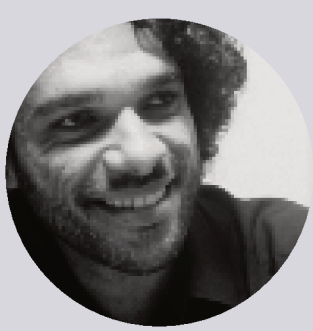
LOGO

# Comparative study of automatic web accessibility evaluators on formative evaluation

## *Estudo comparativo de avaliadores automáticos de acessibilidade web na avaliação formativa*


**Jefferson Ferreira Ribeiro**

Graduando em Tecnologia da Informação,
Universidade Federal do Rio Grande do Norte
Instituto Metrópole Digital, Universidade Federal do Rio Grande do Norte
jfrsnr@gmail.com



**Bruno Santana da Silva**

Doutor em Informática, Pontifícia Universidade Católica do Rio de Janeiro
Instituto Metrópole Digital, Universidade Federal do Rio Grande do Norte
bruno@imd.ufrn.br

## ABSTRACT

Accessibility evaluation is an important activity during digital design process. Although there is software to automatic evaluate web accessibility, little research is done on results from these automatic evaluators, in particular taking into account formative evaluation. This work reports a study that compares results from eight automatic evaluators which are browser's extensions. We analyzed the total amount of reported problems, as well as their stratification by type (errors and warnings), violated principles and criteria, page and occurrence place within the page were analyzed. The results showed that each evaluator stands out in different aspects, being relevant the use of more than one evaluator for a wider coverage in different aspects. The three most promising evaluators were AInspector, Accessibility Checker and Code Sniffer.



## Keywords

Formative Evaluation. Accessibility. Wcag. Emag.



## Resumo

Avaliação de acessibilidade é uma importante atividade no processo de design digital. Apesar de existirem softwares que realizam automaticamente a avaliação de acessibilidade web, pouco se investigou sobre os resultados apresentados por estes avaliadores automáticos, em particular considerando a avaliação formativa. Este trabalho apresenta um estudo que compara os resultados apresentados por oito avaliadores automáticos que são extensão de navegadores. Analisou-se a quantidade total de problemas reportados, bem como sua estratificação por tipo (erros e avisos), princípios e critérios violados, página e local de ocorrência dentro da página. Os resultados encontrados demonstram que cada avaliador se destaca em aspectos diferentes, sendo relevante a utilização de mais de um deles para uma cobertura mais ampla em diferentes aspectos. Os três avaliadores mais promissores foram AInspector, Accessibility Checker e Code Sniffer.



## Palavras Chave

Avaliação Formativa. Acessibilidade. Wcag. Emag.

# 1 INTRODUÇÃO

A avaliação assume papel importante para apoiar o designer verificar se a solução sendo concebida está adequada ao problema de design (Freitas et al., 2013). Ela pode ser considera como formativa ou somativa, dependendo do momento em que é realizada (Barbosa & Silva, 2010). A avaliação formativa é executada enquanto a solução ainda está sendo concebida e, portanto, ainda não foi concluída, nem foi disponibilizada para uso. A avaliação somativa é executada quando a solução já foi concluída e, assim, já poderia estar disponível para uso. No design de informação para web, a avaliação formativa pode se estender até a programação das páginas web, quando o designer pode estudar como suas ideias se materializam nas tecnologias disponíveis. Todavia, neste momento o sistema web ainda não foi disponibilizado para uso; ou seja, ainda é acessado apenas em computadores específicos normalmente internos em uma instituição, e não livremente pela internet. A avaliação formativa deve considerar o sigilo do projeto conforme as estratégias de pessoas e instituições envolvidas.

A avaliação de interfaces pode ser executada de forma manual ou automática (Barbosa & Silva, 2010). Na avaliação manual, o avaliador é responsável por coletar e analisar dados para elaborar uma lista de problemas na interface. Na avaliação automática, por outro lado, um software é responsável por automaticamente coletar e analisar dados para gerar uma lista de problemas, sem intervenção humana no meio do processo. Avaliadores automáticos de interface têm o potencial de reduzir trabalho e tempo necessários para execução da avaliação, bem como reduzir a interferência de falhas humanas causadas por falta de atenção, esquecimento ou equívocos. Contudo, os resultados automáticos são limitados e costumam ser complementados por resultados de avaliações manuais. Assim, é possível delegar a avaliadores automáticos o trabalho básico, pesado e de certa forma mecânico; enquanto que a avaliação manual pode se concentrar em um conjunto menor de atividades mais sofisticadas intelectualmente.

A avaliação de interfaces web pode considerar critérios de qualidade de uso como usabilidade, experiência do usuário, comunicabilidade e acessibilidade (Barbosa & Silva, 2010). Para atender uma demanda social de inclusão de pessoas com deficiência, também exigida por força de leis (Brasil, 2015), a acessibilidade de sistemas web tem recebido maior atenção da academia e da indústria. A acessibilidade possibilita o alcance, a percepção, o entendimento e a interação para a utilização,


e-Revista LOGO - v.8, n.3, 2019 - ISSN 2238-2542
http://doi.org/10.26771/e-Revista.LOGO/2019.3.03

participação e contribuição, com autonomia e segurança, de sistemas web por qualquer pessoa, independente de sua capacidade motora, visual, auditiva, intelectual e social (W3C-Brasil, 2013). Um sistema web acessível possui uma interface com usuário que não impõe obstáculos (ou barreiras) a usuários com diferentes capacidades motoras, de sentido e cognitivas (Barbosa & Silva, 2010).

O World Wide Web Consortium (W3C) é uma organização internacional que define padrões para programação (e.g. HTML, CSS, JavaScript) e diretrizes de acessibilidade web. As diretrizes da W3C são conhecidas pelo acrônimo WCAG, de Web Content Accessibility Guidelines. Provavelmente a versão mais conhecida da WCAG é a versão 2.0 publicada em 2008 (WCAG2, 2008), porém houve uma atualização para a versão 2.1 em 2018 que ainda está sendo difundida. Essas diretrizes têm sido a principal orientação para avaliação de acessibilidade na web. Elas permitiram a criação de softwares capazes de realizar avaliação automática de acessibilidade, apresentando ao designer uma lista de problemas encontrados nas interfaces web.

Atualmente existem muitos softwares de avaliação automática de acessibilidade web, mas ainda sabemos pouco sobre seus impactos na atividade de avaliação. Encontramos na literatura publicações que discutem as diretrizes de acessibilidade (Bach et al., 2009) e uma grande variedade de estudos de caso sobre avaliação de acessibilidade web (Licheski & Fadel, 2013; Mezzaroba et al., 2016; Souza & Malheiros, 2018). Também é possível encontrar estudos que analisam avaliadores automáticos de acessibilidade web (Fuertes et al., 2009; Souza & Mont'Alvão, 2012; Pivetta at al., 2013; Pacheco et al., 2016). Alguns desses estudos consideram poucos avaliadores automáticos, geralmente dois ou três. Outros concentram-se em avaliadores que dão suporte apenas à avaliação somativa, por exigirem que o sistema web já esteja publicado na internet. Em geral, os estudos analisam a quantidade total de problemas encontrados, sem explorar mais profundamente os resultados apresentados. O designer ainda carece de informações mais detalhadas para poder escolher as ferramentas de suporte que vai utilizar durante avaliações formativas.

Este trabalho apresenta um estudo comparativo de avaliadores automáticos de acessibilidade web para suporte à avaliação formativa. Oito extensões gratuitas de navegadores web foram utilizadas para analisar cinco páginas de um Atlas Virtual de Seres Vivos (Marques et al., 2016) que não está disponível na internet. Os problemas identificados pelos avaliadores automáticos foram analisados em termos de quantidade total, classificação como erros ou avisos, princípios e critérios de acessibilidade violados, página e local de ocorrência dentro da página.


e-Revista LOGO - v.8, n.3, 2019 - ISSN 2238-2542
http://doi.org/10.26771/e-Revista.LOGO/2019.3.03

Apesar de as diretrizes básicas desses avaliadores serem as mesmas (ou equivalentes, no caso do eMAG), é evidente a grande diferença nos resultados apresentados pelos avaliadores automáticos.

## 2 Acessibilidade web

A inclusão de pessoas com deficiência na sociedade vem sendo uma reinvindicação de movimentos sociais nas últimas décadas. Conquistas importantes levaram a criação de leis que garantam a acessibilidade em diferentes dimensões: arquitetônica, instrumental, metodológica, comunicacional, digital, entre outras (Brasil, 2015). No âmbito digital, duas iniciativas de promoção da acessibilidade merecem destaque quando tratamos da web. A primeira é a iniciativa internacional promovida pela W3C que tem como principal produto as diretrizes de acessibilidade da WCAG (WCAG2, 2008). A versão 2.0 da WCAG será considerada neste trabalho porque é a versão mais atual suportada pelos softwares analisados. A segunda iniciativa é nacional, promovida pelo Governo Federal que definiu o Modelo de Acessibilidade em Governo Eletrônico (eMAG). A versão mais recente e conhecida do eMAG é a 3.1 (eMAG, 2014).

