenvolver a atividade contextual planejada, armazenando os sinais e o discurso gerado. Para os experimentos, utilizaram-se reuniões colaborativas locais. Se é determinada uma atividade remota, deve-se utilizar uma plataforma como o InCoP [106] para apoiar a atividade colaborativa.

A **coordenação** deve ser conduzida por um mediador (organização das tarefas, descrição das atividades e o controle) e por um assistente (tarefas operacionais). No caso de pelo menos um usuário não se comunicar em LS, um intérprete deve obrigatoriamente participar das atividades. Vale destacar, que o intérprete deve pertencer a mesma comunidade dos demais membros, para evitar conflitos e ruídos de comunicação devido à gírias e aos regionalismos.

No processo de **cooperação**, o mediador deve supervisionar a discussão e quando houver a necessidade, auxiliar na criação de consenso. Durante a atividade, o discurso e as interações devem ser gravadas em um banco de dados como *discurso*.

Nós utilizamos o contexto da Fonologia baseado no processo CCKC (*Collaborative Consensus and Knowledge Creation*), utilizado para a construção colaborativa de consenso [61], para definir os **sinais isolados**. Esta abordagem interconecta o contexto, a colaboração e a computação para gerar uma base de sinais robusta, contextual e representativa.

6.3.3.3 Base de Sinais no CORE-SL

Abordagem. O CORE-SL consiste de um modelo baseado em fonemas *(phoneme-based model)* que segmenta os sinais em sub-unidades que são descritas em uma estrutura de dados. Como visto anteriormente, esta abordagem permite construir sistemas para o reconhecimento dos parâmetros, visando obter um serviço genérico de reconhecimento. Esta abordagem permite definir uma base de sinais representativa, desde que possa ser construída a partir de um conjunto de sinais que cobre (instancia) as sub-unidades do CORE-SL.

Descrição. Cada sinal isolado deve ser descrito por meio do CORE-SL, pois representa a estrutura e as regras computacionais para a descrição dos sinais.

Armazenamento e Recuperação. Cada sinal isolado deve ser armazenado em

uma base de sinais no final de cada atividade, incluindo o vídeo do sinal e a sua descrição utilizando o CORE-SL. Para o armazenamento, pode-se utilizar um sistema que permita a entrada de vídeos bem como uma descrição no CORE-SL, por exemplo, o sistema proposto por [4].

Para os experimentos nós utilizamos uma câmera de baixo custo semelhantes às *web-cams*, frequentemente utilizadas durante as atividades dos alunos surdos em sala de aula para video-chamadas e conversação.

Abordagem Min-Max. Um problema nas bases de sinais existentes é a falta de um método para determinar se o conjunto de sinais é representativo (cobre todas as possibilidades de criação de sinais). O objetivo é possibilitar esta completude por meio de um Conjunto Mínimo Máximo de Sinais (MMSS), que deve minimizar o número de sinais, mas deve maximizar as sub-unidades do modelo, visando reduzir a complexidade, o custo de treinamento e da aquisição dos sinais articulados por novos usuários.

A abordagem Min-Max é definida como: dado como a entrada um conjunto $E = \{e_1, e_2, ..., e_n\}$ das sub-uniades do CORE-SL, e um conjunto de sinais (dicionário) $S = \{s_1, s_2, ..., s_m\}$ onde cada sinal é descrito pela combinação de elementos de E. Encontre um conjunto $C \subseteq S$ tal que $|C|$ deve ser mínimo e seus elementos devem cobrir o máximo de sub-unidades de E.

O MMSS pode ser modelado na forma fo problema da Cobertura de Conjuntos - (*Set Cover Problem* - SCP). Desde que o SCP não possui um algoritmo exato para encontrar a solução ótima em tempo polinomial, utilizou-se um algoritmo de aproximação para a resolução do problema. Assim, um algoritmo guloso foi utilizado [23].

Algorithm 2 Estratégia Gulosa para o MMSS

```
1: function MMSS(E, S)
2:     C ← [ ]                                   ▷ C inicialmente é vazio
3:     while E ≠ [ ] and S ≠ [ ] do   ▷ repita até cobrir todas as sub-unidades de E
4:         escolha S' ∈ S, max(|S' ∩ E|)          ▷ S' deve cobrir o máximo de E
5:         E ← E − S'                    ▷ remover as sub-unidades de S' em E
6:         C ← C ∪ S'                    ▷ C armazena a solução de cada iteração
7:         S ← S − S'                                  ▷ remover S' de S
8:     return C                                        ▷ retorna a solução
```

6.3.3.4 Melhoramento Contínuo

A base de sinais construída pode ser melhorda de forma continuada. Isto pode ser feito por: a) aplicar o *framework* para outros grupos e outras atividades (mesma base); b) incluir o framework no sistema de recohecimento automático de sinais, que pode possibilitar uma ferramenta para cada usuário testar os sinais existentes na base e adicionar novos exemplos.

Neste serviço de reconhecimento, o sistema pode aplicar uma técnica de *feedback* de relevância e utilizar o conhecimento do usuário para avaliar o sistema de reconhecimento com os dados da base de sinais. Se necessário o usuário pode adicionar novos sinais ou descrições na base, que internamente podem ajudar na melhoria do sistema pelo uso desses dados como variações ou margens de erro.

Por exemplo, o sistema pode apresentar as seguintes alternativas de ações:

1. pode mostrar uma lista de sinais ao usuário e pedir para ele corrigir a descrição;
2. pode pedir ao usuário gravar um novo exemplo em vídeo por meio da *webcam*;
3. pode permitir a atualização da descrição por meio do modelo computacional;

Inicialmente, o sistema de reconhecimento pode ser treinado com a base de sinais. O MMSS também pode ser utilizado para definir um conjunto de treinamento e de teste se for necessário.

O sistema funciona por meio da entrada de um sinal. Então processa esta entrada, extrai as características do vídeo e modela a descrição no CORE-SL. Se a descrição retornar um conjunto de sinais de resultados, pode-se então solicitar ao usuário para avaliar a precisão do sistema e, se necessário, o usuário pode entrar com novos exemplos de sinais para ajudar a treinar o sistema - Figura 6.7.

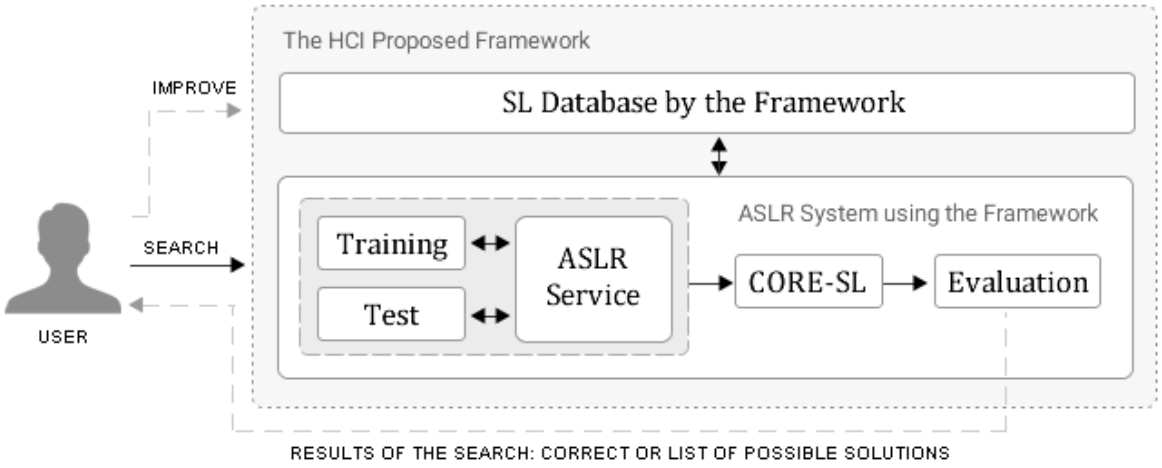


Figura 6.7: O framework proposto aplicado em um serviço de RAS
Fonte: O autor (2014)

6.4 Considerações

Este capítulo teve como objetivo apresentar alguns cenários de aplicação do CORE-SL dentro da Arquitetura HCI-SL, visando o suporte para a resolução de sub-problemas que envolvam o tratamento computacional das LS.

O capítulo faz um fechamento em relação ao uso do CORE-SL na Arquitetura HCI-SL, como abordagem conceitual e técnica para auxiliar o desenvolvimento de aplicações. Assim, além de realizar um estudo sobre os níveis conceitual, formal, interno e externo do CORE-SL, um nível de aplicação ou de uso também foi discutido, mostrando a capacidade de expressividade e de aplicação do modelo proposto.

É importante destacar que as discussões apresentadas neste capítulo foram disseminadas na Ciência da Computação por meio da publicação em diversos fóruns científicos internacionais, com o objetivo de validar estas abordagens propostas e contribuir cientificamente para a solução do problema central das LS.

No próximo capítulo são apresentados alguns resultados preliminares sobre a aplicação do CORE-SL e de suas metodologias em serviços da Arquitetura HCI-SL que estão em desenvolvimento pelo grupo de pesquisa.

CAPÍTULO 7

RESULTADOS

Neste capítulo são apresentados os resultados **primários**, relacionados aos objetivos da tese, os resultados **secundários**, que englobam os trabalhos de pesquisa paralelos que estão utilizando o CORE-SL como abordagem metodológica ou técnica, e resultados **adicionais** na forma das publicações realizadas.

7.1 Resultados Primários - Objetivos

Em relação aos objetivos propostos no Capítulo 1, faz-se uma sumarização dos resultados alcançados por esta tese:

1. Apresentação de um *framework* para a construção de um modelo computacional para a representação de sinais de LS, que agrega e discute propriedades computacionais necessárias para possibilitar o uso deste modelo na arquitetura de trabalho. Dentre essas propriedades, foram estudadas a universalidade do modelo proposto, a recuperabilidade dos sinais, o armazenamento, a reproduzibilidade, dentre outras;

2. Estudo amplo de modelos linguísticos, computacionais e baseados em gestos como forma de entender como os sinais das LS são formados e das possibilidades em relação ao movimento humano. Este conhecimento foi formalizado e utilizado para a construção do CORE-SL e pode auxiliar os trabalhos futuros devido à quantidade de modelos abordados;

3. O desenvolvimento de um processo para a construção de um modelo formal de representação de sinais. Este processo foi aplicado e utilizado para o desenvolvimento do nível formal do CORE-SL. Adicionalmente, o formalismo permitiu descrever diversas propriedades formais tais como a unicidade de representação, a completude e a extensibilidade para outros níveis gramaticas de uma LS.

4. Apresentou-se também uma discussão sobre a usabilidade e a legibilidade de um modelo computacional de representação. Este foi um diferencial desta tese, pois diferentemente dos demais trabalhos da literatura, buscou-se especificar um padrão de documentação que proporcione a facilidade de aprendizado e de uso do modelo por seus usuários. O balanceamento entre as questões computacionais formais e a experiência dos usuários finais foi um grande desafio e o CORE-SL conseguiu proporcionar uma facilidade de entendimento para ambos os extremos.

5. O estudo das propriedades do nível externo ao CORE-SL. Devido a Arquitetura HCI-SL ser orientada à serviços, estudou-se no nível externo do CORE-SL, bem como em outros nívels, estratégias computacionais para facilitar a integração e o uso do modelo em outros serviços que necessitem das funções de um modelo de representação. Neste capítulo discutiram-se estratégias para a alta disponibilidade de uma API para o CORE-SL, bem como as questões de desempenho e de acessibilidade da API.

6. Discussão sobre o problema da busca e da indexação dos sinais. Como resultado desta discussão, apresentou-se uma visão geral sobre o problema da busca no contexto da Arquitetura HCI-SL e definiu-se o conceito de busca por similaridade e funções de distância. O resultado mais importante foi o estudo e a proposta de uma estratégia de indexação de sinais baseada em *clusters*. A estratégia proposta combinou propriedades como eficiência de processamento em um grande conjunto de dados, baixo consumo de recursos computacionais e a escalabilidade do sistema para manter o mesmo desempenho. Adicionalmente, o capítulo discutiu sobre o uma possível adaptação da Distância de Edição para o problema da busca de sinais por similaridade.

