

MARCOS DEVANER DO NASCIMENTO

Programação visual baseada em formulários acessível
para pessoas cegas

São Paulo
2024

MARCOS DEVANER DO NASCIMENTO

**Programação visual baseada em formulários acessível
para pessoas cegas**

Versão Revisada

Tese apresentada à Escola Politécnica da
Universidade de São Paulo para obtenção
do Título de Doutor em Engenharia da
Computação.

São Paulo
2024

MARCOS DEVANER DO NASCIMENTO

**Programação visual baseada em formulários acessível
para pessoas cegas**

Versão Revisada

Tese apresentada à Escola Politécnica da
Universidade de São Paulo para obtenção
do Título de Doutor em Engenharia da
Computação.

Área de Concentração:

Engenharia da Computação

Orientador:

Profª. Dra. Anarosa Alves Franco
Brandão

São Paulo
2024

Autorizo a reprodução e divulgação total ou parcial deste trabalho, por qualquer meio convencional ou eletrônico, para fins de estudo e pesquisa, desde que citada a fonte.

Este exemplar foi revisado e corrigido em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 02 de julho de 2024

Assinatura do autor: Marcos Álvaro do Nascimento

Assinatura do orientador: _____

Documento assinado digitalmente

gov.br

ANAROSA ALVES FRANCO BRANDAO

Data: 02/07/2024 12:09:17-0300

Verifique em <https://validar.iti.gov.br>

Catálogo-na-publicação

Nascimento, Marcos

Programação visual baseada em formulários acessível para pessoas cegas / M. Nascimento -- versão corr. -- São Paulo, 2024.

127 p.

Tese (Doutorado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais.

1.INTERAÇÃO USUÁRIO-COMPUTADOR 2.DEFICIÊNCIA VISUAL
3.LINGUAGEM DE PROGRAMAÇÃO (USO; APLICAÇÕES) I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais II.t.

AGRADECIMENTOS

Sempre fui um homem de muita fé, mas sair da minha terra, deixando a família e amigos me deu um certo medo. Embora eu estivesse cheio de esperança e motivação, o medo de fracassar me assombrava. Antes de partir para São Paulo, fiz uma oração e pedi a Deus que fosse comigo, onde quer que eu fosse. Na mesma hora, lembrei de um trecho da bíblia onde Deus fala para Abraão: - Sai da tua terra, da tua parentela e da casa de teu pai, para a terra que eu te mostrarei. E far-te-ei uma grande nação, e te abençoarei e engrandecerei o teu nome; e tu serás uma bênção (Gênesis 12:1,2). A essa palavra me apeguei e tive coragem de seguir em frente. Por isso, agradeço a Deus pelas portas que me foram abertas, pela força em momentos difíceis e por me levar a lugares que só havia vislumbrado em meus devaneios.

Agradeço a profa. Dra. Anarosa Alves Franco Brandão, minha orientadora, que me acolheu tão bem, orientando com empatia e sabedoria, guiando-me meio em meio ao caos de uma pandemia. Assim, também agradeço ao Prof. Dr. Leônidas de Oliveira Brandão por seus ensinamentos e permitir que seu trabalho (iVProg) fosse a base para a minha tese (iVProg4All). Também agradeço ao Prof. Dr. Leandro Luke que me apoiou e motivou a fazer o doutorado, quando tudo ainda era apenas um sonho.

Mesmo iniciando uma nova jornada, algumas pessoas, que já trilharam outros caminhos comigo, seguiram me apoiando e dando todo suporte necessário. Essas pessoas são o Prof. Dr. Francisco M. B. de Oliveira e Lidiane Castro Silva, a quem agradeço imensamente por toda força e apoio despendidos a mim, durante essa nova jornada. Essas são pessoas que para além da jornada acadêmica se tornaram grandes amigos para vida.

Não poderia deixar de agradecer a minha família e amigos de longas datas, que mesmo de longe sempre se mostraram próximos, dando-me forças para seguir em frente, em meio a saudade e as adversidades. Minha gratidão a Maria de Lourdes e Maria Helena, minhas mães, que me permitiram voar para longe de seus colos, sofrendo a dor da saudade por querer o melhor para um filho.

Minha gratidão a Tiago Bento, Vanderlei Santana, André Santana, Henrique Cesar, Lucimara Medes, Cindy Bastos, Bárbara Santiago e Felipe Gerônimo por me acolheram, trazendo alegria, parceria e aconchego como se fossem minha família em São Paulo. Por fim, agradeço a Thiago Casal e a toda a comunidade de cegos programadores do Brasil e os demais participantes da pesquisa, que ajudaram voluntariamente como participantes da pesquisa. Para eles e com eles, sigo em frente na busca por um mundo mais inclusivo e de oportunidades.

RESUMO

O Relatório Mundial sobre Visão da OMS de 2019 revela que 2,2 milhões de pessoas sofrem de deficiência visual em todo o mundo. Projeções indicam que, nas próximas décadas, 61 milhões ficarão cegas e 474 milhões terão deficiência visual moderada a grave globalmente. No Brasil, segundo dados do IBGE, aproximadamente 8,9% da população com 2 anos ou mais têm algum tipo de deficiência. Dentre esses, 3,1% declaram dificuldade para enxergar, mesmo com óculos ou lentes de contato. Além disso, as pesquisas revelam uma disparidade preocupante: a taxa de analfabetismo entre pessoas com deficiência é significativamente maior, atingindo 19,5%, em comparação com 4,1% entre aqueles sem deficiência. Essa discrepância ressalta a importância crucial de políticas públicas e intervenções adequadas para suprir as necessidades dessa parcela da população. Nesse contexto, a Agenda 2030 para o Desenvolvimento Sustentável, proposta pela ONU em 2015, assume um papel fundamental. Entre os seus objetivos está o compromisso de assegurar uma educação inclusiva e equitativa de qualidade para todos, com um foco especial em pessoas com deficiência. É nesse contexto que esta pesquisa de doutorado busca contribuir, abordando a temática relacionada à criação de sistemas de programação visual acessíveis para pessoas cegas e videntes (aqueles que enxergam). Os estudos iniciais destacaram os inúmeros benefícios que a programação pode proporcionar para alunos, durante o aprendizado do pensamento computacional e da lógica de programação. Identificou-se que essa abordagem não apenas pode aumentar o engajamento dos alunos, mas também reduzir a curva de aprendizado. Diante dessa constatação, surgiu a questão de se essa mesma abordagem poderia beneficiar pessoas cegas. Entretanto, as soluções existentes revelaram algumas limitações em relação ao design universal, que visa garantir a acessibilidade sem a necessidade de adaptações específicas. Em resposta a esses desafios, foi proposto o VProgForms, um modelo genérico de programação visual baseada em formulários, acessível tanto para cegos quanto para videntes. Como prova de conceito desse modelo, foi desenvolvido o iVProg4All, que se mostrou promissor em estudos preliminares. Um estudo abrangente de acessibilidade, usabilidade e desempenho foi então conduzido com usuários cegos e videntes para avaliar a viabilidade do iVProg4All e, conseqüentemente, do VProgForms. Os resultados indicaram que o iVProg4All é acessível e usável para ambos os grupos, embora videntes possam ter vantagem em relação ao desempenho devido ao contexto visual. Isso sugere que o VProgForms pode ser uma contribuição valiosa para o desenvolvimento de sistemas de programação visual acessíveis, permitindo interações eficazes entre usuários cegos e videntes. Essa abordagem pode promover um ambiente de aprendizado inclusivo, especialmente em ambientes educacionais, sem necessidade de adaptações extras. Além disso, o estudo identificou áreas para pesquisas futuras, visando à contínua melhoria da acessibilidade e inclusão na educação e na tecnologia.

Palavras-Chave: Acessibilidade, Design Universal, Deficiência visual, Cegos, Programação visual, Modelo, VProgForms, Sistema, iVProg4All, Inclusão.

ABSTRACT

The 2019 WHO World Report on Vision reveals that 2.2 million people suffer from visual impairment worldwide. Projections indicate that, in the coming decades, 61 million will be blind and 474 million will have moderate to severe visual impairment globally. In Brazil, according to IBGE data, approximately 8.9% of the population aged 2 and older have some form of disability. Among these, 3.1% declare difficulty seeing, even with glasses or contact lenses. Additionally, research reveals a concerning disparity: the illiteracy rate among people with disabilities is significantly higher, reaching 19.5%, compared to 4.1% among those without disabilities. This discrepancy underscores the crucial importance of public policies and appropriate interventions to meet the needs of this population segment. In this context, the 2030 Agenda for Sustainable Development, proposed by the UN in 2015, assumes a fundamental role. Among its objectives is the commitment to ensure inclusive and equitable quality education for all, with a special focus on people with disabilities. It is in this context that this doctoral research seeks to contribute, addressing the theme related to the creation of accessible visual programming systems for blind and sighted individuals. Initial studies highlighted the numerous benefits that programming can provide for students, during the learning of computational thinking and programming logic. It was identified that this approach can not only increase student engagement but also reduce the learning curve. Faced with this realization, the question arose of whether this same approach could benefit blind people. However, existing solutions revealed some limitations regarding universal design, which aims to ensure accessibility without the need for specific adaptations. In response to these challenges, VProgForms, a generic model of form-based visual programming accessible to both blind and sighted individuals, was proposed. As a proof of concept of this model, iVProg4All was developed, which showed promise in preliminary studies. A comprehensive study of accessibility, usability, and performance was then conducted with blind and sighted users to evaluate the feasibility of iVProg4All and, consequently, VProgForms. The results indicated that iVProg4All is accessible and usable for both groups, although sighted individuals may have an advantage in performance due to the visual context. This suggests that VProgForms can be a valuable contribution to the development of accessible visual programming systems, enabling effective interactions between blind and sighted users. This approach can promote an inclusive learning environment, especially in educational settings, without the need for additional adaptations. Additionally, the study identified areas for future research, aiming for continuous improvement of accessibility and inclusion in education and technology.

Keywords: Accessibility, Universal Design, Visual Impairment, Blind, Visual Programming, Model, VProgForms, System, iVProg4All, Inclusion.

LISTA DE FIGURAS

1	Elementos centrais do Modelo-DSR	18
2	Exemplo da soma de dois número com o Blockly, utilizando os blocos " <i>Print</i> " e " <i>Math</i> ".	23
3	Exemplo da soma de dois número com o iVProg, utilizando os blocos de comandos: Variáveis, Atribuições e Escreva.	24
4	Componentes de acessibilidade do Accessible Blockly	40
5	Interface Blocks4all	42
6	Relação entre número de usuário de teste e quantidade de problemas de usabilidade encontrados	50
7	Diagrama de blocos do SysML para definição do Modelo VProgForms	58
8	Definição do Modelo para implementação do iVProg4All	61
9	Modelo de processo evolucionário utilizado para desenvolver o iVProg4All	63
10	Protótipos do iVProg4All, em suas versões 1, 2 e 3.	64
11	Interface do iVProg4all com os blocos de comando adicionados à Função Início: Nova variável, Atribuição/Matemática, Escreva, Se..Então e Repita N vezes.	70
12	Nesta figura, uma representação piramidal é usada para representar a estrutura dos métodos e resultados da pesquisa. Questões de acessibilidade são verificadas na base, se os usuários podem acessar o software; no meio, são verificados os problemas de usabilidade, se os usuários podem usá-los com facilidade e o nível de satisfação; na parte superior o desempenho é verificado durante o processo interativo com o sistema.	74
13	Captura de tela da interação de um participante cego com iVProg4All gravada com , durante o experimento.	75
14	Sumário dos resultados de acessibilidade do iVProg4All, gerados pelo Aces-smonitor	84

LISTA DE TABELAS

1	Leitores de tela mais usados no Brasil e no mundo	32
2	Resumo dos perfis dos participantes sobre a faixa etária. Observe que nesta tabela os valores das quantidades para cada faixa etária estão agrupados nas colunas dos grupos I e II.	78
3	Resumo dos perfis dos participantes sobre o nível de escolaridade. Observe que nesta tabela os valores das quantidades para cada nível de ensino estão agrupados nas colunas dos grupos I e II.	78
4	Resumo dos cursos associados à formação acadêmica dos participantes. Observação: nesta tabela, os valores quantitativos para cada curso mencionado estão agrupados em colunas para os grupos I e II.	78
5	Resumo do nível de conhecimento dos participantes sobre lógica de programação. Observe que, nesta tabela, os valores de quantidade para cada nível de conhecimento em lógica de programação estão agrupados em colunas para os grupos I e II.	79
6	Resumo do número de vezes que uma linguagem de programação foi mencionada. Nesta tabela, os valores para cada linguagem mencionada estão agrupados em linhas para os grupos I e II.	79
7	Resumo do número de vezes que o software foi mencionado como uma ferramenta para aprender programação. Note que, nesta tabela, o número de vezes para cada software está agrupado em colunas para os grupos I e II.	80
8	Resumo da experiência profissional dos participantes, incluindo ocupações mencionadas e se estão atualmente trabalhando.	80
9	Resultados para questões relacionadas ao princípio Perceptível (*NA = Não se aplica)	81
10	Resultados para questões relacionadas ao princípio Compreensível	82
11	Resultados para questões relacionadas ao princípio operável	83
12	Resumo das descobertas de usabilidade derivadas das pontuações SUS obtidas nos Grupos I e II.	85

SUMÁRIO

1	Introdução	12
1.1	Delimitação do problema e relevância	13
1.2	Objetivo primário	15
1.2.1	Objetivos secundários	15
1.3	Estrutura do documento	15
2	Materiais e métodos	17
2.1	<i>Design Science Research</i>	17
2.1.1	Conjecturas comportamentais	18
2.1.2	Problema em contexto	19
2.1.3	Artefato	19
2.1.4	Avaliação empírica	20
2.2	Considerações	20
3	Fundamentos	21
3.1	A Programação Visual	21
3.1.1	Abordagem puramente visual	22
3.1.2	Abordagem baseada em formulário	22
3.1.3	Abordagem orientada a restrições	24
3.1.4	Abordagem baseada em exemplos	25
3.1.5	Abordagem Híbrida	25
3.1.6	Considerações	25
3.2	Diretrizes de Acessibilidade para Conteúdos na WEB	26
3.2.1	Princípio Perceptível	27

3.2.2	Princípio Operável	28
3.2.3	Princípio Compreensível	29
3.2.4	Princípio Robusto	30
3.3	Acessibilidade para cegos em sistemas Web	31
3.3.1	Navegação	32
3.3.2	Representações de conteúdos não textuais	36
3.3.3	Percepção e compreensão de alertas sonoros	37
3.4	Desafios de acessibilidade para cegos em sistemas de programação visual	38
3.4.1	Sistemas de programação visual acessíveis e suas limitações	39
3.4.1.1	Blockly adaptado para cegos	39
3.4.1.2	Blocks4All	41
3.5	Design Participativo, Protótipos e Design Universal	44
3.5.1	Design participativo e o uso de protótipos	44
3.5.2	Design Universal	45
3.6	Avaliação de Acessibilidade, Usabilidade e Análise de Desempenho	47
3.6.1	Avaliação de acessibilidade	48
3.6.2	Avaliação de Usabilidade	48
3.6.3	Análise de desempenho	51
3.7	Considerações	51
4	Um Modelo de Programação Visual Baseado em Formulários Acessível para Pessoas Cegas e Videntes: VProgForms	53
4.1	Contextualização	53
4.2	O processo de ideação e levantamento de requisitos	55
4.3	O modelo VProgForms	57
4.4	Considerações	59
5	Instanciando o modelo VProgForms: iVProg4All	60

5.1	Instanciando o iVProg4All	60
5.1.1	Ciclos de desenvolvimento	62
5.2	Testes preliminares	66
5.2.1	Teste preliminar: Módulo 1	67
5.2.2	Teste preliminar: Módulo 2	68
5.2.3	Teste preliminar: Módulo 3	69
5.2.4	Considerações gerais do testador	69
5.3	Apresentando o iVProg4All	70
5.3.1	Recursos de Acessibilidade	71
5.4	Considerações	72
6	Avaliação de Acessibilidade, Usabilidade e Análise de Desempenho Do iVProg4all	73
6.1	Materiais Métodos	73
6.2	Perfil dos participantes	77
6.2.1	Faixa etária e nível de escolaridade	77
6.2.2	Experiência com programação	78
6.2.3	Experiência profissional	79
6.3	Avaliação de Acessibilidade	80
6.3.1	Resultados relacionados ao princípio Perceptível	80
6.3.2	Resultados relacionados ao princípio Compreensível	81
6.3.3	Resultados relacionados ao princípio Operável	82
6.3.4	Resultados relacionados ao princípio Robusto	83
6.4	Resultados para Usabilidade	84
6.5	Análise de Desempenho	86
6.5.1	Análise de desempenho durante a Atividade 1	87
6.5.2	Análise de desempenho durante a Atividade 2	87

6.5.3	Análise de desempenho durante a Atividade 3	88
6.6	Considerações	88
7	Conclusões	90
7.1	Oportunidades para Trabalhos futuros	93
7.2	Contribuições	93
	REFERÊNCIAS	95
	APÊNDICE A – TERMO DE CONSENTIMENTO LIVRE E ESCLARECIDO	101
	APÊNDICE B – QUESTIONÁRIO SOCIOEDUCATIVO	104
	Anexo A – QUESTIONÁRIO PARA AVALIAÇÃO DE ACESSIBILIDADE	107
	Anexo B – QUESTIONÁRIO <i>SYSTEM USABILITY SCALE</i>	109
	Anexo C – Resultados para análise de acessibilidade do iVProg com Acessmonitor	111
	Anexo D – Parecer consubstanciado do CEP	115
	Anexo E – Relatório de teste - Módulo 1	122
	Anexo F – Resultados para análise de acessibilidade do iVProg4All com Acessmonitor	125

1 INTRODUÇÃO

O Relatório Mundial Sobre Visão, divulgado pela Organização Mundial de Saúde (2019), aponta que a deficiência visual representa um desafio global que afeta a vida de mais de 2,2 milhões de indivíduos. As tendências globais sugerem que 61 milhões de pessoas ficarão cegas e 474 milhões sofrerão de deficiência visual moderada a grave nas próximas décadas, conforme indicado por um estudo publicado na *The Lancet Global Health* (BOURNE et al., 2021).

Segundo o Instituto Brasileiro de Geografia e Estatística (IBGE), a estimativa total da população com deficiência atinge 18,6 milhões de indivíduos, constituindo 8,9% da faixa etária de 2 anos ou mais (ASCOM/MDHC, 2023). Vale ressaltar que o censo realizado pelo IBGE adota uma abordagem que ultrapassa a mera identificação de deficiências, investigando o grau de dificuldade em atividades cotidianas como ouvir, ver ou subir escadas. Dentre os desafios investigados, o mais relatado foi dificuldade para caminhar ou subir escadas (3,4%), seguido de dificuldade para enxergar, mesmo com óculos ou lentes de contato (3,1%).

No terceiro trimestre de 2022, a taxa de analfabetismo dos indivíduos com deficiência era de 19,5%, enquanto entre os sem deficiência essa taxa era de 4,1% (ASCOM/MDHC, 2023). Analisando as informações referentes aos níveis ocupacionais dos indivíduos com deficiência, segundo o tipo de dificuldade funcional, a maior taxa de pessoas com deficiência trabalhando foi observada entre aqueles com dificuldade de enxergar, mesmo com óculos ou lentes de contato, chegando a 30,9%. Essas estatísticas sublinham a natureza multifacetada da deficiência visual e enfatizam a necessidade de intervenções específicas para mitigar os desafios enfrentados pelos indivíduos com deficiência visual, tanto em um contexto global como nacional.

Entre as iniciativas que visam aliviar os diversos desafios enfrentados pelas pessoas com deficiência está a Agenda 2030 para o Desenvolvimento Sustentável, proposta pela Organização das Nações Unidas (2015). Aprovada por unanimidade pelos Estados-Membros das Nações Unidas desde 2015, esta agenda delinea um plano global abrangente com

o objetivo de promover a paz, a prosperidade e a sustentabilidade tanto para os indivíduos como para o planeta. Um dos principais objetivos deste plano é assegurar uma educação inclusiva e equitativa de qualidade, ao mesmo tempo que promove oportunidades de aprendizagem ao longo da vida para todos, com foco especial nos indivíduos com deficiência.

Uma das abordagens para promover a equidade em vários ambientes educativos, sejam físicos ou virtuais, envolve a implementação de princípios do design universal em infraestruturas, atividades e métodos de comunicação, entre outros elementos. O Design Universal é um conjunto de princípios para tornar instalações, informações e atividades acessíveis e utilizáveis para todos, considerando diversos fatores, como línguas nativas, gêneros, origens raciais e étnicas, bem como diversas habilidades e deficiências (BURGSTAHLER, 2019).

Pensando no ensino inclusivo e na aplicação do design universal no contexto de cursos introdutórios de programação, esta pesquisa busca propor um modelo para programação visual, baseado em formulários, definida como VProgForms, que seja acessível para pessoas com deficiência visual, especialmente para cegos, e videntes (pessoas que enxergam). Essa abordagem foi estendida como modelo subjacente a um sistema de programação visual chamado de iVProg4All. O iVProg4All foi utilizado como prova de conceito para estudos de acessibilidade e usabilidade com pessoas cegas e videntes (aquelas que enxergam) a fim de verificar a eficiência da abordagem VprogForms para ambos os grupos.

1.1 Delimitação do problema e relevância

A programação visual usa elementos gráficos para expressar comandos, ao contrário das linguagens de programação tradicionais que usam elementos lexicais. Os comandos costumam ser representados por blocos, aninhados de forma lógica para resolver um problema (MYERS, 1990; BURNETT; MCINTYRE, 1995; BOSHERNITSAN; DOWNES, 1997). Assim, a ideia de produzir uma solução de problema, usando programação visual, reflete a organização lógico-espacial dos elementos gráficos no plano, durante a modelagem da solução. Entre os sistemas de programação visual utilizados para fins educacionais destacam-se o Scratch (SCRATCH FOUNDATION, 2023), Alice (CARNEGIE MELLON UNIVERSITY, 2023), Blockly (GOOGLE, 2023) e IVProg (FELIX et al., 2019) entre outros.

A abordagem de programação visual foi tida como foco deste estudo devido aos bene-

fícios que ela pode trazer para os alunos em diferentes níveis educacionais (DOBESOVA, 2014; CHANG et al., 2017; NASCIMENTO et al., 2019b). Entre esses benefícios estão a redução do esforço cognitivo durante o processo de aprendizagem, maior envolvimento nas atividades práticas, melhoria no desempenho, além de elevar os níveis de satisfação dos alunos (MEERBAUM-SALANT et al., 2010; BRANDÃO et al., 2012; RIBEIRO et al., 2012; CHANG et al., 2017).

Embora a programação visual possa trazer diversos benefícios para aprendizes em cursos introdutórios de programação de computadores, um estudo realizado por Milne e Ladner (2018) indica que uma série de sistemas de programação visual apresentam diversos desafios para pessoas com deficiência visual, que utilizam leitores de tela para interagir com esses sistemas. Alguns desses desafios estão diretamente relacionados ao estilo de interação adotado (clica e arrasta), além da falta de recursos de acessibilidade que auxiliem pessoas cegas durante a interação com esses sistemas.

Algumas iniciativas, como o Blocks4All (MILNE; LADNER, 2018) e o Blockly Acessível (MOUNTAPMBEME et al., 2022), são propostas como soluções de programação visual acessíveis para pessoas cegas. No entanto, os sistemas mencionados ainda apresentam limitações que podem dificultar sua adoção em larga escala por pessoas cegas e videntes. Por exemplo, o Blocks4All é direcionado exclusivamente a crianças, restringindo seu uso ao iPadOS. Em contrapartida, o Blockly Acessível se apresenta como uma solução viável para grupos de usuários mais diversos, mas adota o modelo com representações visuais em forma de quebra-cabeças, o que pode tornar a construção de algoritmos mais desafiadora para usuários de leitores de tela.

Diante deste contexto, o iVProg4All, tendo como pano de fundo o VProgForms, pode ser um diferencial, pois se trata de uma abordagem que aplica o conceito de design universal. Assim, não há necessidade de adaptação para que um grupo de usuários possa utilizar. Além disso, o estilo de interação baseado em formulários costuma fazer parte do cotidiano dos usuários da internet, podendo diminuir a curva de aprendizagem de uso da ferramenta, se levado em consideração o conhecimento prévio dos usuários.

Outro fator que pode ser um potencializador para acessibilidade é o uso de elementos de formulário na Linguagem de Marcação de HiperTexto (HTML), pois a HTML possui uma série de recursos que possibilita a criação de formulários acessíveis (RIVERA, 2009). Sendo assim, sugere-se que uma abordagem de programação visual baseada em formulários com recursos de acessibilidade para usuários cegos pode ser um facilitador para as interações entre alunos (ou professores) cegos e aqueles que enxergam, ampliando

a zona de desenvolvimento proximal (FINO, 2001) dos aprendizes. Como consequência, espera-se possibilitar a realização de tarefas cooperativamente, criando um ambiente de aprendizado inclusivo.

1.2 Objetivo primário

Este estudo busca definir, instanciar e avaliar uma abordagem de programação visual baseada em formulários, alinhada ao design universal passível de utilização por pessoas com e sem deficiência visual, sem a necessidade de adaptações adicionais. Espera-se com isso promover um ensino introdutório de programação que seja equitativo e inclusivo.

1.2.1 Objetivos secundários

1. Propor um modelo de programação visual baseado em formulários que aplique os princípios do design universal, assegurando uma interface acessível tanto para usuários com deficiência visual quanto para aqueles sem essa condição.
2. Implementar e implantar um sistema de programação visual que adote o modelo proposto;
3. Avaliar empiricamente o sistema implementado como forma de meta-avaliação do modelo, com relação a acessibilidade, usabilidade e análise de desempenho.

1.3 Estrutura do documento

Esta tese está organizada em 7 (sete) capítulos, cada um abordando aspectos específicos da pesquisa. O capítulo 2 descreve o método de pesquisa adotado para conduzir a investigação, delineando as escolhas metodológicas fundamentais. No Capítulo 3, são apresentados os fundamentos essenciais para a compreensão da pesquisa, abrangendo conceitos, métodos e a problemática abordada. O Capítulo 4 dedica-se à apresentação do metamodelo VProgForms, desenvolvido neste estudo, descrevendo sua contextualização, métodos e técnicas adotados para sua concepção. No Capítulo 5, introduzimos o iV-Prog4All, um sistema de programação visual acessível que implementa o metamodelo VProgForms. Este capítulo detalha os recursos implementados e apresenta os resultados de testes preliminares realizados. No Capítulo 6, descrevemos os métodos utilizados para a meta-avaliação do modelo VProgForms, considerando aspectos de acessibilidade,

usabilidade e análise de desempenho, por meio do iVProg4All. Finalmente, o Capítulo 7 traz as discussões e conclusões derivadas da pesquisa, além de apontar oportunidades para trabalhos futuros e destacar as publicações realizadas em conferências durante o desenvolvimento deste estudo.

2 MATERIAIS E MÉTODOS

Neste capítulo será apresentado o método de pesquisa aplicado para concepção, desenvolvimento e avaliação de um modelo de programação visual baseado em formulário (VProgForms), servindo este, como um modelo subjacente a um sistema de programação visual (iVProg4All).

2.1 *Design Science Research*

O *Design Science Research* (DSR) constitui um método de pesquisa científica que fornece a base para a criação de artefatos computacionais com o propósito de gerar conhecimento científico (PEFFERS et al., 2007; PIMENTEL, 2017; PIMENTEL et al., 2019). Nesse contexto, busca-se alinhar o progresso tecnológico com a produção de conhecimento científico. No âmbito do DSR, um artefato pode ser concebido como qualquer entidade, seja física ou abstrata, desenvolvida com a finalidade de atingir os objetivos de pesquisa. A criação e investigação de um artefato não se limitam apenas à sua funcionalidade; ao contrário, abrangem uma exploração mais ampla da realidade e do contexto no qual a solução foi implementada (PIMENTEL et al., 2020).

No contexto desta pesquisa, aplicou-se o Modelo-DSR, proposto por Pimentel et al. (2020). Esse modelo proposto para o DSR é uma síntese de diversas abordagens, composto por quatro elementos centrais coerentemente inter-relacionados, especialmente adequado para pesquisas no campo de informática na educação. Esses elementos são: conjecturas comportamentais, problema em contexto, artefato e avaliação empírica é apresentado na figura 1.

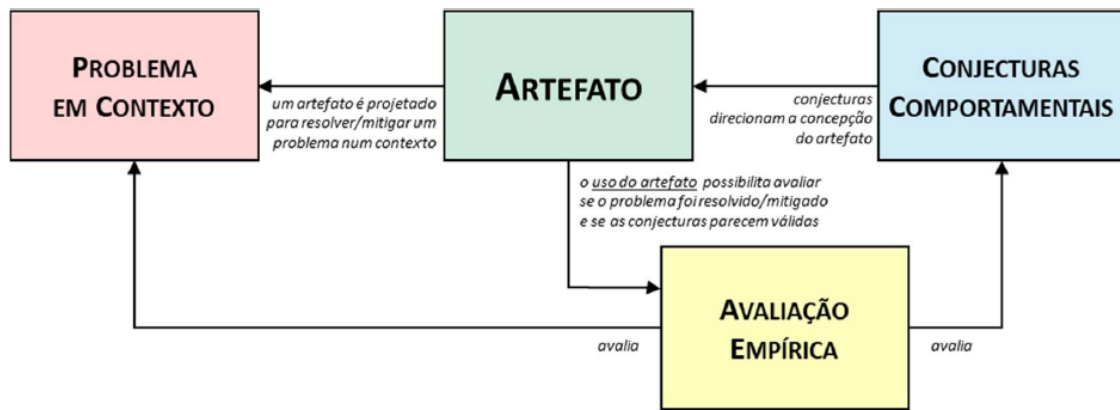


Figura 1: Elementos centrais do Modelo-DSR
 Fonte: Adaptado de Pimentel et al. (2020)

O Modelo-DSR emprega processos iterativos nos quais o desenvolvimento de um artefato tem como fundamento **conjecturas comportamentais** relacionadas ao âmbito humano-organizacional-social considerando os pensamentos, desejos e ações das pessoas no contexto organizacional e social. A partir dessas conjecturas, o **artefato** é concebido como intervenção para um **problema em contexto**. Ao empregar o artefato e conduzir uma **avaliação empírica**, é possível determinar se o problema foi efetivamente resolvido e se as conjecturas que fundamentaram o desenvolvimento do artefato são válidas. Segundo (PIMENTEL et al., 2020), a criação do artefato e a análise de seu uso contribuem para a produção de conhecimentos técnicos e científicos. Além disso, tratando-se de uma pesquisa científica, a revisão da literatura deve fundamentar cada um dos elementos centrais do Modelo-DSR.

Neste estudo, utilizou-se o Modelo-DSR como estrutura metodológica e orientadora para a definição e condução das etapas inerentes à pesquisa em questão. Essas atividades são apresentadas nas seções a seguir.

2.1.1 Conjecturas comportamentais

Nesta fase da pesquisa, foram conduzidas investigações bibliográficas com o propósito de compreender e identificar oportunidades de pesquisa relacionadas à acessibilidade para indivíduos com deficiência visual em ambientes de programação visual. Inicialmente, realizou-se uma revisão do estado da arte referente aos sistemas de programação visual, com o intuito de identificar lacunas e oportunidades de pesquisa nesse domínio (NASCIMENTO et al., 2019b). Além disso, foram empreendidos estudos na literatura específica sobre acessibilidade para pessoas cegas em sistemas web (NASCIMENTO et al., 2019a).

O escopo desse exame acerca da acessibilidade para esse grupo focal buscou elucidar aspectos cruciais para a compreensão desse contexto, visando formular hipóteses pertinentes à acessibilidade para pessoas cegas em ambientes de programação visual. Desse modo, surgiram as seguintes questões de pesquisa:

1. Q1: É possível criar um modelo de programação visual que aplique o conceito de design universal, que possa ser usado por pessoas cegas e videntes sem a necessidade de adaptação?
2. Q2: Um sistema de programação visual pode ser utilizado de forma equânime, no sentido de usabilidade e análise de desempenho por pessoas cegas e videntes?

As questões de pesquisa levantadas foram norteadoras para definir os objetivos desta pesquisa, apontados no capítulo 1. Os resultados da pesquisa bibliográfica realizada nesta etapa são apresentados como fundamentação no capítulo 3, nas seções 3.1, 3.2 e 3.3.

2.1.2 Problema em contexto

As pesquisas bibliográficas realizadas revelaram que a maioria dos sistemas de programação visual amplamente utilizados não eram acessíveis para pessoas cegas (MILNE; LADNER, 2018). O forte apelo visual e o estilo de interação adotado por tais sistemas se apresentaram como limitadores para seu uso por pessoas cegas. Com esse estudo foi possível a compreensão dos desafios encontrados por pessoas cegas em sistemas de programação visual e dos problemas para esse contexto. A explanação desse estudo, bem como seus resultados, são apresentados no capítulo 3.

