Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Towards Enabling Inclusive Conversations: Bridging Accessibility Gaps for the Visually Impaired in a Chatbot Web Chat

João Antônio Desidério de Moraes

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Orientadora
Prof.a Dr.a Edna Dias Canedo

Coorientadora
Prof.a MSc Geovana Ramos Sousa Silva

Brasília
2023

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Towards Enabling Inclusive Conversations: Bridging Accessibility Gaps for the Visually Impaired in a Chatbot Web Chat

João Antônio Desidério de Moraes

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Prof.a Dr.a Edna Dias Canedo (Orientadora)
CIC/UnB

Prof.a Dr.a Fernanda Lima
CIC/UnB

Prof. Dr. Daniel de Paula Porto
CIC/UnB

Prof. Dr. Marcelo Mandelli
Coordenador do Bacharelado em Ciência da Computação

Brasília, 11 de dezembro de 2023

Towards Enabling Inclusive Conversations: Bridging Accessibility Gaps for the Visually Impaired in a Chatbot Web Chat

João Antônio Desidério de Moraes
joao.moraes@aluno.unb.br
University of Brasília (UnB), Department of Computer Science
Brasília, DF, Brazil

Geovana Ramos Sousa Silva
geovannna.1998@gmail.com
University of Brasília (UnB), Department of Computer Science
Brasília, DF, Brazil

Edna Dias Canedo
ednacanedo@unb.br
University of Brasília (UnB), Department of Computer Science
Brasília, DF, Brazil

ABSTRACT

Context: Web chats, pervasive in both corporate and social settings, underscore the critical need for inclusivity in web-based communication platforms. Visually impaired individuals face obstacles in engaging in digital conversations, limiting their participation in the digital sphere. **Problem:** Developing web chats that are fully compatible with screen readers poses a particular challenge owing to the dynamic nature of their element appearance. This paper focuses on addressing specific screen-reader compatibility issues within an open-source web chat designed for chatbot interactions. **Solution:** The proposed solution involves adapting the open-source web chat for screen reader compatibility. Code modifications aim to improve user experience, emphasizing inclusivity for individuals with visual impairments. **Theory of IS:** This research aligns with Design Theory and Equity Theory, incorporating accessibility considerations to ensure equity in using the web chat application. **Method:** The research utilizes a combination of Proof of Concept and Experimentation. It begins with examining the existing web chat, followed by code modifications - including integrating semantic HTML tags, ARIA landmarks, and other accessibility-focused techniques - with validation steps comprising execution and analysis of two use case scenarios. **Summarization of Results:** The study validates the adapted web chat, comparing its usability before and after accessibility-focused code modifications. Screen reader transcriptions reveal significant improvements in user experience, including enhanced navigation and overall announcement of chat updates. **Impact on the IS field:** This research contributes to Information Systems by addressing web chat accessibility gaps. The adapted open-source web chat exemplifies inclusive design, highlighting the impact of minor code modifications on usability for users with visual impairments.

CCS CONCEPTS

• **Human-centered computing** → **Accessibility design and evaluation methods**; **Empirical studies in interaction design**; • **Computing methodologies** → **Natural language generation**.

SBSI 2024, May 20 to 23, 2024, Juiz de Fora, Minas Gerais, Brazil

ACM ISBN …$15.00
https://doi.org/

KEYWORDS

Chatbot, Accessibility, Inclusive Software, Webchat

ACM Reference Format:
João Antônio Desidério de Moraes, Geovana Ramos Sousa Silva, and Edna Dias Canedo. 2024. Towards Enabling Inclusive Conversations: Bridging Accessibility Gaps for the Visually Impaired in a Chatbot Web Chat. In *XX Brazilian Symposium on Information Systems (SBSI 2024), May 20 to 23, 2024,Juiz de Fora, Minas Gerais, Brazil.* ACM, New York, NY, USA, 10 pages. https://doi.org/

1 INTRODUCTION

The growth of the user base in computing systems has brought to light accessibility issues - that is, concerns regarding the possibility of their full utilization by all interested parties, capable of meeting the needs of individuals with specific characteristics [7]. A more specialized audience comprises individuals with visual impairments whose computer usage is complemented by other assistive technologies, among which screen readers are highly popular due to their advantageous cost-effectiveness [13]. In essence, the functioning of a screen reader involves audibly articulating the elements on the screen through voice synthesizers, following the order of arrangement in the underlying code, providing an auditory alternative to visual data output [10].

The pursuit of greater inclusivity on the Internet led to the creation of the Web Accessibility Initiative (WAI) by the World Wide Web Consortium (W3C), aiming to develop strategies, guidelines, and resources to make the Web more accessible to individuals with disabilities [21]. This effort resulted in the Web Content Accessibility Guidelines (WCAG), now considered the accessibility standard by both technical and legal stakeholders [21]. As defined by WAI, web accessibility means that websites, tools, and technologies are designed and developed in a way that individuals with disabilities can use them [1]. The influence of WCAG is such that it inspired the creation of the Electronic Government Accessibility Model by the Brazilian government, with the 'commitment to guide the development and adaptation of federal government digital content, ensuring access for all' in Brazil[8].

Another outcome of the increasing popularity and development of information and communication technologies (ICTs) is the evolution of interaction formats between humans and computers. For this work, the focus is on computational agents known as 'chatbots' - derived from the English words "chat" (conversation) and "bot", a contraction of "robot". This type of agent is defined as a computer

[1] As defined on the WAI website (https://www.w3.org/WAI/fundamentals/accessibility-intro)

program capable of simulating a conversation with a human being [1]. This interaction technology has attracted investors from various sectors - customer service, education, healthcare, and personal assistance - due to the promise of handling labor-intensive tasks at a low cost [15].

Initially implemented solely by pioneering technology and Artificial Intelligence companies, chatbots are now accessible in the market through frameworks and platforms designed to facilitate their development [23]. Rasa is one such platform, providing foundational software for developing the internal components - or backend - of conversational agents within the Python ecosystem. According to the official documentation, Rasa's components do not include a presentation layer, leaving it to the developer to design the interaction with the chatbot [2].

To assist in fulfilling this task, third-party software solutions are available. One of them is a Webchat widget published by Botfront, designed to add a live chat window on web pages for user interaction with the chatbot[3]. The window includes visual elements enabling users to identify the chatbot, compose and send messages, view current and past messages, and be informed when the chatbot is formulating a response.