7. Como último resultado primário, analisou-se a Arquitetura HCI-SL e apresentaram-se dois cenários de aplicação do CORE-SL, em seu nível de uso, como abordagem conceitual e metodológica para auxiliar no processo de desenvolvimento dos serviços relacionados a estes cenários. Um dos contextos discutidos foi o problema do reconhecimento automático de sinais. Esta área de pesquisa utiliza abordagens tradicionais que não têm gerado resultados para o usuário final. Neste caso, sugeriu-se a utilização do CORE-SL como abordagem conceitual para o desenvolvimento de um sistema de reconhecimento. As técnicas algorítmicas continuam as mesmas, mas o processo de desenvolvimento agora utiliza o CORE-SL como uma metodologia alternativa às abordagens tradicionais.

7.2 Resultados Secundários

Durante o desenvolvimento desta tese, o autor explorou o uso do CORE-SL de maneira dinâmica nas pesquisas desenvolvidas pelos demais membros do grupo de pesquisa. Em relação à Arquitetura HCI-SL, alguns dos trabalhos que foram ou estão sendo desenvolvidos com o uso do CORE-SL como abordagem conceitual, metodológica ou técnica são sumarizados a seguir.

7.2.1 Reconhecimento das CM da Libras por Malhas 3D

A pesquisa de mestrado de Porfilio (2013) [92] utilizou a abordagem baseada em fonema (*phoneme-based model*), que o CORE-SL segue para a implementação de um sistema para o reconhecimento das configurações de mão da Libras por meio de malhas 3D.

O trabalho baseou-se a estrutura do CORE-SL em relação às configurações de mão e especifica um conjunto de características a serem reconhecidas pelo sistema.

O trabalho de Porfilio (2013) [92] utilizou como tecnologia um sensor de profundidade como forma de analisar se este tipo de tecnologia pode facilitar a implementação de sistemas de reconhecimento automático de sinais voltados ao usuário final.

Porfilio (2013) [92] desenvolve sua pesquisa em quatro etapas principais:

1. a construção de uma base de sinais com usuários nativos da Libras, apoiado pela estrutura e a metodologia colaborativa do CORE-SL;
2. a reconstrução das configurações de mão da Libras em malhas tridimensionais;
3. a extração de características para distinção das configurações;
4. a classificação de 61 CM da Libras.

Os resultados indicaram um a eficiência na estratégia de reconhecimento proposta e mostraram um primeiro passo para o desenvolvimento de um serviço para o reconhecimento automático de sinais para os usuários finais no contexto da Arquitetura HCI-SL.

7.2.2 Avatar 3D para a Síntese de Sinais da Libras

O trabalho de mestrado de Gonçalves (2012) [58] estuda e desenvolve um avatar 3D para a síntese automática de sinais da Libras, baseado na interpretação da versão em desenvolvimento do CORE-SL. O objetivo do trabalho foi interpretar descrições de sinais feitas por meio do CORE-SL em um formato XML e reproduzí-las em um avatar 3D.

O primeiro passo da pesquisa de Gonçalves (2012) [58] consistiu na modelagem de um Avatar 3D que simulasse os pontos de contato e as articulações do corpo humano, para conseguir representar de maneira realista as configurações de mãos e os movimentos humanos.

Em seguida, Gonçalves (2012) [58] realizou o mapeamento dos parâmetros do CORE-SL em um interpretador, que teve o objetivo de transformar os parâmetros do CORE-SL em comandos para a execução automática de sinais por meio do Avatar 3D. Após este processamento, o Avatar pode realizar a sinalização. A Figura 7.1 apresenta um exemplo de movimento do braço por meio do Avatar 3D desenvolvido.

O trabalho também gerou resultados importantes para a geração de sinais em uma LS de maneira automática, o que pode possibilitar o desenvolvimento de diversos outros

cenários da Arquitetura HCI-SL que necessitam de um Avatar 3D como um mecanismo para gerar a saída de dados. Por exemplo, dicionários e tradutores.

Figura 7.1: Exemplo de Avatar 3D sintetizado a partir do CORE-SL
Fonte: Gonçalves (2012) [58]

7.2.3 Parser para a Geração de Símbolos do SignWriting

A pesquisa de Iatskiu (2014) [68] teve o intuito de gerar um serviço web para a síntese dos símbolos gráficos do SignWriting a partir de uma descrição realizada no CORE-SL por meio de um formato XML. O objetivo do sistema foi, assim como a Avatar 3D, interpretar a descrição feita por meio do CORE-SL e gerar um determinado sinal da Libras por meio da escrita de sinais: o SignWriting.

A Figura 7.2 apresenta um esquema de funcionamento do sistema. O serviço recebe uma descrição no CORE-SL em um formato XML, faz um processamento deste arquivo e extrai as informações principais.

Em um segundo momento o serviço faz um mapeamento interno na forma de um *parsing*, extraindo os dados da descrição do modelo computacional e mapeando os símbolos gráficos do SignWriting para apresentar a saída de maneira escrita.

7.3 Lista de Publicações

Nesta seção são listadas as publicações realizadas no período de doutoramento, que permitiram o compartilhamento das hipóteses e dos resultados encontrados no desenvolvimento do CORE-SL. Estas publicações também possibilitaram o cumprimento de mais um objetivo específico, que consistiu na disseminação do conhecimento gerado para as áreas de pesquisa envolvidas.

Figura 7.2: Exemplo de *Parsing* do CORE-SL para o SignWriting
Fonte: Iatskiu (2014) [68]

A seguir são apresentadas as publicações referentes ao CORE-SL e à Arquitetura HCI-SL:

- **(2015)** - A Context-Based Collaborative Framework to Build Sign Language Databases by Real Users. *Universal Access in Human-Computer Interaction. Access to Interaction, Volume 9176, Lecture Notes in Computer Science, pp 327-338*

- **(2015)** - The Low Use of SignWriting Computational Tools from HCI Perspective. *Universal Access in Human-Computer Interaction. Access to Interaction, Volume 9176, Lecture Notes in Computer Science, pp 373-382.*

- **(2014)** - A CPML-Signwriting Interpreter: A New form to Generate the Graphical Symbols of Signwriting. *Universal Access in Human-Computer Interaction. Design and Development Methods for Universal Access, Volume 8513, Lecture Notes in Computer Science, pp 357-368*

- **(2013)** - HCI Architecture for Deaf Communities Cultural Inclusion and Citizenship. *15th International Conference on Enterprise Information Systems (ICEIS 2013)*

- **(2013)** - Information Challenges of the Deaf in their Health and Social Care Needs. *Book Chapter on Handbook of Research on ICTs for Human-Centered Healthcare and Social Care Services.*

- **(2013)** - Corpus of Semiotic Analysis of Sign Language. *IADIS International Conference - WWW Internet 2013.*

- **(2013)** - Collaborative Consensus and Knowledge Creation: Computer-Mediated Methodology for Sign Language Studies. *Information Systems, E-learning, and Knowledge Management Research, Volume 278, Communications in Computer and Information Science, pp 278-292*

- (**2013)** - Communication and Cooperation Pragmatism: An Analysis of a Community of Practice by Non-deaf and Deaf to Study Sign Language. *Information Systems, E-learning, and Knowledge Management Research, Volume 278, Communications in Computer and Information Science, pp 191-205*

- (**2012)** - Challenges of knowledge management and creation in communities of practice organisations of Deaf and non-Deaf members: requirements for a Web platform. *Behaviour & Information Technology - Taylor & Francis*

- (**2011)** - A Framework to Support Development of Sign Language Human-Computer Interaction: Building Tools for Effective Information Access and Inclusion of the Deaf. *IEEE, RCIS 2011.*

- (**2011**) - Evaluation of a Computational Description Model of Libras (Brazilian Sign Language): Bridging the Gap Towards Information Access. *IEEE, RCIS 2011.*

- **(2011)** - Empowering Collaboration Among the Deaf: Internet-Based Knowledge Creation System. *IADIS International Conference on WWW/Internet.*

CAPÍTULO 8

CONCLUSÕES

O presente trabalho foi motivado pela necessidade do desenvolvimento de ferramentas para as comunidades de surdos que impulsionem de fato a sua acessibilidade de comunicação, na qual a interação e o acesso ao conhecimento sejam mediados pela LS. Neste sentido, este trabalho apresentou um estudo para o desenvolvimento do CORE-SL, um modelo computacional para a representação de sinais.

Como apresentado, esta tese foi desenvolvida com base no contexto da Arquitetura Computacional HCI-SL, que tem o intuito de desenvolver tecnologias que consideram as necessidades dos surdos e buscam um claro entendimento sobre as LS, para promover um tratamento computacional adequado da língua. Nesta arquitetura, uma dependência clara em relação aos seus serviços e entre as suas três camadas foi a necessidade de uma estrutura para a representação computacional dos sinais.

Os modelos computacionais para representação de sinais revisados na literatura, tem apresentado soluções para problemas muito específicos, não realizando estudos mais amplos quanto a aplicabilidade do modelo como subsídio para auxiliar a solução do problema global: o tratamento computacional das LS.

Além disso, estes modelos não exploram sua estrutura em relação às propriedades importantes como a unicidade de representação, a usabilidade, a indexabilidade e uma busca eficiente dos sinais. A maioria dos trabalhos relacionados também são específicos para problemas de síntese (geração) de sinais, mas não estudam quais propriedades que um modelo deve conter e como pode ser utilizado para auxiliar na solução de outros problemas, como reconhecimento, PLN, entre outros.

O desenvolvimento do CORE-SL buscou preencher esta lacuna, apresentando uma proposta de um framework para a construção do modelo computacional de representação, que possui seis níveis: contextual, conceitual (árvore conceitual), formal, interno, externo e de aplicação.

Além do desenvolvimento do modelo, cada nível do CORE-SL foi discutido, bem como as propriedades específicas que um modelo computacional de representação deve incluir para possibilitar uma aplicação e integração nos cenários de uso da Arquitetura HCI-SL.

No nível conceitual, o CORE-SL descreveu toda a metodologia relacionada ao estudo dos modelos linguísticos e computacionais e apresentou um modelo conceitual que permite ao usuário final entender os conceitos (parâmetros) e a relação entre eles. Como propriedades formais, o nível conceitual discutiu sobre a completude (na qual o modelo computacional pode representar qualquer sinal de uma certa língua) e a extensibilidade

(que permite que o CORE-SL possa ser estendido aos demais níveis gramaticais).

Esta última propriedade é fundamental, uma vez o processo pode ser aplicado novamente e novos níveis gramaticais podem ser incluídos no CORE-SL tais como a morfologia, a sintaxe e a semântica. Para isto, o CORE-SL foi desenvolvido com uma abordagem baseada em componentes, permitindo que novos componentes sejam inseridos sem afetar a estrutura dos componentes em uso, proporcionando facilidade e um alto grau de flexibilidade.

No nível formal, o CORE-SL apresentou um formalismo para a representação das regras de produção para a descrição dos sinais. Neste nível, foram discutidas as propriedades de unicidade (não-ambiguidade), a segmentabilidade, a sequencialidade e a simultaneidade que impactam na forma e na corretude de como um sinal pode ser representado computacionalmente.

No nível físico discutiram-se e apresentaram-se estratégias técnicas para o armazenamento dos sinais na Arquitetura HCI-SL. Neste nível foram estudadas propriedades importantes tais como a disponibilidade da informação, a eficiência e o desempenho em relação ao acesso dos dados.

O nível externo do CORE-SL discutiu propriedades como a universalidade do modelo, a legibilidade e a usabilidade para possibilitar a facilidade de uso e de aprendizado, a recuperabilidade e a reproduzibilidade que permitem que o modelo seja reproduzido e, também, a descrição de acessibilidade em relação à API das funções computacionais básicas do CORE-SL.

Portanto, além do desenvolvimento do CORE-SL em si, a tese apresentou um conjunto de processos, metodologias e discussões computacionais sobre os diversos aspectos técnicos relacionados à representação dos sinais, proporcionando resultados importantes para a continuidade das pesquisas na Arquitetura HCI-SL.

