Universidade de São Paulo
Instituto de Ciências Matemáticas e de Computação

Documentação de Software
Joker - An Open Source Card Sorting Application.

SUMÁRIO

1 INTRODUÇÃO

1.1 O QUE É O *JOKER?*

Trata-se de uma ferramenta gratuita e de código aberto voltada para a realização de estudos de usabilidade baseados em Card Sorting, desenvolvida na Universidade de São Paulo no campus de São Carlos por **Anderson Canale Garcia, André de Lima Salgado, Felipe Silva Dias, João Pedro R. Mattos e Renata P. M. Fortes**. O desenvolvimento da ferramenta foi financiado pela Fundação de Amparo à Pesquisa do Estado de São Paulo (**FAPESP**) sob a vigência dos processos **nº 2018/19323-8, nº 2017/15239 - 0 e nº 2015/24525 - 0**.

1.2 SOBRE O *CARD SORTING*

Card Sorting é uma das técnicas mais populares para a avaliação de Arquiteturas de Informação (AI), uma vez que possibilita o agrupamento de informações através do modelo mental dos participantes dos experimentos. Através de *Joker*, o *Card Sorting* pode ser realizado à distância, assim a ferramenta se qualifica como uma alternativa gratuita e disponível em código aberto que visa a contribuir com pesquisas de usabilidade a serem conduzidas futuramente por outros pesquisadores.

1.3 CONTRIBUIÇÕES DA FERRAMENTA

A construção do *Joker* motivou a redação de um artigo[1] intitulado ***"Smart toys and Children's Privacy: usable privacy policy insights from a Card Sorting experiment"***. Esse artigo é um resultado que representa um esforço de todos os envolvidos nos projetos relacionados à utilização da ferramenta de apoio à técnica de *Card Sorting Joker*. O artigo foi submetido e aceito ao evento científico **SIGDOC'2019**[2] (Grupo de Interesse Especial em Design de Comunicação da *Association for Computing Machinery* - ACM) em Outubro de 2019.

Além disso, o benefício oferecido pela ferramenta *Joker* tornou-se de interesse internacional, visto a participação do pesquisador Prof. *Patrick Hung*[3], da *Ontario Tech University* (Canadá), que participou como co-autor no artigo que foi redigido.

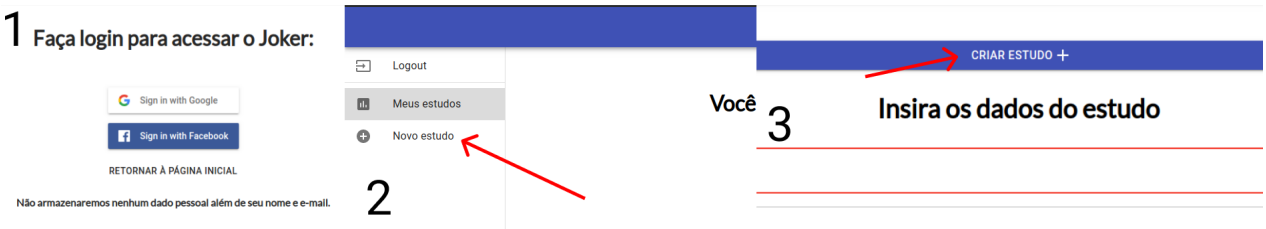
[1] <https://doi.org/10.1145/3328020.3353951>
[2] <https://sigdoc.acm.org/conference/2019/>
[3] <https://businessandit.ontariotechu.ca/people/faculty/networking-and-it-security/patrick-hung-phd.php>

2 GUIA DO USUÁRIO

2.1 CRIAÇÃO DE ESTUDOS

A criação de um novo estudo é realizada, primeiramente, clicando no botão de "Área do Pesquisador"na página inicial. Depois de efetuado o *login* na ferramenta *Joker*, basta acessar a aba "Novo estudo"na lateral esquerda do painel do pesquisador e inserir os dados do estudo. Finalmente, resta clicar em "Criar estudo"para finalizar a criação de um estudo, que já pode ser conferida na página "Meus Estudos".

Figura 1 – Processo de criação de um estudo. As flechas vermelhas indicam os botões citados na subseção 2.1.



Fonte: Elaborada pelo autor.

2.2 PARTICIPAÇÃO NOS ESTUDOS

Para disponibilizar o estudo para participações, basta clicar no símbolo ao lado do estudo desejado dentro da aba "Meus Estudos"no painel do pesquisador e, em seguida, dentro do menu de contexto aberto, clicar em "Copiar Link Para Estudo". Dessa forma, um link para acesso ao estudo será copiado para a área de transferência, de forma a possibilitar seu envio aos participantes, como demonstrado na figura 3.

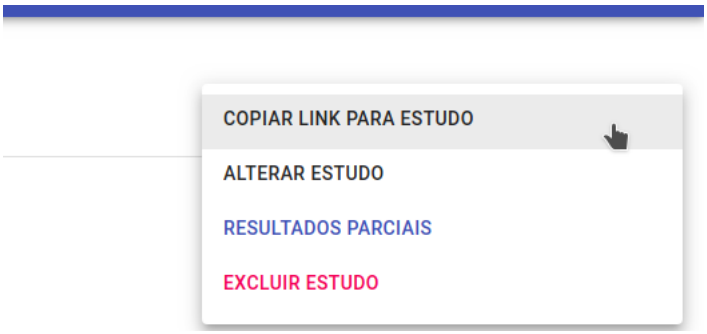
Do ponto de vista do participante, basta clicar no link recebido ou colá-lo no campo presente na página "Área do Participante".

É importante notar que só serão considerados para a "Obtenção de Resultados"as participações que forem devidamente concluídas. Logo, caso um participante feche a tela de participação sem clicar no botão de "Concluir"no canto superior direito, a participação não será considerada para a apuração final de resultados.

2.3 OBTENÇÃO DE RESULTADOS

Os resultados podem ser obtidos a partir do botão "Resultados Parciais"dentro do menu de contexto que se abre quando se clica no símbolo ao lado do estudo desejado, dentro do painel do pesquisador.

Figura 2 – Captura de tela mostrando a cópia para a área de transferência do link que dá acesso ao estudo.



Fonte: Elaborada pelo autor.

Dentro da tela de resultados do *Joker*, é possível realizar o *download* do dendrograma e dos dados de todas as participações do estudo.

Figura 3 – Captura de tela que exibe os botões mencionados na seção 2.3.

Fonte: Elaborada pelo autor.