Os conteúdos da WCAG (WCAG2, 2008) e do eMAG (eMAG, 2014) se assemelham bastante de forma geral (Bach et al., 2009). As principais diferenças aparecem na forma de organização das diretrizes. A WCAG 2.0 organiza suas diretrizes em termos de quatro princípios: perceptível, operável, compreensível e robusto. Estes princípios consideram a acessibilidade da perspectiva de quem usa os sistemas web. Cada princípio está associado a recomendações que, por sua vez, são detalhadas por critérios e especificadas por técnicas. Por exemplo, o Princípio 1, perceptível, está associado à Recomendação 1.1 “fornecer alternativas textuais para qualquer conteúdo não textual” e à Recomendação 1.2 “fornecer alternativas para mídias baseadas em tempo”, dentre outras. A Recomendação 1.1 é detalhada pelo Critério 1.1.1 “Todo o conteúdo não textual que é exibido ao usuário tem uma alternativa textual”, e assim por diante. A numeração dos princípios, recomendações e critérios é definida de forma sistemática e utilizada como índice nos documentos da W3C. Por exemplo, a Diretriz 1.2.3 da WCAG representa o Princípio 1, a Recomendação 2 e o Critério 3. Essa numeração também é utilizada ao longo deste trabalho. Já o eMAG 3.1 organiza suas diretrizes de acessibilidade pela estrutura das páginas web: marcação, comportamento, conteúdo/informação, apresentação/design, multimídia e formulário. Cada grupo


e-Revista LOGO - v.8, n.3, 2019 - ISSN 2238-2542
http://doi.org/10.26771/e-Revista.LOGO/2019.3.03

destes reúne um conjunto de diretrizes específicas no eMAG. Essa forma de organização do eMAG consideram a acessibilidade na perspectiva de quem desenvolve os sistemas web.

Sistemas web isolados costumam ser inacessíveis para vários públicos. Para que eles possam se tornar acessíveis a pessoas com deficiência, eles precisam ser utilizados em conjunto com tecnologias assistivas, como os leitores de tela, por exemplo. Deste modo, é importante que os sistemas web sejam compatíveis com todas as tecnologias assistivas existentes. A principal estratégia para lidar com a diversidade de sistemas web e de tecnologias assistivas tem sido estabelecer e seguir padrões na construção de sistemas web, em particular, na programação em HTML, CSS e JavaScript. Se os navegadores, os sistemas web e as tecnologias assistivas seguirem os mesmos padrões, eles terão a interoperabilidade necessária para promover a acessibilidade web. Os padrões web definidos pela W3C já incluem a interoperabilidade necessária para acessibilidade nas linguagens de programação. Contudo, isso não implica necessariamente na produção de sistemas web acessíveis. As diretrizes de acessibilidade web, como WCAG e eMAG, endereçam exatamente este desafio orientando o design, a construção e avaliação de sistemas web acessíveis de modo a aproveitar os recursos de acessibilidade disponíveis nas linguagens de programação. Algumas dessas orientações podem ser verificadas por softwares que analisam o código-fonte em HTML e CSS dos sistemas web em busca de violações de diretrizes de acessibilidade.

A literatura apresenta uma variedade de estudos de caso com avaliações de acessibilidade baseadas na WCAG ou eMAG ou cujos resultados foram discutidos considerando essas diretrizes (Licheski & Fadel, 2013; Mezzaroba et al., 2016; Souza & Malheiros, 2018). Por exemplo, Licheski e Fadel (2013) avaliaram a acessibilidade do Portal Brasil, o portal oficial do Governo Federal, através de uma verificação manual da interface com um checklist baseado no WCAG e eMAG. Mezzaroba e colegas (2016) avaliaram a acessibilidade do portal do Supremo Tribunal Federal com foco em deficiência visual. Elas tomaram como base a avaliação automática do software ASES. Souza e Malheiros (2018) relatam a avaliação de dois repositórios educacionais abertos, o Portal Domínio Público e o Portal do Professor, considerando pessoas com deficiência motora. Eles utilizaram as diretrizes do eMAG como um checklist aplicado de forma manual. É interessante observar a diversidade de deficiências consideradas e de sistemas web avaliados tendo em vista as diretrizes da WCAG e eMAG. Softwares de avaliação automática foram apenas ferramentas nesses trabalhos, não objetos de estudo.


e-Revista LOGO - v.8, n.3, 2019 - ISSN 2238-2542
http://doi.org/10.26771/e-Revista.LOGO/2019.3.03

A literatura também apresenta outro conjunto de trabalhos relacionados que tem os avaliadores automáticos de acessibilidade como objetos de estudo (Fuertes et al., 2009; Souza & Mont'Alvão, 2012; Pivetta at al., 2013; Pacheco et al., 2016). Fuertes e colegas (2009) comparam funcionalidades de avaliadores automáticos de acessibilidade, tais como plataforma, geração de relatórios e visualização do código com problema de acessibilidade. Essa comparação de característica orientou o desenvolvimento de outro avaliador automático de acessibilidade desenvolvido por eles. Não houve comparação dos resultados de avaliação apresentados pelos avaliadores automáticos. Souza e Mont'Alvão (2012) relatam a comparação dos resultados de avaliação de acessibilidade da página principal do portal do Instituto Nacional da Propriedade Industrial (INPI) por dois avaliadores automáticos: Hera e "daSilva". Eles compararam a quantidade de problemas encontrados como erros e avisos. Pivetta e colegas (2013) analisaram o avaliador automático WAVE, pois foi o único que funcionou adequadamente com sistema que requerem autenticação do usuário (login). Elas analisaram a interface do WAVE e relacionaram os problemas reportados por este avaliador automático com as diretrizes da WCAG. Pacheco e seus colegas (2016) também compararam funcionalidades de três avaliadores automáticos de interface, como diretrizes de acessibilidade utilizadas, exportação dos resultados, indicação visual dos problemas encontrados.

Os estudos anteriores que analisam avaliadores automáticos costumam envolver um número pequeno de avaliadores, não aprofundam a investigação sobre os problemas de acessibilidade relatados, nem comparam os avaliadores em função desses resultados. Além disso, a maioria dos avaliadores automáticos considerados só funciona com sistemas web públicos na web. Deste modo, o designer ainda carece de estudos para orientá-lo sobre o uso de avaliadores automáticos de acessibilidade como ferramenta durante avaliações formativas.

## 3 Avaliadores automáticos de acessibilidade web para avaliação formativa

Para um avaliador automático de acessibilidade web poder ser utilizado durante avaliações formativas, ele precisa funcionar com sistemas web que funcionam apenas em um computador ou dentro de uma rede local. Ele não pode exigir que o sistema web esteja público na internet. Neste caso, o avaliador automático precisa ser um software para desktop ou uma extensão de navegadores web. Com foco na avaliação formativa,


e-Revista LOGO - v.8, n.3, 2019 - ISSN 2238-2542
http://doi.org/10.26771/e-Revista.LOGO/2019.3.03

buscamos softwares desktop e extensões de navegadores na lista de ferramentas de avaliação da W3C (https://www.w3.org/WAI/ER/tools/) que analisem diretrizes da WCAG ou do eMAG, sejam gratuitos e estivessem funcionando em fevereiro de 2019. Além disso, procuramos também extensões de navegadores na loja de extensão do Chrome do Firefox, com os mesmos requisitos. O único software desktop encontrado foi o ASES, porém ele não funcionou depois de instalado em alguns computadores. A Tabela 1 apresenta os oito avaliadores automáticos encontrados com os referidos critérios. Apenas o eScanner verifica o eMAG. Os outros verificam apenas a WCAG. O AInspector foi o único que funciona apenas no Firefox. O eScanner, Site Improve Accessibility Checker e WCAG Accessibility Audit Dev UI funciona apenas no Chrome. Os outros quatro funcionam no Chrome e no Firefox. O nome resumido será utilizado nas imagens e tabelas ao longo do texto.

Tabela 1: Avaliadores automáticos de acessibilidade para avaliação formativa.

| Ferramenta | Diretrizes | Versão | Navegador | Nome resumido |
|---|---|---|---|---|
| AInspector | WCAG | 0.95.0 | Firefox | ainspector |
| aXe Developer Tools | WCAG | 3.7.0 | Chrome e Firefox | axe developer |
| eScanner | eMAG | 2.8 | Chrome | escanner |
| Html_CodeSniffer | WCAG | 2.1.1 | Chrome e Firefox | code sniffer |
| Site Improve Accessibility Checker | WCAG | 117 | Chrome | accessibility checker |
| Tota11y Accessibility Toolkit | WCAG | 1.0.1 | Chrome e Firefox | tota11y |
| Wave Evaluation Tool | WCAG | 1.0.9 | Chrome e Firefox | wave evaluation |
| WCAG Accessibility Audit Dev UI | WCAG | 2.1.2.1 | Chrome | accessibility audit |

Fonte: Elaborado pelos autores.

## 4 Estudo de caso

Este trabalho tem por objetivo comparar os resultados apresentados por oito avaliadores automáticos web indicados na Tabela 1. Serão analisados a quantidade de problemas relatados, o tipo de violação identificada (erro ou aviso), diretriz violada e local em que o problema ocorreu na interface. Também foi considerada a possibilidade de analisar o tempo de execução da análise automática, mas isso foi descartado porque os resultados foram da ordem de milissegundos ou poucos segundos e as diferenças não demonstraram ser significativas.

