Revista brasileira de iniciação científica
ISSN: 2359-232X

VISUALIZAÇÃO DE DADOS CIENTÍFICOS NA WEB: UMA REVISÃO DE RECURSOS SEMÂNTICOS BASEADA EM DESIGN CENTRADO NO USUÁRIO

SCIENTIFIC DATA VISUALIZATION IN THE WEB: A DESIGN-CENTERED USER REVISION OF SEMANTIC RESOURCES.

VISUALIZACIÓN DE DATOS CIENTÍFICOS EN LA WEB: UNA REVISIÓN DE RECURSOS SEMÁNTICOS BASADA EN UN DISEÑO CENTRADO EN EL USUARIO

Paulo Henrique Vieira Cândido[1]
Carlos Henrique da Silva-Santos[2]

Resumo: O desenvolvimento de métodos numéricos pela computação científica tem requerido novas formas de interação na Web. Neste sentido, a disponibilidade de variados recursos para visualização numérica dos dados gerados por estes métodos pode dificultar o trabalho dos desenvolvedores. Assim, este trabalho apresenta uma análise baseada na praticidade de implementação e interação do usuário final com as bibliotecas Plotly, ApexCharts e C3 js, escolhidas utilizando o ranking do Github por *stars* e *issues*, analisando-as em três diferentes cenários para se verificar os resultados de interação e os recursos providos por cada uma delas em aplicações gráficas bidimensionais.
Palavras-chave: Visualização de Dados Científicos. Adaptação de Interfaces. Desenvolvimento Web. Métodos numéricos. Experiência do Usuário (UX).

[1] Graduando em Licenciatura em Física. Instituto Federal de Educação, Ciência e Tecnologia de São Paulo, IFSP, campus Itapetininga. ORCID: https://orcid.org/0000-0002-2333-9520. E-mail: phvcandido@gmail.com.
[2]Doutor em Engenharia Elétrica pela Universidade Estadual de Campinas (Unicamp). Docente de Informática no IFSP, campus Itapetininga. ORCID: https://orcid.org/0000-0002-8786-405X. E-mail: carlos.santos@ifsp.edu.br

Revista brasileira de iniciação científica
ISSN: 2359-232X

Abstract: The computational scientific development of numerical methods has required new interaction forms based on Web technologies. In this context, the availability of various visualization resources for numerical data generated by these methods could difficult the developers' work. Therefore, this work aims to present analysis upon the development practice and users' interaction with the Libraries Plotly, ApexCharts and C3 js, which was chosen by Github search being ranked by stars and questions. They were analyzed in three different scenarios to verify the interactions results and computational approaches allowed for two-dimensional graphic applications.
Keywords: Scientific Data Visualization. Interfaces Adaptability. Web Development. Numerical Methods. User Experience (UX).

Resumen: El desarrollo de métodos numéricos por parte de los científicos ha requerido nuevas formas de interacción en la Red. En este sentido, la disponibilidad de diversos recursos para la visualización numérica de los datos generados por estos métodos puede dificultar el trabajo de los desarrolladores. Así, este trabajo presenta un análisis basado en la practicidad de implementación e interacción del usuario final con las Libraries Plotly, ApexCharts y C3 js, elegidas utilizando el ranking de Github por estrellas y preguntas, analizandolas en tres escenarios diferentes para verificar los resultados de interacción y los recursos que aporta cada uno de ellos en aplicaciones gráficas bidimensionales.
Palabras-clave: Visualización de datos científicos. Adaptabilidad de interfaces. Desarrollo web. Métodos numéricos. Experiencia de usuario (UX).

Submetido 15/01/2021 **Aceito 10/03/2021** **Publicado 11/03/2021**

Revista brasileira de iniciação científica
ISSN: 2359-232X

Introdução

A computação científica, que parte do desenvolvimento e aplicação de métodos numéricos para simulações nas diferentes áreas do conhecimento, tem requerido recursos computacionais cada vez mais sofisticados e diversos para atender as demandas (Santos, 2010; Cândido; Santos, 2020).

Para exemplificar, em telecomunicações pode-se tomar as especificidades das áreas de análise e projeto de antenas, feixes ópticos ou novos materiais em que se tem diferentes simuladores que possibilitam tanto a modelagem por ferramentas CAD quanto à visualização de dados de campos eletromagnéticos processados pelos métodos numéricos deles (Santana et. al, 2014; Cândido et al., 2019).

Para isso, dois cenários são previstos, sendo que no primeiro os desenvolvedores devem desenvolver desde os códigos numéricos até os recursos de visualização dos dados, atentando-se aos padrões e recomendações disponibilizados, assim como recursos disponíveis para essas tarefas. No segundo cenário eles são usuários de software já prontos, como de simuladores em Cândido et. al (2020), Cândido e Santos (2021) e Santana et al. (2014) e precisam apenas conhecer sobre sua aplicação. Neste sentido, objetivando-se pela disponibilidade de mais recursos para atender a esses usuários, tem-se apenas o primeiro cenário como foco deste trabalho, a fim de facilitar o desenvolvimento destes recursos (Harris et al., 2020; Cândido et al., 2019).

No primeiro cenário e considerando a democratização científica e acesso a recursos computacionais, diferentes iniciativas para o desenvolvimento de ambientes de simulação como o Scilab[3], Octave[4] e linguagens de programação de mais alto nível e com variedade de recursos científicos como o R e Python, sendo que, segundo Bocherds (2007) e Harris et al. (2020), esta última linguagem possui interessantes recursos para o desenvolvimento de aplicações científicas, além de ser a segunda linguagem mais utilizada no Github (Github, 2021b).

O Python permite a integração com diferentes pacotes que fornecem ferramentas para realização de cálculos e visualizações numéricas, como Numpy e o Matplotlib. Este segundo, também chamado de Matplot apenas, permite a visualização gráfica de dados numéricos

[3] https://www.scilab.org/
[4] https://www.gnu.org/software/octave/index

Revista brasileira de iniciação científica
ISSN: 2359-232X

implementados para visualização gráfica dos dados de maneira mais simples, permitindo que o desenvolvedor consiga construir suas aplicações com maior liberdade para o uso final (Harris et al., 2020; Cândido et al., 2019).

Essas aplicações com código fonte construído pelos desenvolvedores são interessantes para aplicações isoladas, contudo, quando se pensa na livre distribuição de código e o uso desses métodos por usuários finais não familiarizados com código fonte e o uso de recursos para compilação e edição de código, suas aplicações se tornam mais complexas.

Assim, o uso desses recursos da internet como aplicações Web para o provimento de interface visual aos usuários pode ser uma alternativa interessante. Porém, atentando-se para os requisitos de *User Interface* (UI) e *User Experience* (UX), em que se tem alguns paradigmas que passam a ser apropriados de serem considerados tanto na perspectiva de usabilidade do sistema Web, quanto nos recursos a serem desenvolvidos (Lowdermilk, 2019; Bueno, 2017; Cândido; Bueno; Santos, 2020; Cândido et. al, 2021).

Essa importância se evidencia quando associados às Heurísticas de Nielsen, que permitem identificar e sanar possíveis problemas de interação na usabilidade de interfaces (Nielsen, 2005) no desenvolvimento de recursos oferecidos aos usuários e o suporte à sua autonomia ao visualizar esses dados e interagir com os recursos computacionais.

Para exemplificar, a própria biblioteca Matplotlib do Python por já construir gráficos para a visualização dos elementos da modelagem numérica, inclusive com gráficos interativos que são visualmente atualizados à medida que a solução numérica é executada, proporciona variabilidade de recursos interativos visuais aos usuários (Lowdermilk, 2019; Harris et al., 2020).

Portanto, a UI e a UX estão implicitamente relacionadas ao fluxo de construção de interfaces em que se tem como base o Design Centrado no Usuário (*User-centered Design* - UCD), onde ele é o personagem protagonista, podendo utilizá-la de forma autônoma e seu uso sendo analisado por especialistas em Interação Homem-Máquina (IHC).

Na área de IHC há especificidades de estudos quanto a adaptação de interfaces, a qual envolve tanto as interfaces adaptáveis quanto adaptativas, ou seja, são interfaces que se adaptam às opções do usuário e às formas de interação dele com o dispositivo (*Graphical-User-Interface* - GUI). Uma área de estudos que demonstra importância e aplicabilidade necessária para o

desenvolvimento de interfaces atualmente, como visto em análises de interfaces em Bueno (2017) e Bueno e Zaina (2017), e com aplicação em desenvolvimento em Cândido (2017), Cândido, Bueno e Santos (2020) para o layout de interfaces para desenvolvedores, Cândido et al. (2020), Cândido e Santos (2020) e Santana et al. (2014) para o desenvolvimento de sistemas, e no desenvolvimento de interfaces assistivas, como em Cândido et al. (2021).

Esse trabalho se justifica como uma continuidade dos trabalhos de Bueno e Zaina (2017), Cândido (2017), Cândido, Bueno e Santos (2020) e Jain (2014), buscando verificar os recursos disponibilizados por diferentes bibliotecas e seu impacto na Interação Humano-Computador (IHC) com ênfase na adaptação de interfaces em pacotes e Frameworks CSS para desenvolvimento Web Front-end, acrescentando continuidade também aos trabalhos de Cândido et. al (2019) e Cândido e Santos (2020), onde foram desenvolvidas ferramentas de visualizações de dados científicos baseados em métodos numéricos, também sendo explorados como forma de agregar mais informações e parâmetros para análises dos pacotes aqui utilizados.