2.1.3 Artefato

Com objetivo de verificar as hipóteses levantadas e alcançar os objetivos de pesquisa, foi concebido um Modelo de Programação Visual baseado em formulários denominado VProgForms, apresentado no capítulo 4. A partir desse modelo, foi implementado um sistema de programação visual baseado em formulários denominado iVProg4All, apresentado no capítulo 5.

Para conceber o VProgForms, bem como o iVProg4All (NASCIMENTO et al., 2023), foram realizadas sessões de design participativo com pessoas cegas e videntes. Estes artefatos foram desenvolvidos com propósitos de pesquisa.

2.1.4 Avaliação empírica

Através da implementação do modelo VProgForms por meio do iVProg4All, tornou-se possível conduzir investigações utilizando abordagens destinadas à avaliação de acessibilidade, usabilidade e análise de desempenho dos usuários. Inicialmente, foram conduzidos testes de acessibilidade com um usuário cego, visando analisar a viabilidade do modelo para este perfil de usuário. A descrição e os resultados desses testes foram publicados na conferência *Frontiers in Education 2023* (NASCIMENTO et al., 2023); também são apresentados no capítulo 5, na seção 5.2 deste trabalho.

Após verificar a viabilidade na implementação do iVProg4All, prosseguiu-se com um estudo experimental envolvendo participantes cegos e videntes, cujo propósito foi responder às questões de pesquisa delineadas no âmbito desta investigação. A descrição desse estudo, bem como seus resultados foram submetidos para o *International Journal of Human-Computer Studies*, também são apresentados no capítulo 6 deste trabalho.

2.2 Considerações

Neste capítulo, delineou-se o Modelo de *Design Science Research* como a metodologia fundamental para a concepção e execução da pesquisa em questão. A aplicação do Modelo-DSR, compreendendo a concepção e avaliação de artefatos capazes de contribuir com conhecimento inédito e estabelecer direções para futuras investigações na temática em questão. O embasamento teórico integral, referente à pesquisa, será exposto no capítulo 3.

3 FUNDAMENTOS

Neste capítulo, são expostos os fundamentos essenciais para a compreensão da pesquisa descrita nesta tese. Assim, inicia-se com a apresentação de conceitos relacionados à programação visual, seguidos pelas diretrizes de acessibilidade na web propostas pela *World Wide Web Consortium* (W3C). Em seguida, será abordada a acessibilidade para cegos em ambientes web, assim como os desafios de acessibilidade para cegos em sistemas de programação visual. Também será apresentada a metodologia de Design Participativo para o desenvolvimento de software, destacando o uso de protótipos como ferramenta fundamental para levantamento de requisitos e coleta de *feedback* dos usuários. Além disso, são apresentados os princípios do Design Universal como um meio para a criação de soluções acessíveis para todos. Finalmente, o capítulo é concluído ao apresentar as técnicas para avaliação de usabilidade e acessibilidade adotadas para esta pesquisa.

3.1 A Programação Visual

A programação visual e, paralelamente, as linguagens de programação visual surgem como uma resposta às dificuldades enfrentadas por estudantes no processo de aprendizado de programação por meio de linguagens tradicionais baseadas em texto. Conforme discutido por Glinert e Tanimoto (1984), as linguagens de programação visual representam uma desvinculação dos estilos de programação contemporâneos, almejando assim tornar a programação mais acessível. Nesse sentido, o propósito das linguagens de programação visual é atenuar a acentuada curva de aprendizado associada aos conceitos introdutórios de programação, dada sua complexidade (BOSHERNITSAN; DOWNES, 1997).

De acordo com Myers (1990), a programação visual configura-se como uma classe de linguagens de programação que empregam elementos gráficos para a expressão de comandos. Tal fato contrasta com as linguagens tradicionais que fazem uso de elementos lexicais, visando evitar armadilhas sintáticas, tais como palavras reservadas e comandos, inerentes às linguagens de programação convencionais. Dentre os objetivos primordiais subjacentes

às abordagens de programação visual, destacam-se: facilitar o acesso à programação para um público específico, a exemplo de crianças; aprimorar a precisão na execução de tarefas de programação por parte dos usuários; e incrementar a celeridade na realização de tarefas programáticas (BURNETT; MCINTYRE, 1995). Adicionalmente, a programação visual foi concebida não apenas como um meio de mitigar as complexidades inerentes à sintaxe das linguagens de programação, mas também como um meio de fomentar a motivação e o engajamento dos aprendizes durante o processo de assimilação dos princípios lógicos inerentes à programação (GLINERT; TANIMOTO, 1984).

Com a popularização e surgimento de diferentes abordagens para implementação de linguagens de programação visual, (BOSHERNITSAN; DOWNES, 1997) propõe uma classificação para elas: puramente visuais; baseado em formulário; baseado em restrições; baseado em exemplos; e híbrido, que são brevemente descritos a seguir. Ressalta-se que esta classificação não é exclusiva, podendo uma abordagem ser classificada dentro de uma ou mais das mencionadas.

3.1.1 Abordagem puramente visual

Na abordagem de programação puramente visual, as técnicas visuais são utilizadas ao longo de todo o processo de programação. Além disso, a manipulação de ícones ou outras representações gráficas é utilizada para criar um programa. A depuração e execução ocorrem no mesmo ambiente visual. Além disso, as instruções nunca são transcritas para uma linguagem intermediária baseada em texto. Dentre os sistemas de programação visual que apresentam essas características, é possível mencionar o Scratch (SCRATCH FOUNDATION, 2023) e o Blocks4All (MILNE, 2017). Ambos os sistemas abordam técnicas puramente visuais para construção dos algoritmos e não possuem recursos para conversão da lógica visual para uma linguagem baseada em texto.

3.1.2 Abordagem baseada em formulário

Na abordagem baseada em formulário, é utilizado um grupo de células ou campos interconectados, permitindo ao programador visualizar a execução de um programa como uma sequência de diferentes estados de células ou campos. Por exemplo, ao considerar a metáfora da planilha, é possível organizar as células em um grupo denominado formulário, como exemplificado no sistema Foms/3, proposto por Burnett et al. (2001). Entre outros exemplos que também adotam essa abordagem podem ser citados o Blockly (GOOGLE, 2023) e iVprog (FELIX et al., 2019).

No Blockly, embora possua uma bordagem fortemente baseado em restrições, a lógica de programação também é definida por meio de elementos de formulário, embutidos em cada bloco. Por exemplo, conforme é mostrado na figura 2, para criar e retornar a soma entre dois número e mostrar o resultado da soma, é necessário conectar os blocos "*Print*"(imprimir) e "*Math*"(Matemática) para operações matemáticas. No bloco Math a operação matemática é definida através de elementos de formulários como campos de texto para adicionar valores numéricos e um menu para selecionar o tipo de operação.

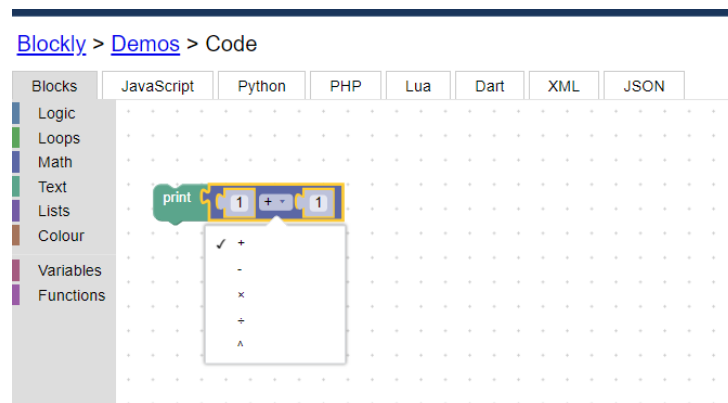


Figura 2: Exemplo da soma de dois número com o Blockly, utilizando os blocos "*Print*" e "*Math*".

Fonte: Adaptado de Google (2023)

Ao contrário do Blockly, o iVProg adota um forte apelo para programação visual baseada em formulários, utilizando a tag `<form>` do HTML (W3C School, 2023) para agrupar elementos de formulário. Toda a lógica é construída por meio de blocos de comandos composto por diversos elementos de formulário como campos de texto, menus, botões, entre outros. Na figura3 é exposto um exemplo onde uma variável global "soma" é criada através do menu para criação de variáveis. O tipo da variável é definido por meio de um menu, onde o usuário pode seleciona se será do tipo texto, inteiro, real, lógico, entre outros. O nome e o valor atribuídos à variável são definidos por meio de campos de texto. A variável global "soma"ficará disponível para que outros blocos à utilizem para para atribuições e operações. Conforme mostra o exemplo da figura 3, também foi criado um bloco de atribuição, usando o menu para criação de blocos de comando. No bloco de atribuição, foi definida uma operação matemática em que a variável "soma"recebe o resultado da operação matemática. Para esta operação., o operador matemático foi selecionado em um menu e os valores foram definidos por meio de campos de texto. Por fim, o resultado da soma é impresso usando o bloco de comando "Escreva", onde a variável "soma"é selecionada através de um menu com as variáveis disponíveis.

Com os exemplos mencionados, é evidente que o iVProg destaca-se por adotar uma

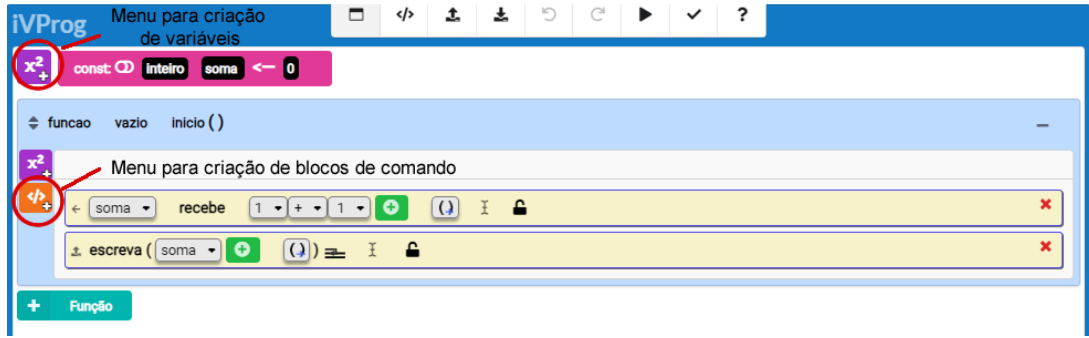


Figura 3: Exemplo da soma de dois número com o iVProg, utilizando os blocos de comandos: Variáveis, Atribuições e Escreva.

Fonte: Adaptado de Felix et al. (2019)

abordagem robusta para a programação visual baseada em formulários. Nesse ambiente, a lógica de programação é inteiramente construída por meio de elementos de formulário, presentes nos blocos de comandos. Esses blocos podem ser entendidos como formulários com "fórmulas" incorporadas que se integram para formar a lógica de programação. Por outro lado, o Blockly possui um forte apelo visual, baseado em restrições, adotando a abordagem baseada em formulários como um recurso complementar.

3.1.3 Abordagem orientada a restrições

Na abordagem de programação visual baseada em restrições, os objetos do mundo real são representados como objetos gráficos, aplicando o conceito de manipulação direta (BARBOSA; SILVA, 2010), sujeita a restrições. Essas características podem ser observadas no sistema Blockly (GOOGLE, 2023), no qual a elaboração de um algoritmo ocorre através da manipulação e interconexão de blocos, assemelhando-se à montagem de peças de um quebra-cabeça. As interligações são sujeitas a restrições, como a conformidade do formato de um bloco para possibilitar o encaixe e a criação da estrutura lógica.

De acordo com Boshernitsan e Downes (1997), essa abordagem também encontra aplicação em sistemas destinados à criação de interfaces gráficas. Na configuração desses sistemas, a disposição dos elementos da interface é estruturada para viabilizar a sua manipulação durante a elaboração da interface gráfica de um sistema. Essas características são evidenciadas no Ambiente de Desenvolvimento Integrado NetBeans ¹, onde é viável construir uma interface gráfica por meio da manipulação de componentes como menus, botões, caixas de seleção, entre outros.

¹<https://netbeans.apache.org/tutorial/main/kb/docs/java/quickstart-gui/>

3.1.4 Abordagem baseada em exemplos

A programação visual baseada em exemplos adota a manipulação de objetos gráficos como método instrucional para capacitar o sistema a realizar tarefas específicas, como orientar um elemento gráfico a efetuar deslocamentos ou gerar emissões sonoras, conforme indicado por Boshernitsan e Downes (1997). Nesse contexto, o Blocks4all (MILNE; LADNER, 2018) e Scratch exemplificam essa abordagem, pois incorporam metáforas, elementos visuais e manipulação de objetos com o propósito de conceber uma estrutura lógica voltada para a realização de ações específicas, que podem resultar em alertas sonoros ou movimentação de personagens.

3.1.5 Abordagem Híbrida

No âmbito da abordagem de programação visual híbrida, ocorre a combinação de elementos visuais e textuais, proporcionando a criação de algoritmos de maneira visual que posteriormente podem ser traduzidos para uma linguagem textual. Essa abordagem viabiliza a construção inicial de um algoritmo por meio de elementos gráficos, seguida da posterior tradução desse algoritmo para uma forma puramente textual. Como exemplos desta abordagem, destacam-se o iVProg (FELIX et al., 2019) e o Blockly (GOOGLE, 2023).

O iVProg trata-se de um sistema programação visual fundamentado em formulários com a possibilidade de conversão para linguagem textual. Enquanto o Blockly permite que os usuários programem utilizando blocos gráficos que representam diferentes comandos e estruturas de controle. Ambos os sistemas permitem que os comandos expressos por gráficos possam ser transcritos para uma linguagem textual.

3.1.6 Considerações

Embora os sistemas de programação visual possam ser eficazes na introdução de conceitos iniciais de programação, é importante observar que alunos com deficiência visual podem não usufruir plenamente dos recursos desse paradigma. A utilização de elementos visuais e estilos de interação específicos, como clicar e arrastar, por exemplo, podem não ser adequados para pessoas cegas. Nesse contexto, a seção 3.4 deste capítulo discutirá os desafios de acessibilidade enfrentados por indivíduos com deficiência visual ao utilizar sistemas de programação visual. Contudo, para uma compreensão mais aprofundada desses desafios, é crucial assimilar os conceitos e diretrizes de acessibilidade apresentados na

seção 3.2.

3.2 Diretrizes de Acessibilidade para Conteúdos na WEB

O Decreto nº 5.296/2004, da lei brasileira de inclusão, define deficiência como “[...] qualquer perda ou anormalidade de uma estrutura ou função psicológica, fisiológica ou anatômica que gere incapacidade para o exercício de uma atividade, dentro do padrão considerado normal para o ser humano” (PRESIDÊNCIA DA REPÚBLICA - CASA CIVIL, 2023). Vale ressaltar que a deficiência é um conceito em evolução. As dificuldades na realização de tarefas estão diretamente relacionadas à falta de recursos de acessibilidade em mídias digitais ou estruturais que possam amenizar as dificuldades enfrentadas pelas pessoas com deficiência em suas tarefas.

A Convenção Internacional sobre os Direitos das Pessoas com Deficiência, adotada pela ONU em 30 de março de 2007, em Nova York, e ratificada pelo Decreto Federal, de lei brasileira, nº 6.949, de 25 de agosto de 2009 (CAIADO, 2009; PRESIDÊNCIA DA REPÚBLICA - CASA CIVIL, 2004), define acessibilidade como a possibilidade de pessoas com deficiência viverem de forma independente e participarem plenamente de todos os aspectos da vida. Para tanto, a lei define que os estados partes devem tomar as medidas cabíveis para assegurar que tenham acesso, em igualdade de oportunidades com os demais, ao meio físico, transporte, informação e comunicação, incluindo sistemas e tecnologias de informação e comunicação, bem como outros serviços e facilidades abertos ou prestados ao público, tanto em áreas urbanas como rurais.

No contexto da acessibilidade para sistemas e tecnologias de informação e comunicação para web, o *World Wide Web Consortium* (W3C) (W3C, 2018) propõe um documento com diretrizes de acessibilidade para o conteúdo da web (WCAG), atualmente em sua versão 2.1. A WCAG 2.1 define quatro princípios para acessibilidade de páginas web: Perceptível, Compreensível, Operável e Robusto. Cada princípio compreende um conjunto de diretrizes que delineiam as ações necessárias e critérios de sucesso que especificam a maneira de implementar um conteúdo na web de modo acessível. Os critérios de sucesso são categorizados em níveis A, AA e AAA, sendo estes os que determinam o nível de conformidade de acessibilidade para conteúdos na web.

Os critérios de sucesso de nível A representam o padrão mais básico, englobando critérios que representam apenas barreiras mais significativas de acessibilidade. Para alcançar o nível A, todos os critérios do nível A devem ser plenamente satisfeitos. O nível AA, por

sua vez, busca atender a maior parte dos usuários, garantindo acesso à grande maioria dos conteúdos. Contudo, para que uma página da web alcance o nível de acessibilidade AA, todos os critérios dos níveis A e AA devem ser integralmente cumpridos. O nível AAA, considerado o patamar mais elevado de acessibilidade, trata-se de refinamentos das anteriores, com especificações mais detalhadas trazendo um nível mais sofisticado de acessibilidade. A W3C (2018) aconselha a não estabelecer a conformidade de nível AAA como uma política abrangente para sites completos, uma vez que, em alguns casos, torna-se inviável atender a todos os critérios de sucesso de nível AAA.

Os princípios, diretrizes e exemplos de critérios de sucesso da WCAG 2.1 são apresentados nas subseções a seguir.

3.2.1 Princípio Perceptível

O princípio perceptível trata da capacidade dos usuários perceberem os conteúdos e elementos de interface em uma página da web. É fundamental que todos os conteúdos de uma página sejam percebidos pelos usuários. Para abordar esse princípio, são listadas as seguintes diretrizes:

1. **Diretriz para Alternativas em Texto:** esta diretriz ressalta a importância de oferecer alternativas textuais para conteúdo não textual, possibilitando a sua conversão em diferentes formatos de acordo com as necessidades individuais dos usuários. Isso abrange adaptações como impressão com fontes ampliadas, transcrição em braille, conversão para fala, representação por meio de símbolos ou utilização de linguagem simplificada. Um exemplo de critério de sucesso, nível A, para essa diretriz, é que todo conteúdo não textual apresentado ao usuário deve prover uma alternativa textual que desempenhe uma função equivalente.
2. **Diretriz para Mídias com Base em Tempo:** esta diretriz está relacionada à disponibilização de alternativas para mídias dependentes de tempo, tais como vídeos e áudios pré-gravados. Um exemplo de critério de sucesso, nível A, é que sejam disponibilizadas legendas para todo o conteúdo de áudio previamente gravado em mídia sincronizada, a menos que a mídia sirva como uma alternativa para o texto e seja claramente identificada como tal. Também constitui-se como critério de sucesso, nível AA, a disponibilização de áudio descrição para todo o conteúdo de vídeo pré-gravado presente em mídia sincronizada.
3. **Diretriz Adaptável:** esta diretriz está associada ao desenvolvimento de conteúdos

que sejam flexíveis em sua apresentação, permitindo diferentes formas de exibição, como um layout simplificado, sem comprometer a informação ou estrutura. Um exemplo de critério de sucesso, nível A, para esta diretriz seria oferecer orientações para compreender e utilizar o conteúdo que não dependam exclusivamente das propriedades sensoriais dos elementos, tais como forma, cor, tamanho, localização visual, orientação ou som. Também constitui-se como exemplo de critério de sucesso, nível AA, não limitar a visualização do conteúdo a uma única orientação de exibição, como retrato ou paisagem, a menos que uma orientação de exibição específica seja considerada indispensável.

4. **Diretriz Discernível:** esta diretriz está relacionada à facilitação da audição e visualização dos conteúdos para os usuários. Isso deve ser feito separando o primeiro plano do plano de fundo. Por exemplo, um dos critérios de sucesso, nível A, para esta diretriz é que não sejam utilizadas cores como único meio para transmitir informações, indicar ações, pedir resposta ou distinguir um elemento visual. Também constitui-se como exemplo de critérios de sucesso, nível AA, o redimensionamento de texto, com exceção para legendas e imagens de texto (texto que foi convertido para um formato não textual). O redimensionamento de texto pode ser feito sem tecnologia assistiva até 200% sem perder conteúdo ou funcionalidade. Além disso, recomenda-se que imagens de texto sejam empregadas exclusivamente por motivos decorativos ou quando uma apresentação específica de texto é crucial para a informação que está sendo comunicada, sendo esta recomendação um critério de sucesso, nível AAA.

3.2.2 Princípio Operável

Para atender ao princípio Operável, os componentes de interface de usuário e a navegação devem ser operáveis. Para esse princípio são elencadas as seguintes diretrizes:

1. **Acessível por Teclado:** esta diretriz ressalta que toda funcionalidade deve estar disponível com o uso do teclado. Um exemplo de critério de sucesso, nível A, para esta diretriz é que todas as operações funcionais do conteúdo possam ser realizadas por meio de um teclado, sem a necessidade de limitações temporais específicas para digitação individual.
2. **Tempo Suficiente:** esta diretriz ressalta que deve ser fornecido aos usuários tempo suficiente para ler e utilizar o conteúdo. Um exemplo de critério de sucesso, nível

AAA, para esta diretriz está relacionado ao elemento temporal não constituir uma parte fundamental do evento ou da atividade representada pelo conteúdo, a menos que esteja relacionado a mídia sincronizada não interativa e eventos em tempo real.

3. **Convulsões e Reações Físicas:** esta diretriz enfatiza que não deve-se criar conteúdo que seja conhecido por desencadear convulsões e respostas físicas adversas. Um exemplo de critério de sucesso, nível A, para esta diretriz está associado à não desenvolver conteúdo que pisque mais de três vezes em um segundo, a menos que o efeito de flash esteja dentro dos limites universais de flash e do flash vermelho. Também constitui-se como critério de sucesso, nível AAA, a capacidade de desativar a animação de movimento acionada por interação está disponível, a menos que a presença da animação seja crucial para a funcionalidade ou para a comunicação eficaz de informações.
4. **Navegável:** esta diretriz está relacionada à disponibilização de métodos que auxiliem os usuários na navegação, localização de conteúdo e na identificação de sua posição na página. Um exemplo de critério de sucesso, nível A, para esta diretriz é que as páginas web devem possuir títulos que descrevem o tópico ou a finalidade. Além disso, outro critério de sucesso, nível AA, estabelece que deve haver mais de uma maneira de localizar uma página da web, exceto quando a página web for o resultado ou uma etapa de um processo. Também pode ser considerado como critério de sucesso, nível AAA, quando os cabeçalhos de seção são empregados para estruturar o conteúdo.
5. **Modalidades de Entrada:** esta diretriz corresponde a facilitar a operação da funcionalidade para os usuários, possibilitando o uso de diferentes métodos de entrada, além do teclado. A exemplo de critério de sucesso, nível A, para esta diretriz é que os elementos de interface do usuário que possuírem rótulos contendo texto ou imagens de texto, o nome deve refletir o conteúdo textual visível. Outro exemplo, nível AAA, está relacionado ao conteúdo da web não limitar o uso das modalidades de entrada de dados disponíveis em uma plataforma, a menos que tal restrição seja essencial para garantir a segurança do conteúdo ou seja necessária para respeitar as configurações do usuário.

3.2.3 Princípio Compreensível

Para o princípio Compreensível estende-se que a compreensão da informação e da operação da interface de usuário é essencial. Para esse princípio são elencadas as seguintes

diretrizes:

1. **Legível:** esta diretriz define que o conteúdo do texto deve ser legível e compreensível. Um exemplo de critério de sucesso, nível A, é que o idioma humano padrão de uma página web pode ser identificada por meio de código de programação. Nesse sentido, também considera-se critério de sucesso, nível AA, que o idioma de cada passagem ou frase no conteúdo, com exceção de nomes próprios, termos técnicos, palavras de idioma indeterminado e expressões que se integram ao contexto do texto que as circunda possam também ser determinado por código de programação. Mecanismos para identificar a pronúncia específica de palavras, onde o significado das mesmas, no contexto, é ambíguo se a pronúncia não for conhecida, também configura-se como um critério de sucesso, nível AAA.
2. **Previsível:** esta diretriz define que as páginas web devem aparecer e funcionar de modo previsível. Por exemplo, um dos critérios de sucesso para esta diretriz, nível A, é que qualquer componente de interface do usuário que recebe o foco, não deve ocasionar uma alteração de contexto. Outro critério de sucesso, nível AA, está relacionado ao fato de que os mecanismos de navegação que se repetem em várias páginas web, dentro de um conjunto de páginas, mantenham a mesma ordem relativa em cada ocorrência, a menos que uma mudança seja iniciada pelo usuário.
3. **Assistência de Entrada:** esta diretriz está relacionada à auxiliar os usuários na prevenção e correção de erros. Um exemplo de critério de sucesso, nível A, para esta diretriz é que se um erro de entrada for automaticamente detectado, o item que apresenta erro deve ser identificado e o erro descrito para o usuário em texto. Nesse sentido, também configura-se como critério de sucesso, nível AA, se um erro de entrada for automaticamente identificado e sugestões de correção forem conhecidas, essas sugestões devem ser fornecidas ao usuário, a menos que isso comprometa a segurança ou o propósito do conteúdo.

3.2.4 Princípio Robusto

O princípio Robusto corresponde à possibilidade do conteúdo ser suficientemente robusto para ser interpretado de maneira consistente por diversos agentes de usuário (elementos de software que atuam em nome de um usuário), abrangendo tecnologias assistivas. Para esse princípio são elencadas as seguintes diretrizes:

Compatível: aprimorar a compatibilidade entre os agentes de usuário atuais e futuros, englobando tecnologias assistivas. Um exemplo de critério de sucesso, nível A, para esta diretriz é que para conteúdos desenvolvidos por meio de linguagens de marcação, por exemplo HTML, a semântica do código deve seguir os padrões definidos para a linguagem. As *tags* de marcação devem possuir abertura e fechamento, sendo adequadamente aninhadas conforme as especificações correspondentes, não apresentando atributos duplicados, os IDs devem ser exclusivos, a menos que as especificações permitam essas características.

É relevante destacar que para cada diretriz há uma variedade de critérios de sucesso, os quais podem ser consultados em <<https://www.w3c.br/traducoes/wcag/wcag21-pt-BR/>>. Os critérios apresentados neste trabalho são exemplos que visam facilitar a compreensão de sua natureza e categorização nos níveis A, AA e AAA.

A WCAG (em todas as versões) trata da acessibilidade de forma ampla e para todos. Porém dado o contexto desta pesquisa, buscou-se na literatura questões de acessibilidade que possam impactar especialmente pessoas cegas, identificando problemas e soluções de acessibilidade que possam afetar a esse grupo de usuários. Sendo assim, na seção 3.3 será apresentado um panorama geral da acessibilidade para cegos em ambientes baseados na Web.

3.3 Acessibilidade para cegos em sistemas Web

As pessoas com deficiência visual severa utilizam leitores de tela para utilizar sistemas computacionais. Esses sistemas incluem aplicativos, sistemas operacionais, software utilitário entre outros. Com isso, os leitores de tela tornam-se ferramentas essenciais para a realização de tarefas cotidianas que envolvem tais sistemas.

Os leitores de tela convertem o conteúdo digital em voz, além de oferecer uma série de funcionalidades para que seus usuários possam navegar e interagir com os aplicativos. Uma pesquisa realizada pela WebAim (2019) aponta o NVDA (*Non Visual Desktop Access*)¹, JAWS² e VoiceOver³ como os leitores de tela mais usados no mundo para sistemas desktop. No Brasil, uma pesquisa realizada pela Everis (2019) também aponta esses três leitores como os mais utilizados. A tabela 1 mostra os resultados comparativos entre as pesquisas realizadas pela WebAIM e Everis Brasil, bem como algumas especificações de cada leitor.

¹<https://www.nvaccess.org/about-nv-access/>

²<https://www.tecassistiva.com.br/catalogo/jaws/>

³<https://support.apple.com/pt-br/guide/iphone/iph3e2e415f/ios>

Vale ressaltar que embora exista o leitor de tela Orca (leitor de tela para Linux) ¹, este leitor não foi contemplado nas pesquisas citadas.

Tabela 1: Leitores de tela mais usados no Brasil e no mundo

Leitor de Tela	Sistema Operacional	Custo	% de respostas sobre o uso	
			No Brasil	No mundo
NVDA	Windows	Gratuito	69,2%	40.6%
JAWS	Windows	Pago	15,7%	40.1%
VoiceOver	Mac OS	Gratuito	7,1%	12.9%

Os resultados da pesquisa mostram que 77% dos brasileiros usam leitores de tela gratuitos, como NVDA e VoiceOver, enquanto a tendência global aponta para 53,5%. As pesquisas realizadas também abordam quais navegadores são mais utilizados com leitores de tela. Segundo a WebAim (2019), o Google Chrome é utilizado por 44,4% dos usuários, Firefox por 27,4%, Internet Explorer 11 por 27,4%, seguido de outros navegadores com porcentagens menores. No Brasil, uma pesquisa realizada pela Everis (2019) mostra que mais de 62% das pessoas cegas usam o NVDA com Google Chrome ou Firefox. Os resultados da pesquisa indicam que a popularidade do Google Chrome, combinada com o custo-benefício do NVDA, tornou o uso dessas tecnologias cada vez mais difundido. Entretanto, para a acessibilidade de sistemas interativos, o uso de leitores de tela não é suficiente, sendo necessário que esses ambientes estejam adequados para interagir com eles, adequando-se as diretrizes de acessibilidade da WCAG (NASCIMENTO et al., 2019a; MILNE; LADNER, 2018).

A acessibilidade é um fator preponderante para que pessoas cegas interajam e naveguem em sistemas computacionais. Esses sistemas são considerados acessíveis quando proporcionam amplo acesso, levando em consideração a autonomia de seus usuários (CONSORTIUM et al., 2013). As atividades de trabalho, educação e lazer, entre outras, são diretamente impactadas se os ambientes propostos não forem acessíveis.

Nesta seção, as barreiras e as recomendações de acessibilidade para cegos foram categorizadas em navegação (subseção 3.3.1), representações de conteúdos não textuais (subseção 3.3.2) e percepção e compreensão de alerta sonoros (subseção 3.3.3).

3.3.1 Navegação

Segundo Billah et al. (2017), existem diferenças em como os indivíduos cegos e pessoas que enxergam navegam em um ambiente baseado na web. O tempo que cegos com uso de

¹https://help.gnome.org/users/orca/stable/introduction.html.pt_BR

leitores levam para executar uma tarefa é relativamente maior que aqueles que enxergam. Essa diferença pode ser ainda maior, caso o sistema não esteja adequado para interagir com os leitores de telas. A falta de indicações e informações para orientar pessoas que utilizam leitores de tela ao interagir com sistemas computacionais pode dificultar a rota de navegação contextual, além de gerar uma carga cognitiva maior para se compreender e se localizar nesses ambientes. Os leitores de tela não serão eficazes para a navegação se os elementos da interface não estiverem bem estruturados ou não tiverem uma boa semântica, dando significados claros aos seus elementos.

As páginas da Web em Linguagem de Marcação de Hipertexto (HTML) (do inglês, *HyperText Markup Language*) podem ser estruturadas usando elementos como `<div>`, usados para separar blocos de informações. Porém se esses elementos não tiverem uma identificação semântica, relacionada ao seu uso, não será possível entender sua função. Uma estratégia para mitigar esse problema é o uso de WAI-ARIA (*Web Accessibility Initiative - Accessible Rich Internet Applications*) ¹. A WAI-ARIA fornece atributos HTML para dar significado aos elementos de interação em uma página da web. Por exemplo, o atributo `role` pode ser usado para indicar o papel de um elemento na estrutura de uma página da web. Por exemplo:

```
<div class="post"role="article"/> <p> text</p> </div>
```

No código acima, o atributo `role` (função) significa um bloco de texto. Em um outro exemplo, no código abaixo, o atributo `role` significando uma barra de menu:

```
<ul role="menubar">
  <li role="menuitem"> Item 1</li>
  <li role="menuitem"> Item 2</li>
  <li role="menuitem"> Item 2</li>
</ul>
```

Desde o HTML 5², foram criados elementos que podem facilitar a estruturação do código. Esses elementos já possuem um significado semântico, o que pode facilitar muito a estruturação da página e a interação com os leitores de tela. Entre esses elementos estão: `<header>`, para cabeçalhos; `<nav>`, para agrupar elementos de navegação; `<main>`, para

¹<https://www.w3.org/WAI/standards-guidelines/aria/>

²<https://developer.mozilla.org/pt-BR/docs/Web/HTML>

mapear o conteúdo principal; `<article>`, para mapear um bloco de conteúdo; `<aside>`, para conteúdo suplementar; `<footer>`, para definir um rodapé.