Metodologia

A metodologia utilizada foi o estudo de caso (Yin, 2003) em que se executou os referidos avaliadores automáticos em um Atlas Virtual de Seres Vivos (Marques et al., 2016). Este sistema web foi escolhido por


e-Revista LOGO - v.8, n.3, 2019 - ISSN 2238-2542
http://doi.org/10.26771/e-Revista.LOGO/2019.3.03

se encontrar num momento adequado à avaliação formativa, sua acessibilidade ainda não tinha sido avaliada e ainda não está disponível na internet. Além disso, os autores deste estudo participam da concepção desta interface e têm acesso ao código-fonte do sistema. Este sistema web também possui uma complexidade interessante, pois suas funcionalidades são diversificadas e vão além de apenas apresentação de informações. A Tabela 2 apresenta a lista das cinco páginas do Atlas Virtual analisadas neste estudo.

Tabela 2: Páginas avaliadas no Atlas Virtual de Seres Vivos.

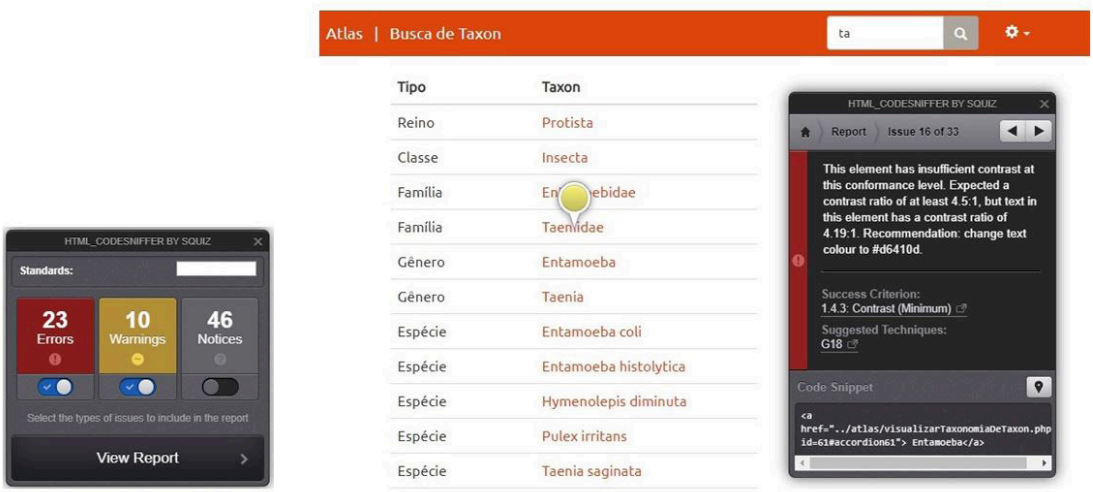
| Página | Funcionalidade |
|---|---|
| Taxonomia | Exibe a taxonomia de seres vivos para consulta, com elementos colapsáveis e expansíveis. |
| Táxon | Exibe informações cadastradas sobre um táxon (e.g. gênero ou espécie específica), em formato de texto e imagens relevantes. |
| Busca | Exibe uma lista de táxon encontrados na busca pelo termo "ta". Também é uma tela prioritariamente de consulta de informações textuais. |
| Login | Tela de *login* para habilitar a edição das informações sobre táxons no sistema. Aqui prioritariamente o usuário precisa realizar entrada de dados. |
| Edição | Tela disponível apenas para usuários autenticados (logados) no sistema. O usuário pode modificar informações em vários campos nos formatos de texto e de imagem. |

Fonte: Elaborado pelos autores.

As páginas deste sistema web possuem muitas estruturas HTML repetidas. Os avaliadores automáticos relatam (tipos e quantidades de) problemas equivalentes para estruturas HTML semelhantes na mesma página. Trabalhar com estas repetições é analisar mais do mesmo, mascarando inclusive diferenças pontuais que seriam mais significativas perante um total menor de problemas. Para facilitar a análise detalhada dos dados, as estruturas HTML semelhantes repetidas foram desconsideradas. Deste modo, os problemas que ocorreram nestes locais foram computados apenas uma vez neste estudo. Por exemplo, foram computados apenas os problemas de uma linha na tabela HTML que aparece na página analisada na Figura 1, pois foram os mesmos para Reino, Classe, Família, Gênero e Espécie. Este descarte foi sistemático em todos avaliadores para manter coerência na comparação dos resultados detalhados entre eles.


e-Revista LOGO - v.8, n.3, 2019 - ISSN 2238-2542
http://doi.org/10.26771/e-Revista.LOGO/2019.3.03

Figura 1. Resultados consolidados (esquerda) e detalhados (direita) do Html_CodeSniffer.




Fonte: Elaborado pelos Autores.

Os dados detalhados foram tabulados em termos de avaliador, página, local de ocorrência do erro dentro da página, diretriz violada, tipo do problema encontrado (erro ou aviso). O local de ocorrência do problema foi definido por números sob uma imagem da página, conforme os problemas foram sendo tabulados. Todas as páginas utilizaram os mesmos números para indicar a violação de diretrizes de acessibilidade naquele local. Semelhante à tabulação dos dados, esses números serão utilizados mais adiante para ilustrar nas figuras o local de ocorrência dos problemas em cada página. Os dados detalhados tabulados foram inseridos em um banco de dados MySQL para facilitar análises quantitativas por meio de consultas SQL. A análise dos dados foi fundamentalmente quantitativa com a contabilização de dados estratificados e de porcentagens. Isso permitiu a construção de gráficos e tabelas apresentados e discutidos a seguir.

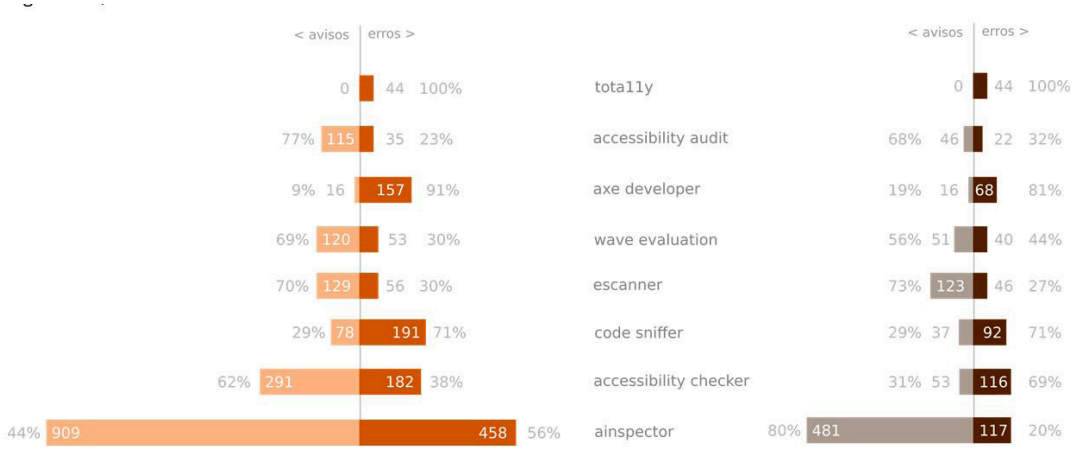
## 5 Resultados

As quantidades totais de problemas consolidadas pelos avaliadores automáticos (considerando repetição nos códigos HTML) são apresentados no lado esquerdo da Figura 2. As quantidades totais de problemas detalhados pelos avaliadores automáticos (desconsiderando repetição nos códigos HTML) são apresentadas no lado direito desta mesma figura. Nos dois casos, os avaliadores estão em ordem crescente da quantidade total de problemas encontrados nas cinco páginas analisadas.


e-Revista LOGO - v.8, n.3, 2019 - ISSN 2238-2542
http://doi.org/10.26771/e-Revista.LOGO/2019.3.03

Tota11y, Accessibility Audit e Axe Developer apresentaram a menor quantidade total de problemas identificados (avisos + erros), com e sem repetições. AInspector, Accessibility Checker e Code Sniffer apresentaram a maior quantidade total de problemas identificados com repetições. Se desconsiderarmos as repetições, o eScanner supera o Code Sniffer pela quantidade de avisos. A comparação fica muito similar quando analisamos apenas a quantidade total de erros. Tota11y, Accessibility Audit e Wave Evaluation retornaram a menor quantidade, com e sem repetições. AInspector, Accessibility Checker e Code Sniffer apresentaram a maior quantidade total de erros, com e sem repetições. Quando analisamos o total de avisos, os extremos se destacam. Tota11y não apresenta nenhum aviso em nenhuma página. AInspector retorna uma quantidade muito superior de avisos, mais de três vezes a quantidade do segundo colocado. Quando se compara as proporções de avisos e erros reportados, Tota11y, Axe Developer e Code Sniffer apresentam bem mais erros do que avisos. O contrário ocorre com Accessibility Audit, eScanner e Wave Evaluation. Poucos avaliadores apresentaram um equilíbrio entre erros e avisos (40%-60%): AInspector com repetição e Wave Evaluation sem repetição.

Figura 2. Quantidade total de avisos e erros encontrados em cada avaliador automático de acessibilidade.




Fonte: Elaborado pelos Autores.