Contudo, neste trabalho intenta-se verificar o uso de pacotes para desenvolvimento de visualizações numéricas na Web. Para isso, avaliações das perspectivas de usuário final e desenvolvedores de software são apresentadas, sendo que para o usuário final considerou-se a diversidade de recursos oferecidos e como eles podem interagir com os gráficos. Da perspectiva do desenvolvedor consideram-se ambientes, a praticidade de codificação e reuso de código. Assim, foram determinados objetivos específicos quanto a se verificar possíveis contribuições, como a busca e classificação dos pacotes de desenho de gráficos de dados científicos para visualização na Web, análise de suas características, disponibilização de exemplos de implementação e análise descritiva das suas funcionalidades e recursos.

Para apresentar essa variedade de estudos e análises, este trabalho inicia-se com a seção de “Fundamentação Teórica”, onde são abordados os conceitos referentes ao desenvolvimento de interfaces. Esses conceitos são explorados na seção seguinte de “Metodologia”, onde é apresentada uma pesquisa na literatura no mesmo tema aqui abordado, tópicos que levaram à escolha dos pacotes, e, por conseguinte reutilizados na seção de Análise dos Pacotes, onde serão realizadas as descrições dos pontos observados durante a “Análise dos Pacotes” e que levam às “Considerações Finais” e “Trabalhos Futuros” deste.

Revista brasileira de iniciação científica
ISSN: 2359-232X

Fundamentação Teórica

Para uma melhor definição dos parâmetros a serem utilizados, bem como as tecnologias, conceitos, metodologia e análises a serem implementadas, buscou-se por trabalhos já desenvolvidos nesta área. Para tanto, utilizou-se os mecanismos de busca do Google, o Google Acadêmico (*Google Scholar*[5]), e o da *Association for Computing Machinery* (ACM), o ACM Digital Library (ACM-DL).

Foi buscado pelos termos “visualização numérica” AND “web” no Google Acadêmico, e o retorno foi de 33 resultados. Com os termos em inglês (“numeric visualization” AND “web”), o retorno foi de 38 resultados. Buscando na ACM-DL, utilizando como query de busca os mesmos parâmetros anteriores (AllField:(("numeric visualization" AND "web") OR ("visualização numérica" AND "web"))) obtiveram-se 5 resultados.

Buscando cercear mais essas buscas, adicionou-se o termo “científic*”, tornando o termo de busca em português "visualização numérica" AND "web" AND 'científic*', com 26 resultados e em inglês “numeric visualization” AND “web” AND "scientific", com 12 resultados. O mesmo processo na ACM-DL, com a query AllField:(("numeric visualization" AND "web" AND "scientific") OR ("visualização numérica" AND "web" AND "científic*")), teve como resultado 1 artigo. Todas as pesquisas nessas bases de dados foram realizadas no dia 22 de fevereiro de 2021.

Com esses resultados encontrados no Google Acadêmico e na ACM-DL, pôde-se perceber a disponibilização de diferentes temas vinculados à visualização numérica científica, sendo que dentre os 39 resultados encontrados na última pesquisa, nenhum se mostrou com alguma relação direta com o assunto a ser tratado neste trabalho.

Quando algum dos artigos encontrados apresentavam alguma linha de desenvolvimento que possuísse alguma relação com o assunto, tratava-se do desenvolvimento de alguma ferramenta para visualização de dados, como o caso de Mcnaney (2016), que aborda o desenvolvimento de uma ferramenta para automonitoramento de pessoas com Mal de Parkinson e utiliza recursos de visualização de dados para seu desenvolvimento, ou Gebremeskel, Hailu e Biazen (2019), focando na realização de análises de dados clínicos com mineração de dados e na modelagem e visualização dos dados. Ambos os artigos fogem do que é abordado aqui.

[5] https://scholar.google.com/

Revista brasileira de iniciação científica
ISSN: 2359-232X

Isso posto, considerando os resultados obtidos, organiza-se aqui uma breve revisão de fundamentos teóricos da área de desenvolvimento Web, atentando-se aos conceitos que são apresentados na seção de Análise dos Pacotes. Assim, as próximas subseções desta trarão os i) Conceitos Básicos da Organização do Desenvolvimento Web e ii) Construção de Interfaces e o uso de Gerenciador de Pacotes.

Conceitos Básicos da Organização do Desenvolvimento Web

No desenvolvimento Web existem diferentes conceitos a serem estudados, variando entre a estrutura de funcionamento de um servidor baseado em processamento assíncrono e banco de dados (Tilkov; Vinoski, 2010; Patil, 2017; Mrozek, 2020) e o desenvolvimento de interfaces e aplicação de tecnologias para navegadores (Shi, 2020; Kyriakidis; Maniatis, 2016; Ellwanger Et. Al, 2015; Bueno; Zaina; 2017).

Partindo desse pressuposto, tratar da construção de interfaces, ou seja, do desenvolvimento dos elementos visuais de um sistema, como botões, fontes de letra, cores e organizações visuais, demanda de pesquisas e tecnologias específicas para esse desenvolvimento, enquanto que no processamento (Tilkov; Vinoski, 2010), armazenamento de dados (Patil et. al, 2017) e controle de fluxos (Kyriakidis; Maniatis, 2016), outras características, como a sensibilidade dos dados, a configuração e segurança do banco de dados, a estrutura e tecnologias do servidor, entre outros fatores tornam-se necessários de serem considerados na parametrização dos requisitos de interface (Bueno, 2017; Cândido, 2017; Lowdermilk, 2019; Tilkov; Vinoski, 2010).

Basicamente há a macro divisão conceitual nomeada back-end e front-end. O processamento de dados, armazenamento de dados e execução de serviços, ou seja tudo aquilo que o usuário não vê diretamente, dá-se o nome Back-end. Por outro lado, tudo aquilo que o usuário vê e interage, dá-se o nome Front-end, sendo este último foco neste trabalho (Kyriakidis; Maniatis, 2016; Cândido, Bueno; Santos, 2020; Lowdermilk, 2019).

Construção de Interfaces e o uso de Gerenciador de Pacotes

A interface do sistema é o meio com que o usuário se “relaciona” com ele e, por consequência, possui diversas abordagens e estudos que criam padrões para a sua estruturação.

Revista brasileira de iniciação científica
ISSN: 2359-232X

O termo interface associa-se à conexão entre dois pontos, ou seja, a interface do sistema é o que faz a conexão entre os recursos e serviços buscados por um usuário e as tecnologias e recursos oferecidos por um sistema, estado associado à área de Interação Humano-Computador (IHC), pela especificidade de Adaptação de Interfaces. Essa especificidade é requerida quando se considera o avanço tecnológico quanto ao surgimento de diferentes dispositivos como, por exemplo, smartphones com tamanhos de tela variados, notebooks com telas de toque, smartwatches e outros tantos dispositivos (Bueno, 2017; Cândido, Bueno; Santos, 2020).

Ao considerar a Web como um ambiente acessível a todos, a adaptação de interfaces coloca em questão o uso do mesmo ambiente em diferentes dispositivos e por diferentes usuários, subdividindo essa área de estudos em dois novos grupos, chamados de interfaces adaptáveis e adaptatividade. Na primeira, as interfaces adaptáveis associam-se à adaptação da interface às opções do usuário, como cores, posicionamento e funcionamento, caracterizando uma personalização por parte do usuário. No segundo grupo tem-se adaptatividade como sendo requisitos ponderados para que a interface se adeque a diferentes dispositivos, múltiplas formas de interação e suporte à sensibilidade ao contexto de execução dos dispositivos em que estão sendo executadas (Bueno, 2017; Bueno; Zaina, 2017; Cândido, Bueno; Santos., 2020).

Outro importante conceito em IHC é quanto ao desenvolvimento das interfaces, perpassando pelos Padrões de Projeto de Design da Interface, em inglês *User Interface Design Patterns* (UIDP). O UIDP auxilia na classificação de componentes visuais da interface, sendo baseados em problemas usualmente encontrados por desenvolvedores e que podem facilitar a descrição e comunicação acerca de recursos visuais.

Por conta disso, diferentes recursos que atentem a esses padrões são implementados e disponibilizados em bibliotecas, como é o caso da UI-Patterns, de Toxboe (2013), que classifica esses UIDP em diferentes categorias, com explicações definidas acerca do título, descrição, definição e exemplos de implementação (Ellwanger et. al, 2015; Cândido et al., 2021).

Essa pluralidade nos requisitos e diferentes estudos para o desenvolvimento de interfaces, implica na constituição de diferentes tecnologias e recursos disponíveis na internet para utilização. Ademais, a integração de novos recursos para o atendimento de novas demandas contribui com o uso de pacotes e bibliotecas. Para os pacotes considera-se a definição de conjunto de funções para realização de uma atividade específica, enquanto às bibliotecas

Revista brasileira de iniciação científica
ISSN: 2359-232X

define-se a disposição de funções que permitam a construção de sistemas com base em um, ou mais, paradigmas (Tilkov; Vinoski, 2010).

Considerando que diferentes recursos possam ser requisitos de dependência e que os sistemas certas vezes precisam ser executados em diferentes equipamentos e plataformas, dificulta para os profissionais da computação e passando a ser necessários recursos que facilitem o gerenciamento de todo esse emaranhado tecnológico. Para isso, usualmente são utilizados gerenciadores de pacotes como, por exemplo, o *Node Package Manager* (NPM) e o Yarn, que permitem a simples adesão, remoção e execução de pacotes em sistemas baseados em Node JS; o Composer permite a mesma integração utilizando recursos para o PHP, permitindo o uso de diferentes bibliotecas e pacotes para construção de sistemas, e o *Package Installer for Python* (PIP) do Python, que permite a adesão de diferentes pacotes no desenvolvimento dos seus códigos (Tilkov; Vinoski, 2010; Harris et al., 2020; Kyriakidis; Maniatis, 2016).