No Capítulo 5, foi estudado o problema de indexação e da busca de sinais que tem um grande impacto e aplicabilidade na arquitetura de trabalho. Neste sentido, foi estudada e apresentada uma estratégia eficiente para a indexação dos sinais baseada em *clusters*. Adicionalmente, foram apresentadas algumas alternativas técnicas para a implementação da solução.

Uma limitação deste estudo foi a falta de definição das operações de transformação para a adaptação da distância de edição ao contexto do CORE-SL. O autor desta tese entendeu que existe a necessidade de um estudo específico sobre estas operações de similaridade por um linguista, que tem o papel de identificar quais as operações relevantes podem ser utilizadas e quais parâmetros podem ser considerados.

Adicionalmente, o Capítulo 6 apresentou um estudo específico sobre dois cenários na Arquitetura HCI-SL na qual o CORE-SL pode contribuir, além da sua estrutura de dados, com uma abordagem metodológica e conceitual, apresentando resultados relevantes na prática.

E para finalizar, no Capítulo 7 apresentaram-se alguns dos resultados parciais em relação aos objetivos da tese, ao uso do CORE-SL na prática em trabalhos relacionados do Grupo de Pesquisa e algumas publicações relacionadas ao modelo desenvolvido.

Destaca-se que esta tese conseguiu alcançar seus objetivos propostos, de estudar e propor um framework para a construção de um modelo computacional de representação; de entender como os sinais são compostos nas LS; como construir um modelo computacional que agregue certas propriedades; e como aplicar o modelo desenvolvido na arquitetura.

Algumas das limitações desta tese são elencadas na forma de trabalhos futuros, ou seja, quais os próximos passos em relação a este problema. Eles incluem:

- Fazer uma validação detalhada do CORE-SL com linguistas e modelar outros níveis gramaticais necessários à arquitetura;
- Documentar todo o CORE-SL conforme o padrão desenvolvido nesta tese para facilitar seu uso e o seu aprendizado;
- Implementar a arquitetura do CORE-SL em relação às operações de CRUD, bem como a estratégia de indexação e de busca;
- Descrever sinais da Libras e de outras LS para formar uma base de sinais e levantar conjuntos específicos como configurações de mão que são específicas de cada LS;
- Elaborar uma interface para facilitar a entrada de sinais no CORE-SL, por exemplo, por meio da manipulação de um avatar 3D;
- Auxiliar os trabalhos da Arquitetura HCI-SL para o uso técnico e conceitual do CORE-SL;

REFERÊNCIAS BIBLIOGRÁFICAS

[1] ISO/IEC 14977. Syntactic Metalanguage – Extended BNF. *ISO - International Organization for Standardization) and IEC - International Electrotechnical Commission*, 1:12, 1996.

[2] Gregory D Abowd, Anind K Dey, Peter J Brown, Nigel Davies, Mark Smith, e Pete Steggles. Towards a Better Understanding of Context and Context-Awareness. *Handheld and Ubiquitous Computing*, páginas 304–307. Springer, 1999.

[3] Mark Ackerman, Trevor Darrell, e Daniel J Weitzner. Privacy in context. *Human–Computer Interaction*, 16(2-4):167–176, 2001.

[4] Diego R. Antunes. Um Modelo de Descrição Computacional da Fonologia da Língua de Sinais Brasileira. Dissertação de Mestrado, Pós-Graduação em Informática, Universidade Federal do Paraná, 2011.

[5] Diego R. Antunes, Cayley Guimarães, Laura Sánchez García, Luiz Eduardo S. Oliveira, e Sueli Fernandes. A Framework to Support Development of Sign Language Human-Computer Interaction: Building Tools for Effective Information Access and Inclusion of the Deaf. *IEEE RCIS 2011. Proceedings of the Fifth IEEE International Conference on Research Challenges in Information Science*, páginas 126–137, Guadeloupe, France, 2011.

[6] Diego R. Antunes, Cayley Guimarães, Daniela G. Trindade, Rafaella A. L. Silva, Laura Sánchez García, e Sueli Fernandes. Evaluation of a Computational Description Model of Libras (Brazilian Sign Language): Bridging the Gap Towards Information Access. *IEEE RCIS 2011. Proceedings of the Fifth IEEE International Conference on Research Challenges in Information Science*, páginas 485–494, Guadeloupe, France, 2011.

[7] Brava Autonomia. Fone fácil. http://www.bravaautonomia.com.br/fonefacil/, Janeiro de 2013.

[8] C. A. Baker. *Microanalysis of the Nonmanual Components of Questions in American Sign Language*. Tese de Doutorado, University of California, Berkeley, 1983.

[9] L Barros. *Suporte a Ambientes Distribuídos para Aprendizagem Cooperativa*. Tese de Doutorado, COPPE/UFRJ, Rio de Janeiro, 1994.

[10] R. Battison. Phonological Deletion in American Sign Language. *Sign Language Studies*, 5:1–19, 1974.

[11] R. Battison. *Analyzing Signs*. In: Valli, C. & C. Lucas (org.) *Linguistics of American Sign Language: an introduction*. Washington, D. C.: Clerc Books / Gallaudet University Press (2002), 1978.

[12] Brasil. Lei n. 10436, de 24 de abril de 2002. Dispõe sobre a Libras - Língua Brasileira de Sinais e dá outras providências. *Diário Oficial da República Federativa do Brasil*, Abril de 2002.

[13] D. Brentari. *A Prosodic Model of Sign Language Phonology*. A Bradford book. MIT Press, 1998.

[14] D. Brentari. Handshape in sign language phonology. *The Blackwell Companion to Phonology. M. van Oostendorp C. Ewen, E. Hume, and K. Rice (ed.)*, páginas 195–222, Oxford, 2011. Blackwell Publishing Ltd.

[15] Lucinda Ferreira Brito. *Por uma Gramática de Línguas de Sinais*. UFRJ, Departamento de Linguística e Filosofia, 1995.

[16] F. C. Capovilla e W. D. Raphael. *Dicionário Enciclopédico Ilustrado Trilíngüe da Língua de Sinais Brasileira. Vol.I: Sinais de A à L*. Editora da Universidade de São Paulo, São Paulo, 2001.

[17] Fernando César Capovilla, Walkiria Duarte Raphael, e Aline Cristina L. Mauricio. *Novo Deit-Libras: Dicionário Enciclopédico Ilustrado Trilíngue da Língua de Sinais Brasileira (Libras) baseado em Linguística e Neuro-Ciências Cognitivas. Sinais de I a Z*, volume 1. São Paulo, 2009.

[18] Fernando César Capovilla, Walkiria Duarte Raphael, e Aline Cristina L. Mauricio. *Novo Deit-Libras: Dicionário Enciclopédico Ilustrado Trilíngüe da Língua de Sinais Brasileira (Libras) baseado em Linguística e Neuro-Ciências Cognitivas. Sinais de A a H*, volume 1. São Paulo, 2009.

[19] Edgar Chávez, Gonzalo Navarro, Ricardo Baeza-Yates, e José Luis Marroquín. Searching in Metric Spaces. *ACM Computing Surveys*, 33(3):273–321, setembro de 2001.

[20] N. Chomsky. *Knowledge of Language: Its Nature, Origin and Use*. Praeger Publishers, New York, 1986.

[21] Hinrich Schutze Christopher D. Manning, Prabhakar Raghavan. *An Introduction to Information Retrieval*. Cambridge University Press, England, 2009. Acessado em Março de 2015.

[22] Hyun-Sook Chung e Yilbyung Lee. MCML: Motion Capture Markup Language for Integration of Heterogeneous Motion Capture Data. *Computer Standards and Interfaces*, 26(2):113–130, 2004.

[23] V. Chvátal. A Greedy Heuristic for the Set-Covering Problem. *Mathematics of Operations Research*, 4(3):233–235, 1979.

[24] J. Clark, C. Yallop, e J. Fletcher. *Introduction to Phonetics and Phonology.* Blackwell, 3 edition, 2007.

[25] A. C. R. Costa e G. P. Dimuro. SignWriting and SWML: Paving the Way to Sign Language Processing. *Traitement Automatique des Langues Naturelles/Workshop on Natural Language Processing of Minority Languages and Small Languages, 2003, Batz-sur-Mer. Acte of TALN 2003. Nantes : Université de Nantes*, 2:193–202, 2003.

[26] Onno Crasborn, Harry van der Hulst, e Els van de Kooij. Phonetic and Phonological Distinctions in Sign Languages. *Intersign, Workshop*, 2, 2002.

[27] Luciana Vargas da Rocha, Nina Edelweiss, e Cirano Iochpe. GeoFrame-T: A Temporal Conceptual Framework for Data Modeling. *Proceedings of the 9th ACM International Symposium on Advances in Geographic Information Systems*, GIS '01, páginas 124–129, New York, NY, USA, 2001. ACM.

[28] D. Dallari. *Direitos Humanos e Cidadania.* Moderna, São Paulo, 1998.

[29] Evani de Carvalho Viotti. *Introdução aos Estudos Linguísticos.* Universidade Federal de Santa Catarina. Curso de Licenciatura em Letras-Libras, Florianópolis, 2008.

[30] R. M de Quadros e L. B. Karnopp. *Língua de Sinais Brasileira: Estudos Linguísticos.* Artmed, Porto Alegre, 2004.

[31] Victoria University of Wellington Deaf Studies Research Unit. NZSL Online: The Online Dictionary of New Zealand Sign Language. http://nzsl.vuw.ac.nz/, May de 2015.

[32] Maria del Puy Carretero, David Oyarzun, Amalia Ortiz, Iker Aizpurua, e Jorge Posada. Virtual characters facial and body animation through the edition and interpretation of mark-up languages. *Computers & Graphics*, 29(2):189–194, 2005.

[33] Anind K. Dey. Understanding and Using Context. *Personal and Ubiquitous Computing*, 5(1):4–7, 2001.

[34] W. M. do Amaral. *Sistema de Transcrição da Língua Brasileira de Sinais Voltado à Produção de Conteúdo Sinalizado por Avatares 3D.* Tese de Doutorado, Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação,

Departamento de Engenharia de Computação e Automação Industrial, Campinas, SP, 2012.

[35] Danilo Assis Nobre dos S. Silva, Tiago Maritan Ugulino de Araújo, Leonardo Dantas, Vandhuy F. Martins, Yúrika Sato Nóbrega, Hozana Raquel Gomes de Lima, e Guido Lemos de Souza Filho. A Formal Language to Describe and Animate Signs in Brazilian Sign Language. *SBC Journal on 3D Interactive Systems*, 3(2):16–26, 2012.

[36] D. Assis Nobre dos S.Silva, T.M.U. de Araujo, L. Dantas, Y.S. Nobrega, H.R.G. De Lima, e G.L. de Souza Filho. FleXLIBRAS: Description and Animation of Signs in Brazilian Sign Language. *Virtual and Augmented Reality (SVR), 2012 14th Symposium on*, páginas 227–236, 2012.

[37] Philippe Dreuw, Carol Neidle, Vassilis Athitsos, Stan Sclaroff, e Hermann Ney. Benchmark Databases for VideoBased Automatic Sign Language Recognition. 2008.

[38] U. ECO. *A theory of Semiotics*. IA: Indiana University Press, 1984.

[39] E. Efthimiou, S.-E. Fotinea, T. Hanke, J. Glauert, R. Bowden, A. Braffort, C. Collet, J. Maragos, e F. Goudenove. Dicta-sign: Sign language recognition, generation, and modelling: A research effort with applications in deaf communication proceedings of the language resources and evaluation. *Conference Workshop on the Representation and Processing of Sign Languages : Corpora and Sign Languages Technologies*, 2010.

[40] P. Ekman. *Facial signs: facts, fantasies and possibilities*. SEBEOK, T. (Ed.). Sight, sound and sense. Indiana University Press, 1978.

[41] Mauro Felice, Tania Mascio, e Rosella Gennari. A Visual Ontology-Driven Interface for a Web Sign Language Dictionary. Mathias Weske, Mohand-Saïd Hacid, e Claude Godart, editors, *Web Information Systems Engineering - WISE 2007 Workshops*, volume 4832 of *Lecture Notes in Computer Science*, páginas 429–440. Springer Berlin Heidelberg, 2007.