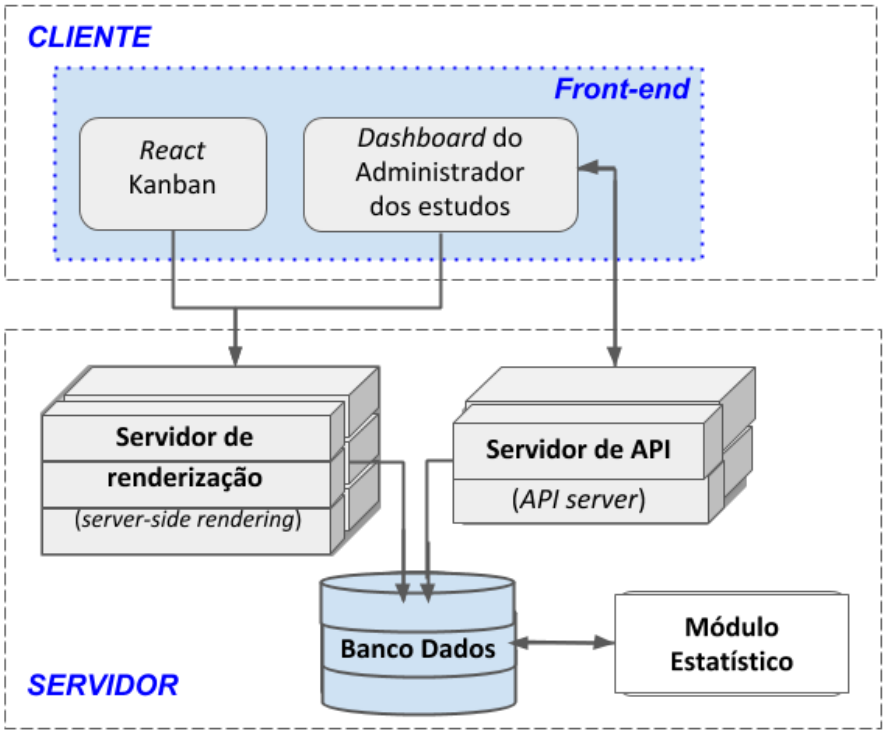
2.4 CONSUMINDO A API - (USUÁRIOS AVANÇADOS)

Através da *Joker API* é possível obter detalhes sobre o sistema e realizar *queries* no banco de dados de Joker. Trata-se de uma API RESTful em HTTP que pode ser facilmente consumida através de um sistema de requisições de qualquer linguagem de programação. Para informações sobre as rotas e tipos de resposta, ver seção 5.2.

3 ARQUITETURA DE SOFTWARE

O objetivo desta seção é fornecer ao leitor uma visão abstraída da funcionalidade de cada módulo da ferramenta *Joker*, além de suas interdependências. A leitura dessa seção é altamente recomendada a futuros contribuidores do projeto, a fim de facilitar a compreensão do funcionamento em alto nível de cada módulo antes de uma abordagem diretamente debruçada sobre a implementação direta dos mesmos em código-fonte.

Figura 4 – Arquitetura da ferramenta *Joker* - prioriza compatibilidade, simplificando assim, a integração entre os módulos do projeto.



Fonte: Elaborada pelo autor.

3.1 LINGUAGENS E *FRAMEWORKS*

O desenvolvimento da ferramenta *Joker* utiliza a biblioteca *open-source React Kanban*[4] como base de componentes, devido a sua semelhança (visual) com o processo de participação de uma sessão de *card sorting*. A partir dos estudos sobre a biblioteca *React Kanban*, foi constatado que ela possuía implementadas apenas as telas de manipulação de cartões, o suporte ao banco de dados *MongoDB*[5] e o *back-end* para *Server-Side Rendering (SSR)* construído usando *NodeJS*[6]. Nesse sentido, optamos por utilizar *ReactJS* para a construção do painel do pesquisador e *NodeJS* para a implementação do servidor de API. Para o módulo estatístico foi utilizada a linguagem *Python 3.7* em conjunto com as bibliotecas *Numpy* e *Matplotlib*. Todos os requisitos para a execução da ferramenta *Joker* podem ser visualizados na página do projeto no *Github*[7];

3.2 JUSTIFICATIVAS DE DECISÕES DE PROJETO

Com base nos expostos da seção 3.1, decidimos que o desenvolvimento da ferramenta *Joker* deveria ser voltado para a integração das estruturas da *React Kanban* com uma

[4] <https://github.com/markusenglund/react-kanban>
[5] <https://www.mongodb.com/>
[6] <https://nodejs.org/en/>
[7] <https://github.com/joaopedromattos/joker#setting-up-joker>

interface de administração e criação de estudos, junto com um Banco de Dados para armazenar e disponibilizar o conteúdo cadastrado por pesquisadores para participantes / usuários do *Card sorting*.

A modelagem da arquitetura da ferramenta *Joker* e a modelagem do Banco de Dados foram definidas com o objetivo de priorizar o aproveitamento de código e escalabilidade.

3.3 SOBRE OS MÓDULOS DO SISTEMA E SUAS RELAÇÕES

Os módulos ilustrados como mostra a Figura 4 foram implementados seguindo as características de sistemas iterativos Web (KUHR; HAMILTON, 2008) (FERNANDEZ; NAVóN, 2010), e têm suas respectivas funcionalidades descritas a seguir em alto nível.

(1.) **módulo *React Kanban*** - O objetivo principal deste módulo é permitir a execução da técnica *Card Sorting* pelos usuários, isto é, consiste nas telas de organização de cartões e de acesso ao estudo. Utiliza rotas do servidor de renderização para realizar a persistência de cada alteração do usuário no banco de dados em tempo real.

(2.) **módulo *Dashboard do Administrador dos estudos*** - consiste nas telas de planejamento e personalização da técnica *card sorting* pelo pesquisador. Neste módulo, o pesquisador poderá verificar os dados coletados e acompanhar os resultados gerados pelo Módulo Estatístico. Comunica-se com o *Servidor de API* para autenticação e armazenamento dos estudos e seus resultados.

(3.) **módulo *Servidor de Renderização*** - trata-se do *back-end* da arquitetura *Server-Side Rendering*, encarregado de tratar todas as requisições do usuário no navegador entregando as páginas web construídas em *React* e definidas nos módulos *React Kanban* e *Dashboard do Administrador de Estudos*. Comunica-se com o banco de dados para realizar a persistência das alterações no quadro de cartões a cada iteração.

(4.) **módulo *Módulo Estatístico*** - representa a execução das análises apropriadas para o agrupamento dos cartões. Este módulo é responsável por gerar a visualização gráfica da Arquitetura de Informação resultante (e.g. dendrogramas). Comunica-se com o banco de dados para recuperar todas as participações de determinado estudo.

(5.) **módulo *Servidor de API*** - encarregado de receber todas as *queries* do banco de dados que foram realizadas pela parte *front-end*. Mais especificamente, trata-se de uma API RESTful que usa o protocolo HTTP. Comunica-se com o banco de dados para realizar a autenticação do pesquisador e para salvar a criação/alteração dos dados de cada estudo.

4 MÓDULOS E COMPONENTES

Esta seção é voltada àqueles que procuram uma compreensão avançada acerca da implementação do código-fonte dos módulos do *Joker*. Nesse sentido, é recomendada a leitura da seção anterior a esta, de forma a possibilitar um primeiro contato com a arquitetura do sistema e seus aspectos técnicos em alto nível antes de iniciar uma abordagem mais detalhada.