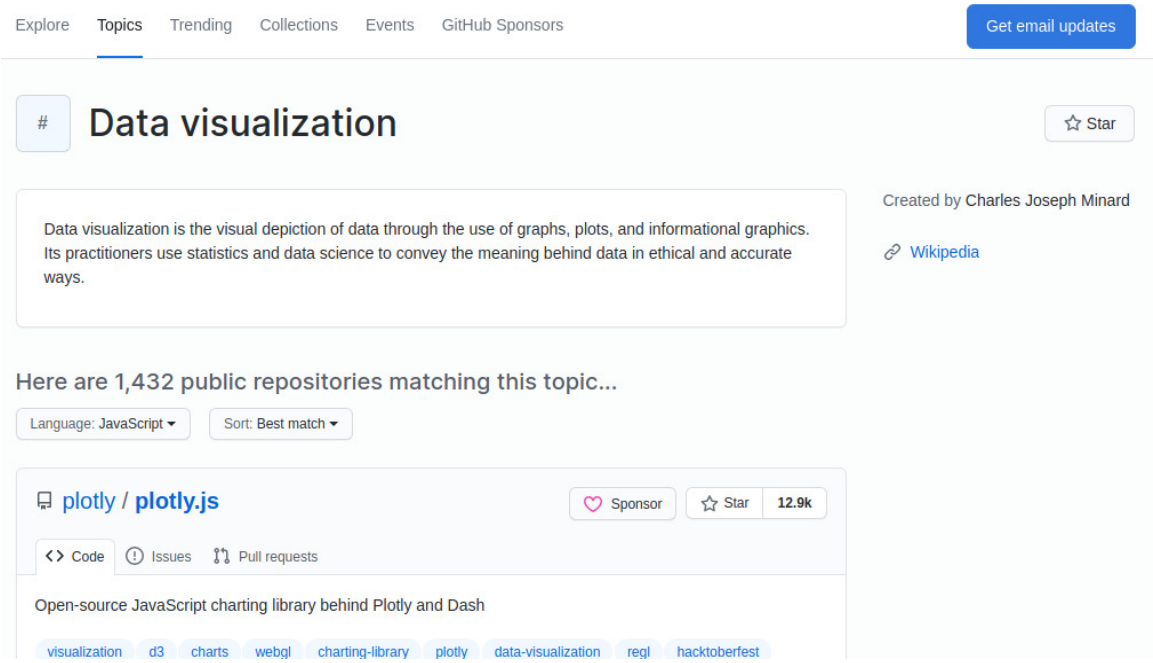
Metodologia

O Github é hoje um dos principais repositórios compartilhados e gratuitos encontrados na Internet. Ele cria um ranking de repositórios por estrelas considerando tópicos de interesse e o código de desenvolvimento do repositório, que permite visualizar a popularidade, uso e relação entre a quantidade de usuários e o código disponibilizado (Github, 2021a).

Utilizando-o como base, buscou-se pelo tópico "*data-visualization*" que possui, até o momento de escrita deste trabalho (fevereiro de 2021), cerca de 8.688 repositórios públicos. Utilizando o recurso de filtro de pesquisa com base em linguagem de programação oferecido pelo Github, utilizando o Javascript como chave, foram encontrados cerca de 1.432 repositórios (mostrado na Figura 1).

Revista brasileira de iniciação científica
ISSN: 2359-232X

Figura 1 - Ranking de repositórios do Github para o tópico "*data-visualization*".



Fonte: Github (2021a).

Com isso, selecionaram-se os 10 primeiros desse ranking que fossem pacotes e não fossem dependentes de outros Frameworks de desenvolvimento Front-end, pois a sua comparação com outros, em termos de configuração, poderiam gerar análises imprecisas, com isso definiram-se os pacotes dispostos na Tabela 1.

Analisando a Tabela 1 é possível levantar algumas considerações acerca dos pacotes listados, os pacotes victory, deck.gl, react-vis, react-map-gl e Historical-ranking-data-visualization-based-on-d3.js, que ocupariam respectivamente posições neste ranking, não foram adicionados devido à sua estrutura vinculada ao Framework ReactJS, o que faz com que a sua implementação e análise comparada a outros pacotes e bibliotecas se torne mais complexa e, por se tratar de algo que foge da *Document Object Model*, o HTML DOM[6], sua adesão à pesquisa torna-se indevida.

[6] Estrutura de objetos listados na página HTML durante a renderização da página pelo navegador.

Revista brasileira de iniciação científica
ISSN: 2359-232X

Tabela 1 - Ranking de Pacotes no Github por estrelas.

Nome do Pacote	**Mantenedor**	***Stars***	***Issues***		***Forks***
			Abertas	***Fechadas***	
plotly.js	Plotly	12.8k	1.001	2.250	1.5k
apexcharts.js	ApexCharts	9.9k	668	1363	681
C3	C3	9k	771	1.509	1.4k
F2	Antvis	7.4k	114	645	525
Raw	rawgraphs	7k	48	127	1.6k
uPlot	leeoniya	5.9k	25	360	183
roughViz	jwilber	5.7k	13	14	188
heatmap.js	pa7	5.4k	95	130	1.2k
GoJS	Northwoods Software	5k	1	99	2.3k
Britecharts	britecharts	3.6k	57	278	238

Fonte: Github (2021a).

Dentre os pacotes listados, torna-se interessante a atenção às questões, tanto abertas quanto fechadas, que dizem respeito às perguntas de possíveis problemas que foram encontrados e resolvidos pelos desenvolvedores, respectivamente. Considerando como parâmetro classificatório, os três pacotes com maiores *issues* fechadas, coincidentemente seriam os que já ocupam os quatro primeiros lugares, sendo que para a classificação em ordem apenas seria necessário trocar de posição o ApexCharts e C3.

Neste sentido, cabe considerar a relação entre as *issues* abertas e fechadas, considerando principalmente a resolução dessas *issues* de acordo com a relação das stars e forks dos pacotes, construindo um novo ranking, como visto na Tabela 2.

Revista brasileira de iniciação científica
ISSN: 2359-232X

Tabela 2 - Ranking utilizando a porcentagem de relação de *issues* resolvidas.

Pacote	**Abertas**	**Fechadas**	**Resolução**	**Não resolvidas**
GoJS	1	99	99%	1%
uPlot	25	360	93,50%	6,50%
F2	114	645	85%	15%
britecharts	57	278	83%	17%
raw	48	127	72,5%	27,5%
Plotly	1001	2250	69%	31%
ApexCharts	668	1363	67%	33%
C3	771	1509	66%	34%
heatmap	95	130	57%	43%
houghViz	13	14	52%	48%

Fonte: Github (2021a).

Com essa relação construída pela Tabela 2, pode ser possível entender a relação que existe entre o suporte oferecido e a complexidade existente na correção das *issues* conforme a construção dos pacotes, o Plotly, que ocupa a primeira colocação na Tabela 1, ocupa a 6ª colocação, estando abaixo inclusive do britecharts, que é o último na Tabela 1.

Por mais que utilizar isso como parâmetro de análise seja interessante para compreender a relação de suporte, estrelas com o tamanho dos pacotes e o contato direto com seus desenvolvedores, ainda se observa a consistência mesmo com um grande fluxo de *issues*. Por mais que o Plotly, C3 e ApexCharts não ocupem as primeiras colocações na Tabela 2, deve-se considerar os parâmetros que corroboram para isso.

O Plotly possui cerca de 6.500 usuários, considerando apenas aqueles que marcam o pacote como sendo usado no Github, enquanto o GoJS é utilizado por cerca de 1.300 usuários, ou seja, o Plotly possui cerca de 5 vezes a quantidade de usuários do GoJS. Isso evidencia, que mesmo com uma grande quantidade de usuários, ainda há uma porcentagem de resolução de 69% das issues, o que torna relevante a sua análise e consideração para pesquisa.

Revista brasileira de iniciação científica
ISSN: 2359-232X

Seguindo o mesmo procedimento de construção, foi possível observar a relação de resolução do ApexCharts, com 67%, e do C3, com 66%, sendo os pacotes mais próximos do Plotly na Tabela 1 e, por conta dessa relação comparativa, percebendo que não existe uma disparidade nos valores de suporte dos três e que todos eles possuem um contínuo desenvolvimento, percebido pela relação de resolução e usuários marcados no Github, com o Plotly, ApexCharts e C3 que, por consequência, foram escolhidos para serem analisados neste trabalho.

Considerando a estrutura dos três pacotes, a análise levou em consideração, principalmente, alguns parâmetros visando melhor assertivos os resultados por serem discutidos posteriormente. Assim, de forma a constituir uma estrutura padrão a ser observada em cada um deles, pode-se observar as diferenças e qualidades nos mesmos aspectos nos três, considerando-se as Heurísticas de Nielsen (Nielsen, 2005) como critério a ser observado, além da adesão de novos tópicos vinculados diretamente ao uso desses pacotes, como a integração dos pacotes às páginas Web, a adaptabilidade da integração dos recursos dos pacotes ao suporte multimodal e sensibilidade ao contexto, atentando-se ainda à uma análise UX baseada em Lowdermilk (2019) de cada um dos pacotes.

Neste contexto, definiu-se como parte da metodologia, a realização de etapas em um roteiro de análise construído a partir de tópicos a serem observados a partir da implementação dos gráficos, sendo: a) Estrutura e uso do sistema pelo usuário; b) Construção visual e sistematização dos recursos; c) Correção, prevenção e tratamento de erros; d) Personalização na implementação e integração ao sistema; e) Suporte e Documentação. Tendo uma descrição breve acerca da implicação de cada um destes na Tabela 3.