Não basta ter uma página web bem estruturada. Além disso, é necessária uma linguagem clara e compreensível para links e botões. Estudos têm mostrado que os desafios de navegação contextual podem ser mitigados usando uma linguagem apropriada, aplicada ao ambiente baseado na web (SCHEUERMAN et al., 2017).

A flexibilidade do HTML 5 permite que os elementos HTML possam ser usados para diferentes funções, em diferentes contextos. Portanto, se esses elementos não tiverem propriedades que lhes deem significado ou semântica extra, podem criar uma barreira de navegação, pois o leitor de tela conseguirá acessar o elemento, mas não conseguirá identificar sua real função. Por exemplo, o código abaixo mostra o código HTML para um botão Fechar:

```
<button aria-label="close"> X</button>
```

No código, apenas o X como rótulo do botão é mostrado ao usuário, portanto, sua real função não é explícita. Frequentemente, para esses botões, os ícones também são usados sem nenhum significado semântico ou textual além de seu significado visual. Por esse motivo, você pode usar o atributo *aria-label* para definir um significado textual para esses elementos.

Alguns elementos da interface, como barras de navegação e menus, possuem uma estrutura hierárquica. Navegar nessa estrutura pode ser desafiador para usuários de leitores de tela. Ferramentas com navegação linear podem tornar a navegação uma tarefa árdua, aumentando a carga cognitiva dos usuários (KHURANA et al., 2018). Além disso, as estruturas visuais bidimensionais presentes nas páginas web são facilmente perdidas, pois o alinhamento com o espaço e as pistas visuais é um desafio para a representação em formato linear. A utilização de teclas de atalho, âncoras e *links* de acesso rápido podem tornar a navegação mais fluida, evitando que o usuário do leitor de tela tenha que navegar por toda a estrutura da página para acessar informações ou funcionalidades (W3C, 2018; NASCIMENTO et al., 2019b).

As tabelas podem ser utilizadas para estruturar conteúdos ou mesmo definir o *layout* de páginas web, apesar de ser mais indicado, atualmente, pela W3C o uso da abordagem *tableless* (layout sem tabelas) ¹. A utilização de tabelas para estruturação de conteúdo em páginas web, além de não ser uma boa prática, também pode ser um grande desafio

¹<https://www.w3.org/WAI/tutorials/tables/>

para cegos, pois como os leitores de tela interpretam as tabelas linha por linha, o conteúdo será lido fora de ordem, prejudicando a navegação e o entendimento (GERALDO, 2016). Além disso, o uso de tabelas para tabulação de dados sem a semântica adequada também pode dificultar a compreensão dos dados (W3C, 2008).

Para mitigar os problemas de navegação com leitores de tela sobre tabelas, é importante definir uma estrutura semântica para que os leitores de tela consigam distinguir os cabeçalhos dos conteúdos e fazer as devidas associações. O W3C em seu Tutorial de Acessibilidade na Web (W3C, 2008) propõe que as células do cabeçalho da tabela sejam marcadas com `<th>` e as células de dados com `<td>`. Para tabelas mais complexas, associações explícitas devem ser feitas usando os atributos `scope`, `id` e `headers`. A seguir, é apresentado um exemplo de tabela em HTML, seguindo boas práticas para acessibilidade.

```
<table>
  <caption>Lista de clientes</caption>
  <tr>
    <th scope="col">Primeiro Nome</th>
    <th scope="col">Sobrenome</th>
    <th scope="col">Cidade</th>
  </tr>
  <tr>
    <td>Marcos</td>
    <td>Nascimento</td>
    <td>Fortaleza</td>
  </tr>
  <tr>
    <td>Marcelo</td>
    <td>Silva</td>
    <td>Fortaleza</td>
  </tr>
  <tr>
    <td>Maria</td>
    <td>Soares</td>
    <td>Fortaleza</td>
  </tr>
</table>
```

No exemplo citado para tabela acessível, é apresentada uma tabela com três colunas, com os cabeçalhos: "Nome", "Sobrenome" e "Cidade". Os dados referentes a cada cabeçalho estão dispostos nas linhas abaixo, respeitando a estrutura das colunas, ou seja, cada coluna corresponde aos dados referentes a cada cabeçalho. Neste exemplo, o atributo `scope` com o valor `col` define a direção das células de cabeçalho e as associa às células de dados correspondentes. O elemento `<caption>` funciona como o título da tabela. Para tabelas complexas, sugere-se também a utilização do atributo `summary` para o elemento `<table>` (`<table summary="texto descritivo" >`) com informações sobre a organização dos dados (W3C, 2019). Dessa forma, embora as tabelas ainda possam apresentar certa complexidade para navegação com leitores de tela, a adequação dessas tabelas para uma estrutura acessível pode amenizar os problemas de navegação e compreensão dos dados.

3.3.2 Representações de conteúdos não textuais

Um dos grandes desafios enfrentados por quem usa leitores de tela, em ambientes computacionais baseados na web, tem sido a falta de texto alternativo associado a conteúdos não textuais como imagens, gráficos, quadros, tabelas, mapas, formulários e conteúdo interativo. Esses tipos de conteúdos tornam-se imperceptíveis para usuários de leitores de tela, restringindo o acesso às informações passadas por eles (PACIELLO, 2000; CONSORTIUM et al., 2013). Para tornar conteúdos não textuais acessíveis, devem ser incluídos atributos descritivos nas *tags* HTML. Caso as imagens sejam meramente decorativas, sugere-se que sejam colocadas como plano de fundo, já que ela não será relevante para cegos (W3C, 2008; PERIŠA et al., 2012). Para descrever um elemento não textual pode-se utilizar o atributo `alt` (para texto alternativo) do HTML. No código HTML, a seguir, é apresentado um exemplo da aplicação desse atributo.

```

```

No código, quando o leitor de tela encontrar a imagem, ele irá ler o conteúdo contido no atributo `alt`. Esse atributo deve fornecer uma representação descritiva da imagem e o que ela transmite visualmente. Caso seja necessário fornecer informações contextuais extras, poderá ser utilizado o atributo de título (`title`). Nesse caso, a maioria dos leitores de tela farão a leitura do texto alternativo, seguido pelo atributo de título e o nome do arquivo. Outra vantagem de uso do atributo `title` é que eles podem ser utilizados para apresentar dicas de ferramentas para as pessoas que enxergam, pois exibem o texto do

título quando estão sobre o mouse. No código abaixo é apresentado um exemplo prático de uso do atributo `title`.

```
<ul>
  <li>
    <a href="#" title="Voltar para pagina inicial"> Inicio </a>
  </li>
  <li>
    <a href="#">Quem Somos</a>
  </li>
  <li>
    <a href="#">Contatos</a>
  </li>
</ul>
```

O avanço da tecnologia possibilitou a criação de legendas automáticas para imagens, mas essas legendas costumam ser imprecisas e não incluem a quantidade e os tipos de detalhes desejados por muitos usuários de leitores de tela. Como cegos não são capazes de usar a imagem visual como ponto de referência, então as legendas devem estar alinhadas com sua cultura e realidade (MACLEOD et al., 2017; MORRIS et al., 2018; DARIN et al., 2017).

3.3.3 Percepção e compreensão de alertas sonoros

No decorrer da interação entre indivíduos cegos e sistemas computacionais, é recorrente a ocorrência de problemas relacionados à perceptibilidade e compreensão. Em diversas situações, os elementos da interface e as notificações são perceptíveis por meio do leitor de tela; entretanto, a finalidade subjacente desses elementos, assim como as mensagens associadas às notificações, frequentemente não são compreensíveis para os usuários cegos (ZHAO et al., 2018; DARIN et al., 2017; BROCK et al., 2018). A título de exemplo, a manifestação de um erro durante a interação com o sistema pode resultar em um alerta sonoro incompreensível para um usuário cego.

Os signos sonoros combinam sons que representam uma ação do usuário ou do sistema para alertá-lo sobre algo. Embora os alertas sonoros possam ser muito úteis, pode ser difícil para os usuários estabelecer uma relação entre o som e a ação por ele representada

(DARIN et al., 2017). Além disso, o abuso de alertas sonoros pode gerar uma “poluição auditiva” e, assim, dificultar a interação. Para que esses alertas sonoros sejam percebidos e compreendidos é necessário levar em consideração as experiências prévias dos usuários, para que possam associar o signo sonoro à ação representada por ele.

Nesta seção foram apresentadas diversas soluções de acessibilidade para que usuários cegos possam interagir com aplicações web. De forma mais ampla, percebe-se que uma série de adequações são necessárias para usuários cegos possam utilizar serviços, produtos, informações entre outras utilidades que a web pode proporcionar. Nesse sentido, levando em consideração sistemas de programação visual, na seção seguinte serão abordados os desafios de acessibilidade que pessoas cegas podem encontrar ao interagir com sistemas de programação visual.

3.4 Desafios de acessibilidade para cegos em sistemas de programação visual

Os sistemas de programação visual em blocos, exemplificadas por Scratch, Blockly, iVProg, entre outros similares, têm experimentado uma disseminação extensiva no âmbito educacional. Eles têm sido empregados para o ensino e aprendizado do pensamento computacional, uma competência progressivamente incorporada às bases curriculares do ensino pré-universitário, conforme discutido por Ludi (2015).

Apesar do grande potencial educacional dos sistemas de programação visual, eles possuem algumas barreiras de acessibilidade que podem impedir ou limitar sua utilidade para pessoas cegas. Um dos principais problemas é que as tecnologias usadas para desenvolvê-los não são bem adequadas para leitores de tela. Além disso, a movimentação dos elementos é restrita ao uso do mouse, através de cliques e arrastes. Adicionalmente, o forte apelo visual na transmissão da informações pode ser imperceptível com leitores de tela. Esses obstáculos precisam ser tratados para garantir que esses sistemas sejam acessíveis para todos, promovendo a igualdade no aprendizado.

Em uma pesquisa conduzida por Milne e Ladner (2018), a acessibilidade de nove sistemas de programação visual foi avaliada. Dentre esses sistemas, incluíram-se aplicações web, a saber: Scratch, Blockly, Blockly Acessível, Tynker e Snap; e aplicações móveis, incluindo Blockly (Android), Scratch Jr (iOS), Hopscotch (iOS) e Tickle (iOS). A avaliação teve como objetivo verificar a conformidade dessas ferramentas com as diretrizes de acessibilidade para o conteúdo da web, especificamente as "Content Accessibility Guide-

lines"(WCAG) 2.0. Além disso, foram realizadas análises com leitores de tela. Para tal análise, empregaram-se os leitores de tela NVDA para as aplicações web; o *VoiceOver*, para aplicativos iOS e o *Talkback* para aplicações Android¹.

Os resultados da pesquisa realizada por Milne e Ladner (2018) revelaram que, em grande parte das aplicações, os blocos não eram perceptíveis para uso efetivo com leitores de tela. Com exceção do Blockly Acessível, todas as ferramentas adotavam a técnica de interação por meio de arrastar e soltar, mostrando-se inviável para usuários de leitores de tela. Também foram identificadas dificuldades na assimilação e determinação da estrutura lógica dos blocos aninhados. Com exceção do Blockly Acessível, as ferramentas apresentavam as informações por meio de formas e cores, imperceptíveis com uso de leitores de tela.

Para que usuários cegos também possam se beneficiar do paradigma de programação visual, pesquisadores têm proposto soluções acessíveis para essa abordagem. Nesse sentido, na seção 3.4.1, serão apresentados sistemas de programação visual concebidos com o propósito de proporcionar acessibilidade a indivíduos cegos. Além de expor as características funcionais destes sistemas, apontando suas limitações no que concerne ao ambiente de aplicação e aos perfis dos usuários.

3.4.1 Sistemas de programação visual acessíveis e suas limitações

Com o intuito de viabilizar o uso da programação visual por pessoas cegas, algumas alternativas têm sido propostas. Dentre elas, destaca-se uma versão acessível do Blockly (LUDI, 2015) e o Blocks4All (MILNE, 2017). As características, recursos de acessibilidade e limitações dessas alternativas serão abordados nas subseções 3.4.1.1 e 3.4.1.2.

3.4.1.1 Blockly adaptado para cegos

O Blockly incorpora uma variedade de elementos que podem ser instrumentalizados para a criação de versões acessíveis destinadas a indivíduos com deficiência visual. A utilização de HTML, CSS, Javascript e SVG em seu desenvolvimento possibilita a implementação de recursos de acessibilidade. A estruturação em HTML, por exemplo, facilita a inclusão de especificações para Aplicações de Internet Rica Acessíveis (ARIA) (MDN, 2023), descrevendo detalhadamente os elementos e suas funcionalidades. Tal abordagem visa proporcionar aos usuários que dependem de leitores de tela uma percepção e com-

¹<https://support.google.com/accessibility/android/answer/6007100?hl=pt-BR>

preensão adequadas dos componentes da interface. Cabe ressaltar que, para efetivar essa adaptação, é imperativo que os desenvolvedores incorporem essas modificações, uma vez que o Blockly não contempla intrinsecamente essa funcionalidade em seu núcleo. As tecnologias empregadas no desenvolvimento do Blockly, aliadas à sua licença Apache 2.0¹, conferiram-lhe a aptidão para servir como um framework viável na elaboração de versões adaptadas e acessíveis destinadas a usuários cegos.

Com o objetivo de tornar o Blockly acessível para usuários com deficiência visual, Mountapmbeme et al. (2022) propuseram uma versão modificada da ferramenta. Nessa versão, foi introduzido o módulo teclado, oferecendo uma alternativa para pessoas que dependem de leitores de tela para navegar pelos blocos. Essa implementação, desenvolvida em JavaScript, encontra-se disponível como um projeto de código aberto, com seu repositório alcançável através do seguinte endereço: <<https://github.com/RITAccess/blockly>>.

Na sua concepção inicial, o Blockly era exclusivamente orientado para interações realizadas por meio do mouse. No entanto, com a introdução do módulo de teclado, o usuário ganha a capacidade de percorrer as categorias de blocos e blocos individuais na caixa de ferramentas usando as teclas do teclado. Além disso, é possível habilitar operações de arrastar e soltar pressionando apenas algumas teclas, simulando a navegação espacial. Essa abordagem permite que o usuário navegue em um programa de cima para baixo, da esquerda para a direita, e entre e fora de blocos aninhados ou contêineres. A figura 4 ilustra os componentes de acessibilidade presentes na versão acessível do Blockly.

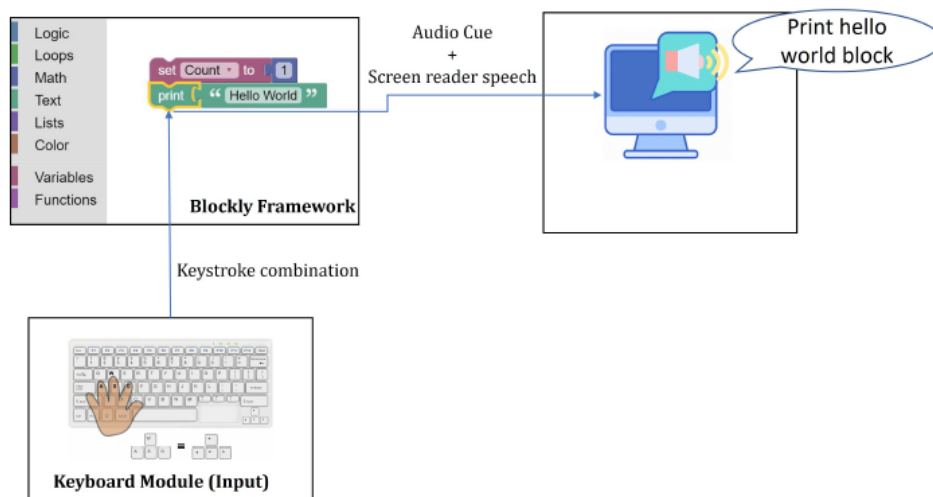


Figura 4: Componentes de acessibilidade do Accessible Blockly
Fonte: Adaptado de Mountapmbeme et al. (2022)

Para transferir os blocos da caixa de ferramentas para a área de trabalho, foi empre-

¹<https://www.apache.org/licenses/LICENSE-2.0>

gada uma técnica que simula a ação de arrastar e soltar, habitualmente realizada com o *mouse*. Inicialmente, o usuário seleciona um local na área de trabalho onde pretende adicionar um novo bloco, utilizando um cursor virtual. Posteriormente, o usuário navega até o menu de ferramentas e seleciona o bloco desejado, adicionando-o ao local previamente indicado com uma tecla específica. A navegação por meio do cursor virtual concede ao usuário a capacidade de alternar entre a caixa de ferramentas e a área de trabalho utilizando exclusivamente as teclas do teclado, de modo análogo à forma como os menus são acessados por meio do leitor de tela.

Conforme apontado por Mountapmbeme et al. (2022), não há, atualmente, um padrão estabelecido para a utilização de texto alternativo que torne os blocos facilmente acessíveis por leitores de tela. Para abordar essa lacuna, foram criadas descrições verbais para cada bloco, utilizando a atribuição de `aria-label`, com o intuito de descrever as representações individuais de cada elemento. Adicionalmente, os pontos de conexão entre os blocos são destacados por meio de informações audíveis. Em outras palavras, quando o foco está em um ponto de conexão específico, uma mensagem de áudio, como "conexão superior", é fornecida para informar sobre a natureza da conexão.

Foi realizado um experimento envolvendo 12 participantes cegos, 10 do sexo masculino, dois do sexo feminino, com idades entre 22 e 60 anos. Todos os participantes possuíam alguma experiência com programação, conforme indicado por Mountapmbeme et al. (2022). Durante esse experimento, os participantes realizaram tarefas de navegação utilizando a versão acessível do Blockly, sendo comparado ao esquema alternativo de navegação do Blockly utilizado como controle.

Os resultados indicam que, embora o Blockly acessível tenha despertado interesse entre os usuários, algumas melhorias relacionadas ao feedback fornecido pelos leitores de tela são necessárias. Por outro lado, os participantes consideraram a estratégia de navegação acessível. Adicionalmente, os participantes expressaram o desejo de contar com um módulo de revisão que lhes permitisse ouvir rapidamente a descrição completa do programa ou de partes específicas. Conforme Mountapmbeme et al. (2022), essa funcionalidade proporcionaria uma vantagem antes de iniciar a exploração dos blocos individuais.

3.4.1.2 Blocks4All

Blocks4All é um sistema de programação visual, proposto por Milne e Ladner (2018), com o intuito de promover o desenvolvimento das habilidades de pensamento computacional em crianças com deficiência visual. Na versão específica para iPad, o Blocks4All utiliza

metáforas, elementos visuais e manipulação de objetos para construir uma estrutura lógica que conduza à execução de ações específicas, resultando em informações acessíveis. A título de exemplo, foi criado um ambiente baseado em blocos para controlar o robô Dash (HUNSAKER,), proporcionando uma saída tangível de informações acessíveis às crianças com deficiência visual.

Com o Blocks4All os usuários podem movimentar robô Dash e acionar emissões sonoras. Para realizar tarefas como mover o robô ou emitir sons, os usuários dispõem de blocos para operações, números (utilizados em conjunto com laços de repetição) e declarações booleanas (utilizadas em conjunto com declarações "SE"). A estrutura do Blocks4All compreende uma caixa de ferramentas (*Toolbox*), um espaço de trabalho (*Workspace*) e a execução do programa (*Run program*), conforme ilustrado na figura 5.



Figura 5: Interface Blocks4all
Fonte: Milne e Ladner (2018)

No estudo conduzido por Milne e Ladner (2018), foram identificadas algumas barreiras de acessibilidade associadas à programação visual utilizando blocos, bem como as soluções adotadas no Blocks4All para superar cada uma dessas barreiras. As barreiras de acessibilidade e as respectivas soluções propostas são:

- **Acessar a saída:** para superar esta barreira de acessibilidade, uma saída tangível foi usada (Ex. movimento ou emissão de som realizada pelo robô);
- **Elementos de acesso:** os blocos são acessados via VoiceOver (leitor de tela integrado ao iOS); ao acessar um bloco, o usuário é informado do nome, localização e tipo do bloco (Ex. “vá em frente, bloco 2 de 4 na área de trabalho, ação”); foram disponibilizadas dicas de como manipular os elementos (Ex. “toque duas vezes para

mover o bloco”); os blocos são colocados na parte inferior da tela para facilitar a localização.

- **Mover os blocos:** os blocos podem ser movidos selecionando o bloco e, em seguida, indicando por meio de um clique na área de trabalho (*Workspace*) onde o bloco selecionado deverá ser inserido. As ações de seleção de um bloco e indicação de onde ele será inserido é informada para os usuário por meio de áudio.
- **Estrutura espacial dos blocos:** os blocos a serem embutidos podem ser acessados a partir de um menu embutido em cada bloco. Quando um bloco é selecionado na caixa de ferramentas, os blocos na área de trabalho que o aceitam reproduzem os mesmos sinais de áudio.
- **Tipo de informação:** o blocos que correspondem a valores numéricos ou booleanos são selecionados no menu, eles reproduzem sinais de áudios distintos, sendo duas notas baixas curtas para os blocos números e duas notas altas curtas para blocos booleanos. Há também uma distinção visual nas formas dos blocos. Por exemplo, o bloco correspondente a declaração "SE" têm uma aba superior triangular e o booleano é um triângulo invertido para indicar que se encaixam. O bloco de repetição têm uma aba superior retangular e o numérico corresponde à um retângulo invertido.

Para avaliar a usabilidade do Blocks4All, um estudo foi conduzido com 5 (cinco) crianças com deficiência visual, sendo 1 cega e 4 com baixa visão (MILNE; LADNER, 2018). As crianças participaram de uma a quatro sessões, cada uma com duração de 60 a 90 minutos, utilizando um robô Dash com o aplicativo executado em um iPad com iOS 10.3. Os resultados do estudo evidenciaram que:

- Os blocos eram mais facilmente acessados quando estavam alinhados na parte inferior da tela, eram redimensionáveis, separados por espaço em branco e acessíveis com o VoiceOver por meio de um teclado;
- Nenhuma das crianças com baixa visão utilizou a função de aproximação (*zoom*) para visualizar os blocos, preferindo o uso do VoiceOver ou segurar o iPad a centímetros do rosto;
- Em entrevistas, as crianças relataram dificuldades ao utilizar o recurso de *zoom*, devido à necessidade de alternar entre uma visão geral do aplicativo e o foco em um elemento específico;

- Todas as crianças enfrentaram desafios com o método guiado por áudio, utilizado para arrastar e soltar;
- Aquelas que mantiveram o iPad próximo ao rosto encontraram facilidade no método de arrastar e soltar;
- O método de primeiro selecionar o local, em seguida, selecionar o bloco e, por último, clicar no local escolhido mostrou-se eficaz, permitindo que as crianças completassem todas as tarefas;
- As dicas de áudio demonstraram eficiência, uma vez que os participantes puderam compreender tanto o espaço quanto as representações sonoras da estrutura do programa.

Embora o estudo apresente achados importantes que podem ser um caminho para a acessibilidade em sistemas de programação visual, os resultados do experimento apresentam lacunas e oportunidades de aprofundamento no estudo. De fato, a dificuldade de recrutamento de participantes fez com que fosse recrutado apenas uma pequena amostra de participantes com ampla faixa etária e habilidades. Isso dificultou as comparações entre os participantes. Além disso, de acordo com Milne e Ladner (2018), ainda há questões a serem exploradas, como como navegar em hierarquias mais complexas de código aninhado e a melhor forma de incorporar comandos ou outros gestos.

3.5 Design Participativo, Protótipos e Design Universal

Esta seção explora os principais aspectos do Design Participativo, ressaltando sua importância e os benefícios associados para esta abordagem. Também aponta o uso de protótipos no processo de design participativo como uma ferramenta fundamental para levantar requisitos e *feedback* dos usuários, durante o desenvolvimento de software. Além disso, são abordados os princípios do Design Universal como requisitos fundamentais para construção de soluções que sejam acessíveis para todos.

3.5.1 Design participativo e o uso de protótipos

No âmbito do desenvolvimento de software, o Design Participativo refere-se à contínua interação com um grupo de usuários considerados como representativos da população-alvo, garantindo acesso constante a suas experiências e perspectivas (BARBOSA; SILVA,

2010; CAMARGO; FAZANI, 2014; STILL; CRANE, 2017). Essa abordagem vai além da simples coleta de feedback, incorporando a participação ativa dos usuários desde as fases iniciais do processo, garantindo que suas experiências, necessidades e perspectivas sejam incorporadas ao longo de todo o ciclo de desenvolvimento.

De acordo com Camargo e Fazani (2014), existem algumas práticas que podem ser consideradas para a aplicação do design participativo. Entre elas estão os depoimentos, oficinas, maquetes, prototipação, entre outras. Por exemplo, Nielsen (1994) enfatiza a necessidade da produção de protótipos, maquetes ou esboços de tela para que as amostras de usuários possam reagir à essas propostas, fornecendo informações relevantes, além de levantar novas questões a acerca das soluções propostas. Borysowich (2007) defende que a utilização de protótipos se destaca em relação aos modelos conceituais e formalismos, uma vez que ativa o conhecimento tácito da atividade, tornando mais fácil a visualização de possíveis cenários futuros, inclusive, no que diz respeito à estrutura social da própria atividade.

Conforme Borysowich (2007), os protótipos podem assumir diferentes formas, como o protótipo de conceito, no qual são delineadas as abordagens do sistema; o protótipo de viabilidade, que avalia a viabilidade de diversas soluções; o protótipo horizontal, que esclarece o escopo e os requisitos ao apresentar vários modelos, embora com poucos detalhes, para testar funções comuns realizadas frequentemente pelos usuários (considerado um protótipo global que abrange todo o sistema em um modelo expandido); o protótipo vertical, que aprimora os requisitos ao oferecer modelos detalhados de algumas características específicas (visto como um protótipo local envolvendo um único componente do sistema); e o protótipo funcional, que determina as sequências utilizáveis para a apresentação de informações.

Durante o processo de design participativo, é importante atentar para o fato de que, embora sejam consideradas amostras de um público-alvo, é necessário levar em consideração os diversos perfis de usuários para esta amostra. Por exemplo, em um grupo de usuários, pode haver pessoas com ou sem deficiência. Com isso, é necessário considerar aspectos relacionados ao Design Universal. Esse assunto será apresentado na subseção a seguir.

3.5.2 Design Universal

As recomendações de acessibilidade propostas pela WCAG (W3C, 2018) definem uma série de critérios aplicados a aplicações web que podem tornar o uso de determinados

softwares mais universal, ou seja, atender a um amplo número de usuários, independentemente de suas limitações e habilidades. Seguindo esse mesmo princípio, o Design Universal tem como objetivo promover a inclusão de todas as pessoas, em seus diferentes perfis, seja com ou sem deficiência, em todas as atividades por meio de um design acessível para todos (PUPO et al., 2006). O design universal pode ser aplicado a ambientes, produtos, serviços, programas e tecnologias acessíveis, visando atender ao maior número de pessoas na maior extensão possível, sem a necessidade de adaptação ou design especializado.

Conforme aponta PUPO et al. (2006), o design universal define sete princípios. Esses princípios correspondem as características a serem aplicadas à ambientes, produtos, serviços, softwares e tecnologias acessíveis. Os princípios são:

Uso de forma equânime (em igual condições): este princípio está relacionado à utilidade e comercialização para pessoas com habilidades distintas. A forma de utilização deve ser atraente para todos, seguindo os mesmos padrões ou equivalente quando não for possível. Além disso, deve-se evitar a marginalização e estigmatização dos usuários, promovendo privacidade, proteção e segurança de forma equânime.

Flexibilidade no Uso: o princípio da flexibilidade no uso visa atender à diversidade e preferências, levando em consideração as habilidades individuais. Para isso, deve ser disponibilizada formas de utilização que atendam aos diferentes perfis de pessoas e suas habilidades. Também, deve-se facilitar a precisão, exatidão e adaptabilidade do usuário. Por exemplo, pessoas destros e canhotos que possuem formas diferentes de utilização de um objeto devem ter flexibilidade para utilizar um objeto conforme suas habilidades.

Uso Simples e Intuitivo: este princípio define que para ser intuitivo, o uso deve ser facilmente compreendido, independentemente da experiência do usuário, formação, idioma ou capacidade de concentração. Para que o Design seja intuitivo, é necessário evitar complexidade desnecessária, manter um padrão lógico que atenda ao modelo mental do usuário, abranger diversos perfis de usuários com diferentes níveis e tipos de formação e proporcionar boa comunicabilidade com o usuário, fornecendo retorno para suas ações.

Informação Perceptível: para atender a este princípio, a informação deve ser transmitida de forma eficaz, independente das condições ambientais, levando em consideração a capacidade sensorial do usuário. Para isso, a informação deve ser transmitida por diferentes meios ou formas, como tátil e verbal.

Tolerância ao Erro: este princípio está relacionado a minimizar o risco e as consequências de ações involuntárias ou imprevistas. Para isso, podem ser disponibilizados elementos como avisos de riscos e erros, além de recursos de segurança.

Baixo Esforço Físico: este princípio visa proporcionar eficiência e conforto durante o uso, evitando a fadiga. Para isso, deve-se levar em consideração questões ergonômicas, evitando o uso de força excessiva, operações repetitivas e esforço físico contínuo.

Tamanho e Espaço para Aproximação e Uso: conforme este princípio, deve-se possibilitar o uso para um amplo número de usuários, independentemente da estatura, postura ou mobilidade da pessoa. Para isso, deve ser proporcionado espaço e dimensões apropriados para interação, alcance, manipulação e uso.

Em suma, a importância do Design Universal revela-se como uma peça-chave na construção de ambientes, produtos, serviços, softwares e tecnologias que transcendem barreiras e promovem a inclusão de todos os usuários, independentemente de suas habilidades ou limitações. Ao incorporar os princípios do Design Universal, as soluções tornam-se mais acessíveis, intuitivas e eficazes, refletindo um compromisso genuíno com a diversidade e proporcionando experiências positivas para um amplo espectro de usuários.

A abordagem do Design Universal não apenas atende às necessidades específicas de grupos marginalizados, como pessoas com deficiência, mas também enriquece a experiência para todos os usuários. Sua aplicação fomenta a criação de ambientes mais equitativos, onde a flexibilidade, simplicidade e tolerância ao erro não são apenas princípios de design, mas também valores que fortalecem a interação humana e a participação de todos na sociedade digital.

3.6 Avaliação de Acessibilidade, Usabilidade e Análise de Desempenho

Nesta seção, serão abordados métodos para avaliação de acessibilidade, usabilidade e desempenho em ambientes web. Inicialmente, é discutido a importância da detecção de obstáculos de acessibilidade por meio de métodos automáticos e de inspeção, complementados por abordagens centradas no usuário. Em seguida, são apresentadas técnicas para avaliação de usabilidade, destacando métodos empíricos como sessões de observação, questionários e pensamento em voz alta. Por fim, adentramos a análise de desempenho, exemplificando a comparação entre grupos de usuários em tarefas específicas.

3.6.1 Avaliação de acessibilidade

De acordo com Junior (2023), a avaliação de acessibilidade na web detecta obstáculos no acesso aos sites para que esses problemas possam ser corrigidos pelos desenvolvedores. Tannus et al. (2021) apontam que os métodos para avaliação de acessibilidade visam identificar questões de acessibilidade, como violações de diretrizes, deficiências na interface ou indicadores de desempenho do usuário, os quais são sintomáticos de níveis deficientes de acessibilidade.

Segundo Barbosa e Silva (2010) os métodos de inspeção permitem avaliar a conformidade com um padrão ou guia de estilo pré-definidos. Com isso, entende-se que esse método também pode ser aplicado para avaliar a acessibilidade de aplicações web, analisando sua conformidade com as diretrizes de acessibilidade definidas na WCAG. Existem ferramentas que fazem essa inspeção de forma automática. Entre elas estão o Accessmonitor¹ e o ASES². Essas ferramentas funcionam como validadores, fazendo uma varredura no código HTML, afim de verificar se os critérios de acessibilidade foram atendidos. Ao inspecionar um site com esses validadores, são obtidas notas em uma escala de 0 a 10 no Accessmonitor e 0% a 100% no ASES. Além disso, as ferramentas apontam os critérios de acessibilidade que não foram atendidos para que possam ser implementados.

Embora a valiação com validadores automáticas sejam essenciais, é importante que este método seja aliado outros métodos, envolvendo a participação direta de pessoas com deficiência. Tannus et al. (2021) destacam que os métodos para avaliação de acessibilidade centrados no usuário, envolvem a observação da interação do usuário com o sistema, proporcionando-lhe uma experiência próxima da realidade na qual o sistema será utilizado. Entre os métodos destacados estão: entrevistas, questionários, grupos focais (agrupa indivíduos que compartilham suas perspectivas sobre produtos ou serviços), além de testes de usabilidade e suas diferentes técnicas de aplicação. Na seção a seguir serão apresentados métodos e técnicas para avaliação de usabilidade, adotados para a pesquisa apresentada nesta tese.