A comparação do total de problemas encontrados pode criar uma expectativa de que comportamento semelhante acorreu em todas as páginas analisadas. A Figura 3 estratifica a quantidade de erros e avisos por cada página, com (esquerda/acima) e sem (direita/abaixo) repetições no HTML. É interessante observar que a ordenação dos avaliadores

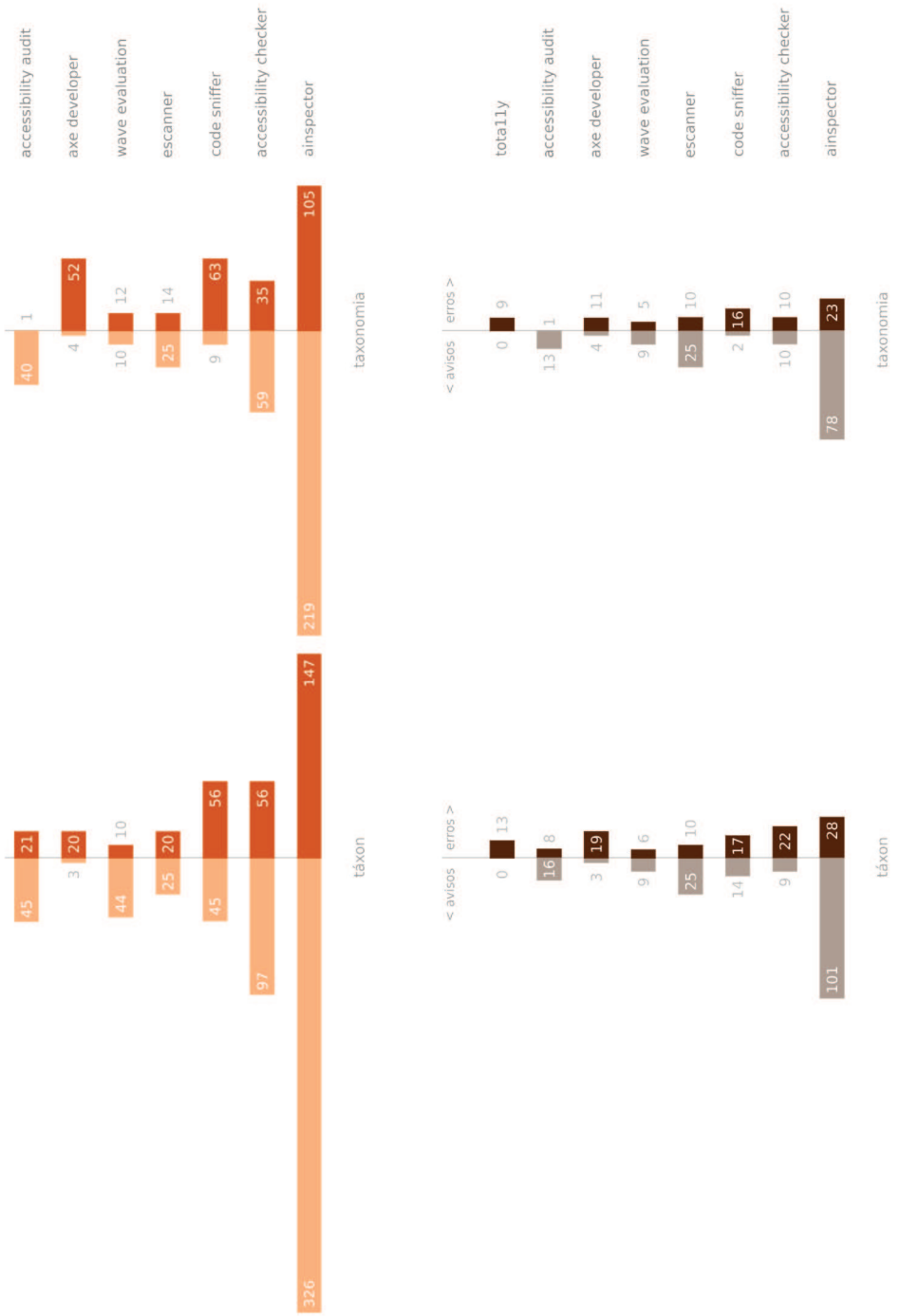

e-Revista LOGO - v.8, n.3, 2019 - ISSN 2238-2542
http://doi.org/10.26771/e-Revista.LOGO/2019.3.03

por quantidade de erros e de avisos variou consideravelmente em todas as páginas, com ou sem repetições no HTML. Apesar de não terem sido resultados uniformes, observamos certos comportamentos relevantes. AInspector apresentou a maior quantidade de erros na maioria das páginas, exceto em login. Code Sniffer e Accessibility Checker alternaram-se em segundo e terceiro com maior quantidade de erros. Axe Developer assumiu um quarto lugar expressivo em quantidade de erros na maioria das páginas.

Apenas quatro avaliadores indicaram explicitamente a diretriz violada nos problemas relatados. A Figura 4 apresenta o total de avisos (esquerda) e erros (direita) de cada avaliador, agrupados pelos princípios da WCAG, boas práticas e ARIA. A grande maioria dos erros identificados foi de percepção, seguidos pelos de compreensão, robustez e operação.

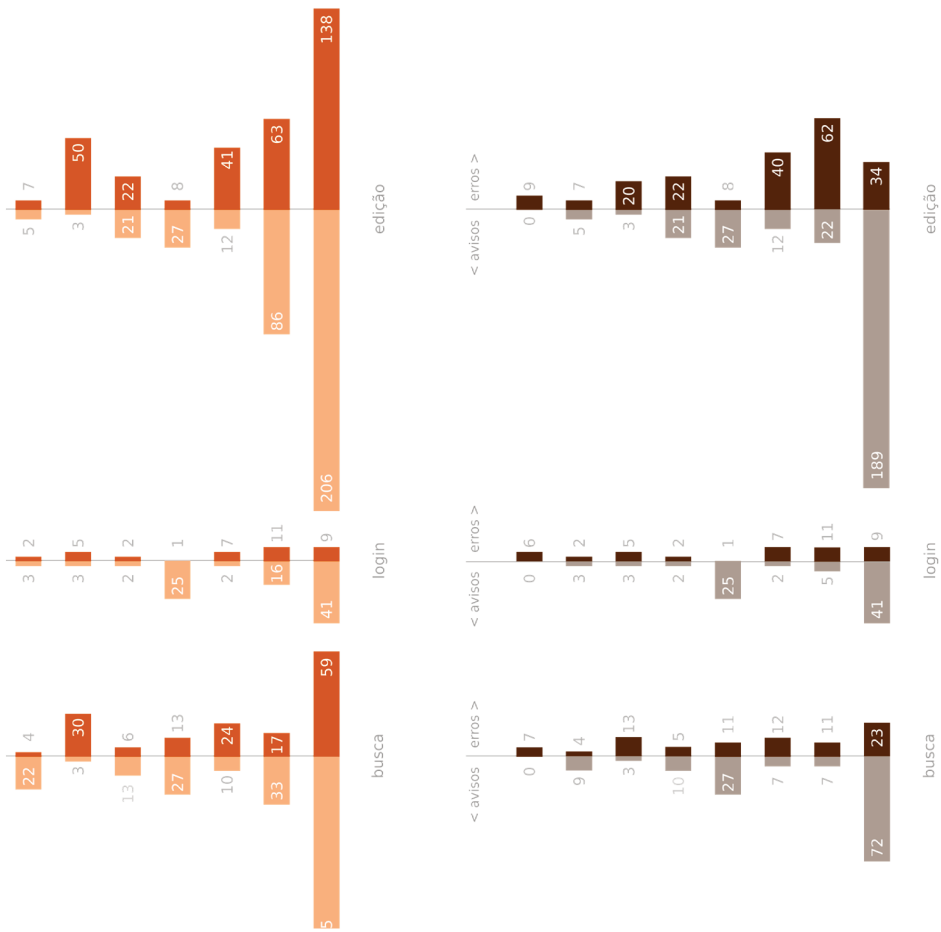

e-Revista LOGO - v.8, n.3, 2019 - ISSN 2238-2542
http://doi.org/10.26771/e-Revista.LOGO/2019.3.03

Figura 3. Quantidade de avisos e erros encontrados por página em cada avaliador






e-Revista LOGO - v.8, n.3, 2019 - ISSN 2238-2542
http://doi.org/10.26771/e-Revista.LOGO/2019.3.03



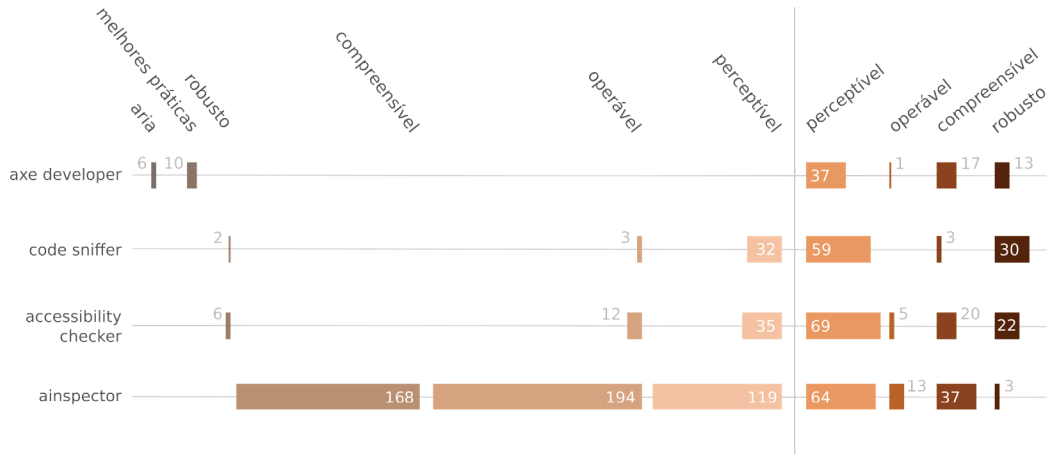

Fonte: Elaborado pelos autores.