4.1 ESTRUTURA DE ARQUIVOS

Podemos dividir a estrutura de arquivos de *Joker* em três grandes localizações iniciais:

Diretório Raiz	Comporta arquivos de configuração do projeto e de suas dependências, além armazenar os diretórios '/src' e '/apiResearcher'.
Diretório `'/src'`	Trata-se do diretório onde estão localizados todos os componentes visuais escritos em *React*, além de seus respectivos reducers (mecanismos de gerenciamento de estado) e servidor. Dessa forma, os arquivos referentes ao **React Kanban**, ao **Dashboard do administrador de estudos** e ao **Servidor de Renderização** são armazenados dentro desse diretório.
Diretório `'/apiResearcher'`	Diretório responsável por armazenar todos os arquivos referentes ao funcionamento do módulo "Servidor de API". A API é escrita segundo o padrão de projeto MVC (Model - View - Controller), logo as pastas subsequentes a esta seguirão esse padrão.

4.2 ESTRUTURA DE ARQUIVOS EM `'/SRC'`

Devido ao fato de o diretório `'/src'` conter mais de um módulo daremos destaque a sua estrutura de arquivos interna, de forma a delinear as fronteiras entre cada uma das partes de *Joker* através da tabela 1.

4.3 PERSISTÊNCIA DE ESTADOS E DADOS DA APLICAÇÃO

Dentro de *Joker*, realizamos dois tipos de gerenciamento de estados e de dados; o primeiro é realizado a partir do *Redux*, uma biblioteca de gerenciamento de estados para React, e o segundo ocorre através das chamadas à API e da atualização dos dados diretamente no MongoDB. Nas duas próximas subsubseções, trataremos em detalhes dos casos de uso relativos a cada mecanismo de controle de dados.

Diretório `'/app'`	Diretório responsável por armazenar todos os componentes dos módulos "Dashboard do Administrador"e "React Kanban".
Diretório `'/assets'`	Conjunto de imagens, ícones e configurações visuais utilizadas para a construção da identidade visual de *Joker*
Diretório `'/server'`	Diretório responsável pelo "Servidor de Renderização", composto por suas rotas e demais configurações.
Arquivo `'client.jsx'`	Arquivo raiz do Virtual DOM do React. O nó inicial da execução do React encontra-se dentro desse arquivo a partir do uso do componente inicial "App".

Tabela 1 – Definição em alto nível dos padrões de projeto utilizados em cada diretório consitituinte da pasta `'/src'`.

4.3.1 REDUX

Durante a execução de *Joker*, utilizamos a biblioteca *Redux*, através de suas instâncias chamadas de *reducers*, para armazenar dados de importância momentânea, isto é, *tokens* de autenticação, estudos de cada usuário, o estado da tela de organização de cartoes, etc.

Não cabe detalharmos os conceitos que abrangem a utilização das funções de gerenciamento de estados do *Redux*, pois podem ser encontrados na documentação oficial da biblioteca.

Ainda assim, é importante detalhar que todos os *reducers* de *Joker* podem ser encontrados sob o diretório `'/src/app/reducers/'`. Lembrando que, apesar de estarem todos em arquivos separados, eles são unidos dentro do arquivo `'/src/app/reducers/index.js'` através da função `'combineReducers'`, também disponível através do *Redux*.

4.3.2 MONGODB

O banco de dados *MongoDB* é utilizado para a persistência dos dados de maior importância para o funcionamento da aplicação, os dados de cada estudo, de suas respectivas participações e do *login* do pesquisador responsável.

Mais detalhes sobre a modelagem do banco de dados, como *Collections* e tipos de dados, estão expostos na subseção seguinte.

4.4 MODELAGEM DO BANCO DE DADOS

Para realizarmos o armazenamento dos dados dos pesquisadores e suas respectivas pesquisas, modelamos o banco de dados *MongoDB* conforme três *models* que, para o banco de dados, serão interpretados como *Collections* (o equivalente às tabelas do modelo modelo relacional).

Os *models* podem ser encontrados sob o diretório `"/apiResearcher/models"`. Dentro da pasta em questão, temos os três modelos utilizados dentro do banco de dados de *Joker*:

(1.) **`apiResearcher/models/board.js`** - Trata-se da *Collection* que modela o tipo de dado responsável por armazenar o estado final (em JSON) de uma participação de uma determinada pesquisa. **Dessa forma, caso exista interesse em realizar algum tipo de processamento de dados com base nos resultados de determinado estudo, será nessa *Collection* que o desenvolvedor deve buscar seus dados.**

Figura 5 – Captura de tela do código-fonte do arquivo `Board.js`.

```
var mongoose = require("mongoose");

var boardSchema = new mongoose.Schema({
    studyId: { type: mongoose.Schema.Types.ObjectId, ref: 'Study' },
    title: String,
    color: String,
    valid: { type: Boolean, default: false },
    lists: [{
        title: String,
        cards: [
            {
                color: String,
                text: String
            }
        ]
    }]
}, { timestamps: true})

module.exports = mongoose.model('Board', boardSchema);
```

Fonte: Elaborada pelo autor.

(2.) **`apiResearcher/models/researcher.js`** - Responsável por armazenar os *ids* dos estudos o `authId` de um pesquisador. O campo *authId* é um dos dados retornados da API do *Firebase*[8], serviço externo que utilizamos para autenticação de usuários. Dessa forma, simplificamos o sistema terceirizando toda a preocupação com gerenciamento de senhas e outros dados de acesso à ferramenta.

(3.) **`apiResearcher/models/study.js`** - Modelo responsável por armazenar os dados de um estudo, como os cartões e o nome do estudo. São aos *ids* desta *Collection* que o campo `studies` da *Collection* `Researcher` faz referência.

[8] <https://firebase.google.com/>

Figura 6 – Captura de tela do código-fonte do arquivo `Study.js`.

```
var mongoose = require('mongoose');

var studySchema = new mongoose.Schema({

    name: String,
    objective: String,
    cards: {
        type: [{ name: String, description: String }]
    },
}, { timestamps: true })

module.exports = mongoose.model('Study', studySchema);
```

Fonte: Elaborada pelo autor.

Figura 7 – Captura de tela do código-fonte do arquivo `Researcher.js`.

```
var mongoose = require('mongoose')

var researcherSchema = new mongoose.Schema({
    authId: String,
    email: String,
    name: String,
    studies: [{ type: mongoose.Schema.Types.ObjectId, ref: 'Study' }]

});

module.exports = mongoose.model("Researcher", researcherSchema);
```

Fonte: Elaborada pelo autor.

4.5 MÓDULO ESTATÍSTICO

Durante a fase de obtenção de resultados da ferramenta *Joker*, o módulo estatístico escrito em *Python 3* é utilizado para gerar o dendrograma e a limpeza dos dados em JSON. Nesse sentido, o objetivo desta seção é explicar como um script em *Python* foi acoplado à API escrita em *Node.js* e conceder uma visão geral do funcionamento do código-fonte dentro do arquivo `/apiResearcher/statisticalModule/cardClustering.py`, responsável por toda a funcionalidade do módulo em questão.