The repository page of the webchat does not mention any accessibility features, which is crucial considering the empowering nature of internet use in social processes for accessing information — a realm where individuals with visual impairments aim to feel included [7].

Thus, this work assessed the WebChat software and implemented the necessary changes to make it accessible to screen reader users according to the standards and technical recommendations outlined in the accessibility literature, with a particular focus on WCAG guidelines. Consequently, this work does not suggest changes to the conversational mechanism on the back-end of the virtual agent itself.

Several steps are necessary to fulfill the overarching goal, such as

- identifying the standards and guidelines for the development of accessible chatbots and instant messaging applications through a review of scientific and grey literature;
- searching for violations of these standards and guidelines within Webchat by inspecting the source code and testing its usability;
- proposal and implementation of the necessary changes to comply with the standards, and finally,
- evaluate the interface with the usage of a screen reader to verify compliance with the guidelines through usability testing and discussing the results.

Section 2 establishes the theoretical background upon which the work was built, including important definitions and related works. Section 3 explains the Design Science methodology, laying the details of problem identification, solution goals, design and development and validation steps. Section 4 discusses the results stemming from implementation and testing. Section 5 exposes some threats to validity, while section 6 concludes the article.

[2]Per the official documentation found on https://rasa.com/docs/rasa/

[3]https://github.com/botfront/rasa-webchat

2 BACKGROUND

This work is based on the social model of disability that emerged in the 1970s in response to moral and medical models of disability. According to psychology professor Rhoda Olkin, the moral perspective on disability sees it as strongly associated with the person's morality. In other words, an individual's physical particularity is viewed as part of their character and purpose in life, its existence justified as a god-given challenge to bear, for example. Under this model, disability is tied to stigma, whether seen positively - as a symbol of resilience, strength, and courage for enduring life despite adversity - or negatively - as a result of sins or moral lapse, bringing shame to the person and its family [18].

The medical model, Olkin explains, focuses on clinical descriptions and regards disability as a pathology, a defective exception to the norms of the human body. Its reliance on medical interventions frames disability as a disease that makes people incapable of participating in society and should be treated, subjecting the individual to cures and ameliorations of the disability to the greatest extent possible, revealing the paternalistic and benevolent undertones of the model. Both moral and medical models share an understanding that disability ultimately resides in the individual.[18]

In contrast to these models, the social model emphasizes the collective aspects of disability, highlighting society's role in excluding individuals with disabilities while identifying this group as a target of systemic oppression. A key argument of this model points to the general population's disregard for certain types of physical characteristics, which leads to the current mode of social activities being intentionally formulated to cover a specific set of needs skills, isolating and excluding people who do not fit that criteria. The social perspective states that disability does not reside within the physical characteristic itself, but rather in the exclusion motivated by it.[22] Therefore, creating an environment where people with different physical and mental features can participate in society is a social task that will demand collective effort to create inclusive ways of interacting.

It is possible to establish a connection between the social model of disability and recent efforts in the field of digital accessibility. As will be discussed further, digital accessibility aims to empower all types of users of digital tools and systems. In an ideally accessible world, physical and psychological needs will have been deliberately considered in the development of civilization, breaking down barriers for the participation of people with disabilities in all social spheres. Therefore, to achieve that result when developing digital technology, stakeholders must accommodate the needs of different groups of people with specific skill sets.

Accessibility is the characteristic of being easily reached, obtained, traversed, or executed without significant barriers. In Brazilian law [4], it is defined as:

> the possibility and condition of reaching for use, safely and autonomously, spaces, furniture, urban equipment, buildings, transportation, and communication systems and means.

When the design of objects disregards the needs of more specific groups, it risks reinforcing barriers to their use. Therefore, the term

[4]Law No. 10,098 of December 19, 2000

also carries an inclusive and civic sense, concerning the possibility for all individuals to enjoy life in society regardless of physical, cognitive, and socio-cultural conditions[17].

This citizen perspective encompasses access to communication means, including digital information systems. The adoption of these systems, driven by the popularization of the internet, enabled stronger connections between individuals and entities worldwide, significantly impacting social organization[7]. The transformative potential of digital tools was recognized by people with special needs, who use digital means to help with their tasks and began to demand appropriate and understandable access to information in this medium[10].

Thus, digital accessibility is the promotion of access to ICTs for the widest possible audience, including people with disabilities. To implement it, aligning software and hardware characteristics with the expected interaction forms for each user group focusing on product usability is necessary. Common features for accessible applications may include a smooth, quick, and easily memorable learning curve, discouraging operational errors, and ensuring efficiency in accomplishing the application's goals, while specific features may vary.[25]

Attention to digital accessibility is a highly relevant topic for Brazilian society. Evidence of this is the governmental effort, dating back to 2004, to regulate accessibility for portals and electronic sites of public entities or those financed by the government. To standardize the creation of inclusive websites, the Brazilian Accessibility Model was formulated, inspired by international guidelines and experiences from other countries.[10]

As previously mentioned, this work focuses on studying an interface for interaction between a human user and a type of virtual agent known as a "chatbot." In an overview article on this technology, Adamopoulou and Moussiades[1] defines this agent as a computer program designed to simulate conversations with humans, tracing the origins of this type of system back to Alan Turing and his famous formulation known as the "Turing Test" regarding the capability of machines to think. The year 1966 witnessed the emergence of the first known chatbot, ELIZA, and the continued development of this interaction format has led to versions embedded in technology company products, such as Siri and Alexa, conversational agents released by Apple and Amazon.

According to the same author, the primary technique behind chatbots is pattern matching - a representation in blocks that groups triggers and expected responses. During the interaction, each user input serves as a trigger to which the chatbot should respond. Responses generated by this technique tend to be straightforward and predictable, a problem alleviated by the increasing use of artificial intelligence, both in understanding natural language triggers and in producing more nuanced responses.

2.1 Related Works

Torres et al. [24] researched the existing literature from 2007 up until 2017 for the state of the art in the intersection of chatbots, conversational interfaces, and the accessibility to visually impaired users. The study highlighted the civic role of designers in developing interfaces that take into account users' difficulties and their specific usage needs. Through a rapid literature review, 95 articles were selected for analysis, of which 25 adhered to the study's requirements. Among these, only 4 exclusively addressed accessibility or assistance for people with disabilities. None of the articles exclusively referred to accessibility in chatbots, which the authors attributed to the incipient studies in this area.