Revista brasileira de iniciação científica
ISSN: 2359-232X

Tabela 3 - Relação de Tópico e Roteiro de aplicação.

Tópico	**Roteiro**	**Base de comparação**
a) Estrutura e uso do sistema pelo usuário;	• Visibilidade do Status do Sistema; • Compatibilidade do sistema com o mundo real; • Controle e liberdade para o usuário;	Considera-se a relação visual oferecida pelo pacote, se existem ícones ao invés de apenas texto, se esses ícones remetem à simbologias utilizadas usualmente, tanto em sistemas quanto ao mundo real e se o usuário pode interagir com os dados gerados através dos recursos do pacote e se sim, quais são esses recursos disponibilizados;
b) Construção visual e sistematização dos recursos;	• Consistência e Padronização; • Reconhecimento em vez de memorização; • Eficiência e flexibilidade de uso; • Estética e design minimalista;	Considera-se a relação de cores, tamanhos de fonte de letra e a exposição clara dos recursos disponibilizados para o usuário, colocando em destaque o reconhecimento de ações e não a memorização, atentando-se para as possibilidades de uso, em diferentes contextos, com diferentes dispositivos, percebendo se o estilo de design é persistente.
c) Correção, prevenção e tratamento de erros;	• Ajude os usuários a reconhecerem, diagnosticarem e recuperarem-se de erros; • Ajuda e documentação;	Perceber se durante a realização das atividades foi percebido algum possível erro e considerar se isso implicaria em maiores complicações para o usuário final, se o pacote seria capaz, de forma autônoma, de resolver o erro ou se seria necessária uma atenção do desenvolvedor.
d) Personalização na implementação e integração ao sistema;	• Como começar: instalação e implementação; • Recursos e comandos do usuário; • Exemplos; • Parâmetros e propriedades claras para implementação;	Perceber durante a realização dos testes se são oferecidos subsídios suficientes para que o usuário possa integrar o pacote à aplicação sem grandes problemas, seja utilizando conexões de terceiros ou gerenciador de pacotes.
e) Suporte e Documentação;	• Documentação oficial; • Pesquisa na documentação; • Contínuo desenvolvimento;	Durante a análise utilizar a documentação oficial para construir e verificar se dentre as informações contidas é possível realizar a implementação sem problemas.

Fonte: O Autor (2021).

Revista brasileira de iniciação científica
ISSN: 2359-232X

De modo que a aplicação desse roteiro de análise nos pacotes Plotly, C3 e ApexCharts é documentada na seção subsequente, de Análises dos pacotes.

Análises dos Pacotes

Seguindo a metodologia especificada na seção anterior, serão explicadas as percepções obtidas a partir das análises realizadas com cada um dos pacotes já definidos, o Plotly, C3 e ApexCharts, com implementações utilizando uma página Web para visualização integral, tendo disponibilizadas todas as visualizações destes testes no Github em um repositório público criado para essa distribuição (https://github.com/paulhenrique/scientific-data-packages-comparison) e registrado no Zenodo sob o *Digital Object Identifier*: 10.5281/zenodo.4580509 e disponibilizado para acesso direto no link: https://scientific-data-packages-comparison.netlify.app/.

Plotly

O Plotly é uma biblioteca de gráficos para Python e Javascript, seus recursos são disponibilizados online e podem ser acessados e baixados sem custo para o usuário, no site principal também é possível acessar a um canal de comunidade que permite o contato com outros usuários da biblioteca através de um fórum de dúvidas (Plotly, 2021).

Para realização dessas análises foram construídos quatro exemplos encontrados na página de exemplos do Plotly, o *Hello World*, *Responsive Plots*, *Line and Scatter Plots* e o *Heat Map Plot*. Todos esses podem ser acessados e visualizados juntos através do link https://scientific-data-packages-comparison.netlify.app/pages/plotly.html e terem seus códigos de construção observados em https://github.com/paulhenrique/scientific-data-packages-comparison.

a) Estrutura e uso do sistema pelo usuário: O Plotly oferece para a visualização dos dados uma série de controles que permitem que o usuário interaja com o gráfico, esses controles podem ser observados na Figura 2.

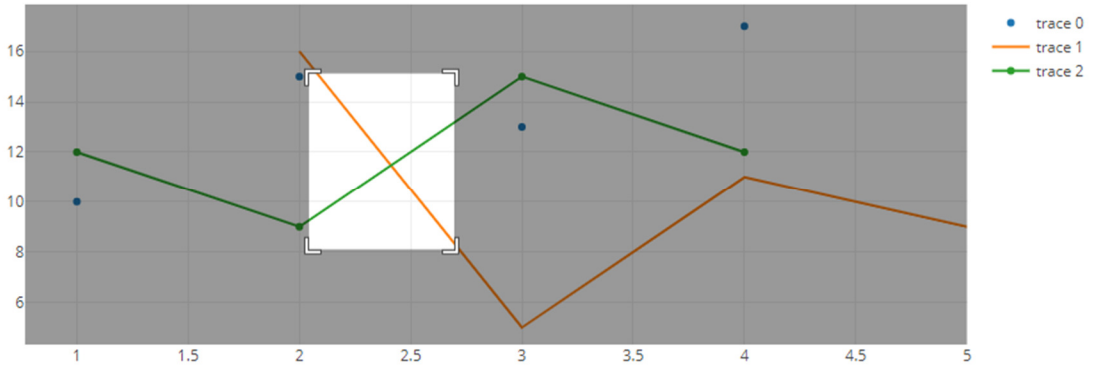
Figura 2 - Controles de configuração do usuário em gráficos gerados pelo Plotly.

Fonte: Plotly (2021).

Revista brasileira de iniciação científica
ISSN: 2359-232X

O usuário pode interagir com o gráfico utilizando o zoom, a modificação de trilhas a serem exibidas no gráfico, alterar e visualizar a centralização original do gráfico, criar recortes com formato livre e recortes retangulares de regiões do gráfico, como se vê na Figura 3, onde realizou-se um corte em um gráfico de espalhamento por linhas.

Figura 3 - Recorte de região de gráfico criado no Plotly para utilização.



Fonte: Plotly (2021).

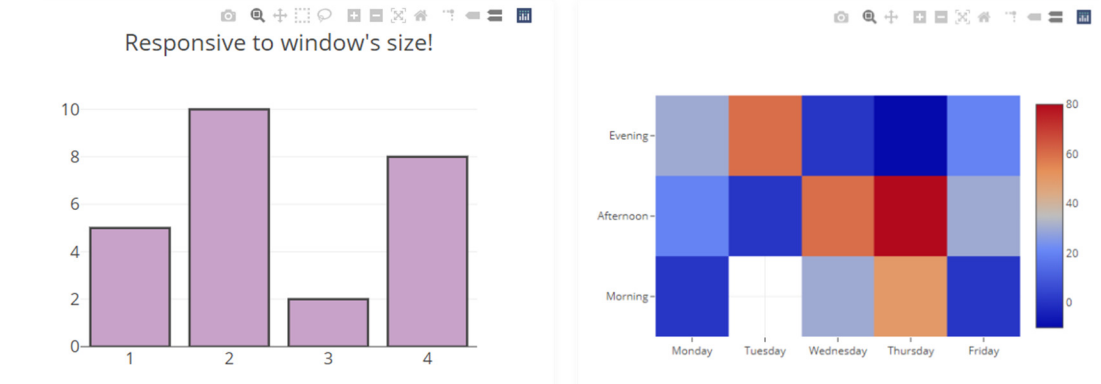
Considera-se através das análises e interações realizadas com o sistema, que é oferecida para o usuário uma visibilidade do status do sistema, perante a persistência dos títulos das linhas posicionados ao lado do gráfico e conforme o usuário movimenta o mouse sobre a visualização, são gerados rótulos para cada uma das linhas, permitindo que o usuário tenha determinado controle e liberdade de suas ações.

Com isso, há a implementação de compatibilidade do sistema com o mundo real por conta dos ícones disponibilizados junto aos controles do gráfico, que permitem que o usuário consiga compreender suas funções antes do uso, além da implementação do UIDP *Tooltips*, que mostra uma pequena descrição acerca da função do botão quando o usuário passa o mouse sobre ele.

b) Construção visual e sistematização dos recursos: Para visualização e consideração deste tópico utilizou-se dos exemplos *Responsive Plot* e *Heat Map Plot*, sendo possível perceber a consistência de utilização dos recursos visuais através da Figura 4, onde são colocados lado a lado ambos os gráficos.

Revista brasileira de iniciação científica
ISSN: 2359-232X

Figura 4 - Consistência e Padrões entre os exemplos *Responsive Plot* e *Heat Map Plot*.



Fonte: O Autor (2021).

Com isso, é interessante notar o uso do reconhecimento em vez da memorização quanto à utilização dos controles do usuário, sendo que para ambos, os controles se mantêm ocultos até que o usuário movimente o mouse sobre a área do gráfico, proporcionando assim maior flexibilidade e eficiência de uso.

Torna-se presente a estética e design minimalista pela consistência nos tamanhos dos ícones e nas suas colorações, além de não ter o uso de fontes de tamanhos muito grandes ou demasiadamente pequenos para visualização nos dispositivos.

c) Correção, prevenção e tratamento de erros: Por oferecer aos usuários formas diferentes de interação com os gráficos, o Plotly emite pequenos alertas, utilizando o UIDP *Toast*, que aparece por alguns segundos e logo é ocultado da tela para atentar os usuários acerca de tentativas de interação.