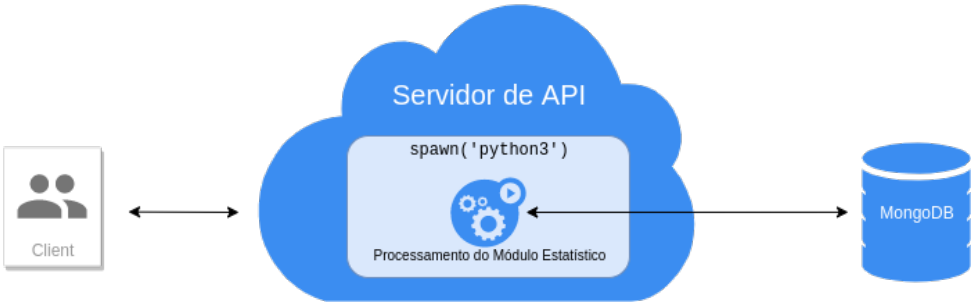
4.5.1 ACOPLANDO O MÓDULO À API

O acoplamento do módulo estatístico foi realizada através da biblioteca `child_process`[9] de gerenciamento e criação de processos do *Node.js*, mais especificamente da função `spawn`, responsável por instanciar um novo processo para a execução de código escrito em *Python*.

Assim que o Servidor de API recebe uma requisição na rota `/getResults/`, ele executa a função `getResults` do *controller* `/apiResearcher/controller/getResults.js`.

Já dentro da `getResults`, a função `spawn` é chamada, de forma a instanciar o script `/apiResearcher/statisticalModule/cardClustering.py`, que realiza o processamento dos dados do estudo desejado, e responde à requisição com um dendrograma e um arquivo JSON, que serão hospedados pelo Servidor de API. Ver figura 8 para uma referência gráfica acerca desse processo de obtenção de resultados.

Figura 8 – Ilustração em alto nível do processo de obtenção de resultados.



Fonte: Elaborada pelo autor.

A figura 9 ilustra em alto nível a ordem dos procedimentos realizados dentro do módulo estatístico, de forma a esclarecer a complexidade das etapas de manipulação de dados que acontecem durante uma requisição para obtenção de resultados.

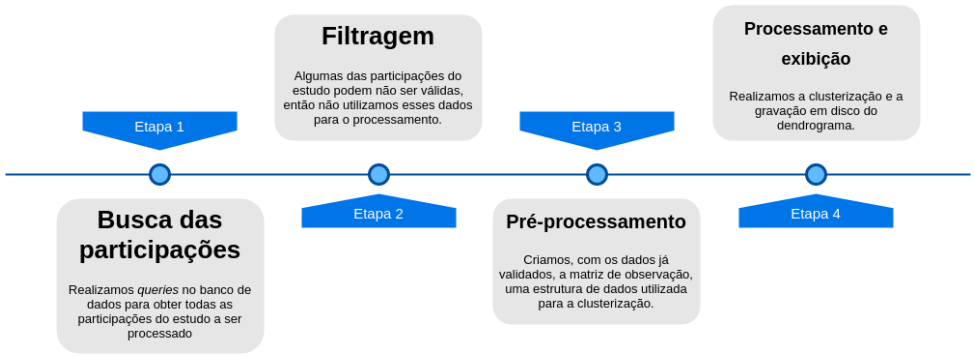
Para obter mais detalhes sobre como o procedimento de processamento de resultados é realizado, incluindo a limpeza dos dados e a utilização da biblioteca *Matplotlib* para a geração dos dendrogramas, consulte o código-fonte original no arquivo `/apiResearcher/statisticalModule/cardClustering.py`.

4.6 VIRTUALIZAÇÃO E *CONTAINERS*

Utilizamos a ferramenta de virtualização *Docker*[10] para modularizar a instalação e a compatibilidade do código fonte da aplicação para futuros *deploys*. Nesse sentido,

[9] <https://nodejs.org/api/child_process.html>
[10] <https://www.docker.com/>

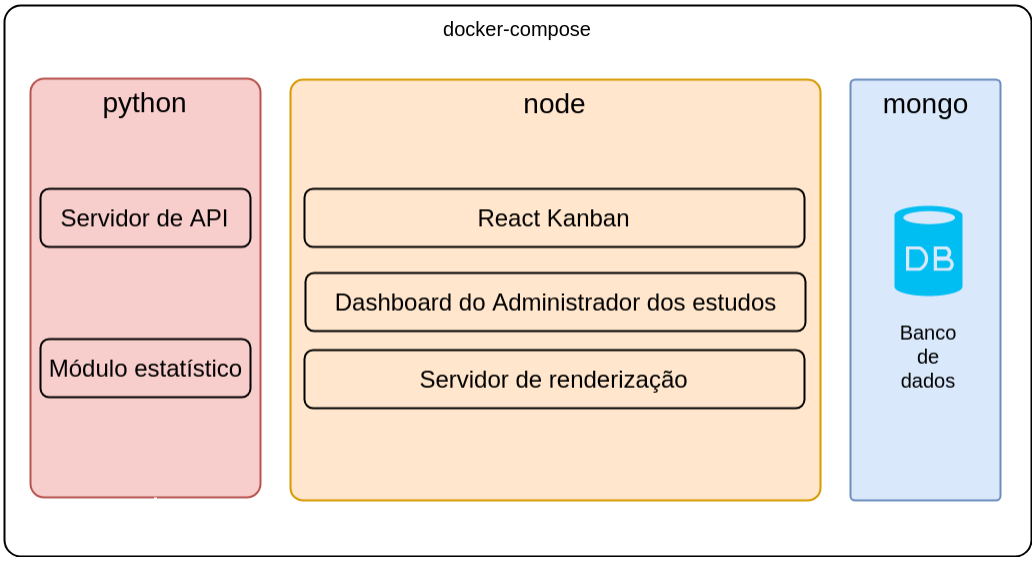
Figura 9 – Ordem dos procedimentos realizados dentro do módulo estatístico.



Fonte: Elaborada pelo autor.

utilizamos três imagens *Docker*: *"mongo"* (MongoDB), *"node"* (NodeJS) e *"python"* (Python).

Figura 10 – Ilustração que relaciona a rede virtual de *contêineres* Docker à arquitetura da ferramenta *Joker*.



Fonte: Elaborada pelo autor.

Além disso, foi utilizada a ferramenta de orquestração *Docker Compose* para criar uma rede virtual que integra os serviços dos três contêineres. A relação dessa rede virtual com os componentes da arquitetura é apresentada na Figura 10.

Todos os `Dockerfiles` e arquivos de configuração do *docker-compose* podem ser encontrados sob os diretórios `'/'` e `'/apiResearcher'`.

5 REFERÊNCIA DA API

5.1 DIRECIONAMENTO DA SEÇÃO

Esta seção é voltada àqueles que desejam realizar uma coleta de dados disponíveis na ferramenta Joker através do acesso à API. Nesse sentido, é recomendada a leitura das seções 3 e 4, além de conhecimentos no consumo de APIs RESTful na linguagem de programação à sua escolha.

5.2 ROTAS E TIPOS DE DADOS

Para uma consulta mais eficiente à esta documentação, recomenda-se ter em mente o *Controller* (disponíveis em `/apiResearcher/controllers`) responsável pela consulta desejada, uma vez que todas as rotas estão dispostas pelas funções dentro de cada um desses arquivos.