More recently, Lister et al. [14] have outlined considerations for accessibility in projects concerning conversational user interfaces, including chatbots. The study is motivated by the initially demonstrated potential for universal access support by these interfaces due to their inherent adaptability to multiple communication modalities, which can be applied to facilitate various tasks for individuals with disabilities. The work aimed to guide designers in inclusively developing such interfaces, taking into account various types of disabilities, and forming an initial framework comprising nine types of disabilities. Addressing visual impairment, recommendations include leveraging voice interaction capabilities, compatibility with screen readers, and the ability to modify the size and colors of on-screen elements. One crucial aspect is the issue of hybrid interfaces, composed of multiple components - attention must be paid to the accessibility of each individual component and how they interrelate.

Melnyk [16] addressed the creation of an accessible instant messenger interface for blind users, focusing on identifying interface sections for effective user perception and handling dynamic behaviors within the interface. Among these, it involved detecting the appearance of the messenger as a widget and identifying various types of content messages may display. The study was validated through tests with blind users who were required to perform conversational scenarios using two interfaces: one enriched with the proposed functionalities and another without these functions. The evaluation revealed a preference among users for the enriched interface.

Calvo et al. [6] introduced a messenger prototype aimed at reducing interface obstacles for screen reader users. The study adopts a user-centered perspective and relies on the WAI-ARIA standard for accessibility in rich applications. Some highlighted key features are the messenger's responsiveness to various window sizes and utilization of HTML5 and CSS3 standards. The main novelty presented was the addition of two functionalities identified in a list of accessibility requirements: allowing users to pause or resume conversation updates, granting control over the pace of incoming interactions if feeling overwhelmed, and the ability to attach files, providing a description and file size beforehand for the recipient to decide whether they wish to receive the file. An evaluation of the implementation's effectiveness was ongoing at the time of publication.

Regarding assistive technology, a study in 2010 investigated navigation strategies using screen readers [3]. As a starting point, the article provides an overview of navigating digital content using a keyboard in conjunction with a screen reader. Information processing speed is highlighted as the main barrier, and its overcoming inspires the emergence of many navigation strategies, such as increasing the reader's speed, inferring control functions, and modulating behavior depending on the task at hand. Dealing with dynamic content is also challenging so it is often overlooked by

users unless strictly necessary. The study found the most widespread strategy for finding relevant content is navigating by headings, with 52% of users stating they do so whenever the content allows. Sequentially navigating through all elements is the least efficient strategy, which users resort to when others fail.

Regarding WCAG 1.0 and 2.0 standards, a 2011 article reports the authors' effort to validate the specifications with users with disabilities [20]. The study was motivated by the lack of experiments to validate widely used specifications. A controlled experiment was conducted involving people with disabilities and a control group, where participants had to complete a series of tasks on two websites and report their experiences. The results identified that 27% of the accessibility issues detected were covered by WCAG 1.0, while version 2.0 would cover 32% of the problems. Analyzing the outcomes, the authors concluded that mere compliance with WCAG specifications is not sufficient to ensure content accessibility. The ideal approach to accessibility should involve using WCAG in conjunction with a user-centered development process, engaging people with disabilities with a wide spectrum of abilities to participate in many steps of development.

3 METHODOLOGY

The adopted methodology is inspired by Design Science Research, defined by Wieringa [26] as a design project aimed at creating or improving an artifact able to align stakeholders with their objectives, along with investigating the performance of this artifact within the context of the problem. According to the author, the meaning of an artifact should be broadly interpreted, encompassing but not limited to components, systems, processes, algorithms, and methods. Wieringa [26] emphasizes the insignificance of the isolated artifact, highlighting the interaction between the artifact and its users in the context of the problem as the true origin of the solution. In this regard, using the same artifact in different contexts might generate distinct effects depending on how these elements interact [26]. This notion adheres to the context of this study since it consists of an evaluation of how an artifact performs when used by different types of users.

Design Science has at its core two interconnected types of inquiry: design problems and knowledge issues. Design problems are well situated within the stakeholders' context, proposing a concrete alteration in the world in the form of a designed artifact aimed at the existing or hypothetical objectives of the stakeholders. Knowledge issues involve an investigation of the world as it presents itself; researchers' findings will be evaluated based on rational truth and may or may not align with the stakeholders' objectives. One type of inquiry often derives from the other: in one sense, we can formulate knowledge issues about the encountered design problems, while conversely, it is possible to envision designs in response to formulated knowledge issues [26].

In their study on Design Science for information systems, Peffers et al. [19] specified six necessary stages in the process: problem identification and motivation, inference of solution objectives, design and development, demonstration, evaluation, and communication [19]. This work will explicitly follow these stages, except for communication, which is implicit in the writing and publication of the paper.

3.1 Problem identification and motivation

The driving knowledge issue for this work is: "Are the capabilities implemented in the Webchat software sufficient for screen reader user accessibility?" Within the context of this issue, the envisaged design problem is to implement accessibility capabilities for screen reader users in the Webchat interface for Rasa conversational agents.

The motivation to investigate Webchat accessibility is rooted in the democratic conception of society, which advocates recognizing differences in creating inclusion policies aimed at expanding equal participation in social processes. Access to information through digital means as a tool for integration among various public entities is part of this set of processes [7].

In the specific context of chatbots, their use is noticeable among service-providing companies as a form of customer service [11]. There are studies aimed at enhancing their usage in various domains, including education [12] and medical assistance [9]. Users of these agents cite advantages such as ease of use, speed, and convenience, saving them from waiting for the availability of a person to assist them [4]. As the prospect of increasingly deployed chatbots emerges, there is an interest in accommodating the needs of the broadest audience possible, as advocated by accessibility guidelines. This aims to enable people with disabilities to benefit from this new communication channel.

3.2 Solution goals

The identification of accessibility capabilities in Webchat is based on WCAG 2.0 guidelines, supplemented by content from scientific and gray literature as needed. The choice of these guidelines as guiding principles was motivated by their status as an established reference for web accessibility, serving as both a practical tool and a set of academic principles [20].

The WCAG specifies four principles that are necessary for achieving web accessibility. For content to be considered accessible, it must be perceivable, operable, understandable, and robust. Each principle encompasses a list of guidelines and their respective success criteria organized into conformance levels: A (lowest), AA, or AAA (highest) [5].

Below are listed the guidelines applicable to the content displayed by Webchat. The solution aims to meet the success criteria of each guideline at least at level A. Discarded guidelines do not apply or address issues outside the scope of this work.