3.6.2 Avaliação de Usabilidade

A usabilidade, no contexto de sistemas interativos, está relacionada à qualidade no uso do sistema, que pode ser determinada mediante os fatores de usabilidade. Os fatores de usabilidade são: facilidade de recordação, eficiência, segurança no uso e a satisfação

¹<https://accessmonitor.acessibilidade.gov.pt/>

²<https://asesweb.governoeletronico.gov.br/>

do usuário (NIELSEN, 1994). Tais fatores podem ser verificados por meio de métodos de inspeção e estudos empíricos (BARBOSA; SILVA, 2010).

Os métodos de inspeção estão relacionados a avaliação da usabilidade por meio da inspeção da interface. Eles permitem verificar a conformidade com um padrão ou guia de estilo. A inspeção é feita por uma pessoa avaliadora que tenta se colocar no lugar do usuário para identificar problemas de usabilidade do sistema (BARBOSA; SILVA, 2010). Por outro lado, os métodos para estudos empíricos contam com a participação dos usuários, pois a avaliação de usabilidade é feita mediante a observação e participação ativa dos usuários, relatando sua experiência de uso, qualificando sua interação com o sistema.

Segundo Barbosa e Silva (2010), os métodos de avaliação empírica, por meio da investigação e observação, podem fornecer resultados mais interessantes. De fato, os usuários podem percorrer caminhos não previstos, interagindo com o sistema de forma criativa e oportunista, proporcionando maior realidade, riqueza e diversidade da experiência de uso. Neste trabalho, foram utilizados métodos empíricos para avaliação de acessibilidade, usabilidade e análise de desempenho dos artefatos gerados pela pesquisa. Os métodos foram estrategicamente combinados, usando o conceito de triangulação (BARBOSA; SILVA, 2010), combinando técnicas de avaliação de acessibilidade, usabilidade e análise de desempenho. Assim, é possível a coleta e análise dos dados, sob diferentes perspectivas, permitindo resultados mais rigorosos e válidos para a pesquisa. Entre os métodos utilizados estão:

1. **Sessões de observação:** esse método consiste na observação da interação dos usuários com o sistema. Essa observação pode ser pura, sem interação do observador com os participantes; observação guiada por um conjunto de tópicos de interesse; ou até mesmo por meio de diários com relatos dos participantes, sem a presença do observador, registrada por meio de vídeo gravados pelos participantes (BARBOSA; SILVA, 2010; COURAGE; BAXTER, 2005).
2. **Questionários:** com esse método, o usuário avalia sua experiência de uso do sistema por meio de questionários, onde são relatados suas opiniões e preferências. A exemplo de questionário de usabilidade pode-se citar o *System Usability Scale - SUS* (LEWIS, 2018). O SUS permite a análise qualitativa por meio das questões relacionadas a usabilidade e quantitativa por meio da pontuação de usabilidade (escore SUS).

Para avaliação de acessibilidade podem ser aplicados questionários como o que é

proposto por Franco (2018), onde abordadas questões específicas relacionadas a recursos de acessibilidade para um determinado tipo de deficiência.

3. **Pensamento em voz alta:** esse método consiste em verbalizar o pensamento durante a interação com o sistema. Assim, a pessoa que está avaliando o sistema vai expondo verbalmente suas impressões, dificuldades e indicação de melhorias durante a interação. Todo o processo de interação pode ser gravado em áudio e imagem, acompanhado remotamente ou acompanhado em laboratório (FAN et al., 2019).

A quantidade de problemas identificados em um teste de usabilidade pode ser influenciada pelo número de participantes envolvidos. Conforme as orientações de Nielsen (2000), a realização do teste com cinco participantes é considerada suficiente para identificar aproximadamente 80% dos problemas relacionados à interface. A figura 6 apresenta graficamente a relação entre o número de participantes do teste e a quantidade de problemas detectados.

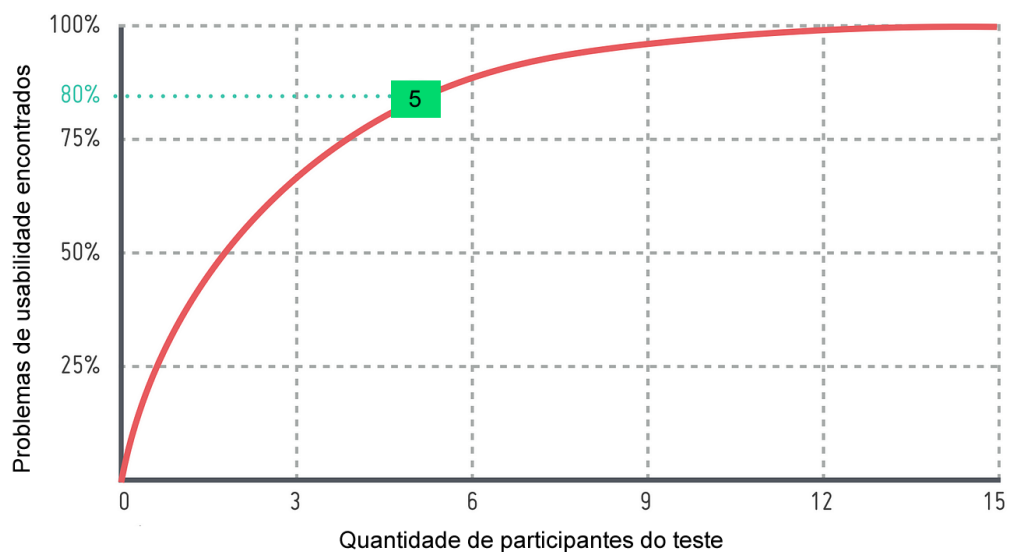


Figura 6: Relação entre número de usuário de teste e quantidade de problemas de usabilidade encontrados

Fonte: Adaptado de Nielsen (2000)

Conforme destacado no gráfico da figura 6, o número de problemas de usabilidade encontrados aumenta a medida em que aumenta o número de participantes do teste. Por exemplo, com 3 participantes podem ser encontrados cerca de 60-70% dos problemas, com 4 participantes cerca de 70-80% e com 5 participantes cerca de 75-85%. A medida em

que o número de participantes passa de 5, os erros começam a se repetir estabilizando a curva de crescimento e diminuindo a taxa de retorno (NIELSEN, 2000).

A usabilidade de uma sistema também pode ser avaliada por meio da análise de desempenho do usuários ao realizar uma tarefa (BARBOSA; SILVA, 2010). Por exemplo, é possível verificar o quanto um sistema é intuitivo ou quanto tempo o usuário leva para aprender a usar o sistema com mais eficiência com base no tempo em que ele leva para realizar as tarefa. A análise de desempenho será abordada na subseção a seguir.

3.6.3 Análise de desempenho

A análise de desempenho durante a utilização de um sistema interativo é conduzida com base nos resultados derivados da observação do uso do sistema para a realização de tarefas específicas (BARBOSA; SILVA, 2010). A análise tem como objetivo avaliar a eficiência e eficácia do sistema ou interface sob a perspectiva do desempenho do usuário. Essa técnica concentra-se na avaliação das interações entre os usuários e o sistema, observando como tarefas específicas são executadas.

Durante esse procedimento, é possível coletar dados relacionados às tarefas executadas como o número **total de erros cometidos pelo usuário**, **a quantificação dos diferentes tipos de erros**, **o tempo necessário para a conclusão de uma tarefa** e **a quantidade de assistências solicitadas**(BARBOSA; SILVA, 2010; PRATES; BARBOSA, 2003).

No contexto da pesquisa apresentada nesta teste, a análise de desempenho foi realizada para medir o nível de desempenho entre dois grupos distintos de usuários, cegos e videntes, ao realizar tarefas usando um sistema de programação visual. A comparação entre os dois grupos foi feita analisando o tempo para realização das tarefas e número de pedidos de auxílio. Essas informações são fundamentais para compreender o nível de equidade e o efeito de aprendizagem do sistema para ambos os grupos.

3.7 Considerações

Neste capítulo foram apresentados conceitos fundamentais para compreensão das diferentes abordagens de programação visual. Além disso, esses fundamentos também perpassam pelas conceitos de acessibilidade, no âmbito geral, por meio da WCAG 2.1. apresentando seus princípios e diretrizes. Dado o contexto do foco desta pesquisa, buscou-se apresentar recomendações de acessibilidade para cegos em ambientes web, servindo de

base para projetar e implementar os artefatos propostos nesta pesquisa. Além disso, foi abordada a problemática da acessibilidade para cegos em sistemas de programação visual, apresentando como soluções o Blockly acessível para cegos e o Blocks4All.

Também foram abordados os métodos essenciais para o desenvolvimento da pesquisa. Tais como Design Participativo, Prototipação e Design universal. Por fim, ainda levando em consideração fundamentos metodológicos, foram apresentados métodos para valiação de acessibilidade, usabilidade e análise de desempenho, sendo estes, aplicados para os estudos empíricos e qualitativos da pesquisa abordada nesta tese.

As soluções aqui apresentadas como o Blockly acessível para cegos e o Blocks4All são individualizadas, não sendo passível de generalização. É fundamental que, a partir dos fundamentos e das soluções para programação visual acessível apresentadas neste capítulo, surjam modelos de programação visual capazes de atender a uma diversidade de perfis de usuários, proporcionando ambientes inclusivos. A partir dos estudos a respeito dos desafios de acessibilidade para cegos em sistemas de programação visual, seria interessante que se concebesse um modelo genérico de interface que pudesse ser instanciado para sistemas de programação visual implementados em HTML5 ou linguagem similar. A abordagem de programação visual baseada em formulários se coloca como candidata para definição de tal modelo. Diante desse contexto, esta pesquisa propõe um modelo de programação visual baseado em formulários com recursos de acessibilidade para cegos. Esse modelo, nomeado de VProgForms, é apresentado no capítulo 4.

4 UM MODELO DE PROGRAMAÇÃO VISUAL BASEADO EM FORMULÁRIOS ACESSÍVEL PARA PESSOAS CEGAS E VIDENTES: VPROGFORMS

Este capítulo apresenta uma contextualização do processo de pesquisa empreendido para concepção e proposição do modelo de programação visual denominado VProgForms. Esse procedimento engloba uma análise de acessibilidade do iVProg, uma solução existente no domínio, examinando seu modelo prevalente e identificando lacunas de acessibilidade. Além disso, incorpora metodologias de design participativo, incluindo sessões com grupos de foco para ideação e prototipação da solução. O modelo resultante, VProgForms, é apresentado por meio do formalismo do diagrama de blocos de definição SysML (HAUSE et al., 2006), detalhando cada elemento para proporcionar clareza e compreensão integral do modelo proposto.

4.1 Contextualização

Os estudos conduzidos por Milne e Ladner (2018) indicam que os sistemas de programação visual amplamente utilizados, conforme documentado, adotam modelos de programação puramente visual, baseados em restrições, exemplos, entre outros. Mesmo quando incorporam recursos de acessibilidade, tais modelos podem apresentar desafios significativos para usuários cegos, demandando adaptações específicas para sua utilização por esses usuários. Nenhum dos sistemas de programação visual avaliados por Milne e Ladner (2018), utilizam um forte apelo para a abordagem baseada em formulários.

Diante do uso frequente de formulários em ambientes web por parte dos usuários que navegam pela internet, juntamente com a capacidade de criar formulários acessíveis, surge a conjectura de que um modelo de programação visual baseado em formulários pode representar uma alternativa viável para o desenvolvimento de sistemas de programação visual acessíveis tanto para indivíduos cegos quanto para aqueles que enxergam. Diante

desse contexto, dado seu forte apelo para a abordagem baseada em formulários, foi realizada uma análise de acessibilidade do iVProg a fim de verificar se o mesmo seria uma alternativa acessível para cegos e videntes.

Uma análise de acessibilidade do iVprog foi conduzida utilizando a ferramenta Accessmonitor, com o intuito de verificar sua conformidade com as diretrizes de acessibilidade estabelecidas pela WCAG 2.1. Essa verificação foi feita por meio do link do sistema disponível em <https://www.usp.br/line/ivprog/>. Os resultados obtidos com o Accessmonitor (Anexo C) revelaram uma pontuação de 7.6 de 10, indicando uma total de oito critérios de sucesso da WCAG 2.1 que não foram atendimentos. Entre os critérios não atendidos pontuados pela ferramenta estão:

1. Seis imagens na página que não contem seu equivalente em texto (Nível A);
2. Dois links compostos por imagens com atributo `alt` (para texto alternativo) vazio (Nível A);
3. A primeira hiperligação da página não permite saltar diretamente para a área do conteúdo principal (Nível A);
4. Quatro casos em que se viola a sequência hierárquica dos níveis de cabeçalho (Nível AAA);
5. 29 casos em que se usa javascript para remover o foco do campo, sempre que o campo recebe o foco (Nível A);
6. Um elemento `<iframe>` sem o atributo `title` (Nível A);
7. Foram localizados 15 combinações de cor cuja relação de contraste é inferior ao rácio mínimo de contraste permitido pelas WCAG, ou seja 3 para 1 para texto com letra grande e 4,5 para 1 para texto com letra normal (Nível AA);
8. Foram encontrados dois links sem o nome acessível (Nível A).

Com os resultados apresentados, é possível observar que, embora o iVProg possua uma nota 7.6 existem problemas acessibilidade de níveis básicos como A e intermediários de nível AA que precisam ser corrigidos. Adicionalmente, foi conduzida uma inspeção da interface do iVProg com o leitor de tela NVDA, onde constatou-se que muitos dos elementos de interface não eram passíveis de leitura pelo NVDA. Tal fato, resulta na inviabilidade de utilização do iVProg por indivíduos com deficiência visual que necessitam do leitor de tela para interagir com o referido sistema.

Dadas as características do iVProg em relação a abordagem baseada em formulários, a flexibilidade do HTML para construção de aplicações acessíveis e seu código aberto, optou-se por partir de seu modelo subjacente para criação de um modelo de programação visual acessível para cegos e videntes. Tal escolha foi embasada na suposição de que um modelo de programação visual fundamentado em formulários poderia representar uma alternativa eficaz para derivar sistemas de programação visual a partir dele. Com isso, iniciou-se o processo de ideação e concepção do modelo VProgForms.

4.2 O processo de ideação e levantamento de requisitos

O processo ideação para concepção do VProgForms foi realizado por meio da metodologia de design participativo, contando com a participação de colaboradores videntes e um cego, todos especialistas da área de computação, com expertise em programação.

Durante essas sessões de grupos de foco, foram discutidas questões relacionadas à viabilidade técnica e acessibilidade para usuários cegos no modelo a ser proposto. Entre as questões levantadas estão:

1. O uso de elementos de formulário para criação de algoritmos é viável para pessoas que utilizam leitor de tela?
2. Quais as estratégias e requisitos para criar uma abordagem de programação visual baseada em formulários acessível para cegos?

Com as discussões em torno dos questionamentos levantados, chegou-se a conclusão de que a programação baseada em formulários poderia ser viável, desde que todos os blocos e elementos fossem acessíveis com leitor de tela. Com isso, foram levantados os requisitos que serviram como fundamentos para a criação de um modelo de programação visual baseada em formulários acessível para pessoas cegos e videntes.

Para a identificação, definição e validação dos requisitos fundamentais, foram empregadas técnicas de *brainstorming* (chuva de ideias) (BARBOSA; SILVA, 2010), juntamente com a validação de protótipos. Os requisitos fundamentais referem-se aos elementos que determinam o padrão de interação adotado, as recomendações de design e as diretrizes de acessibilidade que devem ser seguidas ao implementar o modelo proposto. Esses requisitos são listados a seguir.

1. Não deve ser adotado o estilo de interação "clique duplo" ou "clica e arrasta" por

questões de acessibilidade, embora seja um premissa que todos os elementos possam ser manipulados utilizando o mouse ou teclado.

2. A lógica de programação deve ser inteiramente determinada por meio de elementos de formulário.
3. O algoritmo deve ser construído de forma sequencial, onde cada bloco de comando seja estruturado como uma sequência, executados em ordem, de cima para baixo, um após o outro.
4. A navegação com o teclado deverá ser feita utilizando as teclas **Tab** e as setas direcionais do teclado. A tecla **Tab** deverá ser utilizada para navegar entre os blocos e pelos elementos contidos nos blocos. As setas direcionais deverão ser utilizadas para navegar pelos elementos internos dos menus disponíveis.
5. Ao criar um bloco de comando o foco deverá ser direcionado para o primeiro elemento do bloco.
6. Deverá ter um Terminal(Console), onde os resultados da execução sejam apresentados.
7. O algoritmo criado de forma visual, deverá ser transcrito para um código textual passível de interpretação para gerar e apresentar o resultado no terminal. Para isso, pode-se utilizar um componente de software interpretador de código.
8. Sempre que um algoritmo for executado, o foco deverá ser direcionado para o Terminal, diretamente para o resultado textual apresentado.
9. Os blocos de comandos criados devem poder ser editados ou excluídos.
10. Cada bloco poderá ser lido de forma textual por meio de um recurso para conversão de visual para texto.
11. A interface deve ter disponível links de acesso rápido que direcionem o usuário diretamente para áreas importantes da interface, sem a necessidade de uma navegação linear.
12. Deverá ter disponível um recurso de tradução da interface para diferentes idiomas.
13. Todos os elementos de interface devem poder ser acessados com leitor de tela.
14. A interface deverá atender a todos os critérios de sucesso da WCAG (versão atual) de níveis A e AA.

Baseado nos requisitos apresentados, foi elaborado um modelo genérico passível de implementação como modelo subjacente para sistemas de programação visual baseados em formulários. Esse modelo nomeado VProgForms é descrito na seção 7 deste capítulo. Além disso, uma extensão deste modelo foi implementada em formato de software, conforme apresentado no capítulo 5.

4.3 O modelo VProgForms

Com base nos requisitos elencados na seção 4.2 deste capítulo, foi criado o VProgForms, um modelo de domínio abstrato para a programação visual baseada em formulários, tendo como premissa a implementação das diretrizes de acessibilidade propostas na WCAG 2.1. Para representação do modelo utilizou-se o diagrama de blocos de definição SysML (HAUSE et al., 2006), proposto pelo *Object Management Group* como um diagrama para representar a estrutura de um sistema, descrevendo a hierarquia do sistema e as classificações dos seus componentes (OMG, 2022). O modelo VProgForms é apresentado por meio do diagrama de blocos de definição na figura 7.

O VProgForms, conforme é apresentado na figura 7, adota uma premissa importante relacionada a acessibilidade que deve ser aplicada a todos os elementos de interface. Essa premissa define que devem ser aplicadas as diretrizes de acessibilidade propostas na WCAG (versão atual) para que todos os componentes de interface possam ser lidos de forma concisa por leitores de tela. A partir dessa premissa, define-se um modelo de interface composta por: links de acesso rápido; definição do idioma; função início/Área de trabalho, essa área é composta por um menu com a lista de comandos disponíveis, o botão executar e um botão para excluir os comandos; e um terminal. Cada bloco do modelo será descrito a seguir.

1. **Links de acesso rápido:** Os links de acesso rápido devem direcionar o usuário diretamente para partes importantes do sistema. Por exemplo, pode ser criado um link de acesso rápido que direcione diretamente para o menu de comandos.
2. **Idioma:** A escolha do idioma está relacionada aos recursos de tradução do texto presente na interface e em todos os seus elementos para diferentes línguas.
3. **Função início ou área de trabalho:** A função início ou área de trabalho é o espaço onde os blocos de comandos serão adicionados, por meio do menu `_comandos`. Cada comando adicionado gera uma lista do tipo `Comando` que pode ser composta por um

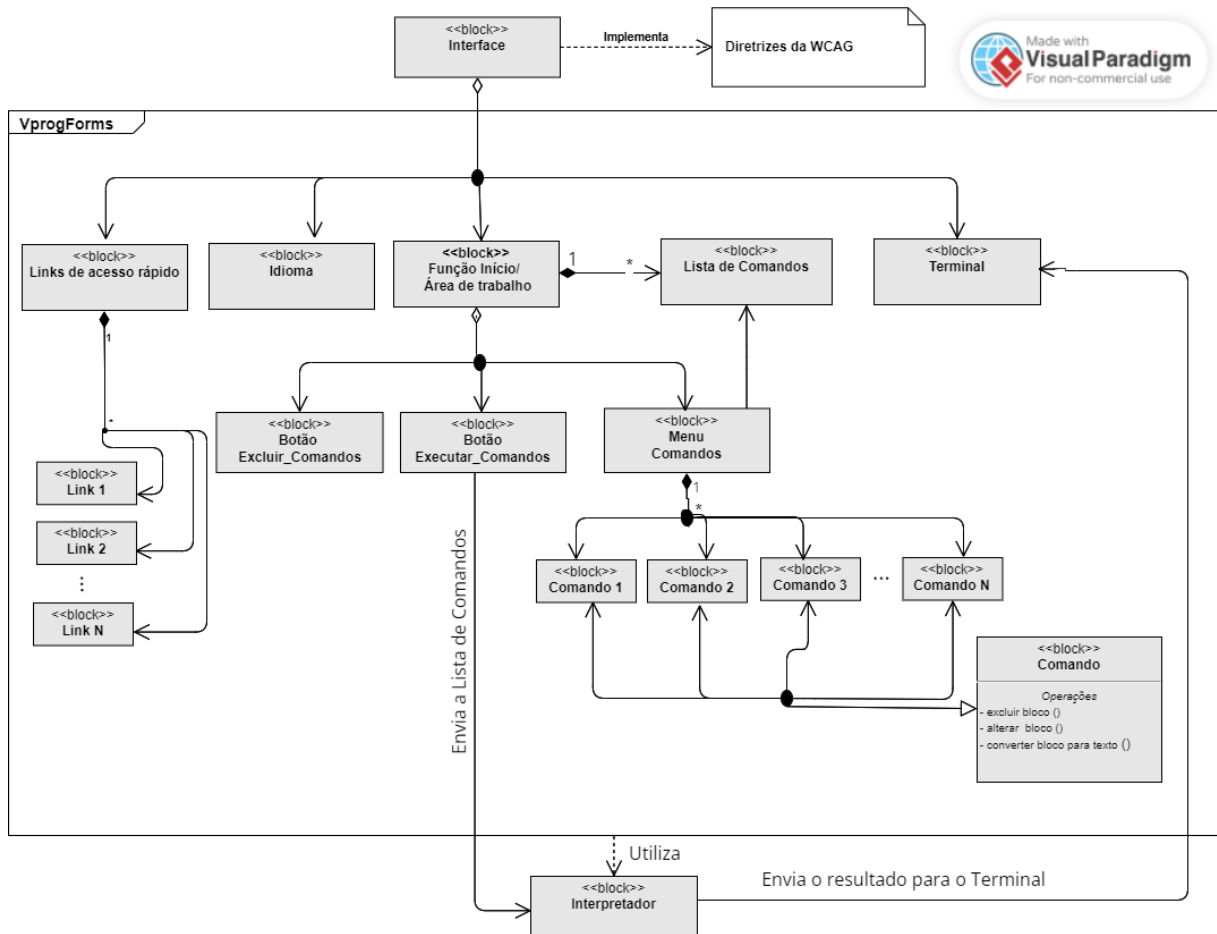


Figura 7: Diagrama de blocos do SysML para definição do Modelo VProgForms
Fonte: Elaborado pelo autor.

ou vários comandos aninhados. Por exemplo, o comando se_então (comando de seleção) pode conter comandos para operações matemáticas (matemática/atribuição), escreva, entre outros.

Além do espaço para os comandos, a função início também possui em seu espaço de trabalho um botão para excluir as listas de comandos criadas (botão excluir_comandos) e um botão para executar o algoritmo criado (botão executar_comandos).

4. **Menu comandos:** O menu_comandos tem como objetivo disponibilizar os comandos para serem adicionados à função início. Ele é composto por 1 ou mais comandos que podem ser criado a do arcabouço Comando. Na figura 7 as generalizações criadas a partir do modelo Comando são representadas pelos blocos Comando 1, Comando 2, Comando 3 e Comando N. Embora cada bloco de comando sejam distintos uns dos outros, algumas propriedades como forma, iconografia e uso de elementos de formulário são comuns entre eles. Além disso, também são comuns as

operações básicas de controle do usuário como excluir bloco de comando, alterar, converter de visual para texto e um menu interno para adicionar comandos ao bloco criado.

5. **Interpretador:** Um dos requisitos apontados para o VProgForms é que seja feita uma conversão da lógica visual para texto. O pseudocódigo textual gerado a partir da lista de comandos deverá ser interpretado por um componente de software (interpretador) capaz de executá-lo.
6. **Terminal:** área destinada a mostrar a saída do algoritmo em formato textual, nos moldes de um terminal de qualquer sistema operacional.

Com intuito de avaliar o VProgForms e verificar sua aplicabilidade, foi desenvolvido o iVProg4All, apresentado no capítulo 5, que implementa o modelo proposto no VProgForms. Além da avaliação do modelo, o sistema permitiu verificar se a abordagem de programação visual baseada em formulários poderia atender a usuários cegos e videntes, abrindo uma nova perspectiva para o design universal em sistemas de programação visual.

4.4 Considerações

Neste capítulo foi apresentada uma contextualização, onde discutiu-se a relevância de modelos de programação visual acessíveis para usuários cegos, considerando as limitações dos modelos existentes. A análise de acessibilidade do iVProg revelou questões significativas, destacando a necessidade de uma abordagem mais inclusiva. Também foi descrito o processo de ideação e levantamento de requisitos para criar o modelo VProgForms, um paradigma de programação visual baseado em formulários. Os requisitos fundamentais foram estabelecidos, priorizando a acessibilidade e a usabilidade. O modelo VProgForms foi apresentado, utilizando um diagrama de blocos SysML para definir a estrutura proposta no modelo. Cada componente do modelo foi descrito de forma genérica para que possam ser implementados por sistemas de programação visual baseados em formulário.

No próximo capítulo, será apresentada uma instância do modelo VProgForms, culminando na criação do iVProg4All. Esta instância permitirá identificar o metamodelo como a estrutura subjacente a um sistema de programação visual fundamentado em formulários, dotado de recursos de acessibilidade específicos para usuários cegos. Este desenvolvimento prático, constituirá uma aplicação concreta do VProgForms.

5 INSTANCIANDO O MODELO VPROGFORMS: IVPROG4ALL

Neste capítulo, é apresentado o iVProg4All, um sistema que se fundamenta no iVProg, adotando o VProgForms como modelo subjacente. São apresentados os métodos e processos de desenvolvimento empregados no projeto. Serão abordados os ciclos de desenvolvimento, organizados em módulos distintos, e a estrutura evolucionária adotada, proporcionando uma visão do processo de construção do iVProg4All. A análise das características do iVProg4All, juntamente com os recursos de acessibilidade e a implementação técnica, estabelece o contexto para a apresentação do sistema, destacando seus principais componentes e funcionalidades. Além disso, são abordadas as técnicas empregadas nos testes preliminares de usabilidade e acessibilidade para cada módulo desenvolvido, bem como os resultados obtidos com os testes.

5.1 Instanciando o iVProg4All

O iVProg4All surge da necessidade de avaliar o modelo VProgForms por meio de sua instanciação para um sistema de programação visual que o adote como modelo subjacente. Dessa forma, buscar alcançar os objetivos proposto na pesquisa, além de responder às questões de pesquisa levantadas no capítulo 2. Após a definição do modelo VProgForms, deu-se início ao processo de desenvolvimento do iVprog4All. Inicialmente, busou-se projetar um modelo para o iVProg4All usando o diagrama de blocos do SysML (HAUSE et al., 2006). Esse modelo foi instanciado e descrito a partir dos elementos e requisitos propostos no modelo VProgForms. O modelo aplicado ao iVpro4All é apresentado na figura 8.

No modelo proposto na figura 8, a interface é composta por:

1. **Links de acesso rápido:** é composto por links que direcionam imediatamente para a Função Início, Menu de Comandos, botão Executar _Comandos, Excluir _Comandos e para o Terminal. Esses links permitem que os usuários, principalmente aqueles que usam leitor de tela, naveguem de maneira não linear, sem a necessidade de

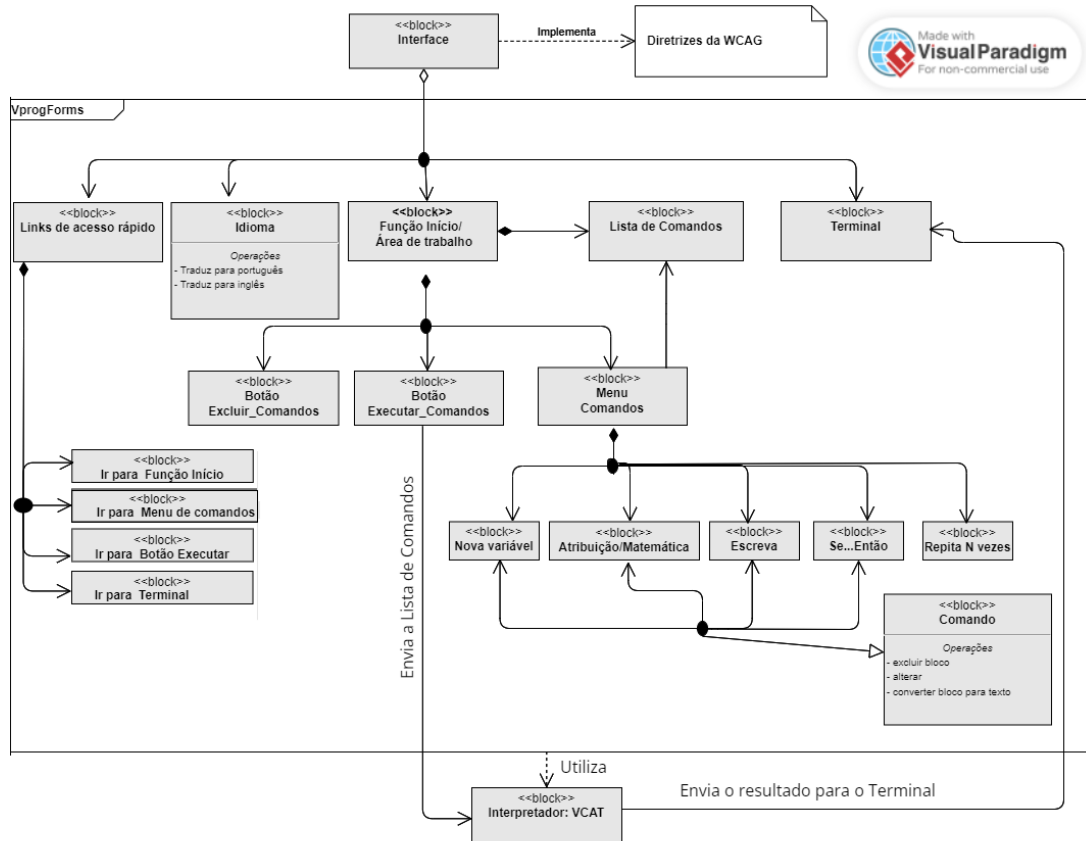


Figura 8: Definição do Modelo para implementação do iVProg4All
Fonte: Elaborado pelo autor.

percorrer toda a interface para alcançar recursos importantes do sistema;

2. **Idioma:** para a tradução da interface e seus elementos para as línguas português e inglês, foi Utilizando a tecnologia ngx-translate¹ do Angular². Existem arquivos de traduções com todos os textos da aplicação e com base na linguagem selecionada pelo usuário, o ngx-translate busca os valores das strings dos respectivos arquivos de tradução;
3. **Função Início:** a função início corresponde a área onde os blocos de comandos serão adicionados e organizados de forma lógica para criação dos algoritmos. Dentro do contexto da função início estão o menu com a lista de comandos, o botão executar e o botão excluir. Esses recursos são descritos a seguir.

3.1. Menu de Comandos: o menu de comandos é composto pelos blocos de comando **Nova variável**, utilizado para criação de variáveis com sua definição de tipo e valor inicial; **Atribuição/Matemática**, utilizado para atribuir valores

¹<https://github.com/ngx-translate/core?tab=readme-ov-file>

²<https://angular.io/>

às variáveis criadas, além de criar expressões matemáticas; **Escreva**, utilizado para retornar valores de variáveis ou texto a serem apresentados no Terminal; **Se...Então**, utilizado para criação de estrutura de seleção do tipo Se (condição) Então (instrução a ser executada) Senão (instrução a ser executada); **Repita N Vezes**, utilizados para criar estruturas de repetição do tipo Repita_para(condição) De (valor) Até (valor) Faça (instrução a ser executada).

3.2. Botão Executar: o botão Executar é utilizado para acionar o interpretador que, por sua vez, gera um resultado a partir da lista de comandos. Esse resultado é apresentado no Terminal.