Uma interação considera que o usuário realiza recortes no gráfico para visualização mais detalhada de uma região e a mensagem “Clique duas vezes sobre o gráfico para retornar à visualização padrão” aparece, dando clareza para o usuário acerca de recursos que podem facilitar a sua interação. Como recurso auxiliar a essa interação também há os botões zoom-out e mover, oferecendo ao usuário formas de movimentar o gráfico por mais que tenha realizado um corte em determinada região que não desejava.

Ainda, no canto direito, o último ícone dos botões de controle do usuário, leva ao site principal do Plotly, o que pode direcionar o usuário à documentação oficial da biblioteca, todavia, isso pode ser um possível problema. Quando selecionado este botão, o usuário pode

Revista brasileira de iniciação científica
ISSN: 2359-232X

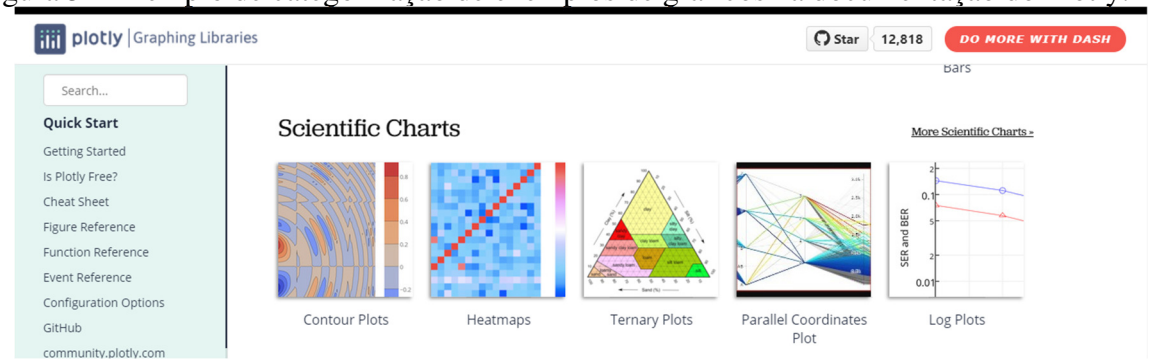
estar buscando por outros recursos da visualização dos gráficos, sendo interessante notar que, por mais que o usuário vá à documentação da biblioteca, não seria de grande ajuda, visto que a documentação é voltada aos desenvolvedores e não para a comum visualização dos gráficos.

d) Personalização na implementação e integração ao sistema: O Plotly oferece um tutorial base para o uso do pacote, sendo que existem diferentes formas de integração à página Web, assim como pela instalação por um gerenciador de pacote para o download dos arquivos ou via link de disponibilização online através da rede de fornecimento de conteúdo (CDN), que foi a aplicada na realização dos testes.

Neste ponto é interessante destacar que existem diversos exemplos na documentação oficial para cada uma das categorias, totalizando cerca de 125 exemplos disponibilizados. A implementação e utilização das funções são explicadas em páginas específicas e com diferentes exemplos de uso para cada um dos parâmetros que podem ser utilizados nas funções de visualização do pacote.

e) Suporte e Documentação: A documentação oficial oferece informações suficientes para a implementação de diferentes gráficos, vide Figura 5, sendo que utilizando os códigos oferecidos diretamente nos exemplos tornou-se simples e rápida a integração à página. Contudo, há de se considerar que a nomenclatura dos identificadores das *tags* em que são implementados nos exemplos pode gerar determinado tipo de confusão aos desenvolvedores não acostumados com o uso dessas implementações em Javascript.

Figura 5 - Exemplo de categorização de exemplos de gráficos na documentação do Plotly.



Fonte: Plotly (2021).

Ademais, nota-se que na página de exemplos disponibilizada na documentação existem imagens e a categorização dos gráficos por *Fundamentals, Basic Charts, Statistical Charts,*

Scientific Charts, Financial Charts, Maps, 3D Charts, Subplots, Custom Chart Events, Transform, Add Custom Controls e *Animations,* que permite uma busca mais fácil de gráficos relacionados ao uso específico do usuário. A Figura 5 demonstra como é essa organização com um exemplo da categoria *Scientific Charts.*

C3

O *Comfortable, Customizable and Controllable* (C3) é uma biblioteca de gráficos Javascript, seus recursos são disponibilizados online e podem ser acessados e baixados sem custo para o usuário, no site principal possível acessar a um canal de fórum que permite o contato com outros usuários da biblioteca através de uma comunidade no Google (Tanaka, 2014).

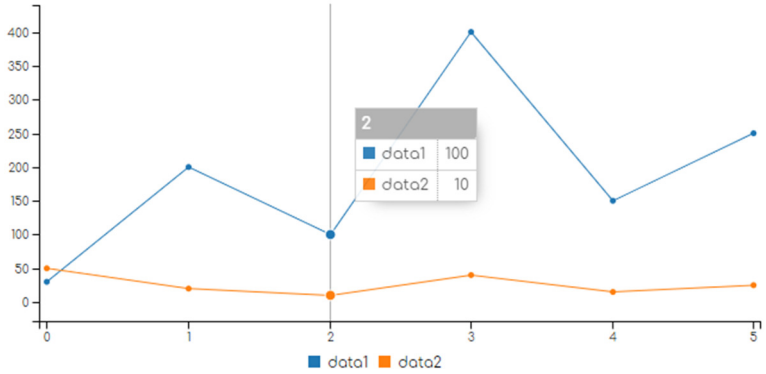
Para realização dessas análises foram construídos três exemplos disponibilizados na página de exemplos do C3, o *Hello World Plot, Line Chart* e *Scatter Chart*. Todos esses podem ser acessados e visualizados juntos através do link https://scientific-data-packages-comparison.netlify.app/pages/c3.html e terem seus códigos de construção observados em https://github.com/paulhenrique/scientific-data-packages-comparison.

a) Estrutura e uso do sistema pelo usuário: O C3 constrói os gráficos na tela sem problemas, permitindo a alteração das linhas a serem visualizadas pelo usuário, além de que, com o movimento do cursor do mouse sobre o gráfico, o usuário consegue visualizar rótulos específicos sob a linha para aquele ponto em questão, vide Figura 6.

Todavia, essas são as únicas formas de interação com a visualização dos dados, portanto, considera-se que não há grande liberdade para o usuário interagir com o gráfico, não podendo fazer alterações, recortes ou escolher qualquer tipo de visualização. Também há de se considerar que todas as informações mostradas através da realização dessas interações são expressas por texto, não mostrando ícones ou relações que forneçam outras formas de leitura mais rápidas ao usuário.

Revista brasileira de iniciação científica
ISSN: 2359-232X

Figura 6 - Visualização do exemplo *Hello World* do C3 com interação do usuário.

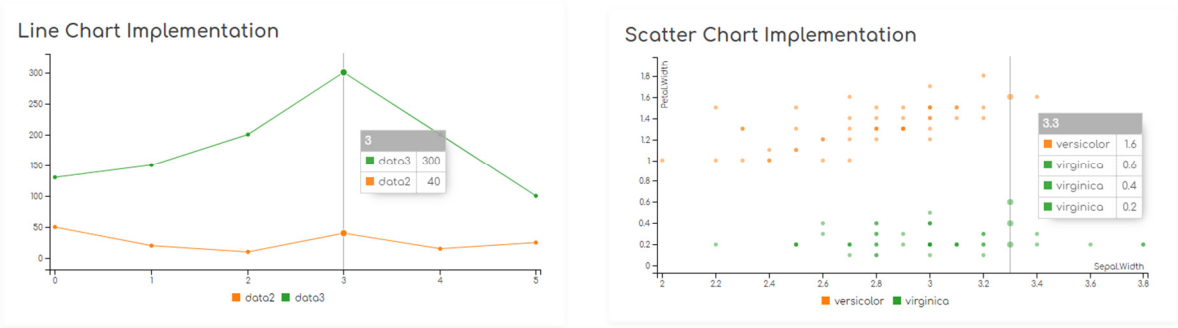


Fonte: O Autor (2021).

Percebe-se também que não existem botões para a interação com o gráfico além dos títulos das linhas exibidas na visualização. A inserção de ícones de visualização, com ícones de olho poderia ser algo muito interessante, pois o usuário poderia compreender e entender a ação do clique sobre esse elemento antes de interagir com ele e, por conta de não oferecer esse tipo de exibição, a visibilidade do sistema como um todo e a compatibilidade dele com o mundo real acaba se tornando menos aparente na sua utilização.

b) Construção visual e sistematização dos recursos: Para visualização deste tópico utilizou-se dos exemplos *Line Plot* e *Scatter Plot*, colocados lado a lado para visualização na Figura 7, onde é possível observar a consistência dos padrões.

Figura 7 - Comparação dos gráficos *Line Chart* e *Scatter Chart* do C3.



Fonte: O Autor (2021).

Revista brasileira de iniciação científica
ISSN: 2359-232X

Porém, apesar das poucas interações possíveis de serem realizadas com os gráficos, existem parâmetros a serem considerados, os gráficos do C3 possuem uma construção CSS além de Javascript, isso permite que o gráfico possua determinado suporte à sensibilidade ao contexto, se adaptando de forma *stand-alone* a diferentes tamanhos de tela de diferentes dispositivos e isso é uma característica interessante a ser esclarecida.

Neste sentido, também se considera a implementação dos gráficos com elementos visuais, cores padrão e estrutura de visualização iniciais comuns aos gráficos, o que define determinada padronização entre a implementação dos gráficos. Por consequência, atentando-se ao estilo, eficiência, flexibilidade de uso e reconhecimento em vez de memorização, torna-se mais complicada a análise.