Accessibility Checker destacou-se em erros de percepção. AInspector em erros de operação e compreensão. Code Sniffer em erros de robustez. A maioria dos avisos foi sobre operação, seguidos por percepção, compreensão, melhores práticas, robustez e ARIA. AInspector destacou-se em avisos sobre percepção, operação e compreensão. Accessibility Checker em avisos sobre robustez. Axe Developer foi o único com avisos sobre melhores práticas e ARIA (Figura 4).



e-Revista LOGO - v.8, n.3, 2019 - ISSN 2238-2542
http://doi.org/10.26771/e-Revista.LOGO/2019.3.03

Figura 4. Quantidade de avisos e erros encontrados por princípio em cada avaliador.




Fonte: Elaborado pelos autores.

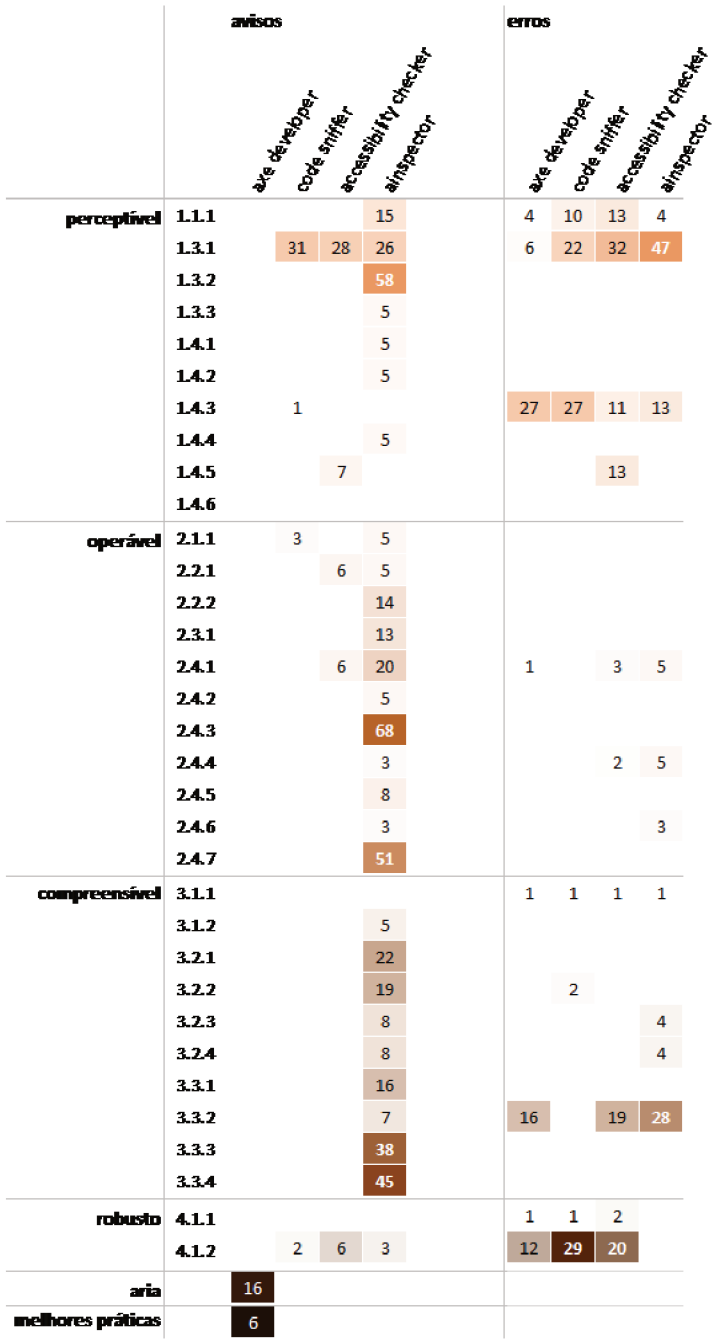
A Figura 5 estratifica por critérios da WCAG a quantidade de erros e avisos encontrados por estes quatro avaliadores. Os critérios 1.3.1 (relação entre informações) e 1.4.3 (contraste) concentraram a maior quantidade de erros de percepção. O critério 2.4.1 (ignorar blocos) apresentou maior quantidade de erros de operação. O critério 3.3.2 (instruções de preenchimento) destaca-se nos erros de compreensão. Quase todos os erros de robustez ferem o critério 4.1.2 (elementos de interface bem definidos). Houve variação significativa na quantidade de erros que violam cada critério entre avaliadores, evidenciando diferenças eles. AInspector foi o avaliador que apresentou a grande maioria dos avisos, bem distribuídos em muitos critérios da WCAG. Apenas nos critérios 1.3.1 e 4.1.2 é que os avisos do Code Sniffer, Accessibility Checker e AInspector se comparam.

As Figuras 6, 7, 8, 9 e 10 comparam o local de ocorrência e a quantidade de erros e avisos em cada página analisada. Cada local foi identificado por um número sequencial com um compartimento superior para os erros e inferior para os avisos. A quantidade de erros e avisos é representada por um círculo preenchido contendo uma letra. Quanto maior for a quantidade, maior será o diâmetro do círculo e mais escura será sua cor de preenchimento.



e-Revista LOGO - v.8, n.3, 2019 - ISSN 2238-2542
http://doi.org/10.26771/e-Revista.LOGO/2019.3.03

Figura 5. Quantidade de avisos e erros encontrados por critério em cada avaliador.



| | | avisos | | | | erros | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | axe developer | code sniffer | accessibility checker | ainspector | axe developer | code sniffer | accessibility checker | ainspector |
| perceptível | 1.1.1 | | | 15 | | 4 | 10 | 13 | 4 |
| | 1.3.1 | 31 | 28 | 26 | | 6 | 22 | 32 | 47 |
| | 1.3.2 | | | 58 | | | | | |
| | 1.3.3 | | | 5 | | | | | |
| | 1.4.1 | | | 5 | | | | | |
| | 1.4.2 | | | 5 | | | | | |
| | 1.4.3 | 1 | | | | 27 | 27 | 11 | 13 |
| | 1.4.4 | | | 5 | | | | | |
| | 1.4.5 | | 7 | | | | | 13 | |
| | 1.4.6 | | | | | | | | |
| operável | 2.1.1 | 3 | | 5 | | | | | |
| | 2.2.1 | | 6 | 5 | | | | | |
| | 2.2.2 | | | 14 | | | | | |
| | 2.3.1 | | | 13 | | | | | |
| | 2.4.1 | | 6 | 20 | | 1 | | 3 | 5 |
| | 2.4.2 | | | 5 | | | | | |
| | 2.4.3 | | | 68 | | | | | |
| | 2.4.4 | | | 3 | | | | 2 | 5 |
| | 2.4.5 | | | 8 | | | | | |
| | 2.4.6 | | | 3 | | | | | 3 |
| | 2.4.7 | | | 51 | | | | | |
| compreensível | 3.1.1 | | | | | 1 | 1 | 1 | 1 |
| | 3.1.2 | | | 5 | | | | | |
| | 3.2.1 | | | 22 | | | | | |
| | 3.2.2 | | | 19 | | | 2 | | |
| | 3.2.3 | | | 8 | | | | | 4 |
| | 3.2.4 | | | 8 | | | | | 4 |
| | 3.3.1 | | | 16 | | | | | |
| | 3.3.2 | | | 7 | | 16 | | 19 | 28 |
| | 3.3.3 | | | 38 | | | | | |
| | 3.3.4 | | | 45 | | | | | |
| robusto | 4.1.1 | | | | | 1 | 1 | 2 | |
| | 4.1.2 | 2 | 6 | 3 | | 12 | 29 | 20 | |
| aria | | 16 | | | | | | | |
| melhores práticas | | 6 | | | | | | | |

Fonte: Elaborado pelos autores.