[42] Tanya A. Felipe. O Discurso VerboVisual na Língua Brasileira de Sinais - Libras. *Bakhtiniana: Revista de Estudos do Discurso*, 8:67–89, 2013.

[43] Tanya Amara Felipe. Escola Inclusiva e os direitos linguísticos dos surdos. *Revista Espaço*, 7:41–46, 1997.

[44] Tanya Amara Felipe. Introdução à Gramática da LIBRAS. *Giuseppe Rinaldi et al.Educação Especial Deficiência Auditiva*, number 4 in Atualidades Pedagógicas, Brasília, 1997. SEESP.

[45] Tanya Amara Felipe. Dicionario Digital da Lingua Brasileira de Sinais. http://www.acessobrasil.org.br/libras/, 2002.

[46] Tanya Amara Felipe. Sistema de Flexão Verbal na LIBRAS: Os classificadores enquanto marcadores de flexão de gênero. *Anais do Congresso Surdez e Pós-Modernidade: Novos rumos para a educação brasileira. I Congresso Internacional do INES. VII Seminário Nacional do INES*, páginas 37–58, 2002.

[47] Tanya Amara Felipe. Os Processos de Formação de Palavra na Libras. *ETD - Educação Temática Digital*, 7(2):199–216, Julho de 2006.

[48] Tanya Amara Felipe. Descrição da Língua de Sinais: desafios teóricos e práticos. *Anais do Congresso Internacional do INES*, 2007.

[49] S. F. Fernandes. Avaliação em língua portuguesa para alunos surdos: algumas considerações. 2007.

[50] S. F. Fernandes. Os Sotaques dos Sinais. *Língua Portuguesa*, 25:28–33, 2007.

[51] Sueli Fernandes. *Educação de Surdos*. Editora Ibpex, Curitiba, 2 edition, 2011.

[52] Susan D. Fischer e Harry van der Hulst. *Sign Language Structures*, capítulo 23, páginas 319–331. Oxford University Press, 2003.

[53] J. B. Freitas e A. C. R. Costa. SWDB: Um Sistema de Dicionários para as Línguas de Sinais usadas pelos Surdos. *Conferência Ibero-Americana WWW/Internet 2004, Madrid - Espanha. Anais do CIAWWWI*, 2004.

[54] L. Friedman. *Formational Properties of American Sign Language*. In. L. Friedman (ed.), On the Other Hand: New Perspectives on American Sign Language. New York: Academic Press, 1977.

[55] E. Fusco e J. R. F. Brega. X-Libras: Uma Proposta de Ambiente de Visualização para Aprendizagem da Língua Brasileira de Sinais. *I CIEPG - I Congresso Internacional de Educação de Ponta Grossa Paraná*, 2009.

[56] Laura S. García, Cayley Guimarães, Diego R. Antunes, e Sueli Fernandes. HCI Architecture for Deaf Communities Cultural Inclusion and Citizenship. *Proceedings of the 15th International Conference on Enterprise Information Systems - ICEIS 2013.*, volume 3, páginas 68–75, Angers, França, Julho de 2013.

[57] John Glauert, Richard Kennaway, Ralph Elliott, e Barry john Theobald. Virtual Human Signing as Expressive Animation. *University of Leeds*, páginas 98–106, 2004.

[58] Diego Addan Gonçalves. Avatar 3D para a Sintese Automática de Sinais da Lingua de Sinais Brasileira. Dissertação de Mestrado, Pós Graduação em Informática da Universidade Federal do Paraná, 2013.

[59] Mira Working Group. Evaluation Frameworks for Interactive Multimedia Information Retrieval Applications. http://www.dcs.gla.ac.uk/mira/, 1996.

[60] N.L. Guarinello. Cidades-Estado na Antiguidade Clássica. *História da Cidadania. Ed. Contexto*, 2003. Org. Jaime e Carla Pinsky.

[61] C.; Guimarães, D. R.; Antunes, S.; Fernandes, L. S.; García, e A. J. Miranda. Empowering Collaboration Among the Deaf: Internet-Based Knowledge Creation System. *IADIS WWW/Internet 2011 Conference. Proceedings of the IADIS International Conference on WWW/Internet*, páginas 137–144, 2011.

[62] Cayley Guimarães. *Arquitetura Pedagógica Computacional para Interações Intelectuais entre Crianças Surdas e Pais Não-Surdos em Libras e Português*. Tese de Doutorado, Pós-Graduação em Informática, Universidade Federal do Paraná, Curitiba, 2013.

[63] J. Hamill e K. M. Knutzen. *Biomechanical Basis of Human Movement*. Lippincott Williams & Wilkins, 2006.

[64] T. Hanke. HamNoSys Representing Sign Language Data in Language Resources and Language Processing Contexts. *Streiter, Oliver, Vettori, Chiara (eds): LREC 2004, Workshop proceedings : Representation and processing of sign languages*, páginas 1–6, Paris : ELRA, 2004.

[65] A. Herrmann e M. Steinbach. *Nonmanuals in Sign Language*. Benjamins Current Topics. John Benjamins Publishing Company, 2013.

[66] H. G. van der Hulst. Units in the Analysis of Signs. *Phonology*, 10(2):209–241, Agosto de 1993.

[67] H. G. van der Hulst. Notation Systems. *Diane Brentari (ed.). Cambridge Survey of Sign Linguistics and Sign Languges*, 2010.

[68] Carlos E. Andrade Iatskiu. Serviço Web para a Interpretação do Modelo Fonológico Computacional da Libras para os Símbolos Gráficos do SignWriting. Dissertação de Mestrado, Pós Graduação em Informática, Universidade Federal do Paraná, 2014.

[69] IBGE. Instituto Brasileiro de Geografia e Estatística. http://www.censo2010.ibge.gov.br/apps/mapa, 2014.

[70] Mohamed Jemni e Oussama Elghoul. A System to Make Signs Using Collaborative Approach. Klaus Miesenberger, Joachim Klaus, Wolfgang Zagler, e Arthur Karshmer, editors, *Computers Helping People with Special Needs*, volume 5105 of *Lecture Notes in Computer Science*, páginas 670–677. Springer Berlin / Heidelberg, 2008.

[71] R. Johnson e S. Liddell. Toward a Phonetic Representation of Signs: Sequentiality and Contrast. *Sign Language Studies*, 11(2):241–274, 2010.

[72] Daniel Jurafsky e James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall Series in Artificial Intelligence. Pearson Education Ltd, London, 2 edition, 2009.

[73] J.Watkins. *Structure and Function of the Musculoskeletal System*. Human Kinetics Publishers, 2010.

[74] L. B. Karnopp. *Aquisição Fonológica na Língua Brasileira de Sinais: Estudo Longitudinal de uma Criança Surda*. Tese de Doutorado, Prontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre, Janeiro de 1999.

[75] R. Elliott; J. Glauert; J. Kennaway e K. Parsons. D52: SiGML Definition. Technical Report working document. Relatório técnico, ViSiCAST Project, 2001.

[76] E. Klima e U. Bellugi. *The Signs of Language*. MA: Harvard University, 1979.

[77] Els Van Der Kooij. *Phonological Categories in Sign Language of the Netherlands - The Role of Phonetic Implementation and Iconicity*, volume 1. LOT, 2002.

[78] Stefan Kopp, Brigitte Krenn, Stacy Marsella, Andrew N. Marshall, Catherine Pelachaud, Hannes Pirker, Kristinn R. Thórisson, e Hannes Vilhjálmsson. Towards a common framework for multimodal generation: The behavior markup language. *International Conference on Intelligent Virtual Agents*, páginas 21–23, 2006.

[79] Alfred Kranstedt, Stefan Kopp, e Ipke Wachsmuth. MURML: A Multimodal Utterance Representation Markup Language for Conversational Agents. AAMAS 02 Workshop Embodied conversational agents - let's specify and evaluate them!, 2002.

[80] S. K. Liddell e R. E. Johnson. *American Sign Language: The Phonological Base*. In: Valli, C. & C. Lucas (org.) *Linguistics of American Sign Language: an introduction*. Washington, D. C.: Clerc Books / Gallaudet University Press (2002)., 1989.

[81] Gary Marchionini. Toward Human-Computer Information Retrieval. Bulletin. Disponível em http://www.asis.org/Bulletin/Jun-06/marchionini.html, Junho de 2006.

[82] A. Marriott. VHML - Virtual Human Markup Language. *Proceedings of Talking Head Technology Workshop*, 2001.

[83] A.M. Martinez, R.B. Wilbur, R. Shay, e A.C. Kak. Purdue RVL-SLLL ASL Database for Automatic Recognition of American Sign Language. *Multimodal Interfaces, 2002. Proceedings. Fourth IEEE International Conference on Multimodal Interfaces*, páginas 167–172, 2002.

[84] L. McCleary e E. Viotti. Transcrição de dados de uma língua sinalizada: Um estudo piloto da transcrição de narrativas na língua de sinais brasileira (lsb). *H.Salles (Org) Bilinguismo dos surdos: questões lingüísticas e educacionais*, páginas 73–96, Goiânia: Cânone Editorial, 2007.

[85] I. Meir, C. Padden, M. Aronoff, e W. Sandler. Body as subject. *Journal of Linguistics*, 43:531–63, 2007.

[86] Mikhaylyna Melnyk, Vira Shadrova, e Borys Karwatsky. Towards Computer Assisted International Sign Language Recognition System: A Systematic Survey. *International Journal of Computer Applications*, 89(17):44–51, March de 2014.

[87] Ruslan Mitkov. *The Oxford Handbook of Computational Linguistics (Oxford Handbooks in Linguistics S.)*. Oxford University Press, 2003.

[88] Kawagashkira Nobuyuki. Moraic Phonology. Linguist List. Disponível em http://www.ling.fju.edu.tw/phono/farrah/Moraic2013.

[89] David Novak. *Similarity Search on a Very Large Scale*. Tese de Doutorado, MASARYK UNIVERSITY, FACULTY OF INFORMATICS, 2008.

[90] J. D. Novak e Alberto J. Cañas. The Theory Underlying Concept Maps and How to Construct Them. Relatório técnico, Technical Report IHMC CmapTools. Pensacola, FL: Institute for Human and Machine Cognition, 2008.

[91] D. Perlmutter. On the Segmental Representation of Transitional and Bidirectional Movements in ASL Phonology. *The Theoretical Issues in Sign Language Research. The University of Chicago Press.*, 1990.

[92] Andres Jesse Porfirio. Reconhecimento das Configurações de Mão da Libras a partir de Malhas 3D. Dissertação de Mestrado, Pós-Graduação em Informática - Departamento de Informática - Universidade Federal do Paraná, 2013.

[93] Jenny Preece, Yvonne Rogers, e Helen Sharp. *Interaction Design*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 2002.

[94] S. Prillwitz, R. Leven, H. Zienert, T. Hanke, J. Henning, e Colleagues. HamNoSys Version 2.0. Hamburg Notation System for Sign Languages: An introductory Guide. *International Studies on Sign Language and the Communication of the Deaf*, páginas 195–278, 1989.

[95] ProDeaf. Prodeaf web. Disponível em http://web.prodeaf.net/CriarSinal, 2015.

[96] Wendy Sandler. *Phonological Representation of the Sign: Linearity and Nonlinearity in American Sign Language*. Dordrecht, 1989.

[97] Adam; Schembri, Jordan; Fenlon, Ramas; Rentelis, Sally; Reynolds, e Kearsy Cormier. Building the British Sign Language Corpus. *Language Documentation & Conservation*, 7:136–154, 2013.

[98] Andrew Sears e Julie A. Jacko. *The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies, and Emerging Applications (Human Factors and Ergonomics Series)*. L. Erlbaum Associates Inc., Hillsdale, NJ, USA, 2007.

[99] N. Shehabuddeen, N. Buddeen, D. Probert, R. Phaal, e K. Platts. Representing and approaching complex management issues: part 1 - role and definition. *Centre for Technology Management (CTM)*, 1999.

[100] C. Skliar. *A Surdez: Um Olhar Sobre a Diferença.* Mediação, Porto Alegre, 1 edition, 1999.