1.1) Text alternatives: "Provide text alternatives for any non-text content so that it can be changed into other forms people need, such as large print, braille, speech, symbols or simpler language." The level A of success criterion for this guideline states that all non-text content that is presented to the user has a text alternative that serves the equivalent purpose, except in specific cases of elements that are controls, input, time-based media, tests, sensory, CAPTCHA, used solely for decoration or formatting or invisible in the page.

1.3) Adaptable: "Create content that can be presented in different ways (for example simpler layout) without losing information or structure." This guideline has three success criteria on its level A. The content's information and relationships established through presentation must be programmatically determined or available in

text. The correct reading sequence for the content must also be programmatically determined. And at last, the sensory characteristics such as but not limited to shape, color, and size must not be the sole source of instruction on how to understand and operate the content.

2.1) Keyboard accessible: "Make all functionality available from a keyboard." The two success criteria for this guideline are as follows: all functionality is available through keyboard interface independent of specific timings for keystrokes - the exception being underlying functions that require path-dependent input - and the content must be free of keyboard traps, meaning focus can be moved to and away from a component using keyboard through standard exit methods.

2.4) Navigable: "Provide ways to help users navigate, find content, and determine where they are." Four success criteria are required to fulfill level A of this guideline. The content must contain a mechanism for bypassing blocks, as well as descriptive page titles, correct focus order for its components when the navigation sequence is relevant, and context for links in the link text alone or its programmatically determined context.

3.1) Readable: "Make text content readable and understandable." The level A success criterion for this guideline states that "the default human language of each Web page can be programmatically determined" in a way that different assistive technologies can understand and use this information.

3.2) Predictable: "Make Web pages appear and operate in predictable ways." The success criteria for this guideline are related to changes of context - meaning major changes in content with the potential to disorient users who cannot perceive the content all at once. To fulfill this guideline, changes of context cannot happen when an user interface component receives focus, or automatically when the user changes a setting.

3.3) Input assistance: "Help users avoid and correct mistakes." The first level A success criterion states that input errors are automatically identified and described to the user in textual form. The second success criterion determines the presence of labels or titles on content when it requires input.

4.1) Compatible: "Maximize compatibility with current and future user agents, including assistive technologies." The fulfillment of this guideline is disputed, in virtue of its definitions existing in possible conflict with the underlying technology best practices, such as the HTML Living Standard. Thus, as stated on the WCAG website: "In practice, this criterion no longer provides any benefit to people with disabilities in itself." However, the work at hand will subscribe to HTML and Javascript best practices to maximize compatibility over time and over different systems.

3.3 Design and development

After acquiring the Webchat code and setting up a local Rasa chatbot server, the code underwent inspection with the support of a specific static analyzer for accessibility in the form of the plugin "eslint-plugin-jsx-a11y," along with usability testing of the features. In Figure 1, we can get an idea of the app's structure. The widget is triggered by a button typically located in the bottom right corner of the page. When opened, it displays three sections: a header, message log, and text field.

The header displays the conversation title. The message log is in the middle portion and visually distinguishes between messages sent by the agent and those sent by the user through colors and positioning. Agent messages appear further left in a grayish bubble, while user messages appear further right against a blue background. The text field is the only element in the overall structure that receives focus through the tab key. This feature is depicted in the image by the area marked with an orange dotted outline.

Figure 1: Simplified structure of the original widget

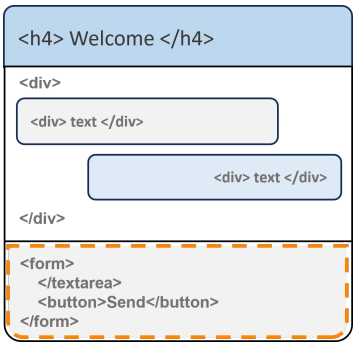


Figure 2: Image depicting a message with response buttons

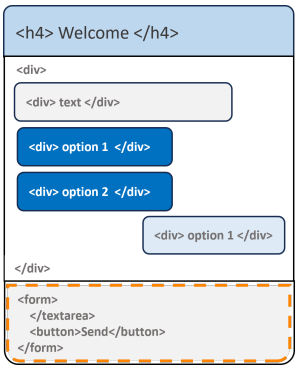
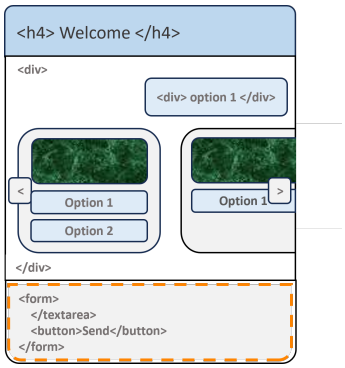


Figure 3: Image depicting a message with carousel enabled



During interaction with the artifact, the following characteristics were identified as not compliant with WCAG 2.0:

3.3.1 Inadequate use of HTML elements and WAI-ARIA roles. The widget has an HTML structure heavily reliant on generic elements such as divs and spans, which are styled through stylesheets. When styled, these elements visually communicate the widget's structure through colors, shapes, and spatial arrangement. However, the screen reader interacts poorly with such elements, natively delivering no information beyond what is explicitly written in the element's content. Accessibility guidelines encourage the use of more specific elements that readily communicate their function - such as list, button, anchor - or at least WAI-ARIA roles.

3.3.2 Absence of element descriptions. When focused during navigation, relevant HTML elements such as the opening button and the carousel failed to provide accurate identification to the screen reader, violating guidelines 1.1 and 1.3. Adding this identification is a minimum measure for visually impaired users to perceive the focused content. The unidentified or generically identified elements include the widget opening button, closing button, header, conversation body, and messages.

3.3.3 No textual alternative for message identification. When using instant messaging widgets, visually perceptive users rely on the graphic hierarchy of elements to distinguish their messages from those sent by the chatbot. However, this organization does not immediately translate to screen reader users. Therefore, the lack of a text alternative for identifying the sender of each message violates guideline 1.1.

3.3.4 Limited keyboard navigation. In its original state, the widget implements significantly limited keyboard navigation. Only the text field receives focus via the tab key. It is possible to navigate elements using arrow keys or some keyboard shortcuts, but some shortcuts exhibit inconsistent behavior. Particularly, the shortcut to navigate through paragraphs only traverses received messages, skipping messages sent by the user. Additionally, some relevant clickable elements cannot be activated via the keyboard, which is the main deficiency of the widget in this aspect. For a new user, this structure leads to an interaction based on trial and error, where many commands will not have an effect. These characteristics directly impact guidelines 2.1 and 2.4.