3.3. Botão Excluir Comandos: é utilizado para "limpar" a função início, excluindo a lista de comandos que foi criada partir da adição dos blocos de comando.

4. **Interpretador:** O iVProg4all tem integrado o interpretador de código (SOUZA, 2023) VCAT. Ele realiza as análises léxica, sintática e semântica do código para poder interpretá-lo. A execução é feita linha por linha (através da árvore sintática abstrata), fazendo uma análise completa do código.
5. **Terminal:** O pseudocódigo executado pelo interpretador gera um resultado para ser apresentado no terminal como saída de dados.

A partir da definição de modelo, deu-se início ao processo de desenvolvimento do iVProg4All. Para o desenvolvimento foi adotado um modelo evolucionário, incremental. Essa abordagem de desenvolvimento foi adotada para que o cada módulo fosse desenvolvido e avaliado antes de ser integrado ao sistema. Os ciclos de desenvolvimento são descritos na subseção a seguir.

5.1.1 Ciclos de desenvolvimento

No processo de desenvolvimento do iVprog4All, optou-se por organizar a estrutura do sistema e suas funcionalidades em módulos distintos. Cada um desses módulos percorreu um ciclo de desenvolvimento, em um processo iterativo. As funcionalidades de cada módulo foram implementadas em Typescript ¹, aliado ao framework Angular v.13². Entre outras tecnologias para o desenvolvimento foram utilizados Bootstrap ³, RxJS⁴ e Karma⁵.

¹<https://www.typescriptlang.org/>

²<https://update.angular.io/?l=3&v=12.0-13.0>

³<https://getbootstrap.com/>

⁴<https://rxjs.dev/guide/overview>

⁵<https://www.npmjs.com/package//karma>

A estrutura do front-end baseia-se na arquitetura padrão do *framework* Angular, onde os componentes estão divididos por pastas. Cada pasta de componente possui um arquivo HTML, CSS e TS. A comunicação com o *back-end* é feita através módulos Javascript externo que é importado para dentro do projeto. O código fonte da aplicação está disponível em <<https://github.com/mdevaner2017/mdevaner2017.github.io>> e o sistema pode ser acessado em <<http://mdevaner2017.github.io>>.

Para o desenvolvimento do iVprog4All foi adotado o modelo evolucionário da engenharia de software (PRESSMAN; MAXIM, 2021), conforme apresentado na figura 9. Este modelo foi empregado como um subprocesso do Modelo-DSR, integrado à etapa de construção do Artefato. Essa abordagem proporcionou uma estrutura metodológica que se alinha aos princípios e diretrizes da engenharia de software, facilitando a evolução iterativa e incremental do sistema durante a fase de desenvolvimento.

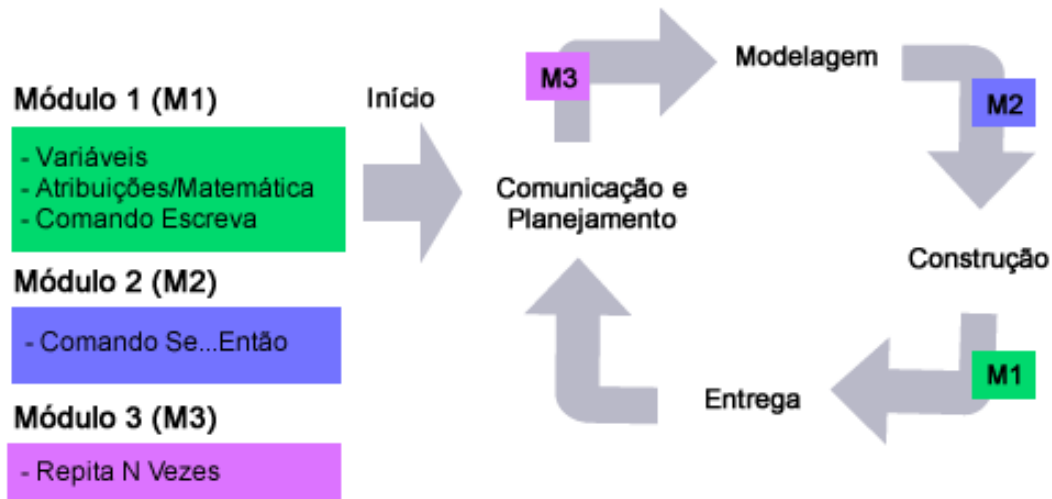


Figura 9: Modelo de processo evolucionário utilizado para desenvolver o iVProg4All
Fonte: Adaptado de Pressman e Maxim (2021)

Conforme apresentado na figura 9, cada módulo do sistema (M1, M2 e M3) passa por esse ciclo em um processo iterativo, incremental e evolucionário. Vale mencionar que o ciclo de entrega, no contexto adotado, representa a etapa de validação de uma versão completa do sistema. A seguir, estão elencados os ciclos de desenvolvimento adotados, juntamente com as atividades correspondentes a cada ciclo.

1. **Comunicação e planejamento:** o desenvolvimento do IVpro4All teve início com o processo de comunicação e planejamento, utilizando a técnica de Design Participativo, realizando sessões de ideação por meio da metodologia de chuva de ideias (BARBOSA; SILVA, 2010). Essas sessões foram conduzidas pelo pesquisador com

as partes interessadas (pesquisadores, colaborador cego e desenvolvedores) para ideiação e levantamento de requisitos.

2. **Modelagem:** no decorrer do processo de modelagem, foram elaborados o modelo instanciado do VProgForms e também protótipos destinados à verificação da viabilidade pelas partes interessadas. Na figura 10 são apresentados os protótipos Versão 1, Versão 2 e Versão 3, gerados ao longo do processo de modelagem até que fosse definida a versão final. Os protótipos foram desenvolvidos utilizando a ferramenta *Jusmind*¹.

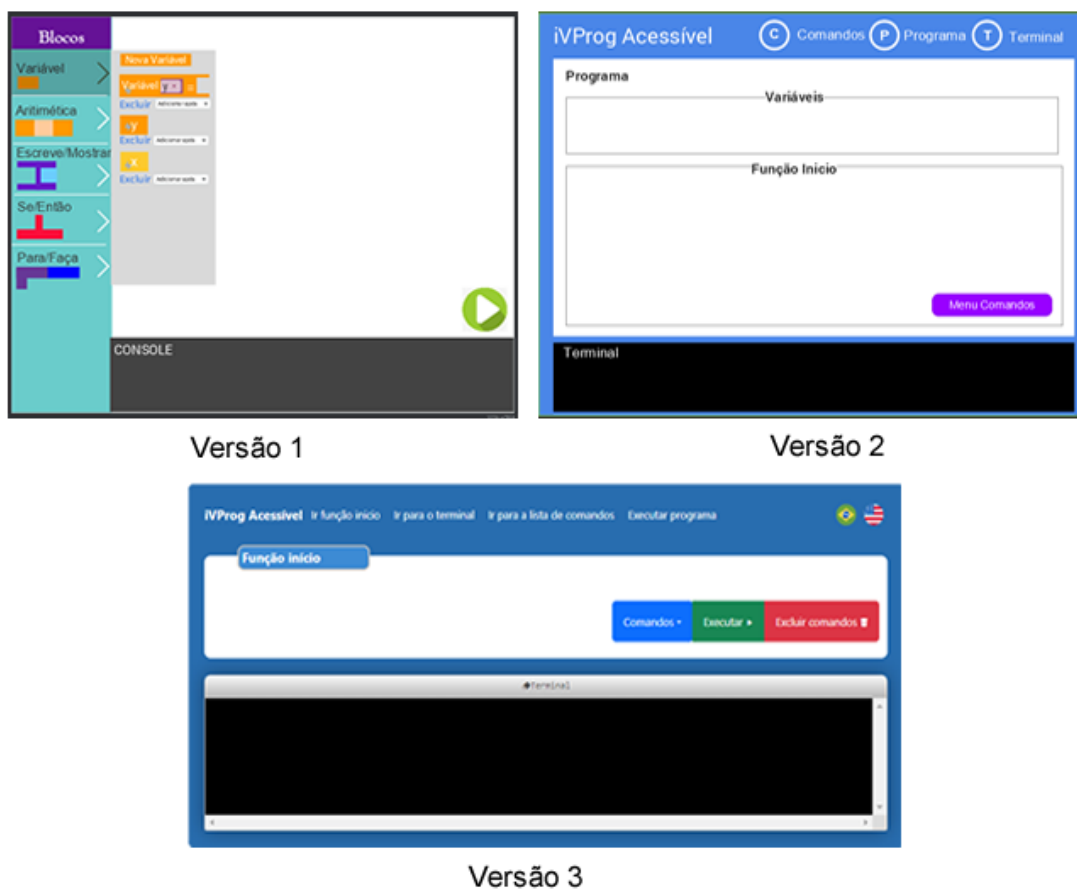


Figura 10: Protótipos do iVProg4All, em suas versões 1, 2 e 3.
 Fonte: Pressman e Maxim (2021)

Conforme apresentado na figura 10, a Versão 1 adotava um modelo mais próximo do Blockly, adotando a metáfora abordada no jogo Tetris² que simula um quebra-cabeça. Nela os blocos seriam selecionados e adicionados a uma área de trabalho, devendo ser encaixados de maneira lógica. Porém, durante as discussões realizadas

¹<https://www.justinmind.com/>

²<https://tetris.com/play-tetris>

com os grupos de foco, verificou-se que essa abordagem seria inviável para implementação do modelo VProgForms. Assim, foi proposta uma segunda versão (Versão 2) com uma interface mais simplificada, mais parecida com o iVProg. Na Versão 2, os comandos seriam adicionados a partir de um menu com os comandos, mas variáveis eram criadas em um contexto fora da função início. Essa versão evoluiu até chegar a Versão 3. Na versão 3, busou-se criar uma interface mais simplificada, onde as variáveis seriam dispostas no mesmo ambiente que os outros blocos de comando. Nela também busou-se atender a todos os requisitos definidos no modelo VProgForms, dando origem ao iVProg4All.

Os protótipos foram validados por meio da criação de versões funcionais em HTML e Javascript que simulavam o desempenho de cada recurso. Nesse processo, as funcionalidades foram executadas exclusivamente na interface, sem integração com o *back-kend*, também não eram gerados pseudocódigo executáveis. Essas versões eram acessíveis com leitor de tela. Além disso, sempre era feita uma descrição da tela para o colaborador cego que ajudou no processo de validação.

3. **Construção:** a construção corresponde aos processos de implementação de código e testes. Desenvolvido por módulos, cada módulo do IVprog4All corresponde a um conjunto de funcionalidades. Os módulos foram divididos da seguinte maneira:

(a) **Módulo 1**

- i. Foi criada a interface do sistema e todos seus elementos como links, botões, Função início e o Terminal.
- ii. Foi criado o bloco de comando Nova variável;
- iii. Foi criado o bloco de comando Atribuição/Matemática;
- iv. Foi criado o bloco de comando Escreva;

(b) **Módulo 2**

- i. Foi criado o bloco de comando Se...Então;

(c) **Módulo 3**

- i. Foi criado o bloco de comando Repita N Vezes.

Durante o desenvolvimento dos módulos, foram conduzidos verificações de acessibilidade por meio da utilização do leitor de tela NVDA. Essas verificações tiveram o propósito de verificar a acessibilidade dos elementos da interface para tal tecnologia assistiva. Adicionalmente, testes automáticos foram realizados com o validador

Acessmonitor para verificar a conformidade com as diretrizes de acessibilidade estabelecidas pela WCAG 2.1. Além disso, a medida em que os módulos eram desenvolvidos, eles eram submetidos para testes de acessibilidade e usabilidade, realizado por um usuário cego, conforme descrito na subseção 5.2 deste capítulo.

4. **Entrega:** nesta fase, uma versão do iVPorg4All com todas as funcionalidades foi apresentada e avaliada na categoria protótipo do concurso App.Edu 2023 ¹, no XII Congresso Brasileiro de Informática na Educação. O concurso App.Edu tem como objetivo avaliar e reconhecer softwares educacionais que oferecem soluções tecnológicas para o processo de ensino e aprendizagem em espaços formais e não formais. No concurso, o iVProg4All foi apresentado como uma solução de software acessível para dar suporte ao ensino e aprendizado do pensamento computacional e da lógica de programação, obtendo terceiro lugar entre os softwares avaliados para a categoria protótipo.

Tendo sido a implementação funcional dos módulos concluída, foi conduzido um teste preliminar para cada módulo do iVProg4All com a participação de um usuário cego. O propósito desse teste foi identificar potenciais problemas de acessibilidade e usabilidade específicos para esse perfil de usuário. Os detalhes do teste preliminar e seus resultados estão expostos na subseção a seguir.

5.2 Testes preliminares

Após o desenvolvimento de cada módulo, as funcionalidades, usabilidade e acessibilidade foram testadas pelo pesquisador responsável (vidente) e depois por um testador cego. Com o testador cego, os testes foram realizados de forma remota e assíncrona.

Para realização do teste foi passado um conjunto de tarefas relacionadas a lógica de programação que o participante teria que resolver utilizando o iVProg4All. As tarefas desenvolvidas foram planejadas para abordar as operações fundamentais da lógica de programação: manipulação de variáveis, atribuições, operações matemáticas, estruturas de seleção e estruturas de repetição.

Foi pedido ao testador (cego) que registrasse toda interação com o iVProg4All em vídeo. Além disso, também foi pedido que utilizasse a técnica de pensamento em voz alta, onde o testador verbalizasse seus pensamentos durante a interação com o sistema, expondo suas impressões, dificuldades e sugestões de melhorias ao longo da interação.

¹<https://cbie.sbc.org.br/2023/apps-edu/>

O participante se identifica com o sexo masculino, possui cegueira congênita, é formado em análise e desenvolvimento de software e atua no mercado como analista de testes. Ele se propôs a colaborar com a pesquisa de forma voluntária. Os testes foram conduzidos com o uso do leitor de tela NVDA versão 2022.4, seguindo um fluxo de tarefas pré-definido pelo pesquisador responsável. As tarefas, assim como as instruções para o teste, foram enviadas ao participante por e-mail.

Por fim, também foi solicitado ao participante que elaborasse um relatório em vídeo ou texto sobre as suas impressões gerais do sistema e sua utilidade. O relatório, enviado em formato de texto, juntamente com a gravação em vídeo da interação do testador, permitiu identificar problemas estruturais de usabilidade e acessibilidade. Dessa forma, foi possível realizar correções e ajustes no sistema antes de realizar estudos empíricos com amostras maiores de usuários. Embora essa solicitação tenha sido feita para todos os módulos, o participante enviou apenas o relatório do Módulo 1 (Anexo E). Os outros resultados foram analisados com base na análise de vídeos gravados da interação do testador com o iVProg4All.

Nas subseções a seguir, serão apresentados uma síntese dos resultados relatados por meio do relatório do módulo 1 e da análise dos vídeos para cada módulo, bem como os problemas de usabilidade e acessibilidade apontados pelo testador.

5.2.1 Teste preliminar: Módulo 1

Para o teste deste módulo foram passadas as seguintes tarefas para o participante:

1. Tarefa 1: crie duas variáveis `x` e `y` e atribua um valor para elas;
2. Tarefa 2: crie a variável `soma` e faça ela receber `x + (mais) y`;
3. Tarefa 3: crie a variável `multiplica` e faça ela receber `x *(vezes) y`;
4. Tarefa 4: crie a variável `subtrai` e faça ela receber `x - (menos) y`;
5. Tarefa 5: crie a variável `divide` e faça ela receber `x / (dividido) y`;
6. Tarefa 6: crie um comando `escreva` para cada variável: `soma`, `multiplica`, `subtrai` e `divide` para mostrar os valores destas variáveis no console.

Durante a análise do vídeo verificou-se que a tarefa 4 (crie a variável `subtrai` e faça ela receber `x - (menos) y`) foi feita de forma incorreta, pois elementos não foram ordenados

na forma correta. Também verificou-se que a tarefa 6 (crie um comando escreva para cada variável: soma, multiplica, subtrai e divide para mostrar os valores destas variáveis no console) não foi realizada. Todas as outras tarefas foram realizadas de forma correta.

Durante o processo de interação e também em relatos textuais enviados pelo participante por e-mail, foram indicadas algumas melhorias. Entre elas estão a alteração de nomenclaturas para torná-las mais compreensíveis para o usuário. Por exemplo, a substituição do termo "String" por "Texto" no menu de tipos de variáveis, a mudança do termo de "Ver código" para "Ler código", a alteração do menu "Atribuições e Matemática" para "Atribuição/Matemática", entre outras.

Também foram identificados problemas de percepção, nos quais certos rótulos não puderam ser lidos pelo leitor de tela. Outro problema de percepção e compreensão estava relacionado ao uso de caracteres especiais. Por exemplo, o hífen (-) deveria ser substituído por um sinal de subtração (Unicode 2212), o asterisco (*) por um sinal de multiplicação (Unicode 00D7) e a barra por um sinal de divisão (Unicode 00F7). Em relação à navegação, foram indicados alguns ajustes para a mudança de foco. Segundo o testador, clicar na opção "Editar Código" para qualquer variável ou comando já inserido nos blocos seria útil se o foco do leitor de tela fosse direcionado para o primeiro item a ser editado nessa linha.

5.2.2 Teste preliminar: Módulo 2

Para o teste deste módulo foram passadas as seguintes tarefas para o participante:

1. Tarefa 1: crie duas variáveis `nota1`, `nota2` e atribua um valor do tipo `real` para elas;
2. Tarefa 2: crie um variável `média` do tipo `real` e atribua a ela $(nota1 + nota2) / 2$;
3. Tarefa 3: crie uma condição `Se...Então` para mostrar "Aprovado", caso o valor da variável `média` seja ≥ 5 , e "Reprovado" caso da `média` seja < 5 . Para mostrar o valor no console utilize o comando `Escreva`.

Durante a análise dos resultados, não foram encontrados muitos problemas na execução das tarefas, pois o testador conseguiu realizar todas as tarefas. No entanto, foi observado que os botões e menus estavam sendo lidos apenas com a tecla TAB, sendo ideal que também fossem lidos com as setas. Outro problema encontrado foi que os

operadores condicionais (aritméticos, lógicos e relacionais) do bloco "Se..Então" não estavam sendo lidos corretamente pelo leitor de tela. Portanto, foi necessário incluir texto alternativo para cada operador.

5.2.3 Teste preliminar: Módulo 3

Para o teste deste módulo foram passadas as seguintes tarefas para o participante:

1. Tarefa 1 - crie uma variável chamada `somatório` e atribua o valor 0 para ela;
2. Tarefa 2 - crie uma estrutura de repetição onde a variável `contador` inicie com 0 e repita uma instrução até 10;
3. Tarefa 3 - no laço de repetição, faça a variável `somatório` receber a soma dos números de 1 a 10 e mostre p valor final no console, por meio do comando `Escreva`.

Considerando o efeito de aprendizado e a correção dos problemas identificados no teste anterior, não foram encontrados muitos problemas na execução das tarefas relacionadas ao bloco de repetição "Repita N vezes". O testador obteve sucesso em todas as tarefas. A única sugestão de melhoria foi relacionada à inclusão de texto alternativo para os menus de incremento e decremento (`++` e `-`).

5.2.4 Considerações gerais do testador

No relatório do Módulo 1, o testador relatou que realizou outros testes com diferentes navegadores, leitores de tela e sistemas operacionais, tais como o navegador MS Edge 109 com o leitor de tela Windows 11 Narrator; o navegador Firefox ESR 102.7 no Linux Debian 11 com o leitor de tela Orca 43; o navegador Google Chrome 109 no Android 13 com o leitor de tela Talkback 13; e o navegador Safari no iOS 16.3 com o leitor de tela Voiceover.

Foi relatado que o iVProg4All surpreendeu positivamente ao ser utilizado por meio do navegador em dispositivos móveis (celulares). Segundo o testador, com essa ferramenta, é fácil aprender em cursos de introdução à lógica de programação usando um celular, pois facilita a interação do usuário. No entanto, tanto com o leitor de tela Narrator (no Windows) quanto com o Orca (no Linux), Talkback (no Android) e Voiceover (no iOS), seja na área de criação de variáveis ou ao utilizar as opções de atribuição/Operações ou de escrita, os quatro leitores de tela mencionados não identificam os nomes dos botões

"Ver código" e "Editar Código", apenas dizem "Botão". O único leitor de tela capaz de identificar tais nomes foi o NVDA.

O processo de desenvolvimento adotado desempenhou um papel crucial na criação da primeira versão do iVProg4All, representando a implementação inicial de um sistema fundamentado no VProgForms. A próxima subseção discorrerá sobre a interface do iVProg4All, apresentando seus elementos e recursos.

5.3 Apresentando o iVProg4All

O iVProg4All é um sistema de programação visual fundamentado em formulários, foi desenvolvido como uma prova de conceito para esta pesquisa, buscando ilustrar a aplicabilidade do modelo proposto, o VProgForms. Além de sua finalidade como prova de conceito, o iVProg4all possui potencial para ser adotado como ferramenta educacional no ensino e aprendizado de programação. A Figura 11 apresenta a interface do sistema juntamente com seus elementos.

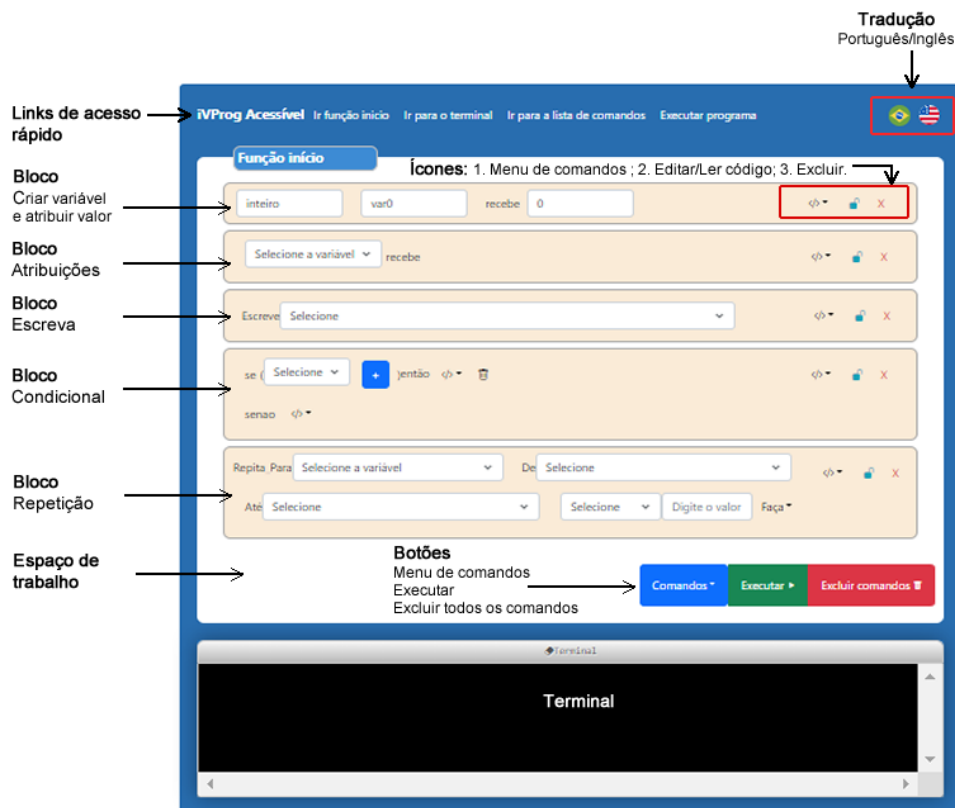


Figura 11: Interface do iVProg4all com os blocos de comando adicionados à Função Início: Nova variável, Atribuição/Matemática, Escreva, Se..Então e Repita N vezes.

Fonte: Elaborado pelo autor.

Na figura 11, é possível observar, no topo, da esquerda para a direita, os links de acesso rápido como (Ir para função inicial, Ir para terminal, Ir para lista de comandos, Executar o programa) que pode ser facilmente utilizado para navegação com *mouse*, teclas de atalho ou leitores de tela. Existem duas áreas separadas: a área de trabalho, onde o algoritmo é criado graficamente usando os blocos de comando, e o terminal, onde a versão em texto do algoritmo e suas saídas são exibidas. O primeiro bloco de comando indica a criação da variável `var0` do tipo `inteiro` e atribui o valor "0" a ela. No lado direito de cada bloco de comando, existem três ícones representando: um atalho para o menu de comandos (representado pelo ícone `</>`), o modo de leitura/edição (representado pelo ícone de um cadeado) e a exclusão do comando (representado pelo ícone X). O segundo bloco de comando indica como é possível selecionar uma variável já existente para atribuir um valor a ela. O terceiro bloco de comando indica o uso do comando `Escreva`, o quarto o uso do bloco condicional `Se ... Então`, e o quinto o uso do bloco de repetição `Repita N Vezes`. Na parte inferior da parte direita da área de trabalho, existem três botões dispostos linearmente. O primeiro abre uma lista de comandos (`Nova Variável`, `Atribuição/Matemática`, `Escreva`, `Se ...Então`, `Repita N vezes`), o segundo executa o código descrito na área de trabalho e o terceiro exclui um comando selecionado.

5.3.1 Recursos de Acessibilidade

Os recursos de acessibilidade implementados no `iVProg4All` são:

1. **Navegação por teclado:** utiliza a tecla `Tab` para ir para o próximo elemento e `Shift + Tab` para retroceder para o elemento anterior; setas para cima (`↑`) e para baixo (`↓`) para navegar em uma lista/menu de seleção, como menu de comandos, menu de tipo de variável, entre outros;
2. **Hierarquia de títulos:** títulos e subtítulos são organizados hierarquicamente nos níveis `h1`, `h2`, `h3...` e assim por diante para facilitar a navegação.
3. **Textos alternativos:** como esta é uma programação visual baseada em formulários, cada elemento do formulário, como botões, campos de texto e menus, possui textos alternativos indicando sua função;
4. **Foco de acordo com o contexto:** durante a execução de uma ação, o foco é direcionado para o elemento subsequente, previsto como consequência da ação. Por exemplo, ao executar um algoritmo, o foco será deslocado para a saída exibida no

terminal. Da mesma forma, ao gerar um bloco de comando, o foco será direcionado para o ponto de início do bloco recém-criado;

5. **Modo de leitura/edição:** Cada bloco é equipado com um comando de leitura/edição, representado por um ícone de cadeado, que permite a leitura de um código baseado em texto gerado pelo bloco de código gráfico.

5.4 Considerações

Neste capítulo foi apresentada uma instancia do VProgForms, cujo resultado concreto foi o iVProg4All, sistema de programação visual baseado em formulário acessível para cegos. O desenvolvimento do iVProg4All segue os ciclos convencionais de desenvolvimento de software, adotando um modelo incremental. Além de incorporar funcionalidades essenciais para a construção de algoritmos, o iVProg4All integra uma variedade de recursos de acessibilidade voltados para usuários cegos, sem comprometer a usabilidade para aqueles que possuem visão.

Os resultados dos testes preliminares conduzidos por um testador cego indicam que, embora alguns problemas tenham sido identificados, o uso do sistema não foi inviabilizado para esse perfil de usuário. As correções sugeridas foram implementadas visando a condução de uma avaliação abrangente abordando acessibilidade, usabilidade e análise de desempenho, contemplando amostras de usuários cegos e videntes. A metodologia para este estudo, sua aplicação e resultados são detalhados no capítulo 6.

6 AVALIAÇÃO DE ACESSIBILIDADE, USABILIDADE E ANÁLISE DE DESEMPENHO DO IVPROG4ALL

Neste capítulo são apresentados os métodos, técnicas e resultados para os estudo de acessibilidade, usabilidade e desempenho relacionados ao uso do iVProg4All por usuários cegos e videntes. A avaliação do iVProg4All e, conseqüentemente, seu modelo subjacente (VProgForms), foi conduzida mediante a aplicação do método de triangulação (ISLAM; BOUWMAN, 2016; BARBOSA; SILVA, 2010). Esse método propicia a análise a partir de diversas perspectivas, integrando abordagens distintas, visando conferir maior confiabilidade aos resultados da pesquisa.

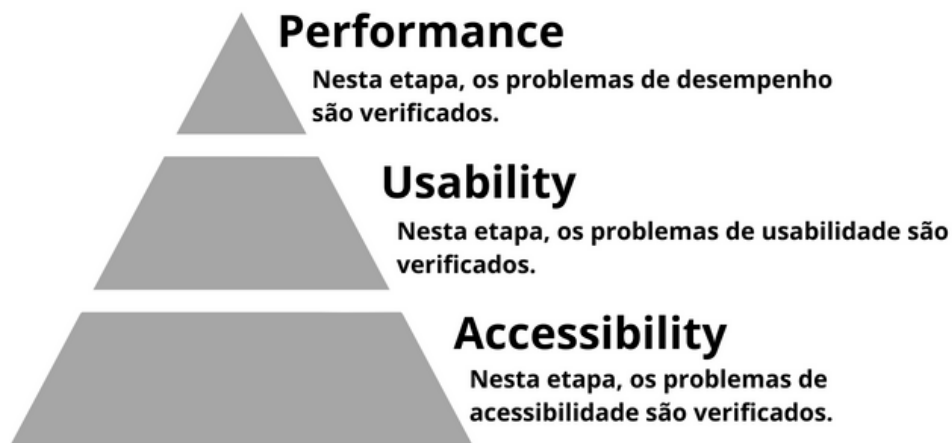
A coleta de dados contou com a participação de indivíduos provenientes de diferentes estados e cidades do Brasil, recrutados por meio de divulgação em instituições, fóruns, grupos *online* e redes sociais. Após aceitarem o convite, todos os participantes tiveram acesso às informações pertinentes e consentiram suas participações voluntariamente por meio de um termo de consentimento livre e esclarecido. Vale ressaltar que o método delineado neste estudo foi submetido e aprovado pelo comitê de ética do Instituto de Psicologia da Universidade de São Paulo, sob a identificação CAEE 31894020.0.0000.5561 (Anexo D).

6.1 Materiais Métodos

Sem acessibilidade, pessoas com deficiência podem ter acesso a sistemas de computador impedido ou prejudicado (NASCIMENTO et al., 2019a). Sendo assim, a acessibilidade configura-se como a base para a interação das pessoas com os sistemas computacionais. Após vencer as barreiras de acessibilidade, é preciso verificar as questões de usabilidade (ROSSON; CARROLL, 2002) para que as pessoas possam utilizar o sistema com mais facilidade e praticidade, conseqüentemente, tendo um bom desempenho no uso. Portanto, nesta pesquisa, para representar os níveis de interação e os métodos aplicados, foi utilizada

uma representação em pirâmide. Conforme apresentado na figura 12, primeiro, verifica-se questões de acessibilidade; depois, verifica-se questões de usabilidade; por fim, é feita uma análise de desempenho quanto ao uso do sistema.

Figura 12: Nesta figura, uma representação piramidal é usada para representar a estrutura dos métodos e resultados da pesquisa. Questões de acessibilidade são verificadas na base, se os usuários podem acessar o software; no meio, são verificados os problemas de usabilidade, se os usuários podem usá-los com facilidade e o nível de satisfação; na parte superior o desempenho é verificado durante o processo interativo com o sistema.



Fonte: Elaborado pelo autor.

Os métodos citados na figura 12 têm suas aplicações e resultados apresentados nas seções 6.3, 6.4 e 6.5 deste capítulo. Para a condução do experimento, foram realizadas sessões observacionais com o propósito de avaliar a usabilidade, acessibilidade e análise de desempenho dos participantes, durante o uso do iVProg4All. Adicionalmente, ao término de cada sessão, foram administrados os questionários de acessibilidade (Anexo A) e usabilidade (Anexo B). Por fim, a análise de desempenho foi feita mediante a análise dos vídeos gravados da interação dos participantes.

A seguir são listadas as técnicas e materiais adotados para condução do estudo.

1. **Sessões de observação:** as sessões de observação foram realizadas de forma remota, no dia e horário previamente agendados com os participantes. Para isso, foi utilizado o Google Meet ¹ para chamada de vídeo com o participante durante a sessão. Foi pedido que o participante compartilhasse sua tela e a câmera enquanto realizava as tarefas. Do lado do pesquisador/observador, toda a sessão foi gravada utilizando um software para gravação de tela. A gravação foi utilizada para análise dos resultados, conforme apresentado na figura 13(para manter a confidencialidade do participante e do pesquisador o rosto de ambos foram borrados).