Essa biblioteca oferece apenas duas formas de interação, ambas não permitindo uma análise específica acerca dos tipos de gráficos, além disso, como já mencionado, a falta de ícones de indicação, ou outros elementos que pudessem cumprir esse papel, faz com que o usuário tenha de memorizar a ação de clique sobre os títulos das linhas para os ocultar na visualização, o que diminui de certa forma a eficiência da implementação dessa aplicação.

Da perspectiva estética, o C3 é bastante conciso, não tendo elementos desalinhados quanto à estrutura do gráfico e não possuindo textos demasiadamente grandes ou pequenos, com leitura visual interessante, caracterizando um design minimalista pela disposição de poucos elementos visuais na implementação.

c) Correção, prevenção e tratamento de erros: As interações com os gráficos gerados pelo C3 são demasiadamente limitadas e, por conta disso, até mesmo a geração de erros torna-se menor.

Todavia, ao clicar sobre os títulos dos elementos para ocultar a sua visualização nos gráficos, mesmo que seja acidentalmente, não existe nenhum tipo de alerta ou visualização que permita que o usuário reconheça rapidamente o seu erro e consiga retornar ao estado anterior, há apenas uma diminuição de opacidade do nome da trilha que foi ocultada.

Por conta disso, considera-se que não é disponibilizada uma correção de erros visual para interações do usuário com o gráfico gerado pela biblioteca.

d) Personalização na implementação e integração ao sistema: A implementação inicial do C3 torna-se mais trabalhosa devido à sua estrutura, inicialmente não é disponibilizada

Revista brasileira de iniciação científica
ISSN: 2359-232X

a integração utilizando CDN, o que faz com que seja necessário o download da release do pacote disponibilizada no Github.

Isso torna mais complicada a compreensão, visto que ao realizar o download da versão, não apenas os arquivos de estilização e script são baixados, mas também todas as dependências e códigos não compilados da biblioteca, além de documentação e páginas Web.

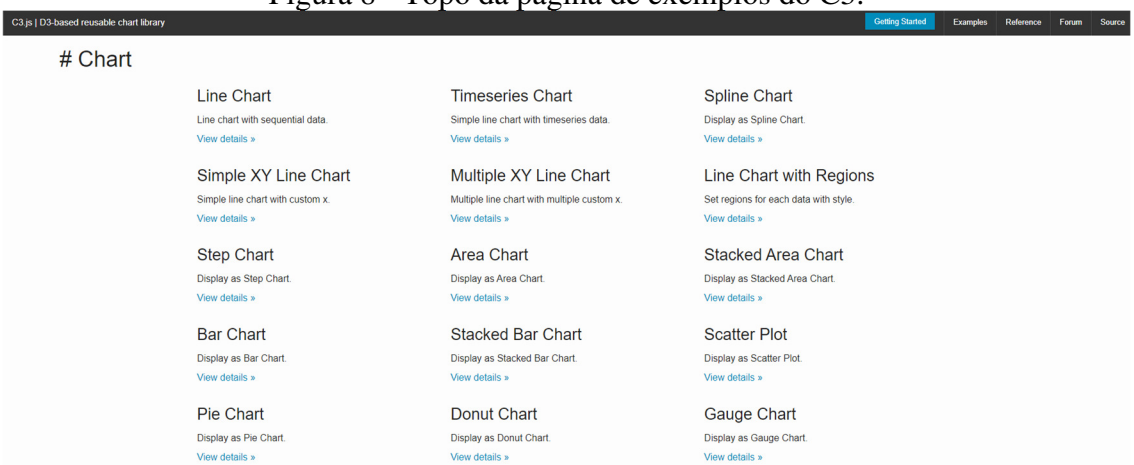
Para a implementação é necessário utilizar tanto o CSS quanto o Javascript aplicado ao sistema, isso pode ser interessante para personalizações, contudo, demanda do desenvolvedor mais atenção quanto à sua integralização na página.

Ademais, considera-se como uma complicação a inserção do D3 como dependência, apesar da documentação do C3 explicar que se deve realizar a importação da biblioteca, a mesma não disponibiliza sequer link para acesso à página do D3 js, que por sua vez permite a inserção de código CDN.

e) Suporte e Documentação: A documentação oficial oferece informações suficientes para a implementação de diferentes gráficos, todavia, os exemplos disponibilizados sempre utilizam a mesma nomenclatura para estruturação dos exemplos, isso faz com que o uso de mais de um desses exemplos na mesma página cause problemas de compatibilidade. O uso, por exemplo, de um identificador único para cada um dos exemplos poderia ser interessante.

A página de exemplos apenas os lista por nomes, sem imagens e pré-visualizações, como pode-se observar na Figura 8. Isso força o usuário que intenta por sua utilização a clicar em cada um dos exemplos para descobrir do que se trata.

Figura 8 - Topo da página de exemplos do C3.



Fonte: O Autor (2021).

Revista brasileira de iniciação científica
ISSN: 2359-232X

A lista desses exemplos é categorizada por *Chart*, *Axis, Dat, Grid, Region, Interaction, Legend, Tooltip, Chart Options, Line Chart Options, Pie Chart Options, API, Transform* e *Style*. Apesar da grande quantidade de categorias, cabe ressaltar que não se trata de diferentes exemplos de tipos de gráficos, mas sim de implementações e configurações diferentes para os gráficos mostrados na categoria *Chart*. Além disso, o C3 não oferece a possibilidade de realização de pesquisa entre os seus exemplos e funções na documentação oficial.

ApexCharts

O ApexCharts é uma biblioteca de gráficos Javascript, seus recursos são disponibilizados online e podem ser acessados e baixados sem custo para o usuário, mas não existe nenhum tipo de link ou direcionamento que leve o usuário à comunidade ou sequer alguma menção da mesma (ApexCharts, 2021).

Para realização dessas análises foram construídos quatro exemplos disponibilizados na página de exemplos do ApexCharts, o *Hello World*, *Line Chart*, *Scatter Chart* e *Heat Map*. Todos esses podem ser acessados e visualizados juntos através do link https://scientific-data-packages-comparison.netlify.app/pages/apexcharts.html e terem seus códigos de construção observados em https://github.com/paulhenrique/scientific-data-packages-comparison.

a) Estrutura e uso do sistema pelo usuário: O ApexCharts permite que o usuário interaja com os gráficos através dos seus controles e com o clique sobre o gráfico, permitindo a realização de recortes em áreas desejadas, fornecendo ainda ícones, que permitem tanto a visibilidade do Status do Sistema quanto a compatibilidade do sistema com o mundo real. Esses controles podem ser observados na Figura 9.

Figura 9 - Controles de gráfico do ApexCharts.

Fonte: O Autor (2021).

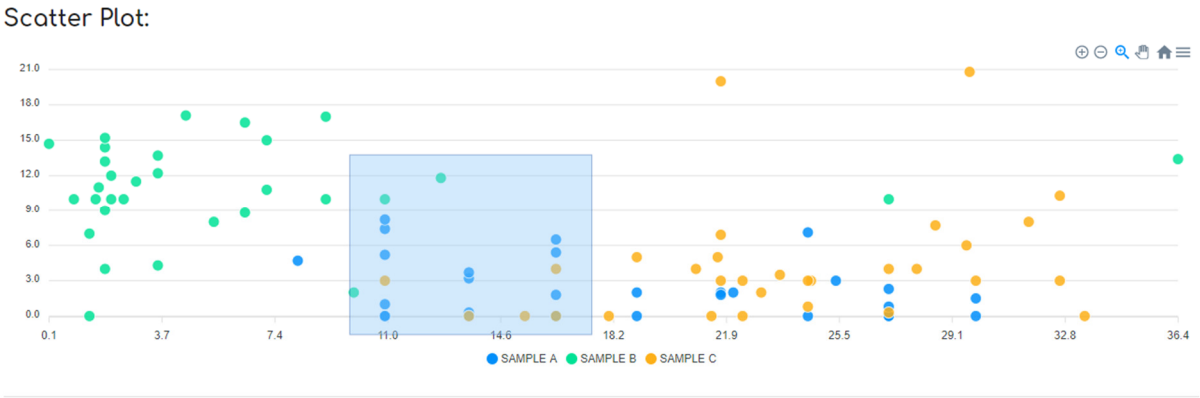
Revista brasileira de iniciação científica
ISSN: 2359-232X

Também é possível realizar o download de imagens dos gráficos através do UIDP *Hamburger Menu*, que permite salvar os recortes em SVG, PNG e CSV em gráficos que a estrutura permita o fazer, como em gráficos de linhas ou espalhamento.

b) Construção visual e sistematização dos recursos: Para visualização deste tópico utilizou-se dos exemplos *Hello World* e *HeatMap*, que tornou possível a percepção da consistência dos elementos visuais em diferentes gráficos do ApexCharts.

Torna-se interessante notar que também há a disposição de um título que permite a ocultação de dados, quando existe mais de uma linha ou grupo de dados a serem exibidos, assim como na Figura 10.