Below are some more relevant characteristics for the accessibility aspect:

3.3.5 Interaction title. The chatbot messenger includes a heading with text that reads 'Welcome'. When searching for page headers, users encounter this minimally descriptive text regarding the referenced functionality. It would be beneficial to have a more descriptive title for the widget's functionality to benefit all users.

3.3.6 Notification for new messages. The presentation of an interface with a chatbot is inherently dynamic, given the immediate response expected after a message is sent. Therefore, an auditory announcement mechanism for received messages can assist screen reader users in keeping pace with the interaction. The widget lacks any form of notification of this nature.

3.3.7 Diverse message formats. In addition to plain text, Webchat can present content in various formats: buttons, carousels (a sequence of cards with images and links), images, videos, and markdown. To support accessibility for screen reader users, it is essential to maintain a consistent interaction pattern across all types of elements displayed. For this work, the behavior of messages with buttons and carousels will be analyzed, leaving the other message types for future efforts.

Messages with buttons serve the purpose of presenting predetermined response options to the chatbot's key conversation points. When activated, the button's content becomes the last message sent by the user in the conversation. In the original implementation, the buttons are styled div elements and do not respond to keyboard input, exclusively triggered by cursor click. This behavior violates keyboard navigability guidelines and needs to be modified to become accessible.

Messages with carousels feature a structure containing cards composed of an image, title, subtitle, and action links. Some accessibility related issues are noteworthy. The structure includes images without text descriptions. Addressing this problem would require more complex refactoring, as the texts and images are received from the Rasa backend. Hence, it would be necessary to amend the entire backend-to-frontend interface to accommodate alternative texts. Another deficiency in the carousel is the presence of several div elements presented as buttons that lack keyboard accessibility. Additionally, there is no non-textual indication of what the carousel is. When presented, screen reader users would perceive only a set of elements inconsistent with the dialogue flow.

3.3.8 Implementation Proposal. To meet the guidelines within the context of these characteristics, the following implementation proposals have been formulated:

(1) **Eliminating all accessibility warnings** All accessibility warnings raised by the static code analyzer will be solved.
(2) **Adaptation of the HTML structure of the message window:** Use appropriate HTML elements to reflect the desired information structure. Represent list elements, buttons, and messages with the correct elements: "ol" (ordered list), 'button' (button), and 'p' (paragraph), respectively. When it is not feasible to use the 'button' element, add the WAI-ARIA 'button' function for a similar effect.
(3) **Enhance element descriptions:** Add text identifications detectable by the screen reader for the widget's open button, widget header, and carousel.
(4) **Enhance keyboard navigation:** Ensure that the new widget structure enables keyboard navigation in an easy and representative manner of the correct navigation order. The adaptation of the HTML structure contributes partially to achieving the goal. Additionally, it is necessary to modify all elements programmed as buttons but not clickable through keyboard input, making them operable, or remove them from screen reader detection when their functionality does not make sense for the user.
(5) **Identification of message senders:** Add an identifier to each message so that the user hears the text 'the agent said' before receiving messages or 'you said' before sending messages.
(6) **Notification of new messages:** Using the WAI-ARIA 'log' function in the message list adds the functionality of detecting and announcing any added child elements within its context. Consequently, the user will perceive the arrival of

new messages without needing to navigate to the end of the message list.

(7) **Add navigation shortcuts:** Include shortcuts allowing the user to move quickly from the top to the bottom of the structure without navigating through the remaining elements of the widget and the page.

3.4 Validation

For the sake of enabling future reproductions of the validation experiments, it is imperative to consider all configurations used in the development and testing environment, and the sequence of actions employed in testing the new version of the Webchat. In its normal use case, Webchat will be installed on a web page following the instructions provided in its GitHub repository. For the widget to appear on the page, the Rasa server must be running. The environment used in development involved the Ubuntu 22.04 operating system on an Intel x86 platform, the Orca screen reader available by default on the OS, and the Firefox 120.0 browser. The Webchat version 1.0.2 served as the base for implementing the proposed functionalities. The Rasa Open Source server in its version 3.2.0 was employed, running the 'retail-demo' agent [5] updated to allow faster access to the carousel.

The experiment conducted to generate the results for this article involves running two dialogue scenarios on Webchat while using a screen reader. The scenarios will be executed with the Webchat in its initial state and then repeated on the version of the Webchat containing the new functionalities. To evaluate the interactivity of each scenario, it will be crucial to detect which elements are being focused on by keyboard navigation and to note the screen reader's auditory output for these elements. The experiment will be recorded using screen capture while the test scenarios are executed, and later transcribed for easier analysis.

Scenario A will follow the "Check status of my order" option, where the user will be asked for their email to find the order and later will need to rate the service using buttons with ratings from 1 to 5. This scenario allows evaluating the basic functionality of the widget, as well as the use of buttons in interactions complementing the use of text. Scenario B should follow the path "Start a return," configured to display a carousel — the more complex message containing cards and links — that must also be examined from the screen reader accessibility perspective.

After running the scenarios, the transcriptions of each execution will be compared to verify if the audio output meets the requirements. It will also be possible to compare the keyboard focus on certain elements through the recorded images. The enhanced version of Webchat is expected to have a higher number of keyboard-navigated elements and a more detailed audio description of the interaction, aiming to better comply with the principles of WCAG 2.0.

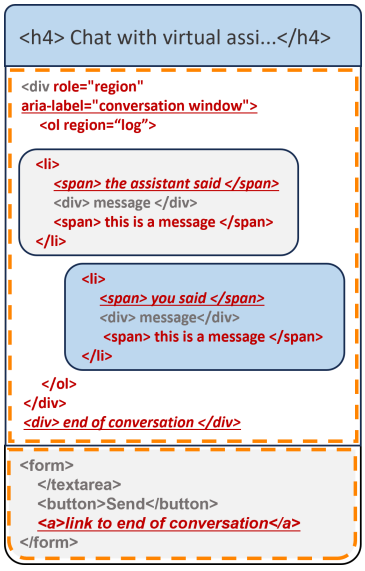
4 RESULTS AND DISCUSSION

In this section, the final product will be showcased, highlighting the techniques employed to achieve the implementation proposals, along with a functional comparison between the two versions of the widget. The widget's source code, including the described changes, is available for inspection and testing on the GitHub repository[6].