¹<https://meet.google.com/>

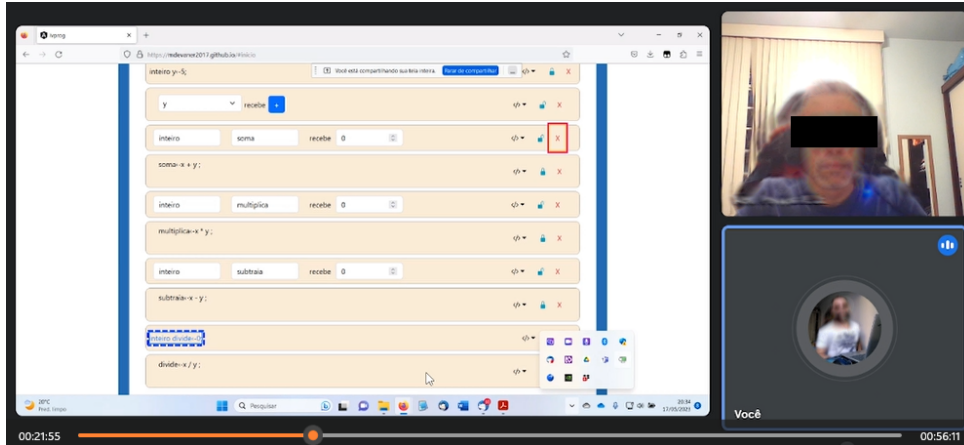


Figura 13: Captura de tela da interação de um participante cego com iVProg4All gravada com , durante o experimento.

Fonte: Elaborado pelo autor.

Foram enviadas as mesmas tarefas definidas nos testes preliminares, a saber:

(a) **Atividade 1 - variáveis, atribuições, operações matemáticas e escrita**

- i. Tarefa 1: crie duas variáveis x e y e atribua um valor a elas;
- ii. Tarefa 2: criar a variável soma e fazê-la receber $x +$ (mais) y ;
- iii. Tarefa 3: criar a variável de multiplicação e fazê-la receber $x *(\text{vezes}) y$;
- iv. Tarefa 4: criar a variável de subtração e fazê-la receber $x -$ (menos) y ;
- v. Tarefa 5: criar a variável divide e fazê-la receber $x /$ (dividido) y ;
- vi. Tarefa 6: criar um comando Escreva para cada variável: soma, multiplica, subtrai e divide para mostrar os valores dessas variáveis no terminal.

(b) **Atividade 2 - se...então estruturas**

- i. Tarefa 1: crie duas variáveis nota1, nota2 e atribua um valor do tipo real a elas;
- ii. Tarefa 2: criar uma variável média do tipo real e atribuir a ela $(\text{nota1} + \text{nota2}) / 2$;
- iii. Tarefa 3: crie uma condição Se...Então para mostrar “Aprovado”, se o valor da variável média for ≥ 5 , e “Reprovado” se a média for < 5 . Para mostrar o valor no console, use o comando Escreva.

(c) **Atividade 3 - repetindo estruturas para...então**

- i. Tarefa 1 - crie uma variável chamada somatório e atribua o valor 0 a ela;
- ii. Tarefa 2 - criar uma estrutura de repetição onde a variável do contador começa com 0 e repete uma instrução até 10;

- iii. Tarefa 3 - no laço de repetição, fazer a variável soma receber a soma dos números de 1 a 10 e mostrar o valor final no console, utilizando o comando Escreva.

Além do papel de observador da interação, o pesquisador poderia tirar dúvidas e dar suporte ao participante durante todo o processo. Foi explicado para os participantes que as atividades não eram obrigatórias, podendo parar no momento em que desejassem. Adicionalmente, ressaltou-se que a avaliação se concentra na usabilidade e acessibilidade do software, não abrangendo a análise de habilidades e conhecimentos técnicos dos participantes.

2. **Pensamento em voz alta:** durante as sessões foi pedido que os usuários fossem expondo verbalmente suas emoções e intenções. Durante esse processo, o participante não foi pressionado a falar. O pensamento em voz alta foi feito de forma voluntária e espontânea.
3. **Questionários de usabilidade e acessibilidade:** ao final de cada sessão, foi pedido aos participantes cegos que respondessem ao questionário de usabilidade SUS e o questionário de acessibilidade adaptado de Franco (2018). Os participantes que enxergam responderam apenas ao questionário de usabilidade. Ambos os questionários estão disponíveis como anexos a esta tese.
4. **Análise de desempenho:** a análise de desempenho dos participantes foi realizada mediante a análise posterior dos vídeos gravados da interação para cada participante. Foi levado em consideração o tempo, em segundos, para realização das tarefas e o número de pedidos de auxílio.

A mobilização de participantes cegos para a pesquisa foi feita por meio do grupo cegos programadores ¹ e entidades que desenvolvem trabalhos com pessoas cegas por todo Brasil. Os participantes que enxergam foram convidados pelos meios de comunicação da Escola Politécnica da Universidade de São Paulo ou por meio das redes sociais. Todos os participantes aceitaram participar de forma voluntária.

Após o convite e aceite dos participantes, foi enviado por e-mail o termo de consentimento livre e esclarecido (TCLE) (Apêndice A) com todas as informações relacionadas ao experimento. O envio do termo e o aceite foram realizados por meio do Google Forms. Após o aceite do termo, foi enviado um e-mail para os participantes pedindo para que

¹https://groups.google.com/u/1/g/cegos_programadores

respondessem ao questionário sócio-educativo, utilizado para levantamento e análise do perfil dos participantes, adaptado de Nogueira et al. (2015)(Apêndice B). Também foi pedido que os participantes respondessem ao e-mail indicando o melhor dia e horário para realização do experimento. O experimento foi realizado de forma síncrona com acompanhamento do pesquisador. Vale ressaltar que, no e-mail enviado, não foi disponibilizado qualquer link para que os participantes pudessem ter acesso prévio ao iVProg4All.

6.2 Perfil dos participantes

O experimento foi conduzido com seis indivíduos cegos e seis videntes, todos do sexo masculino. Todos os participantes cegos relataram possuir uma deficiência visual grave, sendo três delas congênitas e duas adquiridas (um dos participantes não forneceu informações sobre esse ponto). Um ponto de unanimidade entre os participantes cegos, todos relataram o uso do leitor de tela NVDA para utilizar o computador.

Os participantes foram distribuídos em dois grupos: participantes cegos no grupo I e participantes videntes no grupo II. Todos os participantes dos grupos I e II são brasileiros de diferentes regiões. Eles responderam a um questionário socio-educativo (Apêndice B) contendo informações sobre idade, nível de educação, experiência em programação e experiência profissional. Os resultados são apresentados abaixo.

6.2.1 Faixa etária e nível de escolaridade

Conforme apresentado na Tabela 2, observou-se que, em relação à distribuição etária dos participantes, o Grupo I é equilibrado, com 50% dos participantes com idades entre 30 e 59 anos (adultos) e o mesmo percentual entre 21 e 29 anos (jovens adultos). Em contraste, no Grupo II, 75% dos participantes têm idades entre 18 e 29 anos (jovens adultos), enquanto 25% têm idades entre 40 e 49 anos (adultos), indicando uma leve inclinação para um público mais jovem.

Também foram coletadas informações relacionadas à formação dos participantes. Essas informações incluem o nível de educação, apresentado na Tabela 3, e a área de formação, apresentada na Tabela 4, para aqueles com ensino superior ou pós-graduação.

Considerando os valores apresentados nas tabelas 3 e 4, os perfis dos participantes revelam um maior grau de diversidade no Grupo I no que diz respeito ao nível de escolaridade, abrangendo indivíduos com níveis de escolaridade que vão do básico à pós-graduação. Por outro lado, o Grupo II é composto exclusivamente por indivíduos com

Tabela 2: Resumo dos perfis dos participantes sobre a faixa etária. Observe que nesta tabela os valores das quantidades para cada faixa etária estão agrupados nas colunas dos grupos I e II.

Faixa etária (por idade)	Quantidade	
	Grupo I	Grupo II
18 – 20	0	1
21 – 29	3	3
30 – 39	2	0
40 – 49	0	2
50 – 59	1	0

Tabela 3: Resumo dos perfis dos participantes sobre o nível de escolaridade. Observe que nesta tabela os valores das quantidades para cada nível de ensino estão agrupados nas colunas dos grupos I e II.

Nível de escolaridade	Quantidade	
	Grupo I	Grupo II
Ensino médio completo	1	0
Ensino superior incompleto	1	3
Ensino superior completo	2	0
Pós-graduação incompleta	1	0
Pós-graduação completa	1	3

Tabela 4: Resumo dos cursos associados à formação acadêmica dos participantes. Observação: nesta tabela, os valores quantitativos para cada curso mencionado estão agrupados em colunas para os grupos I e II.

Cursos mencionados	Quantidade	
	Grupo I	Grupo II
Sem formação acadêmica	1	0
Análise e Desenvolvimento de Sistemas	3	0
Bacharelado em Ciência da Computação	1	0
Mestrado em Ciência da Computação	1	1
Doutorado em Ciência da Computação	0	1
Engenharia Elétrica	0	3
Mestrado em Física	0	1

ensino superior e pós-graduação. Adicionalmente, o Grupo II goza de uma vantagem em termos de habilitações acadêmicas, uma vez que é composto por um maior número de indivíduos com formação nas áreas de tecnologia e com nível de pós-graduação.

6.2.2 Experiência com programação

O conhecimento dos participantes sobre programação está apresentado na Tabela 5. Assim, é possível observar que o Grupo II pode ter uma leve vantagem sobre o Grupo I,

uma vez que assume um participante, cego, com conhecimento prévio fraco sobre programação. Também foram questionados sobre quais linguagens de programação dominavam.

Tabela 5: Resumo do nível de conhecimento dos participantes sobre lógica de programação. Observe que, nesta tabela, os valores de quantidade para cada nível de conhecimento em lógica de programação estão agrupados em colunas para os grupos I e II.

Conhecimento em lógica de programação	Quantidade	
	Grupo I	Grupo II
Nenhum	0	0
Fraco	1	0
Regular	0	1
Bom	2	2
Muito bom	3	3

Conforme mostra a tabela 6, todos os participantes do Grupo I alegaram proficiência em pelo menos uma linguagem de programação, enquanto no Grupo II, um participante mencionou não ter domínio de nenhuma linguagem de programação. As linguagens de programação comumente mencionadas por ambos os grupos incluem Python, C++ e C. No entanto, o Grupo II citou C++ e C com mais frequência, potencialmente influenciado pela maioria dos participantes com formação acadêmica em na área de tecnologia, onde essas linguagens são comumente enfatizadas.

Tabela 6: Resumo do número de vezes que uma linguagem de programação foi mencionada. Nesta tabela, os valores para cada linguagem mencionada estão agrupados em linhas para os grupos I e II.

	Número de vezes									
	Bash	C	C++	Java	Javascript	Pascal	PHP	Phyton	PySpark	Rust
Grupo I	0	1	1	2	1	1	1	3	0	0
Grupo II	1	5	3	0	0	0	0	6	1	1

Em relação aos softwares usados para aprender programação, ambos os grupos geralmente usam Visual Studio Code e editores de texto como o Notepad. No entanto, o Grupo I relatou o uso de ferramentas adicionais não mencionadas pelo Grupo II, como Eclipse, Visual Studio e Progot. Por outro lado, um membro do Grupo II mencionou o uso de Codeblocks, ferramenta não relatada por nenhum participante do Grupo I.

6.2.3 Experiência profissional

A pesar de a maioria dos participantes dos grupos I e II atuarem na área de informática, conforme apresentado na tabela 8, os grupos possuem perfis diferentes. O perfil do grupo I é mais técnico e do grupo II mais científico. Dois participantes do grupo I e um do grupo II relataram não trabalhar no momento.

Tabela 7: Resumo do número de vezes que o software foi mencionado como uma ferramenta para aprender programação. Note que, nesta tabela, o número de vezes para cada software está agrupado em colunas para os grupos I e II.

Software	Número de vezes	
	Grupo I	Grupo II
Codeblocks	0	1
Eclipse	1	0
Notepad	2	0
Progobot	1	0
Text editor	0	1
Visual Studio Code	3	4

Tabela 8: Resumo da experiência profissional dos participantes, incluindo ocupações mencionadas e se estão atualmente trabalhando.

Pergunta	Opções selecionadas	Quantidade	
		Grupo I	Grupo II
Atualmente trabalhando	Sim	4	5
	Não	2	1
Ocupações mencionadas	Analista de sistemas, técnico	1	0
	Analista de TI	1	0
	Assistente de administração	1	0
	Analista de acessibilidade e programador	1	0
	Cientista e Analista de Dados	0	2
	Pesquisador e professor	0	1
	Pesquisador de Iniciação Científica	0	1
	Desenvolvedor de Software	0	1

6.3 Avaliação de Acessibilidade

A avaliação da acessibilidade foi restrita aos participantes do Grupo I. Dado o caráter do sistema, que é baseado na web, empregamos um questionário de acessibilidade fundamentado nas diretrizes WCAG 2.1, adaptadas de (FRANCO, 2018), disponível no Anexo A. As questões e respostas dos participantes foram categorizadas conforme seu alinhamento com os princípios da WCAG 2.1: Perceptível, apresentado na seção 6.3.1); Compreensível, apresentado na seção 6.3.2; Operável, apresentado na seção 6.3.3 e Robusto, apresentado na seção 6.3.4. Na exposição dos resultados, os participantes são identificados como P1, P2, P3, P4, P5 e P6.

6.3.1 Resultados relacionados ao princípio Perceptível

Para o princípio Perceptível, foram avaliadas três questões específicas. A tabela 9 apresenta tanto essas questões quanto as respostas fornecidas pelos participantes.

Conforme apresentado na tabela 9, a questão 1 teve como objetivo aferir a concordân-

Tabela 9: Resultados para questões relacionadas ao princípio Perceptível (*NA = Não se aplica)

Questões	Respostas às questões					
	P1	P2	P3	P4	P5	P6
1. As informações estavam acessíveis ?	5	3	4	5	4	5
2. Em sua opinião, como estavam as descrições das imagens?	NA	NA	NA	NA	NA	NA
3. Você teve alguma dificuldade para identificar algum elemento da interface?	Sim	Não	Não	Sim	Sim	Sim

cia dos participantes com a acessibilidade às informações, utilizando uma escala de “1” (Discordo totalmente) a “5” (Concordo totalmente). Os resultados revelaram que cinco participantes selecionaram “4” ou “5”, indicando sua concordância, enquanto um participante permaneceu indiferente, optando pela escala “3”. Na questão 2, os participantes foram questionados sobre as descrições das imagens dentro do iVProg4All. Apesar da plataforma conter alguns ícones, a falta de *tags* HTML específicas para essas imagens fez com que todos os participantes escolhessem a opção “Não Aplicável”, o que significa que não perceberam a existência dos ícones. A questão 3 questionou se os participantes encontraram dificuldades em identificar algum elemento da interface. As respostas indicaram que quatro participantes relataram alguma dificuldade, enquanto dois afirmaram não encontrar nenhum desafio, conforme indicado no questionário.

6.3.2 Resultados relacionados ao princípio Compreensível

Para o princípio Compreensível, foram avaliadas quatro questões específicas. A tabela 10 apresenta essas questões e as respostas fornecidas pelos participantes.

Conforme apresentado na tabela 10, na questão 1, é questionado se os links estavam claramente descritos e sugerido seu conteúdo. Novamente, eles responderam em uma escala de “1” (Discordo totalmente) a “5” (Concordo totalmente). Como resultado, quatro dos participantes escolheram a escala “5”, indicando máxima concordância neste ponto, e dois escolheram a escala “4”.

Nas questões 2 e 3, os participantes deveriam avaliar as descrições dos botões e ícones, escolhendo entre “Muito ruim”, “Ruim”, “Razoáveis”, “Boas” ou “Muito boas”. Para a questão 2, dois dos participantes indicaram que eram “Muito boas”, três declararam que eram “Boas” e um “Razoáveis”. Com isso, infere-se que embora as descrições dos botões sejam aceitáveis, elas ainda podem ser melhoradas. Na questão 3, três dos participantes não conseguiram percebê-los, selecionando a opção “Não se aplica”, enquanto um indicou

Tabela 10: Resultados para questões relacionadas ao princípio Compreensível

Questões	Respostas às questões					
	P1	P2	P3	P4	P5	P6
1.Os links são descritos de maneira clara e sugerem seu conteúdo.	5	5	5	4	4	5
2. Em sua opinião, como estavam as descrições dos botões?	Muito boas	Muito boas	Boas	Boas	Razoáveis	Boas
3. Em sua opinião, como estavam as descrições dos ícones?	Muito boas	Não se aplica	Não se aplica	Não se aplica	Razoáveis	Boas
4. Você se identifica com a linguagem textual adotada?	5	4	5	5	5	5

que eram “Muito boas”, um indicou que eram “Razoáveis” e um indicou que eram “Boas”.

Em resposta à questão 4, os participantes selecionaram em uma escala que variava de “1” (Discordo totalmente) a “5” (Concordo totalmente) para indicar seu alinhamento com a compreensão da linguagem textual adotada. Como resultado, cinco dos participantes selecionaram nível de concordância “5” e apenas um selecionou “4”, indicando identificação com a linguagem adotada.

Por fim, foram identificados alguns problemas e melhorias relatados pelos participantes. Entre eles estão: o operador matemático de multiplicação deve ser lido como "operador de multiplicação" em vez de um "operador de asterisco"; os rótulos dos botões de remoção eram ambíguos, portanto não era possível saber o que seria removido; no laço de repetição, o menu de incremento não é intuitivo, lê apenas "selecione"; o botão “Adicionar” para adicionar elementos e montar uma expressão matemática também não é intuitivo, segundo relatos de alguns participantes.

6.3.3 Resultados relacionados ao princípio Operável

As questões relacionadas ao princípio operável, bem como as respostas dos participantes à essas questões foram agrupadas na tabela 11. Na questão 1, os participantes deveriam indicar se os atalhos eram “Extremamente insuficientes”, “Insuficientes”, “Razoáveis”, “Suficientes” ou “Extremamente suficientes”. Na questão 2, os participantes deveriam indicar o nível de facilidade para encontrar as informações, foram disponibilizadas as opções “Extrema facilidade”, “Facilmente”, “Pouca dificuldade”, “Extrema dificuldade”, “Não consigo

encontrar”.

Tabela 11: Resultados para questões relacionadas ao princípio operável

Questões	Respostas às questões					
	P1	P2	P3	P4	P5	P6
1. O que você achou dos atalhos?	Suficientes	Razoáveis	Suficientes	Suficientes	Insuficientes	Insuficientes
2. Você encontra a informação que deseja?	Extrema facilidade	Pouca dificuldade	Extrema facilidade	Facilmente	Facilmente	Facilmente

Conforme apresentado na tabela 11, para a questão 1, relacionada aos atalhos disponíveis, 3 dos participantes apontaram os atalhos como "Suficientes"; 1 "Razoáveis" e 2 "Insuficientes". Esse resultado se reflete no relato dos participantes, onde eles indicaram a necessidade de mais teclas de atalho, principalmente para a criação dos comandos. Para a questão 2, três participantes indicaram encontrar as informações “Facilmente”, 2 com “Extrema facilidade” e apenas 1 com “Pouca dificuldade”. Esses resultados indicam que, embora haja necessidade de mais teclas de atalho, a maioria dos participantes indicou ter encontrado a informação que desejava.

6.3.4 Resultados relacionados ao princípio Robusto

O princípio Robusto aborda a capacidade de interpretação da informação disponível no sistema por uma ampla variedade de agentes, incluindo tecnologias assistivas, com o objetivo de maximizar a compatibilidade (W3C, 2018). Para verificar a conformidade com esse princípio, foram analisados os resultados dos testes conduzidos utilizando o validador Accessmonitor. A análise foi empregada por meio da passagem do link do sistema contendo os blocos de comando, tais como criação de variáveis, expressões aritméticas, instruções de escrita, estruturas condicionais (se/então) e repetições.

Na figura 14 é apresentado um sumário dos resultados de acessibilidade apontados pelo Accessmonitor. Os tópicos relacionados às práticas consideradas "Aceitáveis" na página são identificados pelas linhas iniciadas com um ícone verde. Os tópicos relacionados às práticas consideradas "Não aceitáveis" na página são identificados pelas linhas iniciadas com um ícone vermelho. Os tópicos relacionados às práticas que necessitam de avaliação manual, pois a ferramenta não dispõe de elementos suficientes para avaliá-las automaticamente, são identificados pelas linhas iniciadas com um ícone amarelo.

Conforme apresentado na figura 14, os resultados apontam que o iVProg4All apresenta



Figura 14: Sumário dos resultados de acessibilidade do iVProg4All, gerados pelo ACESSMONITOR

Fonte: Adaptado de AMA (2024)

um excelente nível de cumprimento das WCAG 2.1, obtendo nota 9,7. Dos critérios analisadas pela ferramenta foram considerados aceitáveis 12 (Nível A) e sete de nível AA. A ferramenta indicou para ver manualmente três de nível A e dois de nível AAA. Por fim, considerados como críticos, implementações não são aceitáveis em relação a WCAG 2.1, foi apontado apenas 1 de nível AA. Isso acontece porque foi identificado pela ferramenta que havia um elemento com a semântica de *banner* contido dentro de um elemento com outra semântica. Esses resultados indicam que, embora haja ajustes a serem feitos para atingir a pontuação máxima, o iVprog4All apresenta boa concordância com critérios de acessibilidade estabelecidos pelas WCAG 2.1. O relatório completo gerado pela ferramenta ACESSMONITOR pode ser acessado no Anexo F.

6.4 Resultados para Usabilidade

No contexto da análise de usabilidade, a avaliação incorporou respostas obtidas no questionário do SUS, complementadas por uma investigação subjetiva sobre as impressões dos participantes sobre o iVProg4All. Esses resultados facilitaram um exame comparativo entre os Grupos I e II. A análise utilizou pontuações, notas e adjetivos do SUS, conforme delineado por Bangor et al. (2008), como parâmetros para interpretação dos resultados. Os dados que contrastam as descobertas dos Grupos I e II estão delineados na Tabela 12.

A pontuação média do SUS para o Grupo I foi de 78,3, enquanto o Grupo II registrou pontuação média de 84,2. Este resultado sugere que a usabilidade do iVProg4All pode ser

Tabela 12: Resumo das descobertas de usabilidade derivadas das pontuações SUS obtidas nos Grupos I e II.

Participantes	Grupo I			Grupo II		
	Pontuação SUS	Aceitabilidade	Adjetivo	Pontuação SUS	Aceitabilidade	Adjetivo
P1	92,5	Aceitável	Melhor imaginável	72,5	Aceitável	Bom
P2	60	Marginal/baixa	Bom	85	Aceitável	Excelente
P3	90	Aceitável	Melhor imaginável	90	Aceitável	Melhor imaginável
P4	77,5	Aceitável	Excelente	85	Aceitável	Excelente
P5	72,5	Aceitável	Bom	90	Aceitável	Melhor imaginável
P6	77,5	Aceitável	Excelente	82,5	Aceitável	Excelente

Legenda para pontuação SUS: As pontuações SUS são analisadas em conjunto com a classificação de usabilidade e os adjetivos associados propostos por Bangor et al. (2008) – "Pior imaginável" ≤ 25 ; $25 < \text{"Pobre"} \leq 39$; $39 < \text{"OK"} \leq 51$; $51 < \text{"Bom"} \leq 73$; $73 < \text{"Excelente"} \leq 85$; "Melhor imaginável" > 85

categoricamente caracterizada como “Excelente” para ambos os grupos. Em conjunto com as conclusões quantitativas e qualitativas, os participantes foram incentivados a fornecer relatórios concisos detalhando as suas avaliações do iVProg4All, enfatizando pontos fortes notáveis e áreas potenciais para melhoria do sistema.

No geral, os resultados do teste de usabilidade do iVProg4All indicam pontos positivos e outros a serem melhorados. Considerando a ideia de equidade e inclusão, já que a mesma interface de sistema foi utilizada por ambos os grupos, desatam-se alguns relatos:

Grupo I:

“O programa é muito bom. Sem dúvida eu divulgaria se tivesse permissão”;

“Gostei da iniciativa. Eu o definiria como “ousado”, pois traz um método visual de aprendizagem para o mundo dos deficientes visuais. Possivelmente, pessoas cegas de nascença poderiam se sentir um pouco perdidas (eu me senti um pouco assim), mas é um projeto que pode ajudar no aprendizado de programação”;

“Fluxo prático e de fácil compreensão, necessita de ajustes de usabilidade para retirar a complexidade da operação (não compreensão) e permitir agilidade na execução diária das atividades”;

“É um programa fácil de usar, prático e algo intuitivo”;

“Achei a forma de clicar nos botões muito intuitiva”.

Grupo II:

“A barra de ferramentas é extremamente intuitiva e fácil de usar”;

“A interação visual foi muito boa”;

“Achei a experiência interessante, provavelmente recomendaria para pessoas em fase de aprendizagem por ser uma ferramenta que trabalha os conceitos de forma intuitiva, dispensando a necessidade de conhecer a semântica e sintaxe de outras linguagens mais complexas”;

“Pareceu-me uma ferramenta adequada para aprender programação, principalmente para iniciantes”;

“Achei a ferramenta muito intuitiva, muito simples de usar e entender o que tinha que fazer para realizar as tarefas, e todas as opções de ação estavam sempre à mostra. É ótimo para iniciantes em programação que ainda não têm muito conhecimento de sintaxe, mas querem treinar e aprender mais sobre a própria lógica de programação”;

“Achei a ideia do software muito interessante e acessível”;

“O design é bastante simples (no bom sentido) e fácil de entender”.

Porém, também existem pontos de melhoria de usabilidade, que foram sugeridos em sua maioria pelos participantes do grupo I:

- alterar o texto alternativo do ícone usado no modo `ler/editar`;
- converter o bloco de código em texto pressionando a tecla `Enter`;
- exibir o ícone `Editar bloco` somente enquanto estiver no modo de leitura;
- enumerar os blocos para ajudar os usuários a encontrá-los mais rapidamente;
- fazer atribuições na criação de variáveis;
- criar um atalho para “pular” de um bloco para o outro;
- criar blocos de código em qualquer lugar do algoritmo.

6.5 Análise de Desempenho

A análise de desempenho abrange duas métricas principais: (i) o tempo gasto para executar as atividades descritas na seção 6.1; e (ii) a frequência dos pedidos de assistência durante a sua execução. O processo de contagem foi orientado pelo exame de vídeos gravados. A medição do tempo foi pausada quando os participantes: (a) transitaram para a tela do documento contendo as tarefas; (b) fizeram pausas para observações; ou (c)

receberam instruções após uma solicitação de ajuda relacionada à compreensão da funcionalidade do iVProg4All. Adicionalmente, a contagem de pedidos de ajuda considerou exclusivamente aqueles sobre: (a) utilização do sistema; (b) procedimentos de recuperação de erros do sistema; (c) mal-entendidos durante a construção do algoritmo; e (d) interpretações errôneas dos comandos do sistema.

6.5.1 Análise de desempenho durante a Atividade 1

Inicialmente foram analisadas tarefas consecutivas relacionadas à criação das variáveis x e y , definindo-se o tipo e o valor das mesmas. Assim, foi possível verificar o desempenho dos participantes e o efeito de aprendizagem para esta tarefa. O grupo I obteve um tempo médio de 55,5s para criar a primeira variável (x) e 34,5s para criar a segunda (y). Esses valores indicam uma redução de aproximadamente 38% no tempo gasto para criação de uma variável, após a primeira tentativa. O Grupo II obteve tempo médio de 25,5s para criar x e 14,5s para criar y , indicando uma redução de aproximadamente 43%. Quanto ao número de pedidos de ajuda, os valores foram os mesmos, um pedido para cada grupo.

Considerando o tempo de criação das expressões matemáticas (*e.g.* adição e multiplicação), o grupo I gastou um tempo médio de 129,3s para criar a expressão de adição e 74,3s para criar a expressão de multiplicação. Em contraste, o grupo II gastou 104s para adição e 25,5s para multiplicação. Esses valores indicam uma redução de aproximadamente 43% do tempo para o grupo I e 75% para o grupo II.

Também foi observada disparidade no número de solicitações de atendimento, sendo quatro solicitações no grupo I e apenas uma no grupo II. Embora os resultados sugiram uma vantagem para o grupo II, é essencial observar um fator potencial que influencia o resultado do grupo I. Um participante do grupo I gastou mais tempo na tentativa inicial, optando por excluir a expressão e as variáveis, iniciando novamente o processo. O tempo foi registrado de forma consistente durante esse processo. Apesar disso, os resultados indicam um efeito positivo de aprendizagem para ambos os grupos no que diz respeito à criação de expressões matemáticas.

6.5.2 Análise de desempenho durante a Atividade 2

Ao conduzir a análise da tarefa de criar e usar uma estrutura condicional (*Se... Então*), o acompanhamento do tempo e dos pedidos de assistência são concluídos somente após o cumprimento da tarefa final da atividade. Certos participantes do grupo I

envolveram-se na tarefa várias vezes, motivados por problemas de acessibilidade e usabilidade aparentes no bloco. Apesar de terem concluído uma parte substancial da atividade, alguns participantes optaram por apagar o seu progresso e iniciar a tarefa novamente. Conseqüentemente, o tempo gasto nas tentativas iniciais e subsequentes foi avaliado para esses indivíduos. Esse comportamento não foi observado em nenhum participante do grupo II.

Os resultados revelam que o grupo I alcançou um tempo médio de aproximadamente 264 segundos e sofreu 20 pedidos de ajuda durante a tentativa inicial. Três participantes optaram por refazer a atividade, obtendo tempo médio de 181,6s. Por outro lado, o grupo II alcançou um tempo médio de 123s sem nenhum pedido de assistência. O resultado destaca uma vantagem notável para o grupo II, sugerindo que o bloco *Se...Então* pode representar desafios para o grupo I.

6.5.3 Análise de desempenho durante a Atividade 3

Para a análise do desempenho de criação e utilização da estrutura de repetição, o grupo I obteve tempo médio de aproximadamente 406,1s e 23 pedidos de ajuda, enquanto o grupo II obteve média de 208s e seis pedidos de ajuda. O resultado mostra uma vantagem significativa para o grupo II, indicando que o bloco *Repita N vezes* pode ser problemático para o grupo I. Porém, para realizar esta atividade mesmo o grupo II precisou de mais assistência, percebida pelo número de pedidos de ajuda.

Um fator observado que pode ter impactado nos resultados é a semântica e a sintaxe utilizadas para construir a condição de repetição. Por exemplo, como os participantes já tinham experiência com diferentes linguagens de programação e seus diferentes tipos de sintaxe e semântica, alguns deles tiveram dificuldade de entender como realizar o incremento. Outro problema comum a ambos os grupos está relacionado à adição de um bloco/comando dentro da estrutura de repetição. Não ficou claro para alguns participantes que isso poderia ser feito repetidamente usando o menu *Faça* (Figura 11, canto inferior direito do bloco de repetição).

6.6 Considerações

Neste capítulo, foi apresentado a descrição dos testes realizados por grupos de cegos e videntes, abrangendo as dimensões de acessibilidade, usabilidade e análise de desempenho no âmbito do sistema iVProg4All. Os testes indicam que o iVProg4All é acessível para

cegos e videntes, tendo alcançado valores médios de avaliação bastante competitivos, considerando as métricas adotadas. No capítulo capítulo 7, serão abordadas as discussões e conclusões resultantes deste estudo.

7 CONCLUSÕES

Ao longo desta tese foram realizadas diversas atividades com intuito de responder às seguintes questões de pesquisa:

1. Q1: É possível criar um modelo de programação visual que aplique o conceito de design universal, que possa ser usado por pessoas cegas e videntes sem a necessidade de adaptação?
2. Q2: Um sistema de programação visual pode ser utilizado de forma equânime, no sentido de usabilidade e análise de desempenho por pessoas cegas e videntes?

Para tanto, foram conduzidos estudos que indicaram a adoção do paradigma de programação baseado em formulário como uma solução acessível. Assim, constituiu-se um modelo de programação baseado em formulário (VProgForms), instanciado como modelo subjacente a um sistema de programação visual (iVProg4All) que responderiam positivamente à Q1. De fato, foi implementada uma interface em conformidade com o modelo VProgForms, integrada a um sistema de programação visual (iVProg4All) que foi avaliada por cegos e videntes, mostrando-se acessível para ambos os grupos.

O foco desta tese é propor e avaliar uma abordagem de programação visual que seja acessível para pessoas cegas e videntes. Isso significa que o metamodelo proposto a partir do VProgForms e implementado no iVProg4All visa criar um ambiente de programação visual inclusivo, onde cegos e videntes possam criar seus algoritmos sem a necessidade de adaptação. Para avaliar esta abordagem, foram aplicados métodos empíricos e prospectivos para avaliação de acessibilidade, usabilidade e análise de desempenho.