- `"/researchers"`
  - POST
    `createResearcher()` - Cria um novo perfil de pesquisador conforme dados passados pelo *body* da requisição.

- `"/researchers/email=:email"`
  - GET
    `getResearcher()` - Retorna o perfil de um pesquisador de acordo com o parâmetro da rota.
  - PUT
    `updateResearcher()` - Atualiza o perfil do pesquisador de acordo com os parâmetros passados pelo *body* da requisição.
  - DELETE
    `deleteResearcher()` - Remove um pesquisador do banco de dados através do *id*.

- `"/studies"`
  - POST
    `createStudy()` - Cria um novo estudo com os dados presentes no *body* da requisição.

- `"/studies/_id=:_id"`

  - GET
    `getStudy()` - Retorna um estudo através do *id* passado nos parâmetros da rota.
  - PUT
    `updateStudy()` - Atualiza um estudo com base nos dados passados pelo *body* da requisição.
  - DELETE
    `deleteStudy()` - Remove um estudo do banco de dados através do *id* passado nos parâmetros da rota.

- `"/boards"`
  - GET
    `listBoard()` - Retorna todas as participações em todos os estudos da plataforma *Joker*. **(Cuidado, esta requisição pode demorar muito!)**

- `"/boards/studyId=:studyId"`
  - GET
    `getBoard()` - Retorna todas as participações de determinado estudo.
  - POST
    `createBoard()` - Cria uma nova participação em um estudo.

- `"/boards/_id=:_id"`
  - PUT
    `updateBoard()` - Atualiza uma determinada participação num estudo de acordo com os dados passados pelo *body* da requisição.
  - DELETE
    `deleteBoard()` - Remove uma participação do banco de dados.

- `"/fetchBoard/_id=:_id"`
  - GET
    `createWelcomeBoard()` - Retorna o estado inicial do quadro de participação no experimento de *Card Sorting*.

- `"/getResults/studyId=:studyId"`

  - GET
    `getResults()` - Dispara a atuação do módulo estatístico para o processamento dos dados do estudo passado por parâmetro. **Retorna o endereço do gráfico do dendrograma, hospedado no servidor de API**.

- `"/getResults/dendrogram/studyId=:studyId"`
  - GET
    `downloadDendrogram()` - Realiza o *download* do *plot* do dendrograma do estudo.

- `"/getResults/json/studyId=:studyId"`
  - GET
    `downloadJson()` - *Download* do JSON que contém todos os dados processados do estudo em questão.

- `"/countResults/studyId=:studyId"`
  - GET
    countBoards() - Retorna o número de participações em determinado estudo, cujo *id* é passado por parâmetro.

6 AGRADECIMENTOS

Para a realização do processo de pesquisa e desenvolvimento do projeto *Joker* foi necessária a colaboração de diversas instituições, dentre as quais destacamos e agradecemos:

- Ao Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo (ICMC/USP) contribuiu, durante toda a execução do projeto, com as atividades de pesquisa desenvolvidas, a partir do oferecimento de laboratórios equipados com computadores e componentes de desenvolvimento de software necessários para o projeto;

- À Superintendência de Tecnologia da Informação da Universidade de São Paulo através de um de seus centros, o CeTI-SC[11], que mostrou-se muito solícito durante o processo de hospedagem da aplicação, a partir da abertura de portas nos roteadores da rede da universidade;

[11] <http://cetisc.sti.usp.br/>

- À infraestrutura do Laboratório *Intermídia* que foi absolutamente importante em diversas etapas do projeto, inclusive na hospedagem da ferramenta *Joker* em um de seus servidores;
- À **Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP)** pelos processos nº 2018/19323-8, nº 2017/15239 - 0 e nº 2015/24525 - 0, que financiaram esta pesquisa e o desenvolvimento deste projeto.

REFERÊNCIAS

FERNANDEZ, F.; NAVóN, J. Towards a practical model to facilitate reasoning about rest extensions and reuse. In: **Proceedings of the First International Workshop on RESTful Design**. New York, NY, USA: ACM, 2010. (WS-REST '10), p. 31–38. ISBN 978-1-60558-959-6. Disponível em: <http://doi.acm.org/10.1145/1798354.1798383>.

KUHR, M.; HAMILTON, D. Building executable service-oriented architectures with the ws-management specification. In: **Proceedings of the 2008 Spring Simulation Multiconference**. San Diego, CA, USA: Society for Computer Simulation International, 2008. (SpringSim '08), p. 315–324. ISBN 1-56555-319-5. Disponível em: <http://dl.acm.org/citation.cfm?id=1400549.1400594>.

ANEXOS

- SALGADO, A. L.; DIAS, F. D.; MATTOS, J. P. R.; FORTES, R. P. M.; HUNG, P. C. K. "Smart toys and Children's Privacy: usable privacy policy insights from a Card Sorting experiment". In: Proceedings of the *37th ACM Intl Conf. on Design of Communication.* Portland, OR, USA: ACM, 2019. (**SIGDOC'19**), p. 1–8. (To Appear).

Smart toys and Children's Privacy: usable privacy policy insights from a Card Sorting experiment

André de Lima Salgado
ICMC, University of São Paulo
São Carlos, SP, Brazil
alsalgado@usp.br

Felipe Silva Dias
ICMC, University of São Paulo
São Carlos, SP, Brazil
fsdias@usp.br

João Pedro Rodrigues Mattos
ICMC, University of São Paulo
São Carlos, SP, Brazil
joao_pedro_mattos@usp.br

Renata Pontin de Mattos Fortes
ICMC, University of São Paulo
São Carlos, SP, Brazil
renata@icmc.usp.br

Patrick C. K. Hung
University of Ontario Institute of Technology
Oshawa, ON, Canada
Patrick.Hung@uoit.ca

ABSTRACT

Smart toys are new to the Internet of Things market, and its connectivity to the cloud have raised concerns about children's privacy. Parents and legal guardians have striven to protect the privacy of their owns. However, current approaches for privacy control still lack usability for lay people. In this paper, we have explored the use of Card Sorting to enhance the usability of a privacy control for smart toys. Our goal was to identify and describe benefits of this technique to the design of more usable privacy controls. For this reason, we conducted a case study with voluntarily participants. We chose a parental control model from the literature to be the subject of evaluation for the experiment. Therefore, we extracted 19 units of information from its interface, and put them into cards for the Card Sorting evaluation. After the experiment, we obtained 30 valid responses. From these responses we performed a cluster analysis to understand the best alternative to group privacy related contents. Our contributions include a new model for nutrition label style mobile parental privacy controls for smart toys, suggestion of Google Material Design icons to be applied as indication for groups of privacy policies and, finally, a six steps process to perform Card Sorting with cluster analysis that does not rely on users' discussions to compose the Information Architecture hierarchy.

CCS CONCEPTS

• **Security and privacy** → **Usability in security and privacy**; • **Human-centered computing** → *User models*; *User studies*.