4.1 Demonstration of implemented features

The basic structure of the widget was significantly altered while mostly preserving the original visual characteristics. The following subsections discuss the impacts of the changes on each element of the widget. Figure 4 exemplifies the changes made on the overall structure, highlighting in red the newly added items, framing the elements focusable by tab key in orange dashed lines, and underlining the elements that are not visible on the screen and were added exclusively for screen reader browsing.

Figure 4: Simplified structure of widget with new features



4.1.1 Widget opening button. The widget opening button was already a properly styled button element set to open or close the dialogue windows. However, it did not have any alternative text identification. The element has been modified to include the description 'talk to assistant' to directly communicate the button's function to screen reader users.

4.1.2 Header. The header maintained the same structure but with a different title - 'chat with virtual assistant' - indicating a conversation with the virtual agent would happen under that structure. Among screen reader users, navigating by headers is a widespread method to quickly move to the main content of the page[3], emphasizing the importance of using descriptive headers as there is a high likelihood of users using them to orient themselves on the page.

4.1.3 Conversation window. This part of the widget stands out from the rest as it contains all the communications exchanged. Throughout the interaction, there is an expected growth in the content within this structure, so it should be easy to navigate through the entire message history.

[5]Available at https://github.com/RasaHQ/retail-demo

[6]https://anonymous.4open.science/r/rasa-webchat-3E0B/README.md

The message window was given the 'region' WAI-ARIA role, which highlights it as a relevant element in the structure and allows it to receive keyboard focus via the tab key. The window also received a label with the content 'conversation window' to inform the user textually about the content of the focused element.

The messages themselves are now elements within a list highlighted as a 'log' region. This type of region announces when new elements are added, providing handling for dynamic changes in the chat. Moreover, the list element informs the screen reader user about the number of items in that list and allows navigation using shortcuts to move to the next or previous item in the list. The messages have received a visually hidden but screen-reader-detectable identifier indicating the sender of that message, in the form of 'the assistant said' or 'you said' snippets.

In a conversation, the most recent messages are assumed to be of greater importance as they effectively drive the current moment of the dialogue. To allow easy identification of this point, an element indicating the end of the message window has been added at the bottom. This element is also used as a landing point for another link further ahead to provide quick access to the actual location of the most recent messages.

4.1.4 Text field. The text field already had an HTML structure suitable for screen reader use, so no planned changes were made to this section. Just as an additional feature, a link to the end of the message log was added after the existing elements, allowing users to go directly from there to the area below the last message.

4.1.5 Buttons and carousel. To ensure keyboard navigability, the "button" function was implemented on all div elements styled as buttons in messages and the carousel. This function assists the screen reader in identifying that element as a button, although it does not ensure keyboard click functionality. To achieve this, a treatment of keyboard input was added to the Javascript code. Consequently, all elements that read as buttons can now be focused on and clicked through the keyboard. The carousel is also preceded by a message describing its structure for the screen reader, and its elements are within a list structure to allow the user to navigate the items in the correct order.

4.2 Execution of test scenarios

To validate the implemented functionalities, one of the paper's authors performed the test scenarios described in the methodology section. Therefore, the test subject was a person who possesses full visual function and basic proficiency in using a screen reader. However, a more conclusive validation of the tool would require the participation of a comprehensive group of users with visual impairments and advanced proficiency in using a screen reader at this stage. Constraints in time and human resources did not allow for this type of validation.

The tests were executed on the machine and OS described in section 4.4, recorded, and transcribed. In total, four tests were recorded, comprising the run through of test scenarios A and B using the original version of the Webchat as well as the modified version born from the implementation proposal described in section 4.3.8. The transcription files are available through the Zenodo platform[2].

4.3 Scenario A in original widget

The transcript of this test is recorded in the file "transcript-A-orig.txt". In this scenario, the user is supposed to attempt to check the status of their online store order via the chatbot. The first step to run this scenario was to open the page containing the widget and open it by interacting with its opening button. In the original software, the button lacked any descriptive title, and the screen reader announced it solely as a button landmark, as recorded in second 39 of the transcript.

Upon opening the widget, the bot sent a greeting message and a first set of response options in buttons. None of this information was announced to the user by the screen reader, nor there was any notification sound. To find the greeting message, it was necessary to use the P key, which is used for browsing through paragraphs. The button links were found only using the A key, responsible for toggling between clickable elements. The transcription shows an attempt to find the buttons using other keys that browse specific types of content, such as the K and L keys used to point to links and lists, respectively. Since the widget was on a practically empty page, this attempt did not shift the focus away from the navigation within the widget itself. In a common use case where the widget would be on a page containing other content, using shortcut keys would likely move the screen reader cursor to some unspecified point on the page, interrupting the interaction with the widget. This chain of actions is recorded in the transcript from timestamp 0m58s to 1m46s.

After detecting the buttons, there was an attempt to interact with the "check status of my order" option, but it was not possible to activate any of the buttons via the keyboard. To proceed with the scenario, the message was to be typed into the text entry field. However, the text field and the screen reader did not play well with each other, since focusing on the text field with the tab button and then typing led to the screen reader trying to use the browsing functions for the keys. For example, hitting the C key prompted the screen reader to browse through combo boxes instead of inserting the letter C on the text field. This same behavior happened again on future attempts to use the text field. For the sake of the test, the "Check status of my order" option was clicked using the mouse cursor.

From this point on until the end, browsing the chat widget after finding which keys were most effective was speedier, although the limitations on interaction with buttons and the text field added an unwanted level of challenge to the experience. The dynamic of browsing repeatedly to a message, then to the text field, and back again to the messages proved cumbersome, especially since the screen reader does not announce any change in the conversation context when the user sends or receives messages. The transcript describes the remaining interactions from minute 2m37 to the end.

4.4 Scenario A in modified widget

The transcript for this test run is recorded in the file "transcript-A-mod.txt". Executing the same scenario with the new functionalities revealed the implemented changes. The buttons are now actionable via the keyboard, as seen in the transcript minutes 3m05s and 4m33s. The word "return", captured in the recording, was uttered by the screen reader when pressing the Enter key with a focus on a button,

and it is succeeded by a message, indicating the widget correctly activated the button.

The identification of the sender was also noticed. An example is on minute 1m58s, when the message automatically reads "the assistant said" followed by the actual message. In minutes 3m44s through 3m56s there is also an example of an exchange of messages with the announcement of senders. This behavior creates a textual alternative to a layer of information conveyed solely through visual means in the original widget, which brings the widget to comply with the "Text alternatives" WCAG guideline.