Figura 10 - Realizando recorte em gráfico do ApexCharts;



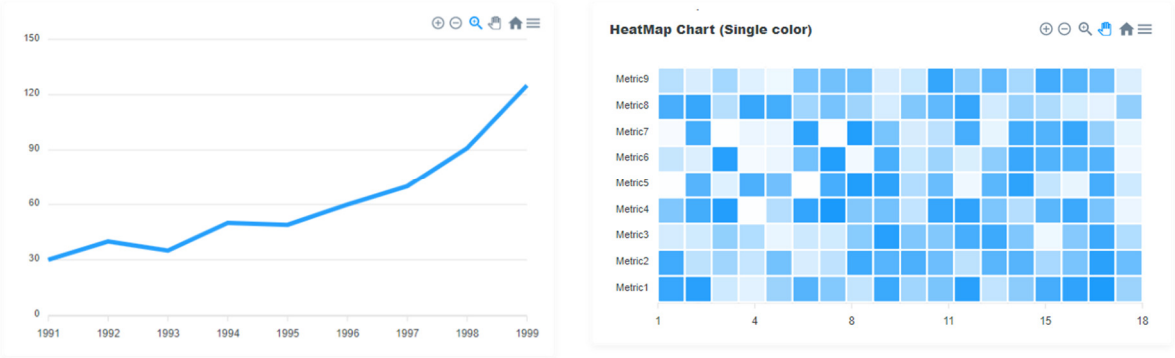
Fonte: O Autor (2021).

Ainda, esse mesmo título é ocultado quando há apenas um grupo de exibição, como pode-se perceber na Figura 11. O ApexCharts também não oculta os controles posicionados acima do gráfico, eles são exibidos a todo momento durante a sua utilização, para os ocultar é necessária determinada alteração dos parâmetros de construção.

O uso de ícones é interessante, mas, em vista de reconhecimento em vez de memorização, a implementação do UIDP *Tooltips*, a fim de propiciar ao usuário reconhecer a ação que acontecerá com o uso de tal botão seria mais interessante. Considera-se também que a flexibilidade e eficiência de uso são interessantes, visto que a interação com os gráficos é bem esclarecida para o usuário.

Revista brasileira de iniciação científica
ISSN: 2359-232X

Figura 11 - Comparação dos gráficos *Hello World* e *HeatMap*.



Fonte: O Autor (2021).

Quanto à estética e design minimalista, considera-se pertinente, visto que há padronização entre as distâncias dos elementos e não são utilizadas fontes de tamanhos muito grandes ou demasiadamente pequenas para visualização nos dispositivos.

c) Correção, prevenção e tratamento de erros: Apesar de oferecer diferentes formas de interação com os gráficos, o ApexCharts não oferece determinados tipos de prevenção de erros, sendo que ao utilizar os exemplos ao realizar recortes o usuário tem de acessar diretamente pelos botões de controle para retornar ao estado inicial de visualização gráfica.

Isso pode causar algum tipo de confusão às ações de recorte e zoom poderem ser aplicadas apenas com o clique e arraste do mouse e o desfazer dessa ação depende de uma ação que, de certa forma, é desconexa à essa interação.

d) Personalização na implementação e integração ao sistema: O ApexCharts permite a integração da biblioteca com o sistema de forma simples utilizando CDN, sendo implementado com um link para um arquivo de Javascript.

Há na página principal, que guia a implementação da biblioteca à página, logo após as instruções de instalação, um tutorial chamado "*Creating Your First JavaScript Chart*", em português Criando seu primeiro Gráfico Javascript, que demonstra como realizar a implementação de um gráfico de linhas podendo ser editado via editor virtual. Também é possível personalizar a exibição de itens do gráfico.

e) Suporte e Documentação: A documentação do ApexCharts é bem-organizada, permite a visualização de diferentes itens e a fácil localização dos assuntos, permitindo inclusive a realização de pesquisa por termos na documentação.

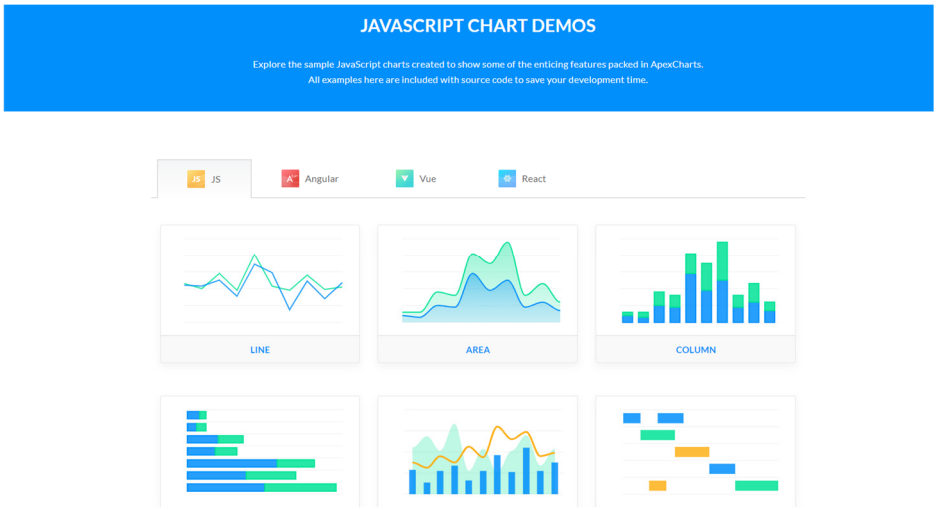
Revista brasileira de iniciação científica
ISSN: 2359-232X

Além dos itens referentes à documentação em si, ou seja, a descrição das funções e exemplos de implementação, existem os exemplos de gráficos, que são chamados aqui de demonstrações, são exibidas apenas duas grandes categorias de gráficos, sendo uma mais básica e outra com recursos mais avançadas construídas junto do *FusionCharts*[7].

Ao todo são 33 exemplos de gráficos dispostos na página, todos com imagens utilizando o UIDP *Cards*, como visto na Figura 12.

Por mais que os exemplos sejam bem estruturados e explicados, alguns utilizam de funções de exemplo que não são contidas na descrição da demonstração, como o *HeatMap*, que foi utilizado como exemplo anteriormente.

Figura 12 - Página de exemplos e demonstrações do ApexCharts.



Fonte: ApexCharts (2021).

Para sua implementação foi necessário construir uma função que o exemplo requisitava, contudo, considerando a estrutura do gráfico e o sentido da demonstração, compreende-se a necessidade de sua implementação, mas um exemplo completo poderia tornar-se mais simples para o desenvolvedor.

[7]Empresa que desenvolve e fornece dashboards para administração de sites : https://www.fusioncharts.com/

Revista brasileira de iniciação científica
ISSN: 2359-232X

Considerações Finais e Trabalhos Futuros

Este trabalho justifica-se a partir da continuidade de análises da perspectiva do desenvolvedor e da disponibilização de recursos para os usuários com análises acerca da perspectiva da adaptação de interfaces e uso das heurísticas de Nielsen.

Para tanto, abordou os conceitos respectivos a realização de uma análise que pudesse abstrair características de três pacotes de visualização de dados numéricos na Web, escolhidos com base no ranqueamento realizado pelo Github acerca dos pacotes mais populares, considerando stars, *issues* e resolução de problemas.

Deste modo, desenvolveu-se na seção de Análises dos Pacotes uma observação através da implementação e uso do Plotly, C3 e ApexCharts, onde pôde-se observar as características individuais de cada um dos pacotes.

Com isso, foi possível definir algumas considerações acerca de cada um dos gráficos para uso em aplicações. O C3 é o pacote que proporcionou menor variabilidade possibilidades de interação a partir do usuário, com menos exemplos de gráficos e com uma abordagem menos direta e simples para os desenvolvedores, contudo, há de se considerar que é baseado em uma biblioteca de construção de visualizações e que, por conta disso, pode ser o único dentre os três pacotes analisados a ser altamente personalizado, mas para isso demanda do desenvolvedor maior dedicação e conhecimento de Javascript e da biblioteca D3.

O Plotly e ApexCharts apresentaram características muito semelhantes acerca das possibilidades de interação dos usuários, sendo que por oferecer maior integração de prevenção de erros o Plotly mostrou maior aderência e, por conta disso, pode proporcionar uma menor quantidade de problemas de interação.

Todavia, há de se considerar que quanto à adaptatividade em diferentes tamanhos de tela e, portanto, a integração de suporte à sensibilidade ao contexto, o ApexCharts mostrou um suporte nativo e uma melhor integração neste sentido.

Por conseguinte, considera-se a qualidade e pertinência dos três pacotes para implementação em cenários e contextos diferentes, em um ambiente onde intenta-se por oferecer apenas a visualização dos gráficos sem suporte ao download dos mesmos e determinada proteção às informações e integridade o C3 pode ser uma melhor alternativa.

Em contextos em que existirá menor interação com o usuário, sendo necessário apenas o uso de recursos rápidos sem grandes necessidades de visualizações específicas do usuário, o ApexCharts se torna uma melhor alternativa.

Contudo, considerando esses três cenários e utilizando das propriedades de personalização e configuração dos gráficos, com base nas análises realizadas e comparações constituídas, o Plotly pode ser considerado o pacote mais versátil e, por consequência, um redutor de possíveis problemas de interação dos usuários.

Assim, considera-se que trabalhos futuros sobre a realização de uma análise com integração de algum destes pacotes com uma ferramenta de visualização numérica com descrição das funções utilizadas, a integração desses pacotes com Frameworks CSS, como o Bootstrap, Materialize ou Foundation, ou Front-end, como VueJS, React ou Angular e a análise de desempenho deles com gráficos com uma grande amostra de dados são interessantes para proporcionar outras formas de análises desses recursos em contextos específicos.

Agradecimentos

Os autores agradecem à Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) pelo suporte financeiro (processo no. 2019/20330-1) e ao IFSP, campus Itapetininga, pelo suporte técnico oferecido.

Referências

APEXCHARTS, Inc. ApexCharts: **Modern & Interactive Open-source Charts**. 2021. Disponível em: <https://apexcharts.com/>. Acesso em: 25 de fev. de 2021.