KEYWORDS

Usability, Privacy, Privacy Policies, Smart Toy, Information Architecture, Card Sorting, Tool

SIGDOC '19, October 4–6, 2019, Portland, OR, USA

ACM ISBN 978-1-4503-6790-5/19/10…$15.00
https://doi.org/10.1145/3328020.3353951

ACM Reference Format:
André de Lima Salgado, Felipe Silva Dias, João Pedro Rodrigues Mattos, Renata Pontin de Mattos Fortes, and Patrick C. K. Hung. 2019. Smart toys and Children's Privacy: usable privacy policy insights from a Card Sorting experiment. In *The 37th ACM International Conference on the Design of Communication (SIGDOC '19), October 4–6, 2019, Portland, OR, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3328020.3353951

1 INTRODUCTION

Usable privacy and security is an emergent cross-disciplinary field, aimed to study usability of systems that help users to secure their sensitive information [11]. Once, usability and security were only seen as antagonistic, but the range of privacy threats has increased, and users required to make security decisions about their data [3, 14, 31]. Governments have also required that companies enable their citizens to be "aware of" or to control their privacy. For example, the General Data Protection Regulation (GDPR) enforced companies in the European Union to enable people to control their data sharing [1]. Usability has been seen as a bridge towards a more secure Internet of Things (IoT).

Smart toys are new to the IoT market and stands as a promising pedagogical approach. Hung [12] defines smart toy as:

> ... a device consisting of a physical toy component that connects to one or more toy computing services to facilitate gameplay in the Cloud through networking and sensory technologies to enhance the functionality of a traditional toy.

Because of smart toy's connectivity to the cloud, which has raised concerns about children's privacy, parents have striven to protect the privacy of their kids [26]. However, current approaches for privacy control (e.g. user authentication, anti-phishing, email security, social media privacy and privacy policies) still lack usability for lay people [4, 11, 22, 23]. Among these approaches, privacy policies are popular [29]. These policies are documents that inform users' consent about information sharing regarding a specific application [11, 30]. Nevertheless, these policies are usually long and complex [7], and designing usable privacy policy tools for laypeople still challenges practitioners [4, 22, 23]. The literature presents

[1] ec.europa.eu/commission/priorities/justice-and-fundamental-rights/data-protection/2018-reform-eu-data-protection-rules_en

advances to the design of usable privacy policy tools, among which the "nutrition label" metaphor, from Kelley *et al.* 2009, is a promising approach. It stands among the few approaches in studies about mobile usability of privacy policy tools [16, 24]. Although the literature presents different approaches to enhance usability of such tools, we observed that studies that explore techniques of Information Architecture (IA) for usable privacy tools are still a gap.

In this study, we aimed to identify: (i) usability improvements for smart toy privacy control interfaces, (ii) adaptations for the nutrition label approach for the domain of smart toys and (iii) indications for more usable privacy policy tools.

To achieve our goals, we conducted a case study with voluntarily participants. The participants used an online tool to perform their Card Sorting individually. Later, we conducted a cluster analysis to group participants' answers and generate the IA. To re-design the parental control, we also employed concepts from the Kelley *et al.*'s [15] nutrition label and Google Material Design guidelines.

The following sections present a literature review on usability, information architecture and privacy policy tools; the methods of this study; the results and discussions; and the conclusions and lessons learned.

2 BACKGROUND

2.1 Smart Toys and privacy issues

Smart toys is a recent topic, concerning connected toys which are new to the IoT market and stands as a promising pedagogical approach. Hung [12] defines smart toy as:

> ... a device consisting of a physical toy component that connects to one or more toy computing services to facilitate gameplay in the Cloud through networking and sensory technologies to enhance the functionality of a traditional toy.

Smart toys are connected to the cloud and, for this reason, they have raised concerns about children's privacy. As consequence, parents have striven to protect the privacy of their kids [26]. Although these new devices have raised such a concern, current approaches for smart toy privacy controls (e.g. user authentication, anti-phishing, email security, social media privacy and privacy policies) still lack usability, which becomes particularly important when users are lay people [4, 11, 22, 23].

Among the applications for privacy controls, privacy policies are popular [29]. These policies are documents that inform users' consent about information sharing regarding a specific application [11, 30]. Governmental acts, as The Personal Information Protection and Electronic Documents Act (PIPEDA)Office of the Privacy Commissioner of Canada at: www.priv.gc.ca/en/privacy-topics/privacy-laws-in-canada/, require the use of privacy policies when businesses are handling personal information for their commercial activity. However, these policies are usually long and complex [7], and designing usable privacy policy tools for laypeople still challenges practitioners [4, 22, 23]. The literature presents advances to the design of usable privacy policy tools, among which the "nutrition label" metaphor, from Kelley *et al.* 2009, is a promising approach. It stands among the few approaches in studies about mobile usability of privacy policy tools [16, 24]. Although the literature presents different approaches to enhance usability of such tools, we observed that studies that explore techniques of Information Architecture (IA) for usable privacy tools are still a gap.

Rafferty *et al.* proposed a model of parental control for smart toys. Their model proposes to provide parents (or legal guardian) with a mobile privacy parental control, with which they would be able to set privacy rules and also check commitment with governmental laws, such as PIPEDA. Analyzing the parental control model presented by Rafferty *et al.* [25, 26], we understand that the Nutrition Label model is more appropriate to the context of our research problem. We made that decision because the parental control of Rafferty *et al.* depends on the use of mobile devices and we believe that the Nutrition Label model is more appropriate to the mobile context [16, 24].

2.2 Usability and information architecture

Usability is defined by ISO/IEC 25066 as "*the degree to which a product can be used by specific users to achieve specific goals with effectiveness, efficiency and satisfaction in a specific context of use*" [9]. In addition, according to Lewis [18], usability is a sensitive concept, and can be defined as summative or formative. The author shows that summative usability aims to obtain measures and reach objectives of related projects to usability. Therefore, summative usability evaluation methods aim to quantify (assign a grade) to applications usability. According to Lewis, formative usability aims the diagnosis of usability problems. It's understood that the first step to form the usability of an application is to diagnose which are the usability problems exist in it to, then, correct them and, thereafter, form an advance in the application's usability. This study focuses in formative usability, since we understand that the state-of-the-art in researches of parental controls still require enhancements to form a more usable model of parental control. Therefore, we have used Card Sorting technique to evaluate the information architecture and, hence, the usability enhancement diagnosis of information architecture from enhancements in the information architecture.

Information Architecture (IA) models "*information and experiences products to support usability and discovery*" [19]. Since privacy policies are, usually, long and complex [29], IA enhancements in the privacy policies interface may offer precise benefits to the usability of them.

Card Sorting[2] has been employed to generate a good Information Architecture (IA). Such technique consists in asking groups of users to group named cards (with contents or features of a system) in categories that make sense for them. Card sorting can be open (all users group cards in not named categories and, then, name them), closed (groups of cards with predefined categories and names) or hybrid (groups of cards in predefined categories and names, but changing or creating a category is allowed) [8].