Another notable change is the automatic announcement of messages by the screen reader. There was no need to navigate to new messages to hear them. Consequently, more information was presented passively without the need for active exploration within the widget, adding to the dynamism of the experience. However, an undesired behavior was observed: the repetition of new messages. A hypothesis for why this occurred is due to the dynamic rendering of elements in the region. As elements are added and removed from the Document Object Model (DOM) to represent visual animations, the screen reader queues up these updates, resulting in the repetition of some messages. To address this functionality, the ideal solution would involve implementing a sensitive area for updates, with its content and behavior defined separately from the visual implementation. Such an implementation was beyond the scope of this work as it would require more comprehensive alterations in the program structure.

Finally, the addition of HTML list elements to depict the conversation was perceived in multiple sentences. On minute 3m32, when moving the cursor to the text field, the screen reader announced the browsing was leaving the list. On minute 4m21s, when navigation returned to the conversation window, the number of items on the list of messages was enunciated, adding context to the status of the conversation. This information is useful since it allows the user to keep track of the dialogue's length, or if it has received any updates.

4.5 Scenario B in original widget

Following the naming convention, the transcription to which this section refers is in the file "transcript-B-orig.txt". Since this scenario was executed after the first one, the tester had already understood the commands for basic browsing through the app. Upon initiating the dialogue and selecting the "start a return" option, again the button did not respond to either the Return or Space key, causing the tester to resort to clicking the button with the mouse. This action prompts the carousel to be displayed, which happens without any automatic update for the screen reader.

While traversing the carousel structure with the directional keys, trying to activate the clickable buttons using the keyboard is ineffective for two of the three buttons displayed, significantly limiting the carousel's functionality for screen reader users. The buttons to move left and right are detected and identified as buttons without any additional context, leaving a gap in user understanding. Nonetheless, these buttons are rendered unnecessary because their only function is to scroll the carousel cards to the sides so the user can see them, for only one card is fully visible at a time. The screen reader user interacts without visual support, so the way to browse through the carousel elements is by using the arrow keys.

4.6 Scenario B in modified widget

This section refers to the file "transcript-B-mod.txt". When performing the same interaction with the added accessibility features, several behavior changes are noticed. Upon the carousel's appearance, the user receives an audio update covering all carousel elements. Navigating to the top element of the carousel prompts the screen reader to enunciate a description of the carousel's structure as appears on the transcript on minute 2m27s. As its cards are contained in a list structure, the screen reader will also say that it is a list with a certain number of elements.

The buttons respond to keyboard input as expected: on the first card, the buttons add messages to the conversation, while the button on the second card opens the card's image in a new tab, as appears on timestamps 3m07s, 3m21s, and 3m40s. there is no mention of the buttons responsible for scrolling the card since they add nothing to the experience via a screen reader, and thus were disabled for assistive technologies through the use of an aria-hidden attribute.

5 THREATS TO VALIDITY

The proposed solution has potential threats worthy of future analysis and addressing. The primary perceived threat is the execution of validation steps exclusively by the same person responsible for developing the solution. Besides, this person is devoid of any visual disability. Testing with blind or impaired users is crucial considering the particular mental model established in their use of assistive technology. Concerning frequent screen reader users, known that the screen reader's configuration and navigation techniques are influenced by visual impairment and proficiency with the screen reader software. Direct testing with these users is the best way to determine if the techniques used in this work truly benefit them. Otherwise, strategies may have been adopted that do not affect or even hinder browsing the Webchat with a screen reader.

Threatening the generality of the proposal is the fact that the solution's construction was entirely based on the Webchat tool designed for agents deployed on the Rasa or Botfront framework. These are not the only types of existing agents, just as Webchat is not the only available presentation layer. Therefore, the results obtained may not generally apply to chats built on other technologies.

Similarly, the experiment was exclusively conducted on the Orca screen reader and Firefox browser, while others are available. Some examples of screen readers include JAWS, NVDA, and VoiceOver, and other well-known and used browsers are Google's Chrome and Microsoft's Edge. Thus, there is a gap in testing, which may lead to questioning if implemented functionalities can be generalized to other screen readers and browsers or if they may perform differently in each of them.

6 CONCLUSION

Inspired by the increasing demands for universal accessibility in technology, this work focused on enhancing the Webchat interface for chatbots for use by visually impaired individuals who rely on screen readers to access digital content. Webchat is a widget capable of displaying messages exchanged with a virtual agent and enabling the sending and receiving of new messages, including content formats such as buttons and carousels. The main issues

detected were inadequate use of HTML structure, limited keyboard navigation, lack of element identification, and absent treatment of dynamic content. The WCAG 2.0 guidelines guided the proposed implementation of changes in various elements of Webchat to make it more perceivable, operable, understandable, and robust. Thus, the main improvements proposed were: eliminating all accessibility warnings raised by the static code analyzer, adequating the HTML structure to accessibility standards, enhancing element descriptions as well as keyboard navigation, adding identification of senders to messages, automatically announcing new messages, and adding navigation shortcuts.

The authors validated the solution and through execution, recording and transcribing of two use cases of interaction with chatbot through the Webchat. Below is a brief discussion of the work's results. In both scenarios, the new features allow for more accurate navigation. Buttons now respond to keyboard input, and key elements such as the opening button and the sender of messages received identification that was correctly announced by the screen reader, eliminating the need for users to guess or test functionalities to discover their purposes. Other elements, whose behavior detracted from the overall experience when browsed by a screen reader, were successfully hidden in the context of assistive technologies, meaning the screen reader ignored them, improving the flow of use. Transcriptions of the use case recordings reveal increased expressiveness of the widget, although interaction may be hindered by repetitive uttering of dynamically added content.

It is worth noting that the scope of this work, as well as the limited time and human resources available, did not allow for consulting with people with disabilities while validating the solution. The authors resorted to researching accessibility standards and best practices to establish solution goals and define the ways to reach them. Hence, future works might involve revisiting the proposed changes alongside users with visual impairment, as well as repeating the experiments to validate the solution.

The resulting software is a much-needed initial step in Webchat's implementation of accessibility, making small changes in the code that can positively impact the use of this interface by visually impaired individuals. The immediate gain in keyboard navigability and translation of visual structure into text descriptions is evident, although further testing and broader experimentation are needed to validate the functionalities. In a future stage, conducting a formal experiment to validate the widget would be interesting, evaluating with users which functionalities fulfill the role of making the widget more accessible, which elements hinder navigation, and which gaps still remain.