[101] W. C. Stokoe. *Sign Language Structure*. Silver Spring: Linstok Press., 1960/1978. Revisto em 1978, Silver Spring, M.D., Linstok Press.

[102] William C. Stokoe, Dorothy Casterline, e Carl Croneberg. *The Dictionary of American Sign Language on Linguistic Principles*. Silver Spring, MD: Linstok Press, Washington, DC, 1965 (Revised 1978).

[103] Thomas A. Sudkamp. *Languages and Machines: An Introduction to the Theory of Computer Science (3rd Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.

[104] T. Supalla e E. Newport. How Many Seats in a Chair? The Derivation of Nouns and Verbs in American Sign Language. *P. Siple, ed., Understanding Language through Sign Language Research*, páginas 91–132., New York, 1978. Academic Press.

[105] Valerie Sutton. SignWriting for Sign Languages. Disponível em http://www.signwriting.org/, Janeiro de 2013.

[106] D. F. G. Trindade. *InCoP: Um Framework Conceitual para o Design de Ambientes Colaborativos Inclusivos para Surdos e Não-Surdos de Cultivo a Comunidades de Prática*. Tese de Doutorado, UFPR, Informatics Program, Curitiba, PR, 2013.

[107] D. Tsichritzis e A. Klug. The ANSI/SPARC DBMS Framework Report of the Study Group on Database Management Systems. *Infosystems*, 3, 1978.

[108] L. Uyechi. Local and global signing space in American Sign Language. *NELS 24. Graduate Linguistic Student Association, University o Massachusetts, Amherst*, 1994.

[109] L. Uyechi. *The Geometry of Visual Phonology*. Tese de Doutorado, Stanford University, Stanford, Calif. [Publicado em 1996 por CSLI Publications, Stanford, Calif], 1995.

[110] Clayton Valli e Ceil Lucas. *Linguistics of American Sign Language: an introduction*. Washington, D. C.: Clerc Books / Gallaudet University Press, 3 edition, 2002.

[111] WFD. World Federation of the Deaf (WFD). Federação Mundial de Surdos. Disponível em http://www.wfdeaf.org, Novembro de 2013.

[112] R. B. Wilbur. *American Sign Language - Linguistic and applied dimensions*. Little, Brown, Boston, 1987.

[113] Niklaus Wirth. What can we do about the unnecessary diversity of notation for syntactic definitions? *Commun. ACM*, 20(11):822–823, novembro de 1977.

[114] André Nogueira Xavier. *Descrição Fonético-Fonológica dos sinais da língua de sinais Brasileira (LIBRAS)*. Dissertação de mestrado, Universidade de São Paulo. Departamento de Linguistica. Pós-graduação em Semiótica e Linguística Geral, São Paulo, 2006.

[115] Kejia Ye, Baocai Yin, e Lichun Wang. CSLML: A Markup Language for Expressive Chinese Sign Language Synthesis. *Computer Animation and Virtual Worlds*, 20(2-3):237–245, 2009.

[116] Beifang Yi. *A Framework for a Sign Language Interfacing System*. Tese de Doutorado, University of Nevada, Reno, 2006.

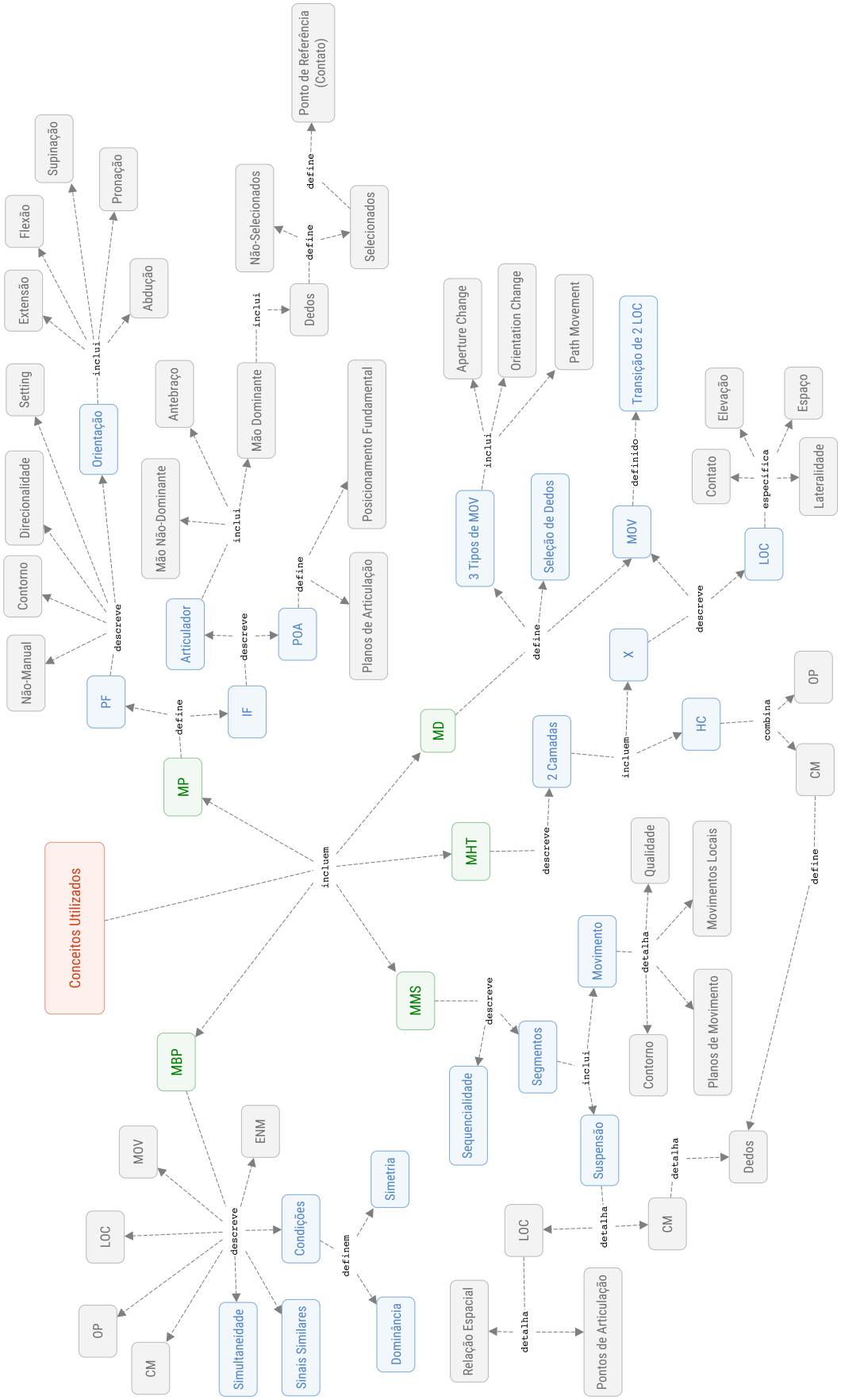
[117] Pavel Zezula, Giuseppe Amato, Vlastislav Dohnal, e Michal Batko. *Similarity Search: The Metric Space Approach*, volume 32 of *Advances in Database Systems*. Springer, 2006.

[118] Águeda A. F. Piedade Ramos. Parametrização da Estrutura de Dados Métrica RLC. Dissertação de Mestrado, Mestrado em Licenciada em Engenharia Informática. Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, 2012.

[119] Ângelo M. L. Sarmento. Estruturas de Dados Métricas Genéricas em Memória Secundária. Dissertação de Mestrado, Mestrado em Licenciada em Engenharia Informática. Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, 2010.

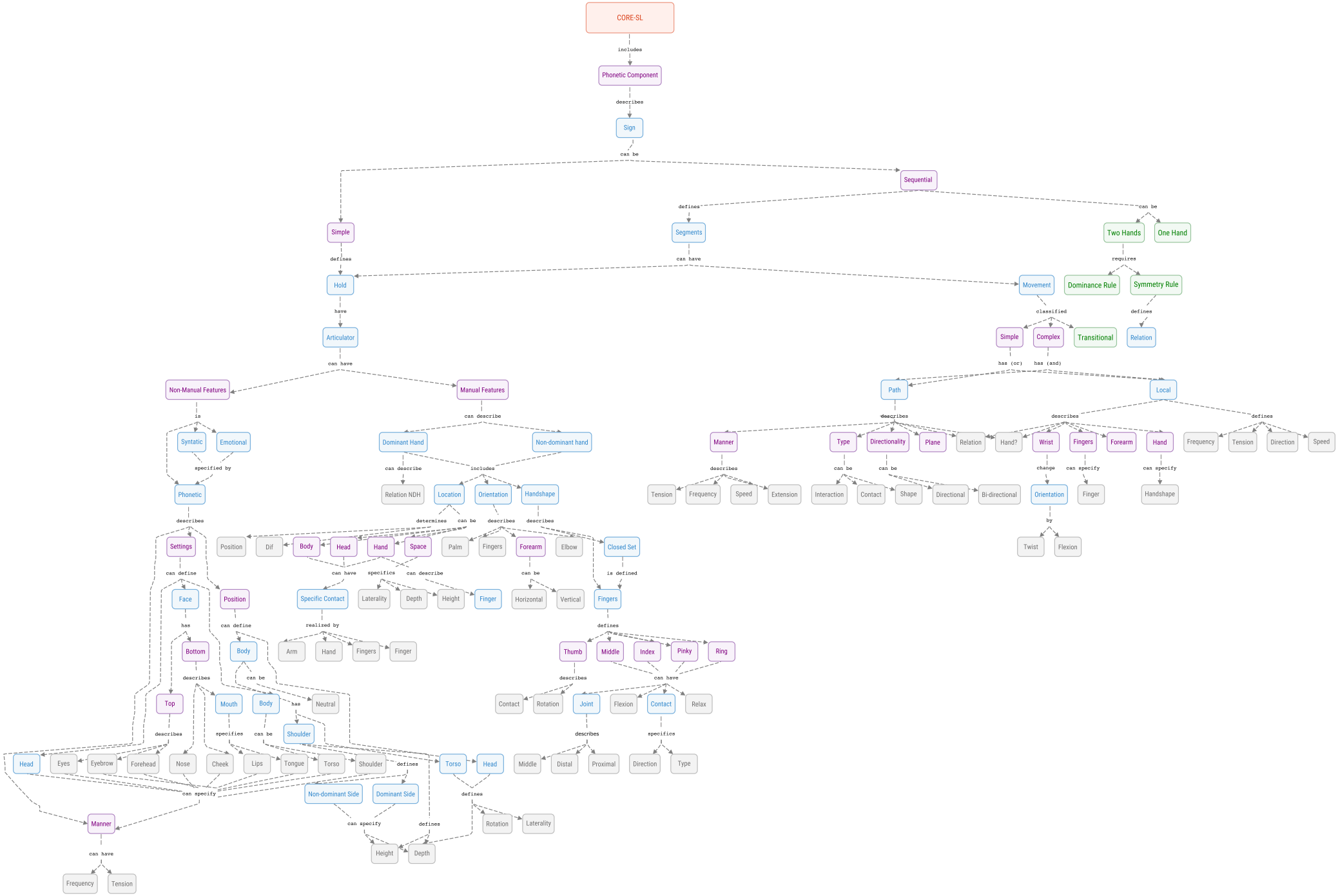
APÊNDICE A

MAPA CONCEITUAL DOS MODELOS FONOLÓGICOS


Conceitos Utilizados
incluem
MP
MD
MHT
MMS
MBP
define
PF
IF
descreve
Não-Manual
Contorno
Direcionalidade
Setting
Orientação
inclui
Supinação
Pronação
Flexão
Extensão
Abdução
Articulador
POA
Mão Não-Dominante
Antebraço
Mão Dominante
Planos de Articulação
Posicionamento Fundamental
Dedos
Não-Selecionados
Selecionados
Ponto de Referência (Contato)
3 Tipos de MOV
Aperture Change
Orientation Change
Path Movement
Seleção de Dedos
2 Camadas
X
MOV
LOC
HC
incluem
combina
OP
CM
Transição de 2 LOC
definido
Contato
Elevação
Espaço
Lateralidade
especifica
descreve
Qualidade
Movimentos Locais
Movimento
Contorno
Planos de Movimento
detalha
Sequencialidade
Segmentos
Suspensão
LOC
CM
Dedos
Relação Espacial
Pontos de Articulação
ENM
MOV
LOC
OP
CM
Condições
Simetria
Dominância
Simultaneidade
Sinais Similares
definem