2.3 Usability and Privacy Policy Tools

Usable Privacy and Security and (UPS) is the research aiming to study usability of systems that assist users to administrate security and privacy of their data [7, 11]. To understand the UPS domain, it is necessary to understand each concept that makes up this field: security, privacy and usability. Information security can be understood as "*preservation of confidentiality, integrity, and availability of*

[2] www.usability.gov/how-to-and-tools/methods/card-sorting.html

information" [10], and privacy shows up as a critical requirement for information security [4]. In this context, privacy refers to control over sharing of personal data [2, 6, 17]. Usability is "*the degree to which a product can be used by specific users to achieve specific goals with effectiveness, efficiency and satisfaction in a specific context of use*" [9].

Initially, usability and security were seen as antagonistic, while users were seen as risks to information security [31]. As software applications became more pervasive, privacy risks have increased [6, 14]. As consequence, users were often required to take decisions about security [14] and seen as a great hope of the area [31]. UPS researchers seek to develop security and privacy technologies that are usable for a variety of users[7].

Whitten and Tygar [32] have shown five security properties make UPS-related problems difficult to work with traditional user interface design: *The Unmotivated User Property*, *The Abstraction Property*, *The Lack of Feedback Property*, *The Barn Door Property* and *The Weakest Link Property*. In this study, we explore the abstraction property. This property states about the abstraction of security and privacy policies and rules [32]. Although the UPS literature has achieved considerable advances, Garfinkel and Lipford [11] show that these properties remains indicative for researchers in the field. Usability of privacy policy tools remains one of the main themes in UPS [11].

To protect citizens' privacy and avoid unauthorized use, the *Federal Trade Commission* of United States of America (USA) demanded that every e-commerce should publish privacy policies related to their practices over the Web [11]. As large Web companies are located in the USA, this determination had an international impact. Consequently, the importance of privacy policy controls has grown. Despite this importance, users rarely read such policies, and the usability of these controls in the applications is one of the biggest challenges in the UPS area [4, 7, 11].

To improve usability of privacy policy tools, studies have explored alternatives to view and configure privacy policies. In this topic, the Expandable Grid [27, 28] has provided great insights for new tools. The Expandable Grid aimed to enhance the visualization and configuration of file permissions for the Windows XP interface. It was a matrix based interface, showing users as the upper axis, files as the vertical axis and permissions (read, write, execute, delete and administrate) as colored squares at the intersection between users and files. Reeder [27] also proposed an adaptation of the Expandable Grid to be in accordance to the W3C Platform for Privacy Preferences Project (P3P) [3].

Later, Kelley *et al.*[15] proposed improvements to the Expandable Grid interface, enhancing its usability. They made such improvements by simplifying it with based on the "nutrition label" paradigm. This paradigm was more familiar to users than the previous grid, because of its employment by the food industry. Kelley *et al.* adopted short labels to describe the vertical and upper axis of the P3P Expandable Grid. Meanwhile, they provided longer definitions at an additional screen, the "useful terms" page. They also adopted colored scales together with privacy symbols to indicate how information is collected and used. A legend provided explanation about the meaning of each symbol. According to them, users realized the benefits of the nutrition label paradigm when comparing policies, which was a positive usability related attribute. Although, they observed that users were confused by the symbols and unfamiliar with terms, which represent usability problems with the interface. The nutrition label interface was adapted to fit mobile interfaces by Vallee *et al.* [24].

[3]www.w3.org/P3P/

3 METHODS

In our study, we aimed to empirically describe: *(i) usability improvements for smart toy privacy control interfaces, (ii) adaptations for the nutrition label approach for the domain of smart toys and (iii) indications for more usable privacy policy tools.* To reach our goals, our process is structured as follows:

(1) Perform an open Card Sorting with a sample of potential users.
   - Provide a feasible amount of cards for users (≤ 20).
   - Each card must contain a short and representative content from the privacy policies. Privacy experts may provide these representative contents.
(2) Perform cluster analysis with the outcomes from the Card Sorting.
(3) Compare the resulted clusters against the nutrition label model [15].
(4) Identify which screens can be abstracted with the nutrition label, and which must be created.
(5) Transform the nutrition label with the mini-IA process (exemplified in Figure 1).
(6) Use the resulted clusters (3) and the transformed nutrition label (6) to prototype the new interfaces.

3.1 Participants

To achieve our goals, we first evaluated the IA of the conceptual parental control proposed by Rafferty *et al.*. For this reason, we conducted an open Card Sorting with potential users. We invited 42 participants to voluntarily take part in the Card Sorting, expressing their public opinion on how the information should be grouped. Because we had difficulties to find parents with available time to voluntarily participate, we invited female university students instead. This was done to be in accordance with the Brazilian Institute for Geography and Statistics (IBGE) statistics of motherhood in Brazil[4]. As indicated by the statistics, and to represent the profile of Brazilian mothers, the invited students had ages ranging from 20 to 29. To the best of our knowledge, there are no statistics on percentage of live births by age of the father at birth. Therefore, we did not sample male students in this study. Our sampling method is defined by this feasibility analysis [5].

3.2 Procedures

We carried out a Card Sorting session for each user, so that a user's opinion would not influence other user's opinion. We asked users to group cards according to their own preferences. Each each card contained a different information about parental control. To compose the cards, we retrieved terms from the parental control of Rafferty *et al.*. They summed 19 terms among all the screens of Rafferty *et*

[4]www.ibge.gov.br/estatisticas-novoportal/sociais/populacao/9110-estatisticas-do-registro-civil.html?=&t=destaques

al.'s model. Table 1 indicates the respective terms that we used as cards for the Card Sorting evaluation.To retrieve the terms from Rafferty's *et al.* prototype, we did not considered those terms at the menus. We did not consider those because they already represented information groups (IA), which was in accordance to the designers of the model. This study aimed to understand the conceptual model of IA from potential users' opinion and, for this reason, we could not consider the conceptual model from the parental control designers.

We decided to conduct a Card Sorting experiment because privacy policies are usually long and complex [29], and IA enhancements of privacy policy interfaces may benefit usability of privacy policy tools. In addition, to the best of our knowledge, this study innovates by exploring the use of Card Sorting to enhance the design of privacy policies. For this reason, we also collected lessons learned from our experiences to further studies. To reduce the required time to collect card sort responses, and because online Card Sorting tools often limit their features for free plans, we have developed our own online Card Sorting tool using the *React Kanban* librarygithub.com/markusenglund/react-kanban. Responses were collected as JavaScript Object Notation (JSON)[5] and, later, converted to Comma Separated Values (CSV) to be analyzed using R packages[6].

ID	Information
I1	Child
I2	Child information
I3	Purpose of Access to Child Data
I4	Contact details of parent/guardian
I5	Privacy Policy
I6	Review Privacy Policy
I7	Agreed with the privacy policy service
I8	Show all privacy rules
I9	Description of the Privacy Rule
I10	Create new privacy rule
I11	Review and add privacy rule
I12	Enable privacy rule
I13	Disable Privacy Rule
I14	Receive updates via email
I15	Authorize access to GPS
I16	Mobile service authorized to access children's data
I17	Obligations and Retention of Data
I18	Main Control of Access to Child Data
I19	Choose Recipients to receive child data

Table 1: Terms used for the cards.