Os avaliadores automáticos foram identificados por letras dentro dos círculos em ordem crescente do total de problemas (erros + avisos) encontrados, como segue:

**a -** Totta11y

**b -** Accessibility Audit



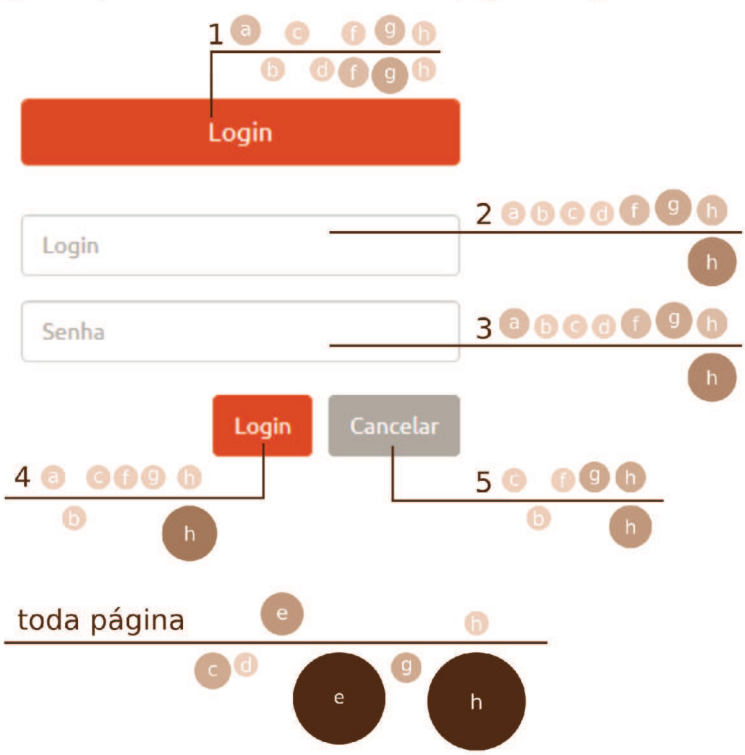
e-Revista LOGO - v.8, n.3, 2019 - ISSN 2238-2542
http://doi.org/10.26771/e-Revista.LOGO/2019.3.03

**c -** Axe Developer
**d -** Wave Evaluation
**e -** eScanner
**f -** Code Sniffer
**g -** Accessibility Checker (improve)
**h –** Ainspector

Para facilitar a comparação entre erros e avisos do mesmo avaliador, reservou-se uma "coluna" (um espaço na vertical) para cada avaliador. "Célula" (espaço) vazia indica que aquele avaliador só apresentou apenas erros ou apenas avisos naquele local.

A Figura 6 apresenta a quantidade de erros e avisos na página de login. Essa visualização evidencia várias diferenças relevantes. Por exemplo, cinco avaliadores apontaram erros no local 4, mas apenas dois relataram avisos e o avaliador (e) eScanner não retornou nem erros, nem avisos, neste local. Houve certa variação na quantidade de erros e avisos apresentados pelos avaliadores no local 1. Os avaliadores (e) eScanner e (h) AInspector retornaram um número muito significativo de avisos em toda a página, apesar de apenas o (h) ter retornado poucos erros neste local.

Figura 6. Quantidade de erros e avisos na página de login.

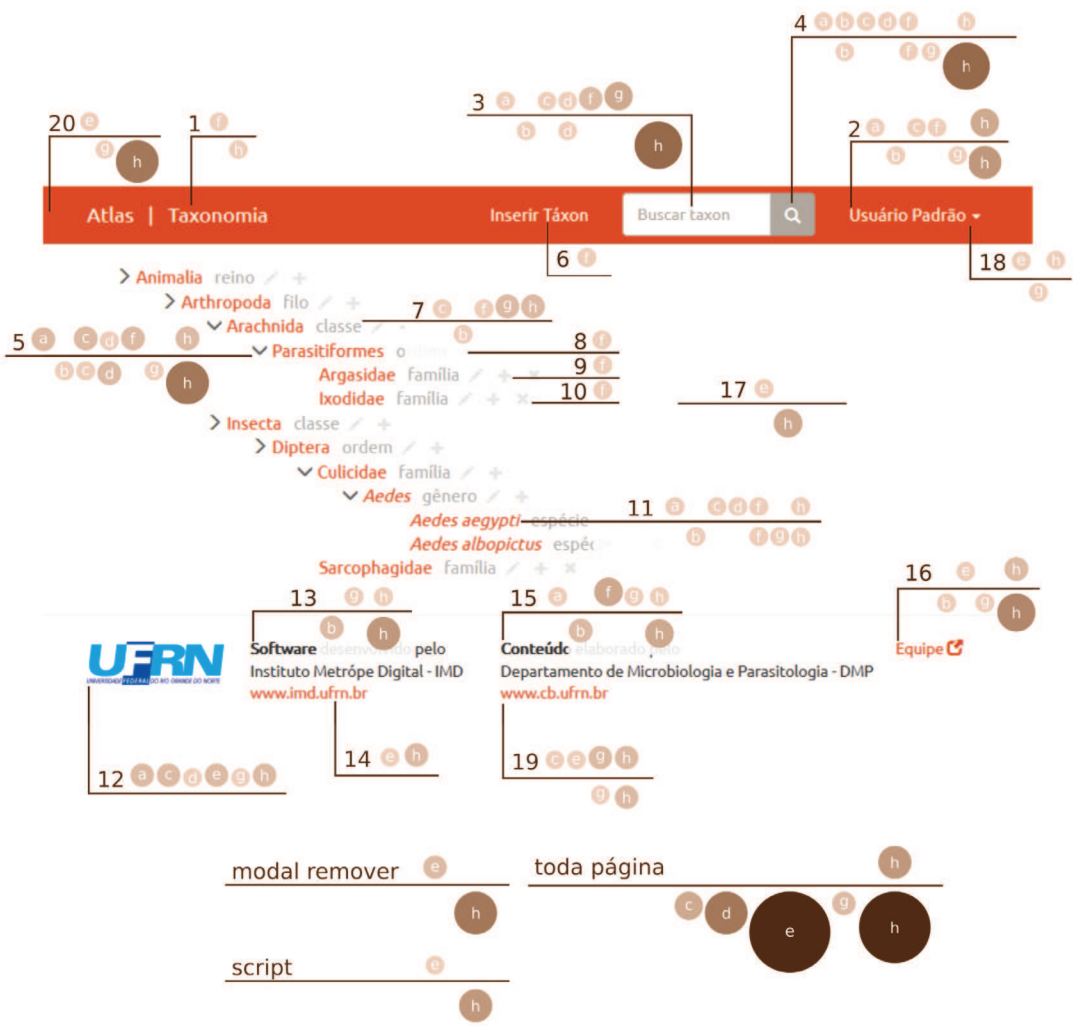



Fonte: Elaborado pelos Autores.


e-Revista LOGO - v.8, n.3, 2019 - ISSN 2238-2542
http://doi.org/10.26771/e-Revista.LOGO/2019.3.03

A Figura 7 apresenta a quantidade de erros e avisos na página de visualização da taxonomia. Aqui também a diversidade de resultados fica evidente. Por exemplo, em alguns locais apenas um avaliador encontrou erros, como nos locais 8, 9 e 10 pelo avaliador (f) Code Sniffer. Já em outros locais a maioria dos avaliadores apresentou erros como em 4 e 12. Também houve diferença entre quantidade dos erros encontrados, como no local 3 onde os avaliadores (a), (c) e (d) retornaram a mesma quantidade de erros, porém menor do que (f), que por sua vez também foi menor do que (g). Houve situações onde um avaliador encontrou erro e não avisos e vice-versa, como (f) e (h) em 1.

Figura 7. Quantidade de erros e avisos na página de taxonomia.