APÊNDICE B

MAPA CONCEITUAL DO CORE-SL


CORE-SL
includes
Phonetic Component
describes
Sign
can be
Sequential
defines
Segments
can be
Two Hands
One Hand
Simple
defines
Hold
have
Articulator
can have
Non-Manual Features
Manual Features
is
Syntatic
Emotional
specified by
Phonetic
describes
Settings
Position
Dif
Body
Head
Hand
Space
Palm
Fingers
Forearm
Elbow
Closed Set
can define
Face
Position
has
Bottom
Body
describes
Top
Mouth
Body
Neutral
can be
has
Shoulder
Head
Eyes
Eyebrow
Forehead
Nose
Cheek
Lips
Tongue
Torso
Shoulder
defines
Torso
Head
Non-dominant Side
Dominant Side
can specify
Manner
can have
Frequency
Tension
can describe
Dominant Hand
Non-dominant hand
includes
Relation NDH
Location
Orientation
Handshape
determines
can be
describes
describes
can have
specifics
can describe
Specific Contact
Laterality
Depth
Height
Finger
can be
Forearm
Horizontal
Vertical
is defined
Fingers
realized by
Arm
Hand
Fingers
Finger
defines
Thumb
Middle
Index
Pinky
Ring
describes
can have
Contact
Rotation
Joint
Flexion
Contact
Relax
describes
specifies
describes
Middle
Distal
Proximal
specifies
Direction
Type
Rotation
Laterality
defines
Height
Depth
requires
Two Hands
Dominance Rule
Symmetry Rule
Movement
defines
Relation
classified
Simple
Complex
Transitional
has (or)
has (and)
Path
Local
describes
defines
Manner
describes
Tension
Frequency
Speed
Extension
Type
Directionality
Plane
Relation
can be
can be
Interaction
Contact
Shape
Directional
Bi-directional
Hand?
Wrist
Fingers
Forearm
Hand
change
can specify
can specify
Orientation
Finger
Handshape
by
Twist
Flexion
Frequency
Tension
Direction
Speed

APÊNDICE C

FORMALIZAÇÃO DO CORE-SL

CORE-SL

core-sl

```
core_sl = (phonetic_component, morphologic_component?);
phonetic_component = identifier, sign;
```

---

sign

```
sign = ('simple', hold | 'sequential', (segment+), hold, | 'spelling', hold+);
segment = hold, movement;
```

---

hold

```
hold = 'articulator', (manual | non_manual | manual, non_manual);
```

---

manual

```
manual = 'one hand', dominant_hand
        | 'two hands', relation_non_dominant?, dominant_hand, non_dominant_hand;

relation_non_dominant = 'up' | 'down' | 'front' | 'back' | ... ;
```

---

symmetry

```
symmetry_rule = 'true', symmetry_relation | 'false', 'passive'
symmetry_relation = 'symmetry' | 'crossed_fingers' | 'palm_palm' | ... ;
```

---

hands

```
non_dominant_hand = symmetry_rule | dominant_hand;
dominant_hand = handshape, orientation, location;

handshape = closed_set | fingers;
```

---

fingers

```
fingers = thumb, ('index',finger), ('middle',finger), ('ring',finger),('pinky',finger);

thumb = thumb_rotation, thumb_contact?;
thumb_rotation = ('lateral' | 'parallel'), thumb_flexion;
thumb_flexion = 'opened' | 'closed' | 'flexed' | 'curved';
```

```
thumb_contact = 'fingertips' | 'digital' | 'nail in digital' | 'digital in nail';

finger = relax, finger_contact, (finger_flexion | joint);
relax = 'true' | 'false';
finger_flexion = 'opened' | 'closed' | 'flexed' | 'curved' | 'flexed (proximal joint'
               | 'flexed (middle joint)' | 'flexed (distal joint)';
finger_contact = contact_type, contact_direction;
contact_type = 'stacked' | 'spread' | 'lateral' | 'crossed' | 'neutral';
contact_direction = 'thumb' | 'pinky' | 'neutral';
```

---

joints

```
joint = joint_proximal, joint_medial, joint_distal;
joint_proximal = 'opened' | 'closed' | 'flexed' | 'curved';
joint_medial = joint_proximal;
joint_distal = joint_proximal;
```

---

orientation

```
orientation = or_palm, or_fingers, or_forearm, or_elbow;

or_elbow = ('lateral' | 'depth'),
           ("0° (body parallel)" | "45° (medial)" | "90° (extended)" | "135° (high)"l

or_fingers = directions, rotation_dif?;
or_palm = directions;

directions = 'front' | 'back' | 'left' | 'right' | 'up' | 'down';
rotation_dif = 'clockwise' | 'counterclockwise';

or_forearm = horizontal | vertical;
horizontal =  '0° (dominant-hand side)' | '45° (diagonal)' | '90° (front)'
            | '135° (diagonal)'
            | '180° (non-dominant-hand side)';
vertical = '90° (up)', '45° (lateral, dominant-hand)' | '45° (diagonal, dominant-hand')
         | '45° (depth, forward)'
         | '45° (diagonal, non-dominant-hand')
         | '45° (lateral, non-dominant-hand)';
```

---

location

```
location = dif?, position, (head | body | hand | space), specific_contact?;
dif = 'up' | 'down' | 'left' | 'right';
position = 'dominant hand' | 'non-dominant hand';

hand = loc_finger | ('palm' | 'back' | 'side' | 'side (thumb)' | 'side (pinky)'
     | 'fingers' | 'fingers (back)' | 'fingers (tip)' | 'base' | 'wrist');

loc_finger = ('thumb' | 'index' | 'middle' | 'ring' | 'pinky'),
            ('palm' | 'back' | 'side (thumb)' | 'side (pinky)' | 'tip' | 'base' | 'nail');

body = 'crook of his arm' | 'navel' | 'elbow' | 'pulse' | 'thigh' | 'abdomen'
     | 'inner side of the arm' | 'outside of the arm' | 'forearm' | 'arm - trícepis'
     | 'waist' | 'chest' | 'stomach' | 'neck' | 'shoulder' | 'leg' | 'waist';

head = 'eyebrow' | 'chin' | 'teeth' | 'cheek' | 'lower lip' | 'upper lip' | 'lips'
     |' tongue '|' tip tongue '|' mouth '|' nose '|' nose side '|' nose tip '|' eyes '
     |' ear '|' lower face '|' top face '|' face '|' tests '|' side of forehead '
     |' head '|' top head ';

specific_contact = ('finger', ('index' | 'middle' | 'ring' | 'pinky')
                 | 'fingers' | 'hand' | 'arm'),
                   ('side (thumb)' | 'side (pinky) | 'back' | 'palm' | 'base' | 'dorso'
                 | 'tip' | 'nail', 'elbow');

space = laterality, height, depth;
depth = 'proximal' | 'distal' | 'medial' | 'extended';
height = 'head' | 'hand' | 'forehead' | 'eye' | 'eyebrow' | 'nose' | 'mouth'
        | 'chin' | 'cheek' | 'ear' | 'side of the head' | 'neck' | 'chest'
        | 'shoulder' | 'stomach' | 'abdomen';
laterality = 'distal in space sideways' | 'parallel to shoulder' | 'parallel to chest'
           | 'parallel to midline';
```

---

non-manual features

```
non_manual = phonetic | emotional | syntactic;

phonetic = manner_exp?, settings, position?;
```

```
settings = body?, head?, face?;

head = 'inclination left and right'|'inclination back''|'inclination to the right side'
     | 'inclination to the left side'|'bending forward'|'rotate sideways (not)'|'
     | 'balancing forward and back (yes)';

face = top | bottom | (top, bottom);

top = forehead?, eyes?, eyebrow?;
forehead = 'furrowing';
eyes = 'closed' | 'semi-closed' | 'blink' | 'wide eyes' ;
eyebrow = 'furrowing' | 'raised - left' | 'raised - right';

bottom = cheek?, nose?, mouth?;
cheek = 'inflated cheeks and blowing'|'blowing'|'inflated cheeks'|'breathe out'
      | 'cheek contracted' | 'cheek contracted left'|'cheek contracted right'
      | 'inflated cheek left' | 'inflated cheek right';
nose = 'furrowing';
mouth = tongue?, lips? | ('open' | 'closed' | 'semi opened');
tongue = 'rub the teeth' | 'show the tip' | 'showing the tongue'|
       | 'pressing the tongue in cheek'|'move the tongue in cheek'
       | 'open mouth, tongue on the upper teeth';
lips = 'bite the upper lip' | 'bite the lower lip' | 'tight lips' | 'opened lips';



manner_exp = frequency, tension;
```

---

movement

```
movement = simple | complex;
simple = local | path;
complex = local, path;
```

---

local movement

```
local = hand?, symmetric_movement? , direction?, frequency?, speed?, tension?,
        mov_hand?, mov_wrist?, mov_forearm?, mov_fingers?;

mov_hand = ('rub' | 'vibration') | ('variation', handshape);
mov_wrist = wrist_twist | wrist_flexion;
```

```
mov_forearm = 'rotation' | sideways swing' | 'forward swing' | 'twist';
mov_fingers = finger_spec+ | mov_fingers;

mov_fingers = ('symmetric' | ('alternate', ('by pinky' | 'by thumb')?))?,
              ('vibration' | 'circles' | 'join', ('by side' | 'by fingertips')
              | 'spread' | 'flexion'| 'extension');
finger_spec = ('thumb' | 'index' | 'middle' | 'ring' | 'pinky'), mov_fingers;

wrist_twist = 'twist' | 'rotation' | 'rotation left' | 'rotation right';
wrist_flexion = 'flexion' | 'extension' | 'flexion to thumb' | 'flexion to pinky'
              | 'flexion and extension' | 'flexion to thumb and to pinky';

direction = 'clockwise' | 'counterclockwise';
frequency = [1-9] | 'repeated';
speed = 'slow', 'normal', 'fast', 'high';
tension = 'touch' | 'knock' | 'rub';

symmetric_movement = 'symmetric' | 'alternate' | 'simultaneous';

hand = 'non-dominant hand' | 'dominant hand';
```

---

path movements

```
path = hand?, symmetric_movement? , type, plane, directionality, manner;

type = shape | interaction | contact;
shape = 'zig zag' | 'circular' | 'arc' | 'straight' | 'spiral' | 'wave';
interaction = 'crossed' | 'insertion' | 'separation' | 'approximation' | 'alternate';
contact = 'brush' | 'scratch' | 'touch' | 'rub' | 'grab' | 'link' | 'slide';

plane = ('vertical' | 'horizontal' | 'obliquo' | 'middle line') | surface;
surface = 'surface', ('head' | 'hand' | 'forehead' | 'eye' | 'eyebrow' | 'nose' | 'mouth'
        | 'chin' | 'cheek' | 'ear' | 'side of the head' | 'neck' | 'arm' | 'chest'
        | 'shoulder' | 'stomach' | 'abdomen');

directionality = unidirectional | bidirectional;
unidirectional = 'up' | 'down' | 'left' | 'right' | 'forward' | 'backward' | 'up-left'
               | 'up-right' | 'up-forward' | 'up-forward-left' ... ; (*XYZ combinations*)
bidirectional = 'up and down' | 'forward and backward' ...;
```

```
manner = speed?, frequency?, tension?, extension?;
extension = 'normal' | 'short' | 'long' ;
```

---

identifier

```
identifier = (string | integer)+
string = [a-Z];
integer = [0-9];
```