Os resultados da avaliação mostraram que embora os participantes cegos tenham achado as informações acessíveis, alguns participantes tiveram alguma dificuldade para identificar alguns elementos da interface. Por outro lado, a avaliação quanto à descrição dos links, botões e linguagem adotada foi positiva. Observou-se também que para alguns participantes as teclas de atalho eram insuficientes, o que pode ter impactado no

desempenho desses participantes. No entanto, a maioria dos participantes indicou encontrar informações com facilidade. No que diz respeito à robustez do iVProg4All e ao cumprimento das diretrizes de acessibilidade WCAG 2.1 e eMAG, a verificação com Accessmonitor (WCAG 2.1) indica um nível de conformidade de 9,7 em 10 com verificação via link e 9,3 com envio de arquivo HTML. Já com ASES (eMAG) tem um percentual de 86,07% através do link e 91% através do arquivo HTML. Os resultados indicam que embora ainda existam ajustes a serem feitos em relação à acessibilidade, o iVProg4All apresenta possui um bom nível de acessibilidade levando em consideração a WCAG 2.1 e o eMAG.

Ao avaliar a usabilidade, os resultados indicam uma média de pontuação SUS de 78,3 pelos cegos e 84,2 pelos videntes. A média da pontuação SUS no grupo de cegos alcançou 78,3, ao passo que o grupo de videntes apresentou uma média de pontuação de 84,2. Esses dados indicam que a usabilidade do iVProg4All pode ser categorizada como "Excelente" para ambos os grupos (BANGOR et al., 2008). Porém observa-se que o grupo de videntes fez uma avaliação melhor em relação à usabilidade. Isso pode ter ocorrido dadas as dificuldades encontradas pelos participantes cegos na execução das tarefas e ajustes ainda necessários no iVprog4All para este grupo.

Muitos dos problemas apontados por ambos os grupos estão relacionados a melhorias que precisam ser implementadas para tornar a navegação mais fluida e melhorar a comunicação através de sinalização sonora (para o grupo de cegos) e visual (para o grupo de videntes). Contudo, pode-se inferir que, conforme a avaliação de ambos os grupos, o iVProg4All apresenta boa usabilidade, sendo apropriado tanto para cegos como para videntes.

Durante a análise de desempenho, observou-se que o efeito aprendizagem pode ser um fator positivo para o grupo de cegos, pois se percebeu melhora no tempo e nos pedidos de ajuda quando uma tarefa foi realizada mais de uma vez. Por outro lado, levando em consideração o tempo para realização das tarefas e os pedidos de ajuda durante as tarefas, o grupo de videntes levou vantagem, que foi ainda maior nas atividades com estruturas de seleção e repetição. Isso pode ter ocorrido pela necessidade de maior tempo de configuração e percepção de todo o ambiente utilizando o leitor de tela. Além disso, percebemos que os problemas de acessibilidade e usabilidade presentes no iVProg4All impactaram diretamente nos resultados deste grupo. Assim, entendemos que a correção desses problemas e melhorias em resposta à usabilidade e acessibilidade podem mitigar a diferença de desempenho entre os grupos cegos e videntes.

Ao revisitar os objetivos (capítulo 1) e questões de pesquisa levantados (capítulo 2), visando a verificação se os objetivos foram alcançados e as questões respondidas com os resultados obtidos, possível concluir que:

1. O objetivo de propor um modelo de programação visual baseado em formulários, alinhado aos princípios do design universal para garantir uma interface acessível tanto para usuários com deficiência visual quanto para aqueles sem tal condição, foi alcançado. O modelo proposto foi descrito no capítulo 4 e uma instanciação do mesmo foi descrita no capítulo 5 e avaliada no capítulo 6. Desta avaliação, evidencia-se o êxito do modelo aplicado ao iVProg4All, pois cegos e videntes utilizaram o iVProg4All com uma avaliação de usabilidade positiva. Essa conquista atende à questão de pesquisa, demonstrando que é viável criar um modelo de programação visual que adote os preceitos do design universal, proporcionando uma utilização igualmente eficaz por parte de pessoas cegas e videntes, sem a necessidade de adaptações.
2. Os resultados apresentados indicam que a avaliação empírica da eficiência do modelo VProgForms, por meio do iVProg4All, utilizando métricas de usabilidade, acessibilidade e desempenho, foi conduzida com resultados positivos, especialmente no que se refere à acessibilidade e usabilidade. Portanto, os achados sugerem que a abordagem proposta, utilizando o modelo VProgForms, é promissora como um modelo para instâncias de sistemas de programação visual baseada em formulário, que sejam acessíveis para cegos e videntes. Com base nesses resultados, pode-se afirmar que o segundo objetivo desta pesquisa foi alcançado.

Em resposta à questão de pesquisa Q2, que indaga sobre a extensão na qual um sistema de programação visual pode ser utilizado de maneira equitativa por pessoas cegas e videntes, os resultados revelam que, embora haja uma disparidade de desempenho entre os dois grupos em termos do número de solicitações de auxílio e do tempo necessário para concluir as tarefas, as avaliações de usabilidade para ambos os grupos sugerem um certo grau de equidade nos resultados. Isso é evidenciado pela classificação "Excelente" atribuída ao iVProg4All com base na média dos escores do Sistema de Usabilidade (SUS) para ambos os grupos. No entanto, nota-se que melhorias são necessárias para garantir que a interação de pessoas cegas com o sistema não seja prejudicada, contribuindo assim para o aprimoramento de seus níveis de desempenho.

Por fim, conclui-se que embora sejam necessários ajustes e melhorias em relação à usabilidade e acessibilidade do sistema iVprog4All, o modelo subjacente de programação

visual baseado em formulários (VProgForms) pode ser uma contribuição para criação de novos sistemas de programação visual acessíveis, possibilitando as interações entre usuários cegos e aqueles que enxergam. Por exemplo, em um contexto educacional, alunos (ou professores) cegos e aqueles que enxergam poderão realizar tarefas cooperativamente, usando a mesma ferramenta sem a necessidade de adaptação, criando um ambiente de aprendizado inclusivo.

7.1 Oportunidades para Trabalhos futuros

Nesta seção são propostas algumas sugestões de continuidade da pesquisa, podendo ser realizadas como pesquisas futuras na graduação e pós graduação. Entre as sugestões estão:

1. implementar as melhorias de usabilidade e acessibilidade indicadas neste trabalho para uma nova avaliação, a fim de verificar se essas melhorias podem impactar positivamente na usabilidade e acessibilidade para usuários cegos;
2. realizar um estudo da aplicabilidade e desempenho do iVProg4All em cursos introdutórios de programação;
3. realizar um estudo de usabilidade e acessibilidade com usuários que vivem com outros tipos de deficiência, como surdez, deficiência motora, entre outras;
4. Implementar outras propostas de sistemas de programação visual baseada em formulários a partir do metamodelo VProgForms;
5. Evoluir o metamodelo VProgForms visando atender a pessoas com outros tipos de deficiência;

7.2 Contribuições

Como contribuição científica desta tese tem-se a definição do modelo VProgForms. A contribuição tecnológica, destaca-se o desenvolvimento do iVProg4All. Como forma de divulgar o conhecimento gerado ao longo desta teste, foram produzidos os seguintes artigos científicos:

1. NASCIMENTO, Marcos; BRANDÃO, Anarosa. Um modelo de acessibilidade para cegos em sistemas de programação visual. In: Anais dos Workshops do Congresso

Brasileiro de Informática na Educação. 2019. p. 1467.

<<http://milanesa.ime.usp.br/rbie/index.php/wcbie/article/view/9120>>

2. DO NASCIMENTO, Marcos Devaner et al. Which visual programming language best suits each school level? A look at Alice, iVProg, and Scratch. In: 2019 IEEE World Conference on Engineering Education (EDUNINE). IEEE, 2019. p. 1-6.
<<https://ieeexplore.ieee.org/document/8875788>>
3. DO NASCIMENTO, Marcos D. et al. Overcoming Accessibility Barriers for People with Severe Vision Impairment in Web-based Learning Environments: A Literature Review. In: 2019 IEEE Frontiers in Education Conference (FIE). IEEE, 2019. p. 1-8.
<<https://ieeexplore.ieee.org/document/9028484>>
4. M. D. Do Nascimento, A. A. F. Brandão, L. De Oliveira Brandão and T. M. Casal, "Towards iVProg4All: An Accessibility Test with Blind," 2023 IEEE Frontiers in Education Conference (FIE), College Station, TX, USA, 2023, pp. 01-05, doi: 10.1109/FIE58773.2023.10343224. <<https://ieeexplore.ieee.org/document/10343224>>

Ainda, o iVProg4All foi agraciado com a terceira colocação na categoria protótipo do concurso App.Edu, no CBIE 2023.

REFERÊNCIAS

AMA, A. p. a. M. A. **Uma ferramenta do ecossistema do acessibilidade.gov.pt.** 2024. <https://accessmonitor.acessibilidade.gov.pt/>.

ASCOM/MDHC. **Brasil tem 18,6 milhões de pessoas com deficiência, indica pesquisa divulgada pelo IBGE e MDHC.** 2023. Disponível em: <<https://agenciadenoticias.ibge.gov.br/agencia-noticias/2012-agencia-de-noticias/noticias/37317-pessoas-com-deficiencia-tem-menor-acesso-a-educacao-ao-trabalho-e-a-renda>>. Acesso em: 08 dez 2023.

BANGOR, A.; KORTUM, P. T.; MILLER, J. T. An empirical evaluation of the system usability scale. **Intl. Journal of Human–Computer Interaction**, Taylor & Francis, v. 24, n. 6, p. 574–594, 2008.

BARBOSA, S.; SILVA, B. **Interação humano-computador.** [S.l.]: Elsevier Brasil, 2010.

BILLAH, S.; ASHOK, V.; PORTER, D.; RAMAKRISHNAN. Ubiquitous accessibility for people with visual impairments: Are we there yet? In: **CHI 2017 - Proceedings of the 2017 ACM SIGCHI Conference on Human Factors in Computing Systems.** [S.l.]: Association for Computing Machinery, 2017. (Conference on Human Factors in Computing Systems - Proceedings), p. 5862–5868.

BORYSOWICH, C. Prototyping: Types of prototypes. **toolbox.com**, v. 2011, 2007.

BOSHERNITSAN, M.; DOWNES, M. S. **Visual programming languages: A survey.** [S.l.]: Computer Science Division, University of California, 1997.

BOURNE, R.; STEINMETZ, J. D.; FLAXMAN, S.; BRIANT, P. S.; TAYLOR, H. R.; RESNIKOFF, S.; CASSON, R. J.; ABDOLI, A.; ABU-GHARBIEH, E.; AFSHIN, A. et al. Trends in prevalence of blindness and distance and near vision impairment over 30 years: an analysis for the global burden of disease study. **The Lancet global health**, Elsevier, v. 9, n. 2, p. e130–e143, 2021.

BRANDÃO, L. de O.; RIBEIRO, R. da S.; BRANDÃO, A. A. A system to help teaching and learning algorithms. In: IEEE. **2012 Frontiers in Education Conference Proceedings.** [S.l.], 2012. p. 1–6.

BROCK, M.; QUIGLEY, A.; KRISTENSSON, P. O. Change blindness in proximity-aware mobile interfaces. In: ACM. **Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems.** [S.l.], 2018. p. 43.

BURGSTHALER, S. **Equal access: Universal design of cyber learning projects.** [S.l.]: Seattle, WA: University of Washington. Retrieved from <https://www...>, 2019.

- BURNETT, M.; ATWOOD, J.; DJANG, R. W.; REICHWEIN, J.; GOTTFRIED, H.; YANG, S. Forms/3: A first-order visual language to explore the boundaries of the spreadsheet paradigm. **Journal of functional programming**, Cambridge University Press, v. 11, n. 2, p. 155–206, 2001.
- BURNETT, M. M.; MCINTYRE, D. W. Visual programming. **Computer-Los Alamitos-**, IEEE INSTITUTE OF ELECTRICAL AND ELECTRONICS, v. 28, p. 14–14, 1995.
- CAIADO, K. R. M. Convenção internacional sobre os direitos das pessoas com deficiências: destaques para o debate sobre a educação. **Revista Educação Especial**, Universidade Federal de Santa Maria, v. 22, n. 35, p. 329–338, 2009.
- CAMARGO, L. S. de A.; FAZANI, A. J. Explorando o design participativo como prática de desenvolvimento de sistemas de informação. **InCID: Revista de Ciência da Informação e Documentação**, v. 5, n. 1, p. 138–150, 2014.
- CARNEGIE MELLON UNIVERSITY. **How Tos**. 2023. Disponível em: <<http://www.alice.org/resources/alice-3-how-tos/>>. Acesso em: 23 de fevereiro.
- CHANG, C.-K.; YANG, Y.-F.; TSAI, Y.-T. Exploring the engagement effects of visual programming language for data structure courses. **Education for Information**, IOS Press, v. 33, n. 3, p. 187–200, 2017.
- CONSORTIUM, W. W. W. et al. Cartilha de acessibilidade na web-w3c brasil. **Retrieved April**, v. 5, p. 2017, 2013.
- COURAGE, C.; BAXTER, K. **Understanding your users: A practical guide to user requirements methods, tools, and techniques**. [S.l.]: Gulf Professional Publishing, 2005.
- DARIN, T. G.; ANDRADE, R.; MERABET, L. B.; SÁNCHEZ, J. H. Investigating the mode in multimodal video games: Usability issues for learners who are blind. In: **ACM. Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems**. [S.l.], 2017. p. 2487–2495.
- DOBESOVA, Z. E-learning for visual programming language. In: **IEEE. 2014 IEEE 12th IEEE International Conference on Emerging eLearning Technologies and Applications (ICETA)**. [S.l.], 2014. p. 103–108.
- EVERIS. **Pesquisa brasileira do uso de leitores de tela**. 2019. Disponível em: <https://mwpt.com.br/wp-content/uploads/2019/07/Pesquisa-LDT_Relatorio.pdf>. Acesso em: April 23th, 2022.
- FAN, M.; LIN, J.; CHUNG, C.; TRUONG, K. N. Concurrent think-aloud verbalizations and usability problems. **ACM Transactions on Computer-Human Interaction (TOCHI)**, ACM New York, NY, USA, v. 26, n. 5, p. 1–35, 2019.
- FELIX, I.; SOUZA, L.; BRANDÃO, L.; FERREIRA, B.; BRANDÃO, A. ivprog: Programação interativa visual e textual na internet. In: **Anais dos Workshops do Congresso Brasileiro de Informática na Educação**. [S.l.: s.n.], 2019. v. 8, n. 1, p. 1164.

FINO, C. N. Vygotsky e a zona de desenvolvimento proximal (zdp): três implicações pedagógicas. **Revista Portuguesa de educação**, v. 14, p. 273–291, 2001.

FRANCO, R. P. **Audiodescrição Em Objetos De Aprendizagem Na Plataforma Ead Dell Accessible Learning**. Dissertação (Mestrado) — Universidade Estadual do Ceará, Fortaleza-CE, 2018.

GERALDO, R. J. **Um auxílio à navegação acessível na web para usuários cegos**. Tese (Doutorado) — Universidade de São Paulo, 2016.

GLINERT, E. P.; TANIMOTO, S. L. Pict: An interactive graphical programming environment. **Computer**, IEEE Computer Society, v. 17, n. 11, p. 7–25, 1984.

GOOGLE. **Experimente o Blockly**. 2023. Disponível em: <<https://developers.google.com/blockly?hl=pt-br>>. Acesso em: 23 de fevereiro.

HAUSE, M. et al. The sysml modelling language. In: **Fifteenth European Systems Engineering Conference**. [S.l.: s.n.], 2006. v. 9, p. 1–12.

HUNSAKER, E. Dash.

ISLAM, M. N.; BOUWMAN, H. Towards user-intuitive web interface sign design and evaluation: A semiotic framework. **International Journal of Human-Computer Studies**, v. 86, p. 121–137, 2016. ISSN 1071-5819. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S1071581915001615>>.

JUNIOR, M. A. L. D. **Análise de métodos de avaliação de acessibilidade em sites para deficientes visuais**. Dissertação (B.S. thesis) — Universidade Tecnológica Federal do Paraná, 2023.

KHURANA, R.; MCISAAC, D.; LOCKERMAN, E.; MANKOFF, J. Nonvisual interaction techniques at the keyboard surface. In: ACM. **Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems**. [S.l.], 2018. p. 11.

LEWIS, J. R. The system usability scale: Past, present, and future. **International Journal of Human-Computer Interaction**, Taylor Francis, v. 34, n. 7, p. 577–590, 2018. Disponível em: <<https://doi.org/10.1080/10447318.2018.1455307>>.

LUDI, S. Position paper: Towards making block-based programming accessible for blind users. In: IEEE. **2015 IEEE Blocks and Beyond Workshop (Blocks and Beyond)**. [S.l.], 2015. p. 67–69.

MACLEOD, H.; BENNETT, C. L.; MORRIS, M. R.; CUTRELL, E. Understanding blind people's experiences with computer-generated captions of social media images. In: ACM. **Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems**. [S.l.], 2017. p. 5988–5999.

MDN, M. **ARIA**. 2023. Url <https://developer.mozilla.org/pt-BR/docs/Web/Accessibility/ARIA>.

MEERBAUM-SALANT, O.; ARMONI, M.; BEN-ARI, M. Learning computer science concepts with scratch. In: **Proceedings of the Sixth international workshop on Computing education research**. [S.l.: s.n.], 2010. p. 69–76.

- MILNE, L. R. Blocks4all: making block programming languages accessible for blind children. **ACM SIGACCESS Accessibility and Computing**, ACM New York, NY, USA, n. 117, p. 26–29, 2017.
- MILNE, L. R.; LADNER, R. E. Blocks4all: overcoming accessibility barriers to blocks programming for children with visual impairments. In: **Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems**. [S.l.: s.n.], 2018. p. 1–10.
- MORRIS, M. R.; JOHNSON, J.; BENNETT, C. L.; CUTRELL, E. Rich representations of visual content for screen reader users. In: ACM. **Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems**. [S.l.], 2018. p. 59.
- MOUNTAPMBEME, A.; OKAFOR, O.; LUDI, S. Accessible blockly: An accessible block-based programming library for people with visual impairments. In: **The 24th International ACM SIGACCESS Conference on Computers and Accessibility**. [S.l.: s.n.], 2022. p. 1–15.
- MYERS, B. A. Taxonomies of visual programming and program visualization. **Journal of Visual Languages & Computing**, Elsevier, v. 1, n. 1, p. 97–123, 1990.
- NASCIMENTO, M. D. D.; BRANDÃO, A. A.; BRANDÃO, L. D. O.; CASAL, T. M. Towards ivprog4all: an accessibility test with blind. In: IEEE. **2023 IEEE Frontiers in Education Conference (FIE)**. [S.l.], 2023. p. 01–05.
- NASCIMENTO, M. D. do; BRANDÃO, A. A.; BRANDÃO, L. de O.; OLIVEIRA, F. C. de M. Overcoming accessibility barriers for people with severe vision impairment in web-based learning environments: A literature review. In: IEEE. **2019 IEEE Frontiers in Education Conference (FIE)**. [S.l.], 2019. p. 1–8.
- NASCIMENTO, M. D. do; FÉLIX, I. M.; FERREIRA, B. M.; SOUZA, L. M. de; DANTAS, D. L.; BRANDÃO, L. de O.; BRANDÃO, A. de O. Which visual programming language best suits each school level? a look at alice, ivprog, and scratch. In: IEEE. **2019 IEEE World Conference on Engineering Education (EDUNINE)**. [S.l.], 2019. p. 1–6.
- NIELSEN, J. **Usability engineering**. [S.l.]: Morgan Kaufmann, 1994.
- _____. Why you only need to test with 5 users. **Nielsen Norman Group, Nielsen**, 2000.
- NOGUEIRA, T. d. C. et al. Estudo comparativo da experiência de usuários cegos e videntes no design web responsivo e não responsivo. Universidade Federal de Goiás, 2015.
- OMG, O. M. G. **Unified Architecture Framework**. 2022. Disponível em: <<https://www.omg.org/spec/UAF/1.2>>. Acesso em: 30 ago 2022.
- Organização das Nações Unidas, O. **Agenda 2030 para o Desenvolvimento Sustentável**. 2015. Disponível em: <<https://brasil.un.org/pt-br/91863-agenda-2030-para-o-desenvolvimento-sustent%C3%A1vel>>. Acesso em: 30 jan 2023.
- Organização Mundial de Saúde. World report on vision. Organização Mundial de Saúde, 2019.

- PACIELLO, M. **Web accessibility for people with disabilities**. [S.l.]: Crc Press, 2000.
- PEFFERS, K.; TUUNANEN, T.; ROTHENBERGER, M. A.; CHATTERJEE, S. A design science research methodology for information systems research. **Journal of management information systems**, Taylor & Francis, v. 24, n. 3, p. 45–77, 2007.
- PERIŠA, M.; PERAKOVIĆ, D.; REMENAR, V. Guidelines for developing e-learning system for visually impaired. **Universal Learning Desing**, v. 2, p. 167–173, 2012.
- PIMENTEL, M. Design science research e pesquisas com os cotidianos escolares para fazer pensar as pesquisas em informática na educação. In: **Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)**. [S.l.: s.n.], 2017. v. 28, n. 1, p. 414.
- PIMENTEL, M.; FILIPPO, D.; SANTORO, F. M. **Design Science Research: fazendo pesquisas científicas rigorosas atreladas ao desenvolvimento de artefatos computacionais projetados para a educação**. [S.l.]: Metodologia de Pesquisa Científica em Informática na Educação, 2019. v. 1.
- PIMENTEL, M.; FILIPPO, D.; SANTOS, T. M. dos. Design science research: pesquisa científica atrelada ao design de artefatos. **RE@ D-Revista de Educação a Distância e eLearning**, v. 3, n. 1, p. 37–61, 2020.
- PRATES, R. O.; BARBOSA, S. D. J. Avaliação de interfaces de usuário–conceitos e métodos. In: SN. **Jornada de Atualização em Informática do Congresso da Sociedade Brasileira de Computação, Capítulo**. [S.l.], 2003. v. 6, p. 28.
- PRESIDÊNCIA DA REPÚBLICA - CASA CIVIL. **DECRETO Nº 5.296 DE 2 DE DEZEMBRO DE 2004**. 2004. Disponível em: <http://www.planalto.gov.br/ccivil_03/_ato2004-2006/2004/decreto/d5296.htm>. Acesso em: 17 de outubro.
- _____. **DECRETO Nº 6.949, DE 25 DE AGOSTO DE 2009**. 2023. Disponível em: <http://www.planalto.gov.br/ccivil_03/_ato2007-2010/2009/decreto/d6949.htm>. Acesso em: 17 de outubro.
- PRESSMAN, R. S.; MAXIM, B. R. **Engenharia de software-9**. [S.l.]: McGraw Hill Brasil, 2021.
- PUPO, D. T.; MELO, A. M.; FERRÉS, S. P. Acessibilidade: discurso e prática no cotidiano das bibliotecas. **Campinas: Unicamp/Biblioteca Central Cesar Lattes**, v. 31, 2006.
- RIBEIRO, R. S.; BRANDÃO, L. O.; RODRIGUES, P. A.; BRANDÃO, A. A.; ISOTANI, S. ivprog e itarefa: aprimorando o ensino de algoritmos e programação para iniciantes. In: **Anais dos Workshops do Congresso Brasileiro de Informática na Educação**. [S.l.: s.n.], 2012. v. 1, n. 1.
- RIVERA, J. F. Wai-aria, una aproximación. **No Solo Usabilidad**, n. 8, 2009.
- ROSSON, M. B.; CARROLL, J. M. Chapter 7 - usability evaluation. In: ROSSON, M. B.; CARROLL, J. M. (Ed.). **Usability Engineering**. San Francisco: Morgan Kaufmann, 2002, (Interactive Technologies). p. 227–271. ISBN 978-1-55860-712-5. Disponível em: <<https://www.sciencedirect.com/science/article/pii/B9781558607125500084>>.

SCHEUERMAN, M. K.; EASLEY, W.; ABDOLRAHMANI, A.; HURST, A.; BRANHAM, S. Learning the language: The importance of studying written directions in designing navigational technologies for the blind. In: ACM. **Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems**. [S.l.], 2017. p. 2922–2928.

SCRATCH FOUNDATION. **Acerca do Scratch**. 2023. Disponível em: <<https://scratch.mit.edu/about>>. Acesso em: 23 de fevereiro.

SOUZA, L. M. de. **VCAT: An automatic assessment model for visual programming languages**. Dissertação (Mestrado em Ciências da Computação) — Universidade de São Paulo, São Paulo, 2023.

STILL, B.; CRANE, K. **Fundamentals of user-centered design: A practical approach**. [S.l.]: CRC press, 2017.

TANNUS, A. M.; GONÇALVES, A. F.; CARVALHO, A. E.; GUEDES, I. O.; HONORATO, S. D. Métodos de avaliação de usabilidade e acessibilidade para inclusão digital de portadores de deficiência visual. 2021.

W3C. Web content accessibility guidelines (wcag) 2.1. **WWW Consortium (W3C)**, v. 290, p. 1–34, 2008.

_____. **Web Content Accessibility Guidelines (WCAG) 2.1**. 2018. Disponível em: <<https://www.w3.org/TR/WCAG21/>>. Acesso em: 17 de outubro.

_____. **Tables Tutorial**. 2019. <https://www.w3.org/WAI/tutorials/tables/>.

W3C School. **HTML <form> Tag**. 2023. Disponível em: <https://www.w3schools.com/tags/tag_form.asp>. Acesso em: accessed 28 December 2003.

WEBAIM. **Screen Reader User Survey**. 2019. Disponível em: <<https://webaim.org/projects/screenreadersurvey8/#primary>>. Acesso em: April 23th, 2022.

ZHAO, Y.; BENNETT, C. L.; BENKO, H.; CUTRELL, E.; HOLZ, C.; MORRIS, M. R.; SINCLAIR, M. Enabling people with visual impairments to navigate virtual reality with a haptic and auditory cane simulation. In: ACM. **Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems**. [S.l.], 2018. p. 116.

APÊNDICE A – TERMO DE CONSENTIMENTO LIVRE E ESCLARECIDO

Prezado (a) Senhor (a), O (a) Sr (a). está sendo convidado a participar da pesquisa: “Uma abordagem de programação visual acessível para cegos” que tem por objetivo Propor e avaliar uma abordagem para sistemas de programação visual que dê suporte a pessoas cegas e aquelas que enxergam durante o processo de aprendizado do pensamento computacional e da lógica de programação. Essa pesquisa será realizada com dois grupos de pessoas, o primeiro grupo (GRUPO 1) de pessoas cegas e o segundo grupo (GRUPO 2) de pessoas que enxergam. Para participar da pesquisa, essas pessoas devem possuir conhecimento prévio de lógica de programação, seja ele acadêmico ou profissional; e a disponibilidade deverá ser voluntária.

Os participantes irão realizar um estudo de usabilidade e acessibilidade do sistema de programação visual iVProg4all. O estudo será realizado de forma remota, no dia e horário da preferência do participante. Para isso, será necessário uma conta Google, para acessar o Google Meet. O estudo terá uma duração de aproximadamente 1h. Serão passados 3 tarefas, uma relacionada a variáveis e atribuições, uma relacionada a estrutura se... então e uma relacionada a laço de repetição para... então. A execução dessas tarefas será acompanhada por um pesquisador-observador que irá catalogar as suas dificuldades durante a realização das tarefas usando o iVProg4all. Para o estudo de usabilidade e acessibilidade serão utilizados os métodos de pensamento em voz alta, onde ao utilizar o sistema você deverá relatar em voz alta as dificuldades encontradas para realização das tarefas. Também serão aplicados questionários de usabilidade e acessibilidade. Os participantes que enxergam devem responder apenas ao questionário de usabilidade.

Toda interação dos participantes com o iVProg4all será gravada, utilizando um programa gravador de tela, para análises futuras. Não será divulgada nenhuma imagem ou identificação dos participantes. Vale ressaltar que a competência técnica dos participan-

tes não será avaliada. Deixamos explícito neste documento a seguinte afirmação: o que estará sendo avaliado será a acessibilidade e usabilidade do iVProg4all e não o aprendizado e competências técnicas dos participantes. Os participantes ficarão resguardados das seguintes garantias:

1. Será garantida a plena liberdade do participante da pesquisa, de recusar-se a participar ou retirar seu consentimento, em qualquer fase da pesquisa, sem penalização alguma;
2. Ressarcimento das despesas tidas pelos participantes da pesquisa e dela decorrentes;

Entre os possíveis riscos da pesquisa estão a possibilidade do participante não conseguir executar as atividades solicitadas utilizando os sistemas e sentir-se constrangido por essa razão. Para minimizar esse risco, será oferecido suporte por meio do pesquisador. Além disso, será esclarecido que as habilidades e conhecimento técnico não estarão sendo avaliados, e sim a interação dos participantes com os sistemas avaliados. Há também a possibilidade de estresse durante a aplicação do estudo. Para minimizar o estresse e garantir o bem estar dos participantes, será preparado um ambiente agradável, acessível e uma condução amigável do estudo. O risco de constrangimento ao responderem as perguntas pessoais será minimizado garantindo que, no momento dessas coletas de dados, estarão presentes apenas o participante e o pesquisador.

Declaramos que os dados pessoais dos participantes não serão divulgados e que terão o direito a uma cópia deste Termo no formato digital acessível. Todas as despesas tidas pelos participantes da pesquisa e dela decorrentes serão ressarcidas, caso se aplique. Sempre que necessário, você poderá entrar em contato com os pesquisadores para esclarecimento de eventuais dúvidas: MSc Marcos Devaner do Nascimento, telefone (11) 947742707 e e-mail: marcos.devaner@usp.br integrante do Laboratório de Técnicas Inteligentes - LTI, Escola Politécnica da Universidade de São Paulo - EPUSP, situado na Av. Prof. Luciano Gualberto, trav.3, n.158, Prédio de Engenharia de Eletricidade, sala C2-50, Cidade Universitária, CEP: 05508-970 - São Paulo - SP/Brasil, Phone: x55-11-3091-5397 Fax: x55-11-3091-5294.

Sua participação é importante e voluntária e vai gerar informações que serão úteis para a criação de uma abordagem de programação visual acessível que atende a usuários cegos e não cegos. Após o aceite, uma cópia com a data e local do aceite será enviada para o e-mail de acesso ao Google Forms.

De posse das informações sobre a pesquisa intitulada “Uma abordagem de programa-

ção visual acessível para cegos” concordo voluntariamente em participar dela, de forma livre e esclarecida.

APÊNDICE B – QUESTIONÁRIO SOCIOEDUCATIVO

1. Qual o seu gênero?

- Masculino()
- Feminino()
- Prefiro não dizer ()
- Outro:

2. Qual sua faixa etária?

- Entre 18 e 20 anos()
- Entre 21 e 29 anos()
- Entre 40 e 49 anos()
- Entre 50 e 59 anos()
- 60 ou mais()

3. Qual cidade e estado você mora atualmente?

4. Qual seu grau de escolaridade?

- Ensino fundamental completo ()
- Ensino médio incompleto ()
- Ensino médio completo ()
- Ensino superior incompleto ()
- Ensino superior completo ()
- Pós-graduação incompleto ()
- Pós-graduação completo ()

Para o superior concluído ou em andamento, qual seu curso?

5. Há quanto tempo utiliza computador?

Menos de um ano () Entre 1 e 2 anos () Entre 2 e 4 anos () Mais de 4 anos ()

6. Onde você utiliza computador? Observação: Você poderá selecionar mais de uma opção.

Casa () Trabalho () Escola/faculdade () Outros ()

Caso selecione a opção Outros, onde costuma utilizar?

7. Com que frequência utiliza o computador?

- Diariamente () Quantas horas por dia?
- Semanalmente () Quantas vezes por semana?

8. Há quantos anos ou meses você utiliza a internet?

9. Qual seu nível de conhecimento em lógica de programação?

- Nenhum()
- Pouco()
- Regular()
- Bom()
- Muito bom()

10. Você domina alguma linguagem de programação?

sim() não()

Qual?

11. Qual software você costumava utilizar para aprender a programar?

12. Você trabalha atualmente?

sim() não()

Caso sua resposta seja sim, qual função você exerce?

13. Você possui algum tipo de deficiência?

Sim () Não ()

Caso sua resposta seja sim, qual o tipo e a severidade da deficiência?

QUESTÕES DIRECIONADAS PARA PARTICIPANTES CEGOS

A sua deficiência visual foi:

Congênita () Adquirida ()

Caso seja adquirida, há quanto tempo você vive com essa deficiência?

Qual leitor de tela você mais utiliza com mais frequência

DOSVOX () NVDA () JAWS () Virtual Vision () Outro ()

Caso, selecione a opção "Outro", qual você utiliza?