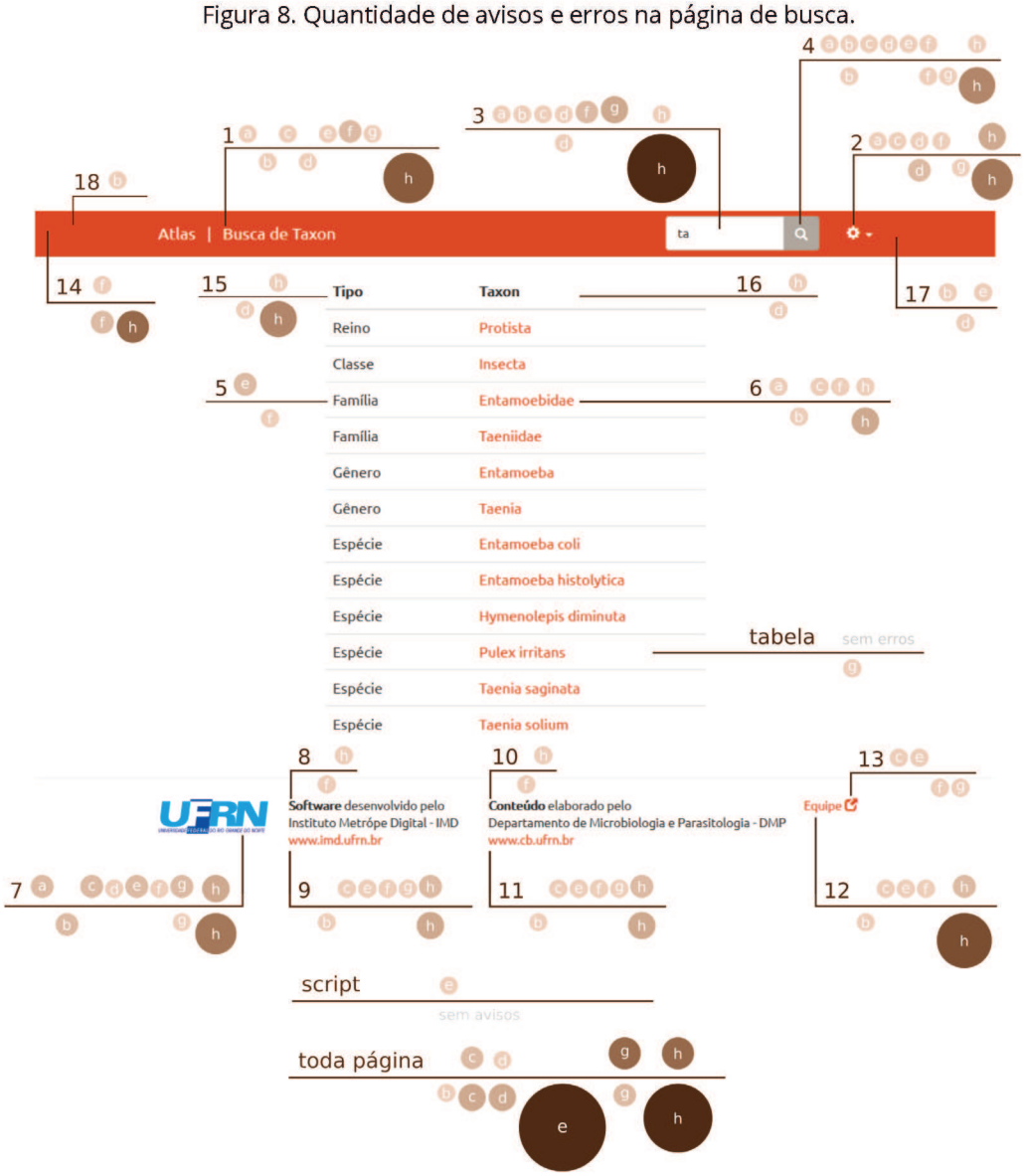

Fonte: Elaborado pelos autores.



e-Revista LOGO - v.8, n.3, 2019 - ISSN 2238-2542
http://doi.org/10.26771/e-Revista.LOGO/2019.3.03

A Figura 8 também segue a diversidade de resultados apresentados pelos avaliadores automáticos distribuídos pelos locais da página. Contudo, vale reparar que apesar de existir alguma variação na quantidade de erros entre os avaliadores, ela não foi muito significativa (pouca variação de tamanho entre círculos na parte superior). Já a quantidade de avisos variou bastante entre avaliadores, como ocorreu em partes do código que referenciava toda página de taxonomia. Aqui o avaliador (b) retornou quantidade bem pequena de avisos e o avaliador (e) retornou quantidade muito maior de avisos.

Figura 8. Quantidade de avisos e erros na página de busca.



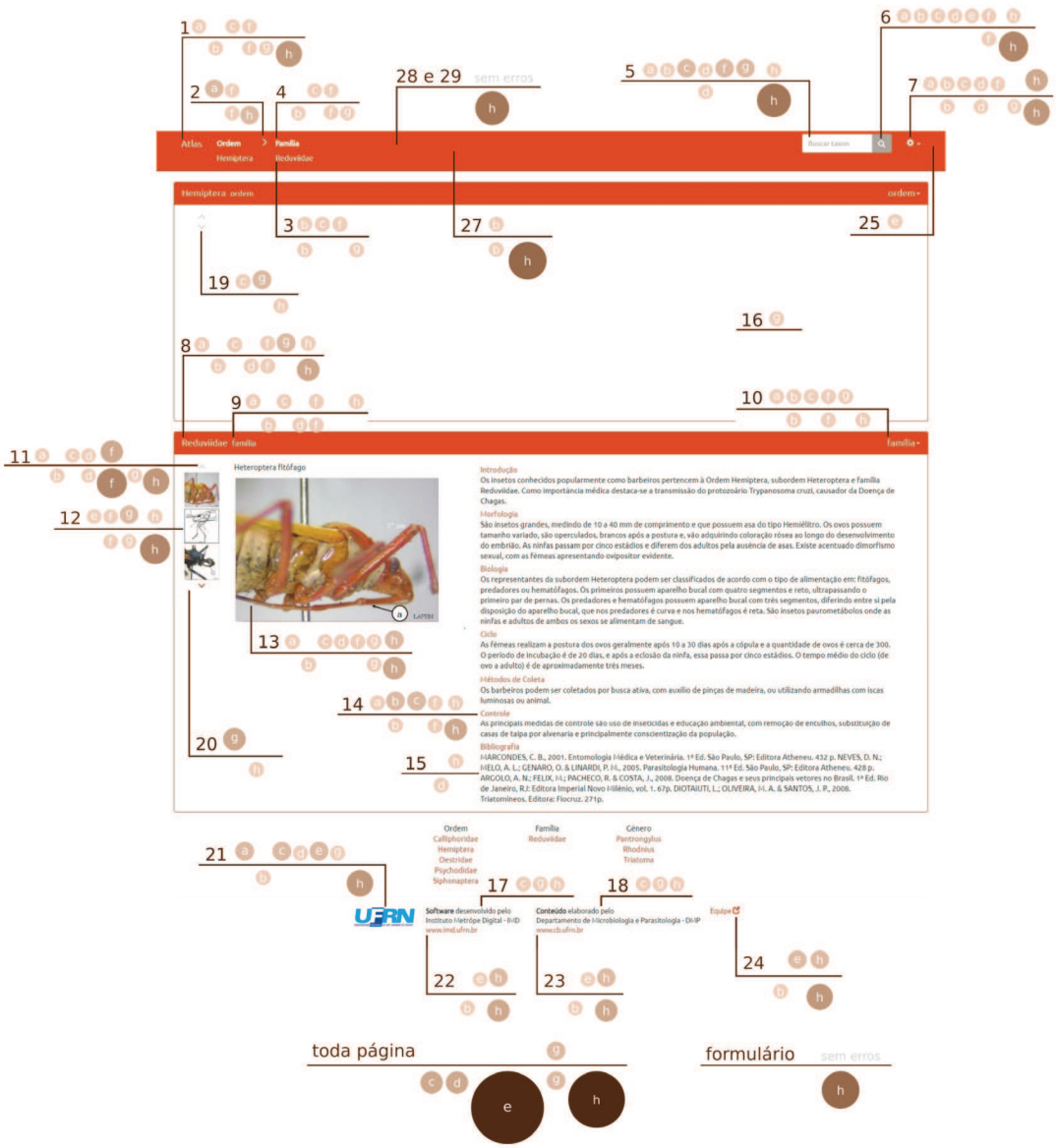

Fonte: Elaborado pelos autores.



e-Revista LOGO - v.8, n.3, 2019 - ISSN 2238-2542
http://doi.org/10.26771/e-Revista.LOGO/2019.3.03

Os resultados apresentados na página de consulta de táxon (Figura 9) e de edição de táxon (Figura 10) apresentaram o mesmo comportamento diversificado em relação à quantidade de erros e avisos encontrados pelos avaliadores em cada local da página.

Figura 9. Quantidade de avisos e erros na página de táxon.




Fonte: Elaborado pelos autores.



e-Revista LOGO - v.8, n.3, 2019 - ISSN 2238-2542
http://doi.org/10.26771/e-Revista.LOGO/2019.3.03