---

handshape > closed set

```
closed_set = handshape_1 | handshape_2 | .... handshape_n;

handshape_1 = (*consiste de uma sentença derivada e válida sobre fingers*)
```

```
handshape_1 = [{
    "thumb": {
        "thumb_rotation": {
          "thumb_rotation": "lateral",
          "thumb_flexion": "opened"
        }
    },
    "index": {
      "relax": "false",
      "finger_contact": {
        "contact_type": "spread",
        "contact_direction": "thumb"
      },
      "finger_flexion": "opened"
    },
    "middle": {
      "relax": "false",
      "finger_contact": {
        "contact_type": "spread",
        "contact_direction": "neutral"
      },
      "finger_flexion": "opened"
    },
    "ring": {
      "relax": "false",
      "finger_contact": {
        "contact_type": "spread",
        "contact_direction": "pinky"
      },
      "finger_flexion": "opened"
    },
    "pinky": {
      "relax": "false",
      "finger_contact": {
        "contact_type": "spread",
        "contact_direction": "pinky"
      },
      "finger_flexion": "opened"
    }
  }]
```

```
[{
    "core_sl": {
        "phonetic_component": {
          "identifier": "acabar",
          "sign": {
            "type": "sequential",
            "segment[1]": {
              "hold":{
                "manual" : {
                  "type": "two hands",
                  "dominant_hand": {
                  },
                  "non_dominant_hand":{
                  }
                }
              },
              "movement": {
              }
            },
            "segment[2]": {
            },
            "hold":{
            }
          }
        }
    }
  }]
```

APÊNDICE D

JSON PARA O SINAL AFOGAR

```
[
  {
    "core_sl": {
      "phonetic_component": {
        "identifier": "afogar",
        "sign": {
          "type": "sequential",
          "segment[1]": {
            "hold": {
              "manual": {
                "type": "two hands",
                "dominant_hand": {
                  "handshape": {
                    "fingers": {
                      "thumb": {
                        "thumb_rotation": {
                          "thumb_rotation": "lateral",
                          "thumb_flexion": "opened"
                        }
                      },
                      "index": {
                        "relax": "false",
                        "finger_contact": {
                          "contact_type": "spread",
                          "contact_direction": "thumb"
                        },
                        "finger_flexion": "opened"
                      },
                      "middle": {
                        "relax": "false",
                        "finger_contact": {
                          "contact_type": "spread",
                          "contact_direction": "neutral"
                        },
                        "finger_flexion": "opened"
                      },
                      "ring": {
                        "relax": "false",
                        "finger_contact": {
                          "contact_type": "spread",
                          "contact_direction": "pinky"
                        },
                        "finger_flexion": "opened"
                      },
                      "pinky": {
                        "relax": "false",
                        "finger_contact": {
                          "contact_type": "spread",
                          "contact_direction": "pinky"
                        },
                        "finger_flexion": "opened"
                      }
                    }
                  },
                  "orientation": {
                    "or_palm":"down",
                    "or_fingers":"left",
                    "or_forearm": {
                      "type": "vertical",
                      "value": "45Â° (lateral, non-dominant-hand)"
                    },
                    "or_elbow":{
                      "type":"depth",
                      "value": "45Â° (medial)"
```

```
    }
  },
  "location": {
    "position": "dominant hand",
    "space": {
      "laterality": "parallel to midline",
      "height": "neck",
      "depth": "proximal",
    }
  }
},
"non_dominant_hand": {
  "handshape": {
    "fingers": {
      "thumb": {
        "thumb_rotation": {
          "thumb_rotation": "parallel",
          "thumb_flexion": "opened"
        }
      },
      "index": {
        "relax": "true",
        "finger_contact": {
          "contact_type": "spread",
          "contact_direction": "thumb"
        },
        "finger_flexion": "flexed"
      },
      "middle": {
        "relax": "true",
        "finger_contact": {
          "contact_type": "spread",
          "contact_direction": "neutral"
        },
        "finger_flexion": "flexed"
      },
      "ring": {
        "relax": "true",
        "finger_contact": {
          "contact_type": "spread",
          "contact_direction": "pinky"
        },
        "finger_flexion": "flexed"
      },
      "pinky": {
        "relax": "true",
        "finger_contact": {
          "contact_type": "spread",
          "contact_direction": "pinky"
        },
        "finger_flexion": "flexed"
      }
    }
  },
  "orientation": {
    "or_palm":"down",
    "or_fingers":"forward",
    "or_forearm": {
      "type": "vertical",
      "value": "45Â° (diagonal, dominant-hand)"
    },
    "or_elbow":{
      "type":"lateral",
      "local": "0Â° (body parallel)"
    }
```

```
        },
        "location": {
          "position": "non-dominant hand",
          "space":{
            "laterality": "parallel to shoulder",
            "height": "chest",
            "depth": "medial",
          }
        }
      }
    }
  },
  "movement": {
    "simple":{
      "path":{
        "type":{
          "shape":"arc"
        },
        "plane":"vertical",
        "directionality":{
          "unidirectiona":"up"
        },
        "manner":{
          "extension":"short"
        }
      }
    }
  }
},
"segment[2]": {
  "hold": {
    "manual": {
      "type": "two hands",
      "dominant_hand": {
        "handshape": {
          "fingers": {
            "thumb": {
              "thumb_rotation": {
                "thumb_rotation": "lateral",
                "thumb_flexion": "opened"
              }
            },
            "index": {
              "relax": "false",
              "finger_contact": {
                "contact_type": "spread",
                "contact_direction": "thumb"
              },
              "finger_flexion": "opened"
            },
            "middle": {
              "relax": "false",
              "finger_contact": {
                "contact_type": "spread",
                "contact_direction": "neutral"
              },
              "finger_flexion": "opened"
            },
            "ring": {
              "relax": "false",
              "finger_contact": {
                "contact_type": "spread",
                "contact_direction": "pinky"
              },
              "finger_flexion": "opened"
```

```
          },
          "pinky": {
            "relax": "false",
            "finger_contact": {
              "contact_type": "spread",
              "contact_direction": "pinky"
            },
            "finger_flexion": "opened"
          }
        }
      },
      "orientation": {
        "or_palm":"forward",
        "or_fingers":"up",
        "or_forearm": {
          "type": "vertical",
          "value": "0Âº (up)"
        },
        "or_elbow":{
          "type":"lateral",
          "0Âº (body parallel)"
        }
      },
      "location": {
        "position": "dominant hand",
        "space":{
          "laterality": "parallel to shoulder",
          "height": "ear",
          "depth": "proximal",
        }
      }
    },
    "non_dominant_hand": {
      "symmetry": "true",
      "symmetry_relation": "symmetry"
    }
  },
  "non-manual": {
    "phonetic": {
      "manner": {
        "frequency": "repeated";
      },
      "settings": {
        "head": "inclination left and right",
        "face": {
          "top":{
            "eyes":"wide eyes"
          },
          "bottom":{
            "mouth":{
              "tongue":"open mouth, tongue on the upper teeth"
            }
          }
        }
      }
    }
  }
},
"movement": {
  "symmetric_movement":"alternate",
  "complex":{
    "local": {
      "mov_wrist":{
        "wrist_twist": "twist"
      }
```

```
      },
      "path":{
        "type":{
          "shape":"straight"
        },
        "plane":"obliquo",
        "directionality":{
          "bidirectional":"foward-backward"
        },
        "manner":{
          "extension":"short"
        }
      }
    }
  }
},
"hold": {
  "manual": {
      "type": "two hands",
      "dominant_hand": {
        "handshape": {
          "fingers": {
            "thumb": {
              "thumb_rotation": {
                "thumb_rotation": "lateral",
                "thumb_flexion": "opened"
              }
            },
            "index": {
              "relax": "false",
              "finger_contact": {
                "contact_type": "spread",
                "contact_direction": "thumb"
              },
              "finger_flexion": "opened"
            },
            "middle": {
              "relax": "false",
              "finger_contact": {
                "contact_type": "spread",
                "contact_direction": "neutral"
              },
              "finger_flexion": "opened"
            },
            "ring": {
              "relax": "false",
              "finger_contact": {
                "contact_type": "spread",
                "contact_direction": "pinky"
              },
              "finger_flexion": "opened"
            },
            "pinky": {
              "relax": "false",
              "finger_contact": {
                "contact_type": "spread",
                "contact_direction": "pinky"
              },
              "finger_flexion": "opened"
            }
          }
        },
        "orientation": {
          "or_palm":"forward",
          "or_fingers":"up",
```

```
                    "or_forearm": {
                      "type": "vertical",
                      "value": "0Â° (up)"
                    },
                    "or_elbow":{
                      "type":"lateral",
                      "0Â° (body parallel)"
                    }
                  },
                  "location": {
                    "position": "dominant hand",
                    "space":{
                      "laterality": "parallel to shoulder",
                      "height": "ear",
                      "depth": "proximal",
                    }
                  }
                },
                "non_dominant_hand": {
                  "symmetry": "true",
                  "symmetry_relation": "symmetry"
                }
              },
          }
        }
      }
    }
  }
]
```

APÊNDICE E

JSON PARA O SINAL LIKE

```
[
  {
    "core_sl": {
      "phonetic_component": {
        "identifier": "like",
        "sign": {
          "type": "sequential",
          "segment[1]": {
            "hold": {
              "manual": {
                "type": "one hand",
                "dominant_hand": {
                  "handshape": {
                    "fingers": {
                      "thumb": {
                        "thumb_rotation": {
                          "thumb_rotation": "lateral",
                          "thumb_flexion": "opened"
                        }
                      },
                      "index": {
                        "relax": "false",
                        "finger_contact": {
                          "contact_type": "lateral",
                          "contact_direction": "neutral"
                        },
                        "finger_flexion": "opened"
                      },
                      "middle": {
                        "relax": "false",
                        "finger_contact": {
                          "contact_type": "lateral",
                          "contact_direction": "neutral"
                        },
                        "finger_flexion": "opened"
                      },
                      "ring": {
                        "relax": "false",
                        "finger_contact": {
                          "contact_type": "lateral",
                          "contact_direction": "neutral"
                        },
                        "finger_flexion": "opened"
                      },
                      "pinky": {
                        "relax": "false",
                        "finger_contact": {
                          "contact_type": "lateral",
                          "contact_direction": "neutral"
                        },
                        "finger_flexion": "opened"
                      }
                    }
                  },
                  "orientation": {
                    "or_palm":"back",
                    "or_fingers":"left",
                    "or_forearm": {
                      "type": "vertical",
                      "value": "45Â° (lateral, non-dominant-hand)"
                    },
                    "or_elbow":{
                      "type":"lateral",
                      "value": "0Â° (parallel body)"
```

```
          }
        },
        "location": {
          "position": "non-dominant hand",
          "body":"chest"
        }
      }
    }
  },
  "movement": {
    "simple":{
      "local":{
        "frequency": "3",
        "speed":"normal",
        "tension":"touch",
        "mov_wrist":{
          "wrist_twist": "twist"
        }
      }
    }
  }
},
"hold": {
    "manual": {
      "type": "one hand",
      "dominant_hand": {
        "handshape": {
          "fingers": {
            "thumb": {
              "thumb_rotation": {
                "thumb_rotation": "lateral",
                "thumb_flexion": "opened"
              }
            },
            "index": {
              "relax": "false",
              "finger_contact": {
                "contact_type": "lateral",
                "contact_direction": "neutral"
              },
              "finger_flexion": "opened"
            },
            "middle": {
              "relax": "false",
              "finger_contact": {
                "contact_type": "lateral",
                "contact_direction": "neutral"
              },
              "finger_flexion": "opened"
            },
            "ring": {
              "relax": "false",
              "finger_contact": {
                "contact_type": "lateral",
                "contact_direction": "neutral"
              },
              "finger_flexion": "opened"
            },
            "pinky": {
              "relax": "false",
              "finger_contact": {
                "contact_type": "lateral",
                "contact_direction": "neutral"
              },
              "finger_flexion": "opened"
```

```
                    }
                  }
                },
                "orientation": {
                  "or_palm":"back",
                  "or_fingers":"left",
                  "or_forearm": {
                    "type": "vertical",
                    "value": "45Â° (lateral, non-dominant-hand)"
                  },
                  "or_elbow":{
                    "type":"lateral",
                    "value": "0Â° (parallel body)"
                  }
                },
                "location": {
                  "position": "non-dominant hand",
                  "body":"chest"
                }
              }
            }
          }
        }
      }
    }
  }
]
```