ANEXO A – QUESTIONÁRIO PARA AVALIAÇÃO DE ACESSIBILIDADE

1. As informações estavam acessíveis?

Discordo totalmente Discordo Indiferente Concordo Concordo totalmente

2. Em sua opinião, como estavam as descrições das imagens?

Muito ruins Ruins Razoáveis Boas Muito boas Não se aplica

3. Em sua opinião, como estavam as descrições dos ícones?

Muito ruins Ruins Razoáveis Boas Muito boas

4. Em sua opinião, como estavam as descrições dos botões?

Muito ruins Ruins Razoáveis Boas Muito boas

5. Os links são descritos de maneira clara e sugerem seu conteúdo

Discordo totalmente Discordo Indiferente Concordo Concordo totalmente

6. O que você achou dos atalhos?

Extremamente insuficiente Insuficiente Razoáveis Suficiente Extremamente suficiente

7. Você encontra a informação que deseja?

Com extremamente facilidade Facilmente Com um pouco de dificuldade Com extrema dificuldade Não encontro

8. Você se identifica com a linguagem textual adotada?

Discordo totalmente Discordo Indiferente Concordo Concordo totalmente

Se a resposta for negativa, comente.

9. **Você teve alguma dificuldade para identificar algum elemento da interface? Se sim, faça um pequeno relato.**

10. **Você encontrou dificuldades para navegar?**

Não Sim

Se a resposta for sim, quais foram as dificuldades encontradas? Qual dificuldade?

11. **Faça um pequeno relato sobre suas impressões a respeito do iVProg4All. Você pode destacar pontos de sucesso e melhorias, caso se apliquem.**

ANEXO B – QUESTIONÁRIO *SYSTEM USABILITY SCALE*

1. **Eu acho que gostaria de usar esse sistema com frequência**
 Discordo totalmente Discordo parcialmente Indiferente Concordo parcialmente Concordo totalmente
2. **Eu acho o sistema desnecessariamente complexo**
 Discordo totalmente Discordo parcialmente Indiferente Concordo parcialmente Concordo totalmente
3. **Eu achei o sistema fácil de usar**
 Discordo totalmente Discordo parcialmente Indiferente Concordo parcialmente Concordo totalmente
4. **Eu acho que precisaria de ajuda de uma pessoa com conhecimentos técnicos para usar o sistema**
 Discordo totalmente Discordo parcialmente Indiferente Concordo parcialmente Concordo totalmente
5. **Eu acho que as várias funções do sistema estão muito bem integradas**
 Discordo totalmente Discordo parcialmente Indiferente Concordo parcialmente Concordo totalmente
6. **Eu acho que o sistema apresenta muitas inconsistências**
 Discordo totalmente Discordo parcialmente Indiferente Concordo parcialmente Concordo totalmente
7. **Eu imagino que as pessoas aprenderão como usar esse sistema rapidamente**
 Discordo totalmente Discordo parcialmente Indiferente Concordo parcialmente Concordo totalmente

8. **Eu achei o sistema confuso de usar**

Discordo totalmente Discordo parcialmente Indiferente Concordo parcialmente Concordo totalmente

9. **Eu me senti confiante ao usar o sistema**

Discordo totalmente Discordo parcialmente Indiferente Concordo parcialmente Concordo totalmente

10. **Eu precisei aprender várias coisas novas antes de conseguir usar o sistema**

Discordo totalmente Discordo parcialmente Indiferente Concordo parcialmente Concordo totalmente

11. **Em uma escla de 0 a 10, qual é a probabilidade de você recomendar o iVProg4all a outras pessoas?**

0 () 1 () 2 () 3 () 4 () 5 () 6 () 7 () 8 () 9 () 10 ()

**ANEXO C – RESULTADOS PARA ANÁLISE
DE ACESSIBILIDADE DO
IVPROG COM
ACCESSMONITOR**



URI
https://usp.br/line/ivprog/


































Título
IVProg - LInE (free educational software and contents)

32 práticas encontradas

	A	AA	AAA
✓ Aceitáveis 19	10	9	0
↔ Para ver manualmente 5	2	0	3
✗ Não aceitáveis 8	6	1	1
	18	10	4

Avaliação

Prática encontrada	Nível	Ver detalhe
✗ ✓ Encontrei <u>6</u> imagens na página que não têm o necessário equivalente alternativo em texto.	A	☰Q
✗ ✓ Encontrei <u>2</u> links cujo conteúdo está vazio. Ou melhor, são compostos apenas por uma imagem e a imagem tem um equivalente textual alternativo vazio (i.e. <code>alt=""</code>).	A	☰Q
↔ ✓ Encontrei <u>2</u> grupos de links com o mesmo texto mas cujo destino é diferente.	AAA	☰Q
✗ ✓ Constatei que a primeira hiperligação da página <u>não permite saltar</u> diretamente para a área do conteúdo principal.	A	☰Q
↔ ✓ Encontrei <u>8</u> cabeçalhos na página.	AAA	☰Q

	✓	Encontrei 4 casos em que se viola a sequência hierárquica dos níveis de cabeçalho.	AAA	
	✓	Encontrei 29 casos em que se usa javascript para remover o foco do campo, sempre que o campo recebe o foco.	A	
	✓	Encontrei 1 elemento <code><iframe></code> sem <code>title</code> .	A	
	✓	Localizei 15 combinações de cor cuja relação de contraste é inferior ao rácio mínimo de contraste permitido pelas WCAG, ou seja 3 para 1 para texto com letra grande e 4,5 para 1 para texto com letra normal.	AA	
	✓	Perguntei ao validador de HTML do W3C e constatei que há 3 erros de HTML.	A	
	✓	Constatai que não há elementos obsoletos usados para controlo visual da apresentação.	A	
	✓	Verifiquei que o idioma principal da página está marcado como "en".	A	
	✓	Encontrei um título na página e ele parece-me correto.	A	
	✓	Constatai que todos os cabeçalhos desta página têm nome acessível	A	
	✓	Verifiquei que todos os atributos aria-* estão de acordo com a especificação ARIA.	A	
	✓	Verifiquei que todos os elementos <code><button></code> têm nome acessível.	A	
	✓	Não encontrei nenhum elemento com o atributo aria-hidden que tenha conteúdo focável	A	
	✓	Não encontrei elementos marcados como decorativos que tenham sido expostos a Tecnologias de Apoio	A	
	✓	Constatai que todos os elementos com um papel semântico que confere aos seus descendentes um papel decorativo, não têm descendentes focáveis	A	
	✓	Constatai que todos os elementos com papel semântico explícito têm os necessários estados e propriedades.	A	
	✓	Constatai que nesta página não há atributos id repetidos .	A	
	✓	Encontrei 2 ligações sem nome acessível.	A	

	✓	Constatei que o elemento meta não impede o utilizador de fazer zoom.	AA	
	✓	Verifiquei que todos os atributos role têm um valor válido	AA	
	✓	Localizei 15 combinações de cor cujas relações de contraste são inferiores ao rácio de contraste otimizado sugerido pelas WCAG, ou seja 4,5 para 1 para texto com letra grande e 7 para 1 para texto com letra normal.	AAA	
	✓	Constatei que o elemento com a semântica de contentinfo não está contido dentro de nenhum elemento com outra semântica	AA	
	✓	Constatei que o elemento com a semântica de main não está contido dentro de nenhum elemento com outra semântica	AA	
	✓	Encontrei um elemento com a semântica de banner .	AA	
	✓	Encontrei um elemento com a semântica de contentinfo .	AA	
	✓	Encontrei um elemento com a semântica de main .	AA	
	✓	Verifiquei que todos os elementos estão contidos dentro de uma lista.	AA	
	✓	Verifiquei que todas as listas só contêm itens de lista.	AA	

**ANEXO D – PARECER
CONSUBSTANCIADO DO CEP**

PARECER CONSUBSTANCIADO DO CEP

DADOS DA EMENDA

Título da Pesquisa: Uma abordagem de programação visual acessível para cegos

Pesquisador: MARCOS DEVANER DO NASCIMENTO

Área Temática:

Versão: 3

CAAE: 31894020.0.0000.5561

Instituição Proponente: UNIVERSIDADE DE SAO PAULO

Patrocinador Principal: Financiamento Próprio

DADOS DO PARECER

Número do Parecer: 5.993.592

Apresentação do Projeto:

Trata-se Projeto de Pesquisa referente ao Doutorado de Marcos Devaner do Nascimento em Engenharia da Computação Programa de Pós Graduação em Engenharia Elétrica da Escola Politécnica .

O projeto é apresentado:

O pensamento computacional está relacionado à formulação de um problema e aplicação de uma solução por meio de estruturas lógicas.

Essencialmente é a introdução da modelagem e solução de problemas através de algoritmos. Em um mundo onde cada vez a computação tem sido inserida no cotidiano, o desenvolvimento do pensamento computacional tem se tornado uma habilidade valiosa. O pensamento computacional tem sido adotado como matéria em diversos cursos, em diferentes níveis de escolaridade. Com isso, a abordagem de programação visual, que usa elementos gráficos para expressar comandos de programação, tem se mostrado eficiente para aprendizes de todos os níveis de escolaridade, apresentando impactos positivos em relação à aprendizagem e motivação dos alunos. Por outro lado, aprendizes cegos podem não ter esses mesmos benefícios, pois o termo “programação visual” pressupõe que a habilidade de enxergar é necessária para utilizar e se beneficiar dessa abordagem.

Por essa razão, esse trabalho tem como objetivo, por meio de testes de usabilidade e

Endereço: Av. Prof. Mello Moraes, 1721 - Bloco G - Sala 27

Bairro: Cidade Universitária

CEP: 05.508-030

UF: SP

Município: SAO PAULO

Telefone: (11)3091-4182

E-mail: cep.ip@usp.br

Continuação do Parecer: 5.993.592

acessibilidade com usuários cegos e não cegos, avaliar um sistema de programação visual existente e, assim, verificar se esse sistema atende a cegos e, não cegos, dando-lhes as mesmas oportunidades de aprendizado no que se refere ao pensamento computacional e à lógica de programação.

Objetivo da Pesquisa:

Partindo da Hipótese:

Aprendizes cegos podem adotar um sistema de programação visual, desde que esse sistema aplique as adaptações necessárias para atenuar as barreiras de acessibilidade e proporcione um estilo de interação focado em suas habilidades.

O autor descreve os objetivos:

Objetivo Primário:

Propor e avaliar sistema de programação visual acessível para cegos, sendo este, utilizado para o ensino e aprendizado do pensamento computacional e da lógica de programação.

E os Objetivos Secundários:

1. Propor um sistema de programação visual acessível para cegos;;
2. Avaliar com usuários cegos e não cegos a usabilidade e acessibilidade do sistema de programação visual proposto;
3. Propor um estilo de interação, em sistemas de programação visual, que atenda a usuários cegos e não cegos.

Avaliação dos Riscos e Benefícios:

São descritos:

Riscos:

Existe a possibilidade do participante não conseguir executar as atividades solicitadas utilizando o software e sentir-se constrangido por essa razão. Para minimizar esse risco, será oferecido suporte por meio do pesquisador, garantindo o auxílio para a execução das tarefas utilizando o

Endereço: Av. Prof. Mello Moraes, 1721 - Bloco G - Sala 27

Bairro: Cidade Universitária

CEP: 05.508-030

UF: SP

Município: SAO PAULO

Telefone: (11)3091-4182

E-mail: cep.ip@usp.br

Continuação do Parecer: 5.993.592

software. Também será explicado que a atividade com o uso do software não é obrigatória. Além disso, será esclarecido que as habilidades e conhecimentos técnicos não estarão sendo avaliados, e sim a interação dos participantes com o software. Há também a possibilidade de estresse durante a aplicação do estudo. Para minimizar o estresse e garantir o bem estar dos participantes, será preparado um ambiente agradável, acessível e uma condução amigável do estudo. O risco de constrangimento ao responderem as perguntas pessoais será minimizado garantindo que, no momento dessas coletas de dados, estarão presentes apenas o participante e o pesquisador.

Benefícios:

Entre os benefícios estão a oportunidade de participar de um estudo, relatando problemas e sugerindo adaptações para a criação de um sistema de programação visual que atenda a usuário cegos, tornando esse paradigma inclusivo. Como consequência, espera-se elevar as oportunidades de aprendizado e contribuir com o desenvolvimento profissional da população de cegos no Brasil.

Comentários e Considerações sobre a Pesquisa:

Metodologia Proposta:

Para esta pesquisa foi adotada a metodologia Design Science Research, onde os artefatos gerados podem ser qualquer coisa física ou abstrata, projetada para alcançar um objetivo (PIMENTEL et al., 2019). Peffers et al. (2007) propõem um modelo de processos para realização da pesquisa.

Espera-se que no processo de avaliação do DSR seja realizado um estudo de usabilidade e acessibilidade do iVProg4all. A avaliação será realizada com 7 participantes cegos e 7 que enxergam. Os participantes convidados devem possuir conhecimento em programação (experts), seja ele acadêmico ou profissional; todos devem ser maiores de 18 anos, convidados por meio de grupos, fóruns de programadores cegos (incluir quais) ou pelos meios de comunicação da escola politécnica da universidade de São Paulo. Para participar da pesquisa, todos devem aceitar o termo de consentimento livre esclarecido e responder a um questionário sócio-educativo.

Após o convite e aceite dos participantes, será enviado por e-mail o termo de consentimento livre esclarecido (TCLE) (Apêndice A) com todas as informações relacionadas ao experimento. O envio do termo e o aceite serão realizados por meio da ferramenta Google Forms. Após o aceite do TCLE, será enviado um e-mail para os participantes pedindo para que respondam ao questionário sócio-educativo, utilizado para levantamento e análise do perfil dos participantes (Apêndice B). Também será pedido que os participantes respondam ao e-mail, indicando o melhor dia e horário para realização da avaliação, a ser realizada de forma síncrona e remota. Vale ressaltar que, no e-mail enviado, não será disponibilizado qualquer link para que os participantes

Endereço: Av. Prof. Mello Moraes, 1721 - Bloco G - Sala 27

Bairro: Cidade Universitária

CEP: 05.508-030

UF: SP

Município: SAO PAULO

Telefone: (11)3091-4182

E-mail: cep.ip@usp.br

Continuação do Parecer: 5.993.592

possam ter acesso prévio ao iVProg4all. Isso acontecerá para que os participantes não tenham acesso prévio ao sistema a ser avaliado, impactando nos resultados da avaliação.

Para realização do experimento, serão realizadas sessões de observação para avaliação da usabilidade, acessibilidade e desempenho dos participantes (em relação ao uso do sistema e não do seu conhecimento). Além disso, serão aplicados os questionários de usabilidade (apêndice C) e acessibilidade (apêndice D) ao final de cada sessão.

Sessões de observação: as sessões de observação serão realizadas de forma remota, no dia e horário previamente agendados com os participantes. Para isso, será utilizado o Google Meets para chamada de vídeo com o participante durante a sessão. Será pedido que o participante compartilhe sua tela e a câmera enquanto realiza as tarefas. Do lado do pesquisador/observador, toda a sessão será gravada utilizando um software para gravação de tela. A gravação será utilizada para análise dos resultados.

Além do papel de observar a interação, o pesquisador/observador poderá tirar dúvidas e dar suporte ao participante durante todo o processo. Será explicado para os participantes que a atividade não é obrigatória, podendo parar no momento em que desejar. Além disso, será esclarecido que as habilidades e conhecimentos técnicos não serão avaliados, e sim a usabilidade e acessibilidade do sistema.

Pensamento em voz alta: durante as sessões será pedido que os usuários exponham verbalmente suas emoções e intenções. Durante esse processo, o participante não será pressionado a falar, o pensamento em voz alta deverá ser feito de forma voluntária e espontânea;

Questionários de usabilidade e acessibilidade: ao final de cada sessão, será pedido aos participantes cegos que respondam ao questionário de acessibilidade e usabilidade.

Os participantes não cegos deverão responder apenas ao questionário de usabilidade.

Os resultados gerados, não identificarão qualquer participante, garantindo sua privacidade. Espera-se que a aplicação dos métodos de pesquisa aplicados, permitam identificar o nível de equidade no uso de um sistema de programação visual por usuários cegos e não cegos. Também permitirá identificar problemas de usabilidade e acessibilidade.

Metodologia de Análise de Dados:

O software de programação visual adotado para pesquisa deverá atender aos critérios de acessibilidade propostos pelas diretrizes de acessibilidade para web, propostos na WCAG 2.1. Será garantido que toda a interação com software poderá ser realizada utilizando um leitor de tela.

Endereço: Av. Prof. Mello Moraes, 1721 - Bloco G - Sala 27

Bairro: Cidade Universitária

CEP: 05.508-030

UF: SP

Município: SAO PAULO

Telefone: (11)3091-4182

E-mail: cep.ip@usp.br

Continuação do Parecer: 5.993.592

Antes da aplicação do estudo, será realizado um piloto com 1 participante cego. Após a execução do piloto será realizada a pesquisa com até 7 participantes cegos e 7 não cegos. Serão registrados tempo de execução da atividade, número de auxílios e conclusão da atividade. Além disso, também serão gravados e analisados vídeos com os relatos dos participantes, durante a avaliação do sistema; também serão aplicados os questionários de perfil sócio-educativo, usabilidade e acessibilidade. Com isso, será possível fazer uma análise qualitativa e quantitativa da usabilidade e acessibilidade do sistema avaliado.

Considerações sobre os Termos de apresentação obrigatória:

O TCLE tem todas as informações necessárias , e os documentos estão adequadamente apresentados

Recomendações:

Não há .

Conclusões ou Pendências e Lista de Inadequações:

O projeto pode ser aprovado

Este parecer foi elaborado baseado nos documentos abaixo relacionados:

Tipo Documento	Arquivo	Postagem	Autor	Situação
Informações Básicas do Projeto	PB_INFORMAÇÕES_BÁSICAS_2099511_E1.pdf	15/03/2023 13:47:05		Aceito
Folha de Rosto	plataforma_15032023_133012.pdf	15/03/2023 13:42:54	MARCOS DEVANER DO NASCIMENTO	Aceito
TCLE / Termos de Assentimento / Justificativa de Ausência	APENDICE_A_TCLE.pdf	13/03/2023 17:21:38	MARCOS DEVANER DO NASCIMENTO	Aceito
Outros	APENDICE_D.pdf	13/03/2023 17:20:28	MARCOS DEVANER DO NASCIMENTO	Aceito
Outros	APENDICE_C.pdf	13/03/2023 17:19:55	MARCOS DEVANER DO NASCIMENTO	Aceito
Outros	APENDICE_B.pdf	13/03/2023 17:19:29	MARCOS DEVANER DO NASCIMENTO	Aceito
Declaração de Pesquisadores	DECLARACAO_DE_COMPROMISSO.pdf	13/03/2023 17:11:37	MARCOS DEVANER DO NASCIMENTO	Aceito
Projeto Detalhado / Brochura Investigador	Projeto_de_pesquisa_detalhado.pdf	13/03/2023 16:53:19	MARCOS DEVANER DO NASCIMENTO	Aceito

Endereço: Av. Prof. Mello Moraes, 1721 - Bloco G - Sala 27

Bairro: Cidade Universitária

CEP: 05.508-030

UF: SP

Município: SAO PAULO

Telefone: (11)3091-4182

E-mail: cep.ip@usp.br

INSTITUTO DE PSICOLOGIA
DA UNIVERSIDADE DE SÃO
PAULO - USP



Continuação do Parecer: 5.993.592

Situação do Parecer:

Aprovado

Necessita Apreciação da CONEP:

Não

SAO PAULO, 10 de Abril de 2023

Assinado por:

**Leila Salomão de La Plata Cury Tardivo
(Coordenador(a))**

Endereço: Av. Prof. Mello Moraes, 1721 - Bloco G - Sala 27

Bairro: Cidade Universitária

CEP: 05.508-030

UF: SP

Município: SAO PAULO

Telefone: (11)3091-4182

E-mail: cep.ip@usp.br

ANEXO E – RELATÓRIO DE TESTE - MÓDULO 1



Protocolo de testes - Módulo 1

Tiago Melo Casal <tiagocasal@yahoo.com.br>
Para: marcos.devaner@usp.br

8 de fevereiro de 2023 às 12:54

Olá Devaner,

Como vai?

Interessante o primeiro módulo!

Parabéns pela ideia! A Kaiara está de parabéns pelo bom trabalho de desenvolvimento!

No vídeo que enviei o link pelo WhatsApp, do 1º teste realizado em Fevereiro/2023 usando Windows 11 com Firefox ESR 102.7 e NVDA 2022.4, seguem alguns pontos para apreciação:

- Em "Comandos", entrando em qualquer das 3 opções, de nova variável, ou de atribuição/Operações ou a opção escreva, fiquei com uma dúvida sobre o sisteminha, se o que eu fosse alterando nas caixas e digitando nos campos já estariam "valendo", se já estavam sendo digitados no código e eu poderia partir para outra opção quando quisesse, ou se para concluir precisava de clicar na opção "Visualizar código";

- Criando uma variável, ou usando opção de atribuição/Operações ou a opção escreva, ao clicar em Visualizar código, o código exibido mostra um caractere de atribuição pronunciado de forma incompleta pelo leitor de tela, formado pelo caractere unicode 2039 seguido de um tracinho "-", sugestão é trocar o caractere 2039 por sinal de menor "<", o sinal de atribuição ficaria "<", o ruim do conjunto com o sinal de menor seria escutá-lo no meio de uma linha que tenha um menor usado para comparação de duas variáveis (muito sinal de menor sendo falado pelo leitor de tela), outra alternativa é trocar este conjunto de dois caracteres pelo caractere de seta pra esquerda, Unicode 2190, outra opção ainda seria substituir isso pelo sinal de igual "=", tem linguagens de programação como Python que usam o sinal de igualdade como sinal de atribuição e por um lado acho uma boa opção (a leitura fica como uma equação do primeiro grau, quem é $x = \text{variável1} + \text{variável2}$), o ruim do igual para atribuição é quando vai se fazer comparação e em vez de apenas um sinal de igual se usa dois em sequência, talvez seta para esquerda seja uma boa para atribuição mesmo, não sei, fica aí as 3 sugestões;

Link para informação sobre o Caractere unicode 2039: <http://unicode-table.com/pt/2039/>

Link para informação sobre o Caractere unicode 2190: <http://unicode-table.com/pt/2190/>

- Pra qualquer variável ou comando já colocadas no código, ao clicar na opção "Editar código" ajudaria se o foco do leitor de tela fosse puxado para o primeiro item a ser editado da quela linha, normalmente é uma caixa;
- Entrar em "Comandos" -> "Atribuição e Operações", a primeira caixa combinada em que se escolhe a variável a receber o resultado da operação, quando nada foi escolhido nela fica "Selecione a variável", especificamente para essa caixa em questão poderia ter um texto que indicasse melhor o propósito da escolha, algo do tipo "Selecione variável que receberá resultado da operação" ou algum texto melhor;
- Lá em "Comandos" -> "Atribuição e Operações", tive um pouco de dificuldade e só percebi o comportamento da falha estudando de forma mais calma após a gravação, estando com o foco na primeira caixa combinada em que se escolhe a variável a receber o resultado da operação, ao mexer uma vez com alguma seta, o sisteminha escolhe o primeiro item e pula o foco do leitor de tela imediatamente para a segunda caixa, isso confunde o usuário, não deveria pular para a outra caixa, tendo vários itens na caixa o usuário pode querer navegar com setas pelos itens e escolher qualquer um deles, após navegar e escolher é que deseja ir para a próxima caixa teclando tab, mesma falha ocorre em outras caixas nessa área, as caixas em que a pessoa escolhe se quer operar com "Variável" ou com "Valor", para resolver pode-se verificar a área do comando Escreva a caixa que tem por lá para escolher entre Variável e Texto não está com esse problema;
- Tendo entrado em "Comandos" -> "Atribuição e Operações", após escolher na primeira caixa combinada alguma variável, ao navegar com tab e/ou shift+tab o foco do leitor de tela está pulando o texto "recebe", é importante que o leitor de tela focalize tal texto para que ele pronuncie e o usuário perceba que a caixa anterior recebe o resultado e as caixas posteriores são da operação, tal falha não acontece na opção de "Nova Variável", lá o texto "recebe" está sendo focalizado;
- Ainda em "Comandos" -> "Atribuição e Operações", o botão nomeado com o sinal de mais "+", não é intuitivo, poderia ter um nome melhor, só não sei qual, como uma das operações é com o sinal "+" não acho legal o nome do botão assim, fica ambíguo, talvez mudar o nome dele para "+ operação/variável", "Mais operação e variável", ou algum nome melhor;
- Pra finalizar os comentários sobre os recursos de "Comandos" -> "Atribuição e Operações", tendo escolhido uma variável na primeira caixa e tendo clicado no botão "+", a penúltima caixa combinada é para se escolher o tipo de operação, inicialmente pensei que seria interessante trocar o tracinho (hífen) pelo sinal de subtração (unicode 2212), o sinal de asterisco pelo de multiplicação (unicode 00D7) e o barra pelo de divisão (unicode 00F7), para que o leitor de tela pronuncie melhor esses itens em tal caixa, é claro que no código deve continuar os operadores corretos "- * /", mas verifiquei que nem todos leitores de tela pronunciam o caractere de subtração e outros sugeridos, então deixa do jeito que está mesmo, ou outra opção seria trocar na caixa os caracteres "+ - * /" por nomes como "Adição subtração multiplicação divisão", "Operador de adição, operador de subtração, operador de multiplicação, operador de divisão", etc, é algo a ser estudado se troca ou deixa como está mesmo;

- Na opção "Comandos" -> "Escreva", escolhendo "Texto" na caixa, digitando algo no campo de edição, ao clicar em "Visualizar código" aparece o comando escreve seguido do caractere de atribuição e por fim o texto digitado com o ponto e vírgula encerrando, tal texto digitado para diferenciar de um comando Escreve com variável. Deveria estar entre aspas genéricas ("), no comando Escreve pegando duma variável é que permaneceria sem aspas;
- Também indo na opção "Comandos" -> "Escreva" e ter realizado a adição do comando no código, após clicar em "Visualizar código" o código mostrado tem um caractere de atribuição entre o comando Escreve e o que ele deve imprimir na tela, tal caractere ao que parece não se aplica nessa situação já que Escreve não é variável;
- Após usar a opção "Comandos" -> "Escreva" e realizar o procedimento pra adicionar tal comando no código, a opção específica para removê-lo deveria ser trocada o nome de "Excluir variável" pra "Remover o comando" (por não se tratar de variável, nome "Remover comando" para não ficar parecido com o comando geral "Excluir comandos");
- Durante o teste gravado em vídeo, quando cliquei em "Executar" não funcionou e pensei que não estava acessível a saída no terminal, após a gravação fiz testes e constatei que a falha está no sistema não prevenir uma divisão por zero como coloquei pra teste, é preciso colocar para o sistema verificar quando houver divisão se o divisor é "0", ou se for usado um divisor que recebe resultado de outro cálculo checar se no divisor será atribuído 0, pois divisão com zero não existe e trava os programas, nesse caso é preciso escolher alguma estratégia para o sisteminha, se ele vai emitir no terminal uma mensagem de erro "Código com falha: divisão por zero não é permitida" (ou alguma mensagem melhor), se além da mensagem o que será feito, ou parar a execução dos próximos comandos, ou continuar a execução atribuindo como resultado para aquela operação inválida o valor 0...

Após a gravação do vídeo fiz outros testes com navegadores, leitores de tela e sistemas diferentes: navegador Ms Edge 109 com leitor de tela Narrador no Windows 11; navegador Firefox ESR 102.7 em Linux Debian 11 com leitor de tela Orca 43; navegador Google Chrome 109 em Android 13 com leitor de tela Talkback 13; navegador Safari em iOS 16.3 com leitor de tela Voiceover.

Antes de prosseguir, quero dizer que me surpreendeu positivamente utilizar a ferramenta via navegador em dispositivos móveis (celulares), com um assistente desses fica fácil aprender em cursos de introdução à lógica de programação usando um celular por facilitar a interação do usuário, o aluno só vai digitar algo quando realmente necessário, em computador de mesa/notebook digitar não é problema, já em dispositivos móveis como celulares digitar é cansativo e desmotivador.

Em anexo estão capturas de tela (prints) do seguinte ponto observado nos testes após a gravação:

- Tanto com leitor de tela Narrador (em Windows), como com Orca (no Linux), Talkback (em Android) e Voiceover (no iOS), seja na área de criação de variável, seja usando opção de atribuição/Operações ou a opção escreva, os quatro leitores de tela mencionados não identificam o nome dos botões "Visualizar código" e "Editar código", diz apenas "Botão" sem falar o nome deles (os outros botões estão certinhos para eles), o único leitor de tela que está conseguindo identificar tais nomes é o NVDA, analisei o código HTML rapidamente e não consegui perceber a causa.

Abraços,

Tiago

-----Mensagem original-----

De: Marcos Devaner do Nascimento <marcos.devaner@usp.br>

Enviada em: Quinta, 2 de Fevereiro de 2023 14:22

Para: tiagocasal@yahoo.com.br

Assunto: Protocolo de testes - Módulo 1

Oi, Tiago! Tudo bem?

Primeiro, muito obrigado por colaborar com minha pesquisa. Sua participação é muito importante para a conclusão do meu doutorado e te agradeço demais! Seguem as instruções para os testes.

1.

Sobre a ferramenta a ser testada

O iVProg4all trata-se de uma versão simplificada do iVProg, sendo esta, uma ferramenta de programação visual com objetivo de facilitar o aprendizado dos conceitos básicos de programação. O iVProg4all possui recursos de acessibilidade para cegos, além de aplicar as diretrizes propostas pela WCAG 2.1. Em sua primeira versão, será possível criar variáveis, atribuir valores, realizar operações aritméticas, criar estruturas se..então, além de criar estruturas de repetição para...então e escreva.

1.

Instruções para gerais o teste

**ANEXO F – RESULTADOS PARA ANÁLISE
DE ACESSIBILIDADE DO
IVPROG4ALL COM
ACESSMONITOR**



URI
https://mdevaner2017.github.io/#comands

Título
lvprog

85
Elementos (x)HTML

11 KB
Tamanho da página

25 práticas encontradas

	A	AA	AAA
Aceitáveis 19	12	7	0
Para ver manualmente 5	3	0	2
Não aceitáveis 1	0	1	0
	15	8	2

Avaliação

Prática encontrada	Nível	Ver detalhe
Constatei que o primeiro link da página nos <u>permite saltar</u> para o conteúdo principal.	A	
Encontrei <u>4</u> links para contornar blocos de conteúdo.	A	
Encontrei <u>2</u> cabeçalhos na página.	AAA	
Perguntei ao validador de HTML do W3C e constatei que <u>não existem erros</u> de HTML.	A	
Constatei que <u>não há</u> elementos obsoletos usados para controlo visual da apresentação.	A	
Verifiquei que o idioma principal da página <u>está marcado</u> como " <u>pt-BR</u> ".	A	
Encontrei <u>um título</u> na página e ele parece-me correto.	A	
Constatei que todos os cabeçalhos desta página <u>têm</u> nome acessível	A	
Verifiquei que <u>todos</u> os estados e todas as propriedades ARIA têm um tipo de valor válido.	A	
Verifiquei que <u>todos</u> os estados e todas as propriedades ARIA são permitidos.	A	
Verifiquei que <u>todos</u> os atributos aria-* estão de acordo com a especificação ARIA.	A	
Verifiquei que <u>todos</u> os elementos <u><button></u> têm nome acessível.	A	
Não encontrei <u>nenhum elemento</u> marcado com aria-hidden que tenha conteúdo focável	A	
Constatei que todos os elementos com um papel semântico que confere aos seus descendentes um papel decorativo, não têm descendentes focáveis	A	

✓	✓	Constatei que nesta página <u>não há atributos id repetidos</u> .	A	🔍
✓	✓	Verifiquei que <u>todas</u> as ligações têm nome acessível.	A	🔍
✓	✓	Constatei que o elemento <u>meta não impede</u> o utilizador de fazer zoom.	AA	🔍
⊖	✓	Localizei <u>6</u> combinações de cor cujas relações de contraste são inferiores ao rácio de contraste otimizado sugerido pelas WCAG, ou seja 4,5 para 1 para texto com letra grande e 7 para 1 para texto com letra normal.	AAA	🔍
✗	✓	Constatei que o elemento com a semântica de <u>banner está</u> contido dentro de um elemento com outra semântica	AA	🔍
✓	✓	Constatei que o elemento com a semântica de <u>contentinfo não está</u> contido dentro de nenhum elemento com outra semântica	AA	🔍
✓	✓	Encontrei <u>um</u> elemento com a semântica de <u>banner</u> .	AA	🔍
✓	✓	Encontrei <u>um</u> elemento com a semântica de <u>contentinfo</u> .	AA	🔍
✓	✓	Encontrei <u>um</u> elemento com a semântica de <u>main</u> .	AA	🔍
✓	✓	Verifiquei que <u>todos</u> os elementos <u></u> estão contidos dentro de uma lista.	AA	🔍
✓	✓	Verifiquei que <u>todas</u> as listas só contêm itens de lista.	AA	🔍

[Acessibilidade](#)
[Termos e Condições](#)
[Política de privacidade](#)
[Github](#)
[Glossário](#)
[Opções de visualização](#)








© 2021 AMA - Agência para a Modernização Administrativa, I.P. Todos os Direitos Reservados.