ACKNOWLEDGMENTS

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001.

REFERENCES

[1] Eleni Adamopoulou and Lefteris Moussiades. 2020. An overview of chatbot technology. In *Artificial Intelligence Applications and Innovations*. Ilias Maglogiannis, Lazaros Iliadis, and Elias Pimenidis, (Eds.) Springer International Publishing, Cham, 373–383. ISBN: 978-3-030-49186-4.

[2] Anonymous Anonymous. 2023. Rasa webchat tests for screen reader accessibility project. (Dec. 2023). DOI: 10.5281/zenodo.10234060.

[3] Yevgen Borodin, Jeffrey P Bigham, Glenn Dausch, and IV Ramakrishnan. 2010. More than meets the eye: a survey of screen-reader browsing strategies. In *Proceedings of the 2010 International Cross Disciplinary Conference on Web Accessibility (W4A)*, 1–10.

[4] Petter Bae Brandtzaeg and Asbjørn Følstad. 2017. Why people use chatbots. In *Internet Science: 4th International Conference, INSCI 2017, Thessaloniki, Greece, November 22-24, 2017, Proceedings 4*. Springer, 377–392.

[5] Ben Caldwell, Michael Cooper, Loretta Guarino Reid, Gregg Vanderheiden, Wendy Chisholm, John Slatin, and Jason White. 2008. Web content accessibility guidelines (wcag) 2.0. *WWW Consortium (W3C)*, 290, 1–34.

[6] Rocío Calvo, Ana Iglesias, and Lourdes Moreno. 2013. An accessible chat prototype for screen reader users in mobile devices. In *HCI International 2013-Posters' Extended Abstracts: International Conference, HCI International 2013, Las Vegas, NV, USA, July 21-26, 2013, Proceedings, Part I 15*. Springer, 216–220.

[7] Débora Conforto and Lucila Maria Costi Santarosa. 2002. Acessibilidade à web: internet para todos. *Informática na educação: teoria & prática. Porto Alegre. Vol. 5, n. 2 (nov. 2002), p. 87-102.*

[8] BRASIL. Ministério da Gestão e da Inovação em Serviços Públicos. 2014. E-MAG: modelo de acessibilidade em governo eletrônico. Versão 3.1. Brasília-DF. (Apr. 2014). Retrieved July 11, 2023 from http://www.governoeletronico.gov.br/documentos-e-arquivos/eMAGv31.pdf.

[9] Rashmi Dharwadkar and Neeta A Deshpande. 2018. A medical chatbot. *International Journal of Computer Trends and Technology (IJCTT)*, 60, 1, 41–45.

[10] Simone Bacellar Leal Ferreira. 2015. E-acessibilidade: tornando visível o invisível. *Revista Morpheus - Estudos Interdisciplinares em Memória Social*, 6, 10, (Mar. 2015). http://seer.unirio.br/morpheus/article/view/4780.

[11] Asbjørn Følstad, Cecilie Bertinussen Nordheim, and Cato Alexander Bjørkli. 2018. What makes users trust a chatbot for customer service? an exploratory interview study. In *Internet Science: 5th International Conference, INSCI 2018, St. Petersburg, Russia, October 24–26, 2018, Proceedings 5*. Springer, 194–208.

[12] Guruswami Hiremath, Aishwarya Hajare, Priyanka Bhosale, Rasika Nanaware, and KS Wagh. 2018. Chatbot for education system. *International Journal of Advance Research, Ideas and Innovations in Technology*, 4, 3, 37–43.

[13] Jonathan Lazar, Aaron Allen, Jason Kleinman, and Chris Malarkey. 2007. What frustrates screen reader users on the web: a study of 100 blind users. *International Journal of Human–Computer Interaction*, 22, 3, 247–269. eprint: https://doi.org/10.1080/10447310709336964. DOI: 10.1080/10447310709336964.

[14] Kate Lister, Tim Coughlan, Francisco Iniesto, Nick Freear, and Peter Devine. 2020. Accessible conversational user interfaces: considerations for design. In *Proceedings of the 17th international web for all conference*, 1–11.

[15] Bei Luo, Raymond YK Lau, Chunping Li, and Yain-Whar Si. 2022. A critical review of state-of-the-art chatbot designs and applications. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 12, 1, e1434.

[16] Valentyn Melnyk. 2014. Accessible web chat interface. Rochester, New York, USA, (2014). DOI: 10.1145/2661334.2661415.

[17] Anthony Robert Joseph Nicholl. 2001. O ambiente que promove a inclusão: conceitos de acessibilidade e usabilidade. *Revista Assentamentos Humanos*, 2, 3.

[18] Rhoda Olkin. 2002. Could you hold the door for me? including disability in diversity. *Cultural Diversity and Ethnic Minority Psychology*, 8, 2, 130.

[19] Ken Peffers, Tuure Tuunanen, Charles E Gengler, Matti Rossi, Wendy Hui, Ville Virtanen, and Johanna Bragge. 2020. Design science research process: a model for producing and presenting information systems research. (2020). arXiv: 2006.02763 [cs.SE].

[20] Dagfinn Rømen and Dag Svanæs. 2012. Validating wcag versions 1.0 and 2.0 through usability testing with disabled users. *Universal Access in the Information Society*, 11, 375–385.

[21] Richard Rutter et al. 2007. *Web accessibility: Web standards and regulatory compliance*. Apress.

[22] Tom Shakespeare et al. 2006. The social model of disability. *The disability studies reader*, 2, 197–204.

[23] Sandeep A Thorat and Vishakha Jadhav. 2020. A review on implementation issues of rule-based chatbot systems. In *Proceedings of the international conference on innovative computing & communications (ICICC)*.

[24] Cecília Torres, Walter Franklin, and Laura Martins. 2019. Accessibility in chatbots: the state of the art in favor of users with visual impairment. In *Advances in Usability, User Experience and Assistive Technology*. Tareq Z. Ahram and Christianne Falcão, (Eds.) Springer International Publishing, Cham, 623–635. ISBN: 978-3-319-94947-5.

[25] Elisabeth Fátima Torres, Alberto Angel Mazzoni, and João Bosco Da Mota Alves. 2002. A acessibilidade à informação no espaço digital. *Ciência da Informação*, 31, 3, (Sept. 2002), 83–91. DOI: 10.1590/S0100-19652002000300009.

[26] Roel J Wieringa. 2014. *Design science methodology for information systems and software engineering*. Springer.