APÊNDICE F

JSON PARA O SINAL CLOCK

```
[
  {
    "core_sl": {
      "phonetic_component": {
        "identifier": "clock",
        "sign": {
          "type": "sequential",
          "segment[1]": {
            "hold": {
              "manual": {
                "type": "two hands",
                "dominant_hand": {
                  "handshape": {
                    "fingers": {
                      "thumb": {
                        "thumb_rotation": {
                          "thumb_rotation": "parallel",
                          "thumb_flexion": "curved"
                        }
                      },
                      "index": {
                        "relax": "true",
                        "finger_contact": {
                          "contact_type": "neutral",
                          "contact_direction": "neutral"
                        },
                        "finger_flexion": "curved"
                      },
                      "middle": {
                        "relax": "true",
                        "finger_contact": {
                          "contact_type": "neutral",
                          "contact_direction": "neutral"
                        },
                        "finger_flexion": "curved"
                      },
                      "ring": {
                        "relax": "true",
                        "finger_contact": {
                          "contact_type": "neutral",
                          "contact_direction": "neutral"
                        },
                        "finger_flexion": "curved"
                      },
                      "pinky": {
                        "relax": "true",
                        "finger_contact": {
                          "contact_type": "neutral",
                          "contact_direction": "neutral"
                        },
                        "finger_flexion": "curved"
                      }
                    }
                  },
                  "orientation": {
                    "or_palm":"down",
                    "or_fingers":"down",
                    "or_forearm": {
                      "type": "horizontal",
                      "value": "0Âº (lateral, dominant-hand)"
                    },
                    "or_elbow":{
                      "type":"lateral",
                      "value": "0Âº (body parallel)"
```

```
    }
  },
  "location": {
    "position": "dominant hand",
    "hand": "wrist"
  }
},
"non_dominant_hand": {
  "handshape": {
    "fingers": {
      "thumb": {
        "thumb_rotation": {
          "thumb_rotation": "parallel",
          "thumb_flexion": "opened"
        }
      },
      "index": {
        "relax": "true",
        "finger_contact": {
          "contact_type": "spread",
          "contact_direction": "thumb"
        },
        "finger_flexion": "flexed"
      },
      "middle": {
        "relax": "true",
        "finger_contact": {
          "contact_type": "spread",
          "contact_direction": "neutral"
        },
        "finger_flexion": "flexed"
      },
      "ring": {
        "relax": "true",
        "finger_contact": {
          "contact_type": "spread",
          "contact_direction": "pinky"
        },
        "finger_flexion": "flexed"
      },
      "pinky": {
        "relax": "true",
        "finger_contact": {
          "contact_type": "spread",
          "contact_direction": "pinky"
        },
        "finger_flexion": "flexed"
      }
    }
  },
  "orientation": {
    "or_palm":"down",
    "or_fingers":"down",
    "or_forearm": {
      "type": "horizontal",
      "value": "0Âº dominant-hand side"
    },
    "or_elbow":{
      "type":"lateral",
      "local": "0Âº (body parallel)"
    }
  },
  "location": {
    "position": "dominant hand",
    "space":{
```

```
            "laterality": "parallel to midline",
            "height": "chest",
            "depth": "medial"
          }
        }
      }
    }
  },
  "movement": {
    "simple":{
      "local":{
        "frequency": "2",
        "tension" : "touch",
        "mov_forearm": "forward swing"
      }
    }
  }
},
"hold": {
    "manual": {
      "type": "two hands",
      "dominant_hand": {
        "handshape": {
          "fingers": {
            "thumb": {
              "thumb_rotation": {
                "thumb_rotation": "parallel",
                "thumb_flexion": "curved"
              }
            },
            "index": {
              "relax": "true",
              "finger_contact": {
                "contact_type": "neutral",
                "contact_direction": "neutral"
              },
              "finger_flexion": "curved"
            },
            "middle": {
              "relax": "true",
              "finger_contact": {
                "contact_type": "neutral",
                "contact_direction": "neutral"
              },
              "finger_flexion": "curved"
            },
            "ring": {
              "relax": "true",
              "finger_contact": {
                "contact_type": "neutral",
                "contact_direction": "neutral"
              },
              "finger_flexion": "curved"
            },
            "pinky": {
              "relax": "true",
              "finger_contact": {
                "contact_type": "neutral",
                "contact_direction": "neutral"
              },
              "finger_flexion": "curved"
            }
          }
        },
        "orientation": {
```

```
      "or_palm":"down",
      "or_fingers":"down",
      "or_forearm": {
        "type": "horizontal",
        "value": "0Âº (lateral, dominant-hand)"
      },
      "or_elbow":{
        "type":"lateral",
        "value": "0Âº (body parallel)"
      }
    },
    "location": {
      "position": "dominant hand",
      "hand": "wrist"
    }
  },
  "non_dominant_hand": {
    "handshape": {
      "fingers": {
        "thumb": {
          "thumb_rotation": {
            "thumb_rotation": "parallel",
            "thumb_flexion": "opened"
          }
        },
        "index": {
          "relax": "true",
          "finger_contact": {
            "contact_type": "spread",
            "contact_direction": "thumb"
          },
          "finger_flexion": "flexed"
        },
        "middle": {
          "relax": "true",
          "finger_contact": {
            "contact_type": "spread",
            "contact_direction": "neutral"
          },
          "finger_flexion": "flexed"
        },
        "ring": {
          "relax": "true",
          "finger_contact": {
            "contact_type": "spread",
            "contact_direction": "pinky"
          },
          "finger_flexion": "flexed"
        },
        "pinky": {
          "relax": "true",
          "finger_contact": {
            "contact_type": "spread",
            "contact_direction": "pinky"
          },
          "finger_flexion": "flexed"
        }
      }
    },
    "orientation": {
      "or_palm":"down",
      "or_fingers":"down",
      "or_forearm": {
        "type": "horizontal",
        "value": "0Âº dominant-hand side"
```

```
                    },
                    "or_elbow":{
                      "type":"lateral",
                      "local": "0Âº (body parallel)"
                    }
                  },
                  "location": {
                    "position": "dominant hand",
                    "space":{
                      "laterality": "parallel to midline",
                      "height": "chest",
                      "depth": "medial"
                    }
                  }
                }
              }
            }
          }
        }
      }
    }
  }
]
```

APÊNDICE G

JSON PARA O SINAL ARVORE

```
[
  {
    "core_sl": {
      "phonetic_component": {
        "identifier": "arvore",
        "sign": {
          "type": "sequential",
          "segment[1]": {
            "hold": {
              "manual": {
                "type": "two hands",
                "dominant_hand": {
                  "handshape": {
                    "fingers": {
                      "thumb": {
                        "thumb_rotation": {
                          "thumb_rotation": "parallel",
                          "thumb_flexion": "opened"
                        }
                      },
                      "index": {
                        "relax": "true",
                        "finger_contact": {
                          "contact_type": "spread",
                          "contact_direction": "neutral"
                        },
                        "finger_flexion": "flexed"
                      },
                      "middle": {
                        "relax": "true",
                        "finger_contact": {
                          "contact_type": "neutral",
                          "contact_direction": "neutral"
                        },
                        "finger_flexion": "flexed"
                      },
                      "ring": {
                        "relax": "true",
                        "finger_contact": {
                          "contact_type": "neutral",
                          "contact_direction": "neutral"
                        },
                        "finger_flexion": "flexed"
                      },
                      "pinky": {
                        "relax": "true",
                        "finger_contact": {
                          "contact_type": "spread",
                          "contact_direction": "neutral"
                        },
                        "finger_flexion": "flexed"
                      }
                    }
                  },
                  "orientation": {
                    "or_palm":"up",
                    "or_fingers":"up",
                    "or_forearm": {
                      "type": "vertical",
                      "value": "90Â° (up)"
                    }
                  },
                  "location": {
                    "position": "dominant hand",
```

```
    "hand": "fingers",
    "specific_contact": "elbow"
  }
},
"non_dominant_hand": {
  "handshape": {
    "fingers": {
      "thumb": {
        "thumb_rotation": {
          "thumb_rotation": "parallel",
          "thumb_flexion": "opened"
        }
      },
      "index": {
        "relax": "true",
        "finger_contact": {
          "contact_type": "neutral",
          "contact_direction": "neutral"
        },
        "finger_flexion": "opened"
      },
      "middle": {
        "relax": "true",
        "finger_contact": {
          "contact_type": "neutral",
          "contact_direction": "neutral"
        },
        "finger_flexion": "opened"
      },
      "ring": {
        "relax": "true",
        "finger_contact": {
          "contact_type": "neutral",
          "contact_direction": "neutral"
        },
        "finger_flexion": "opened"
      },
      "pinky": {
        "relax": "true",
        "finger_contact": {
          "contact_type": "neutral",
          "contact_direction": "neutral"
        },
        "finger_flexion": "opened"
      }
    }
  },
  "orientation": {
    "or_palm":"down",
    "or_fingers":"right",
    "or_forearm": {
      "type": "horizontal",
      "value": "0Âº dominant-hand side"
    },
    "or_elbow":{
      "type":"depth",
      "local": "0Âº (dominant-hand side)"
    }
  },
  "location": {
    "position": "dominant hand",
    "space":{
      "laterality": "parallel to shoulder",
      "height": "stomach",
      "depth": "medial"
```

```
          }
        }
      }
    }
  },
  "movement": {
    "simple":{
      "local":{
        "frequency": "1",
        "direction": "counterclockwise",
        "mov_forearm": "rotation"
      }
    }
  }
},
"hold": {
    "manual": {
      "type": "two hands",
      "dominant_hand": {
        "handshape": {
          "fingers": {
            "thumb": {
              "thumb_rotation": {
                "thumb_rotation": "parallel",
                "thumb_flexion": "opened"
              }
            },
            "index": {
              "relax": "true",
              "finger_contact": {
                "contact_type": "spread",
                "contact_direction": "neutral"
              },
              "finger_flexion": "flexed"
            },
            "middle": {
              "relax": "true",
              "finger_contact": {
                "contact_type": "neutral",
                "contact_direction": "neutral"
              },
              "finger_flexion": "flexed"
            },
            "ring": {
              "relax": "true",
              "finger_contact": {
                "contact_type": "neutral",
                "contact_direction": "neutral"
              },
              "finger_flexion": "flexed"
            },
            "pinky": {
              "relax": "true",
              "finger_contact": {
                "contact_type": "spread",
                "contact_direction": "neutral"
              },
              "finger_flexion": "flexed"
            }
          }
        },
        "orientation": {
          "or_palm":"back",
          "or_fingers":"up",
          "or_forearm": {
```

```
      "type": "vertical",
      "value": "90Â° (up)"
    }
  },
  "location": {
    "position": "dominant hand",
    "hand": "fingers",
    "specific_contact": "elbow"
  }
},
"non_dominant_hand": {
  "handshape": {
    "fingers": {
      "thumb": {
        "thumb_rotation": {
          "thumb_rotation": "parallel",
          "thumb_flexion": "opened"
        }
      },
      "index": {
        "relax": "true",
        "finger_contact": {
          "contact_type": "neutral",
          "contact_direction": "neutral"
        },
        "finger_flexion": "opened"
      },
      "middle": {
        "relax": "true",
        "finger_contact": {
          "contact_type": "neutral",
          "contact_direction": "neutral"
        },
        "finger_flexion": "opened"
      },
      "ring": {
        "relax": "true",
        "finger_contact": {
          "contact_type": "neutral",
          "contact_direction": "neutral"
        },
        "finger_flexion": "opened"
      },
      "pinky": {
        "relax": "true",
        "finger_contact": {
          "contact_type": "neutral",
          "contact_direction": "neutral"
        },
        "finger_flexion": "opened"
      }
    }
  },
  "orientation": {
    "or_palm":"down",
    "or_fingers":"right",
    "or_forearm": {
      "type": "horizontal",
      "value": "0Â° dominant-hand side"
    },
    "or_elbow":{
      "type":"depth",
      "local": "0Â° (dominant-hand side)"
    }
  },
```

```
                "location": {
                  "position": "dominant hand",
                  "space":{
                    "laterality": "parallel to shoulder",
                    "height": "stomach",
                    "depth": "medial"
                  }
                }
              }
            }
          }
        }
      }
    }
  }
]
```