## 6 Considerações finais

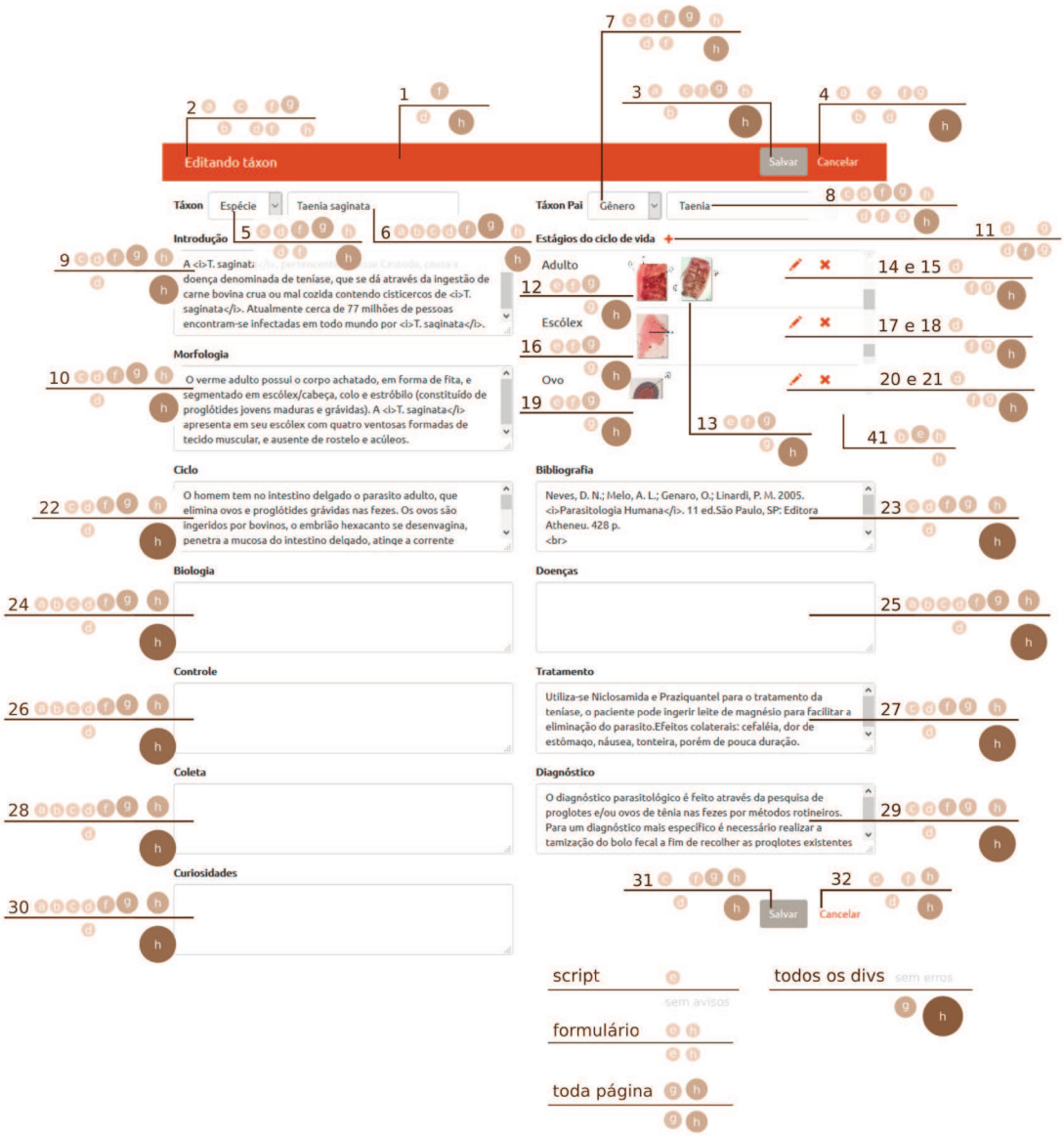
Este artigo comparou os problemas relatados por oito avaliadores automáticos de acessibilidade web que são extensão de navegadores, tendo em vista a avaliação formativa. Considerando que WCAG e eMAG têm conteúdos semelhantes e que todos os avaliadores foram construídos com base nestas diretrizes, é razoável imaginar que os resultados apresentados por estes softwares sejam similares, a ponto de se diferenciarem principalmente pela usabilidade e as funcionalidades que oferecem. As poucas diferenças imaginadas entre os resultados geralmente dizem respeito à quantidade total de problemas. Este estudo trouxe novos elementos a essa discussão com uma análise aprofundada dos problemas de acessibilidade reportados.

Os oito avaliadores analisados apresentaram resultados com diferenças importantes não apenas na quantidade total de problemas reportados, mas também nos tipos de problemas (erro ou avisos), nas diretrizes violadas, bem como nos locais da interface em que ocorreram os problemas. Cada avaliador se destacou por aspectos diferentes, sendo relevante empregar mais de um deles na avaliação formativa para uma cobertura mais ampla. A quantidade total de problemas nem sempre é mais importante porque pode mascarar os diferentes tipos de verificação realizados. Os três avaliadores mais promissores foram AInspector, Accessibility Checker e Code Sniffer.

Outros estudos semelhantes devem ser realizados no futuro com outros sistemas web para ampliar a compreensão sobre os resultados oferecidos por essas ferramentas. Também é relevante analisar os resultados apresentados por outros avaliadores automáticos que funcionam com sistemas web na internet.


e-Revista LOGO - v.8, n.3, 2019 - ISSN 2238-2542
http://doi.org/10.26771/e-Revista.LOGO/2019.3.03

Figura 10. Quantidade de avisos e erros na página de edição.




Fonte: Elaborado pelos autores.



e-Revista LOGO - v.8, n.3, 2019 - ISSN 2238-2542
http://doi.org/10.26771/e-Revista.LOGO/2019.3.03

## Referências

BACH, C.F. FERREIRA, S.B.L.; SILVEIRA, D.S.; NUNES, R.R. Diretrizes de Acessibilidade: Uma Abordagem Comparativa entre WCAG e e-Mag. **RESI: Revista Eletrônica de Sistemas de Informação**, v. 8, p. 1-14, 2009.

Barbosa, S.D.J.; Silva, B.S. **Interação Humano-Computador.** Rio de Janeiro: RJ, Elsevier, 2010.

BRASIL. Presidência da República. Lei nº 13.146, de 6 de julho de 2015. **Institui a Lei Brasileira de Inclusão da Pessoa com Deficiência**. Disponível em: <http://www.planalto.gov.br/ccivil_03/ _ato2015-2018/2015/lei/l13146.htm>. Acesso em junho de 2019.

EMAG. **Modelo de Acessibilidade em Governo Eletrônico** (eMAG) 3.1. 2014. Disponível em: <http://emag.governoeletronico.gov.br/ >. Acesso em junho de 2019.

FREITAS, R.F.; COUTINHO, S.G.; NÓBREGA WAECHTER, H. Análise de Metodologias em Design: a informação tratada por diferentes olhares. **Estudos em Design,** 21(1), p.1-15, 2013.

FUERTES, J.L.; GONZÁLEZ, R., GUTIÉRREZ, E.; MARTÍNEZ, L. Hera-FFX: a Firefox add-on for semi-automatic web accessibility evaluation. In **Proceedings of the 2009 International Cross-Disciplinary Conference on Web Accessibililty** (W4A) (W4A '09). ACM, New York, NY, USA, pp. 26-35, 2009.

LICHESKI, L.C.; FADEL, L.M. (In) acessibilidade digital. InfoDesign-**Revista Brasileira de Design da Informação,** 10(2), 104-122, 2013.

MARQUES, V.G.; SILVA, B.S.; GAMA, R.A. Atlas Virtual de Parasitologia e Entomologia. In: **Anais do Congresso Regional sobre Tecnologias na Educação** (Ctrl+E 2016), 2016.

MEZZAROBA, M. P.; DE ALMEIDA, T. C., ULBRICHT, V.R., VANZIN, T.; FADEL, L.M. Acessibilidade em portais de Governo Eletrônico do Poder Judiciário. **InfoDesign-Revista Brasileira de Design da Informação**, 13(1), 93-106, 2016.



e-Revista LOGO - v.8, n.3, 2019 - ISSN 2238-2542
http://doi.org/10.26771/e-Revista.LOGO/2019.3.03

PACHECO, H.S.; AMORIM, P.F.; BARBOSA, P.G.; FERREIRA, S.B. Comparative Analysis of Web Accessibility Evaluation Tools. In Proceedings of the **15th Brazilian Symposium on Human Factors in Computing Systems**. ACM, 2016.

PIVETTA, E.M.; FLOR, C.; SAITO, D.S.; ULBRICHT, V.R. (2013). Analysis of an automatic accessibility evaluator to validate a virtual and authenticated environment. **International Journal of Advanced Computer Science and Applications** (IJACSA), Vol. 4, No.4, pp. 15-22.

SOUZA, E.R.; MONT'ALVÃO, C. Web accessibility: evaluation of a website with different semi-automatic evaluation tools. **Work**, 41(Supplement 1), p. 1567-1571, 2012.

SOUZA, E.; MALHEIROS, N. Avaliação de Acessibilidade Digital para Pessoas com Deficiência Motora em Repositórios Educacionais Abertos. **Revista Brasileira de Informática na Educação** – RBIE, 26(03), 1, 2018.

W3C-BRASIL. **Cartilha de Acessibilidade na Web** - W3C Brasil. 2013. Disponível em: <http://www.w3c.br/pub/Materiais/PublicacoesW3C/cartilha-w3cbr-acessibilidade-web-fasciculo-I.html>. Acesso em junho de 2019.

WCAG2. **Web Content Accessibility Guidelines** (WCAG) 2.0. 2008. Disponível em: <https://www.w3.org/Translations/WCAG20-pt-br/ >. Acesso em junho de 2019.

YIN R.K. **Case study research. Design and methods,** 3rd editon. London: SAGE Publications, 2003.

## Jefferson Ferreira Ribeiro

Granduando em Tecnologia da Informação pela Universidade Federal do Rio Grande do Norte. Tem interesse por acessibilidade digital e desenvolvimento web.

## Bruno Santana da Silva

É bacharel em Ciência da Computação pela Universidade Federal Fluminense (2003), mestre (2005) e doutor (2010) em Informática pela Pontifícia Universidade Católica do Rio de Janeiro. Desde 2012 é professor no Instituto Metrópole Digital da Universidade Federal do Rio Grande do Norte. Atua no Programa de Pós-graduação em Design da mesma universidade. Tem experiência nas áreas de Informática e Design, com ênfase no projeto, avaliação e desenvolvimento de interfaces com usuário de sistemas computacionais interativos, atuando principalmente nos seguintes temas: interação humano-computador, design de interfaces, acessibilidade digital e ergonomia. Colabora e coordena projetos de pesquisa e desenvolvimento com forte caráter multidisciplinar, envolvendo áreas como Tecnologia da Informação, Design, Biologia, Enfermagem, Fonoaudiologia, Educação, dentre outras.


**Recebido em: Setembro, 2019**
**Aceito em: Dezembro, 2019**



e-Revista LOGO - v.8, n.3, 2019 - ISSN 2238-2542
http://doi.org/10.26771/e-Revista.LOGO/2019.3.03