BORCHERDS P. H., **Python: a language for computational physics**, Computer Physics Communications, Volume 177, Issues 1–2, 2007, Pages 199-201, ISSN 0010-4655. doi: [https://doi.org/10.1016/j.cpc.2007.02.019] Disponível em: <https://www.sciencedirect.com/science/article/pii/S0010465507000732>. Acesso em: 23 de fev. de 2021.

BUENO, Danilo Camargo. **HyMobWeb: uma abordagem para a adaptação híbrida de interfaces Web móveis sensíveis ao contexto e com suporte à multimodalidade.** 2017. Disponível em: <https://repositorio.ufscar.br/handle/ufscar/9157>. Acesso em: 23 de fev. de 2021.

BUENO, Danilo Camargo; ZAINA, Luciana Martinez. **HyMobWeb: A hybrid adaptation of context-sensitive Web interfaces with multimodality support in mobile devices.** SBC Journal on Interactive Systems, v. 8, n. 2, p. 20-34, 2017. Disponível em: <https://www.seer.ufrgs.br/jis/article/view/73165>. Acesso em: 23 de fev. de 2021.

CÂNDIDO, Paulo Henrique Vieira; **Problemas de interação na adaptação de interface Web móvel em Frameworks Front-end**. Trabalho de Conclusão de Curso - Instituto Federal de Educação, Ciẽncia e Tecnologia de São Paulo - IFSP Campus Itapetininga, 2017. Disponível em: <https://drive.ifsp.edu.br/s/CoQpCEY3zC00jw6>. Acesso em: 23 de fev. de 2021.

CÂNDIDO, Paulo Henrique Vieira et al. **Panda: Uma Nova Ferramenta Web Responsiva Para Auxiliar no Ensino e Comunicação de Pessoas com Limitações Psicomotoras**. Revista Brasileira de Informática na Educação, v. 29, p. 25-47, 2021. Disponível em: <https://www.br-ie.org/pub/index.php/rbie/article/view/v29p25>. Acesso em 25 de fev. de 2021.

CÂNDIDO, Paulo Henrique Vieira, SILVA-SANTOS, Carlos Henrique da, VOTTO, Luíz Felipe. and AMBROSIO Leonardo Ambrosio, **Semantic Web-based System for Light Scattering Using the Generalized Lorenz-Mie Theory**, 2019 PhotonIcs & Electromagnetics Research Symposium - Spring (PIERS-Spring), Rome, Italy, 2019, pp. 3217-3224, doi:[10.1109/PIERS-Spring46901.2019.9017438]. Disponível em: <https://ieeexplore.ieee.org/document/9017438>. Acesso em 23 de fev. de 2021.

CÂNDIDO, Paulo Henrique Vieira; BUENO, Danilo Camargo; SANTOS, Carlos Henrique Silva. **Análise da utilização de framework front-end em sistema web adaptativo: um estudo da perspectiva do desenvolvedor**. Revista Brasileira de Iniciação Científica, v. 7, n. 1, p. 31-54, 2020a. Disponível em: <https://periodicos.itp.ifsp.edu.br/index.php/IC/article/view/1592/0>. Acesso em: 23 de fev. de 2021.

CÂNDIDO, Paulo Henrique Vieira; SANTOS, Carlos Henrique da Silva. **OptX.SaaS: Novo Simulador Óptico em Nuvem**. In: Anais do MOMAG 2020 - 19º SBMO: Simpósio Brasileiro de Micro-ondas e Optoeletrônica e 14º CBMag - Congresso Brasileiro de Eletromagnetismo. São Caetano do Sul: SBMO, 2020b. p. 896-898. Disponível em:<https://www.sbmo.org.br/evento/55/momag-2020>. Acesso em: 23 de fev. de 2021.

ELLWANGER, Cristiane; DA ROCHA, Rudimar Antunes; DA SILVA, Régio Pierre. **Design de interação, design experiencial e design thinking: a triangulação da interação humano-computador (IHC).** Revista de Ciências da Administração, v. 17, n. 43, p. 26-36, 2015. Disponível em:<https://www.redalyc.org/pdf/2735/273543309003.pdf>. Acesso em: 23 de fev. de 2021.

GEBREMESKEL, Gebeyehu Belay; HAILU, Birhanu; BIAZEN, Belete. **Architecture and optimization of data mining modeling for visualization of knowledge extraction: Patient safety care**. Journal of King Saud University-Computer and Information Sciences, 2019. Disponível em:<https://www.sciencedirect.com/science/article/pii/S1319157819308754>. Acesso em: 23 de fev. de 2021.

GITHUB, Inc. **GitHub: Where the world builds software**. Disponível em: <https://github.com/>, 2021a, Acesso em: 17 de fev. de 2021.

GITHUB, Inc. **The State of the Octoverse | The State of the Octoverse explores a year of change with new deep dives into developer productivity, security, and how we build communities on GitHub.** Disponível em: <https://octoverse.github.com/>, 2021b. Acesso em: 17 de fev. de 2021.

HARRIS, C.R., MILLMAN, K.J., WALT, S.J. van der et al. **Array programming with NumPy.** Nature 585, 357–362 (2020). doi:[10.1038/s41586-020-2649-2], Disponível em: <https://doi.org/10.1038/s41586-020-2649-2>. Acesso em: 23 de fev. de 2021.

JAIN, Nilesh. **Review of different responsive css front-end frameworks**. Journal of Global Research in Computer Science, v. 5, n. 11, p. 5-10, 2014. Disponível em: <https://www.rroij.com/open-access/review-of-different-responsive-css-frontend-frameworks.php?aid=52575>. Acesso em: 24 de fev. de 2021.

KYRIAKIDIS, Alex; MANIATIS, Kostas. **The Majesty of Vue. js**. Packt Publishing Ltd, 2016.

LOWDERMILK, Travis. **Design Centrado no Usuário: um guia para o desenvolvimento de aplicativos amigáveis**. Novatec Editora, 2019.

MCNANEY, Róisín et al. **Speeching: Mobile crowdsourced speech assessment to support self-monitoring and management for people with Parkinson's**. In: Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems. 2016. p. 4464-4476. doi:[10.1145/2858036.2858321].

MROZEK, Dariusz. **A review of Cloud computing technologies for comprehensive microRNA analyses**. Computational Biology and Chemistry, p. 107365, 2020. Disponível em:<https://www.sciencedirect.com/science/article/abs/pii/S1476927120307696>. Acesso em: 23 de fev. de 2021.

NIELSEN, Jakob. **Ten usability heuristics**. 2005. Disponível em: <https://pdfs.semanticscholar.org/5f03/b251093aee730ab9772db2e1a8a7eb8522cb.pdf>. Acesso em: 23 de fev. de 2021.

PATIL, Mayur M. et al. **A qualitative analysis of the performance of MongoDB vs MySQL database based on insertion and retriewal operations using a web/android application to explore load balancing—Sharding in MongoDB and its advantages**. In: 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC). IEEE, 2017. p. 325-330. Disponível em:<https://ieeexplore.ieee.org/abstract/document/8058365>. Acesso em: 23 de fev. de 2021.

PLOTLY. **Plotly: visualize data, together**. 2021. Disponível em: <https://plotly.com/javascript/>. Acesso em: 19 de fev. de 2021

SANTANA, Daniela Santos; DA SILVA SANTOS, Carlos Henrique; FIGUEROA, Hugo Enrique Hernandez. **Human-computer interface techniques to design and evaluate an electromagnetic simulator.** IEEE Latin America Transactions, v. 12, n. 4, p. 725-732, 2014. Disponível em:<https://ieeexplore.ieee.org/abstract/document/6868876>. Acesso em: 23 de fev. de 2021.

SANTOS, Carlos Henrique da Silva. **Computação bio-inspirada e paralela para a análise de estruturas metamateriais em microondas e fotônica.** 2010. 155 p. Tese (doutorado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação, Campinas, SP. Disponível em: <http://www.repositorio.unicamp.br/handle/REPOSIP/260809>. Acesso em: 23 fev. de 2021.

SCHMIDT, Vincent A. **User interface design patterns**. AIR Force Research Lab Wright-patterson Afb Oh Human Effectiveness Directorate, 2010. Disponível em: <https://apps.dtic.mil/sti/citations/ADA530798>. Acesso em: 23 de fev. de 2021.

SHI, Yan et al. **Human-computer interaction based on face feature localization**. Journal of Visual Communication and Image Representation, v. 70, p. 102740, 2020. Disponível em: <https://www.sciencedirect.com/science/article/pii/S104732031930361X>. Acesso em: 23 de fev. de 2021.

TANAKA, Masayuki. **Comfortable, Customizable, Controllable (C3): D3-based reusable chart library,** 2014. Disponível em: <https://c3js.org/>, Acesso em: 19 de fev. de 2021.

TILKOV, Stefan; VINOSKI, Steve. **Node. js: Using JavaScript to build high-performance network programs**. IEEE Internet Computing, v. 14, n. 6, p. 80-83, 2010. Disponível em: <https://ieeexplore.ieee.org/abstract/document/5617064>. Acesso em: 23 de fev. de 2021.

TOXBOE, Anders. **User interface design pattern library**. UI Patterns, 2013. Disponível em: <http://ui-patterns.com/>. Acesso em: 23 de fev. de 2021.

TSAY, Jason; DABBISH, Laura; HERBSLEB, James. **Influence of social and technical factors for evaluating contribution in GitHub**. In: Proceedings of the 36th international conference on Software engineering. 2014. p. 356-366. Disponível em: <https://dl.acm.org/doi/abs/10.1145/2568225.2568315>. Acesso em: 23 de fev. de 2021.