In sequence, we adapted the nutrition label interface [15] using the mini-IA process [21] to compose a Android mobile interface. We understand that the Nutrition Label model is appropriate to visualize and configure privacy policies in mobile devices, because it has been used as basis to support the design of mobile privacy policy tools [24]. In addition, the parental control model of Rafferty *et al.* depends on the use of mobile devices. The mini-IA process is based on the logical reading order (Western style) to turn the table in a unique column. Figure 1 illustrates the mini-IA process, transforming a tabular interface (columns and rows) into a mobile interface with a unique column with hierarchically displayed information. In addition to the adaptations made to the nutrition label, we also defined interface elements using Google Material Design guidelines for mobile devices [7]. Finally, by conducting these steps, we have observed lessons learned in this study to indicate how the (re)design processes might be structured for further practical cases.

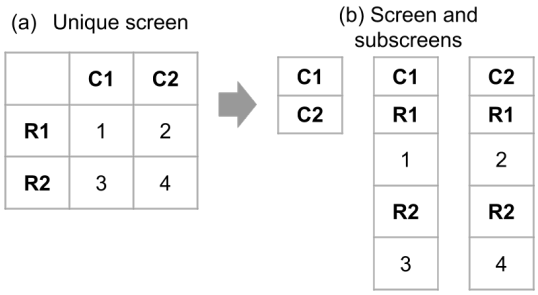


Figure 1: The mini-IA process illustrated. Transforming (a) table columns (C#) and rows (R#) in (b) unique column screens for mobile device interfaces.

The following section shows the results from our methods and provide deeper details on the process to re-design the parental control model.

4 RESULTS AND DISCUSSIONS

4.1 Card Sorting Experiment

We collected 30 valid responses from the Card Sorting sessions. These responses had, at least, two categories of cards grouped by the participants. From 42 participants, we took out 12 registered answers that were not completed (when participants started the grouping process but did not use all terms). To analyze the results from the Card Sorting, we used the Ward method to group participants' answers based on Euclidean distance [20]. Euclidean distance is suitable for the case because the results of Card Sorting can be represented in a two dimensions Euclidean space. We decided to employ cluster analysis to evaluate the outcomes from the Card Sorting because we could not have all participants together to discuss the final IA.

The results obtained were analyzed and are presented at Figure 2. As shown at the table, we observed two main ramifications in the resulted dendrogram. The lower branch presents a greater degree of similarity between the terms. It includes terms related to privacy policies and creation of privacy rules. The upper branch presents less similarity between its terms. In it, a subgroup stands out with greater similarity between the information, represented by the information I1 (Child), I2 (Child information) and I4 (Details of the father, mother or guardian).

[5] www.json.org/
[6] stat.ethz.ch/R-manual/R-patched/library/stats/html/hclust.html
[7] material.io/design/

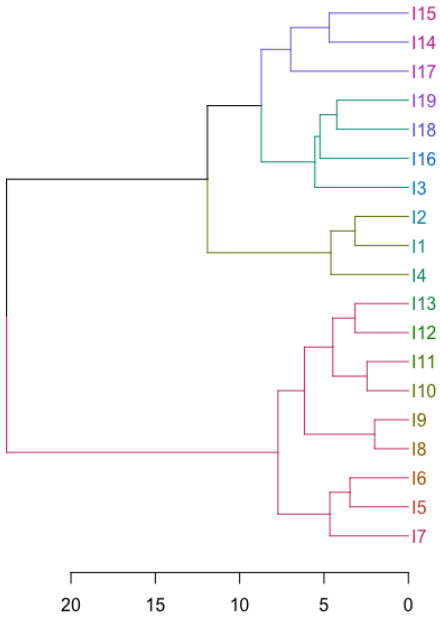


Figure 2: Dendrogram representing the IA as resulted from the Card Sorting and cluster analysis.

The results showed that some features that were presented in more than one screen of the prototype of Rafferty *et al.* could, according to the observations of this study, be presented on the same screen. Some examples of that are: the layout in different privacy policy display screens (I5); task of reviewing the privacy policy (I6); description of the privacy rule (I9); review and add privacy rule (I11); contact details of the parent or guardian (I4); and information of the child (I2).

4.2 Prototyping the new Parental Control

To prototype our parental control, we first created an Android like version of Rafferty *et al.*'s model (e.g. see Figure 3). This was done to enable necessary comparisons in this study. For this version, we performed as few adaptations as possible, so that the model could remain as much original as possible. Thereafter, we adapted the nutrition label model and considered the adaptation results to construct the new prototype.

To adapt the nutrition label model, we considered the results from the open Card Sorting and cluster analysis. We compared these results against the nutrition label model. We found that the terms I15 (Authorize access to GPS), I18 (Main control of access to the child's data) and I3 (Purpose of access to the child's data) could be removed, because the nutrition label already contains those terms among its terms and symbols. For this reason, we understood that repeating these terms would represent excessive information in the interface. As indicated by the usability principles of Nielsen [13], any excess information competes with relevant information. The results of the removal of excessive information for the contents represented by the nutrition label model can be visualized in the comparison between the interfaces represented at Figure 3 (Model of Rafferty *et al.*) and at Figure 4 (our prototype).

Figure 3: Main sections of the "Privacy Rules' category of the prototype proposed by Rafferty *et al.* .

In sequence, we adopted the strategy for mini-IA [21] to allocate the information in appropriate groups, logically displayed according to users' reading style (Western style). Figure 1 illustrates the process of reshaping the (a) nutrition label tabular interface into (b) multiple single-linear interfaces, as recommended by Nielsen and Budiu [21]. We call this the mini-IA process. As shown at Figure 1, we had to divide the table into more than one screen: one root screen containing column's labels only (C#) and multiple sub-screens containing the row's labels (R#) and the values (e.g. 1 to 4). At Figure 4, we show the result of the mini-IA process of Kelley's *et al.* nutrition label table [15] (see Figure 5 reported in the Kelley *et al.* study). For such, we re-organize the information according to the logical sequence of reading of the Western world (left to right followed by top to bottom), as suggested by Nielsen and Budiu [21].

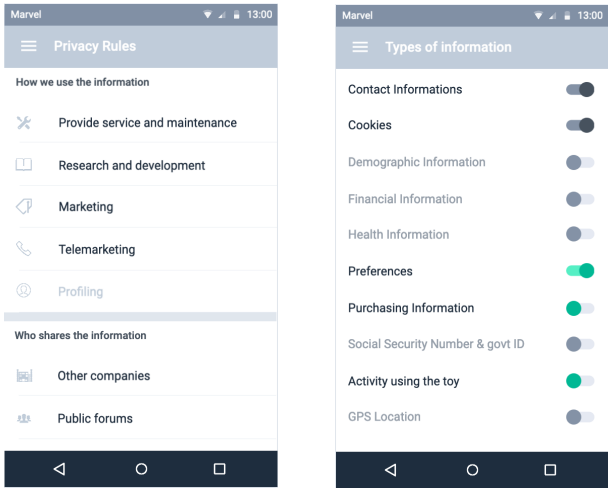


Figure 4: Our new prototype.

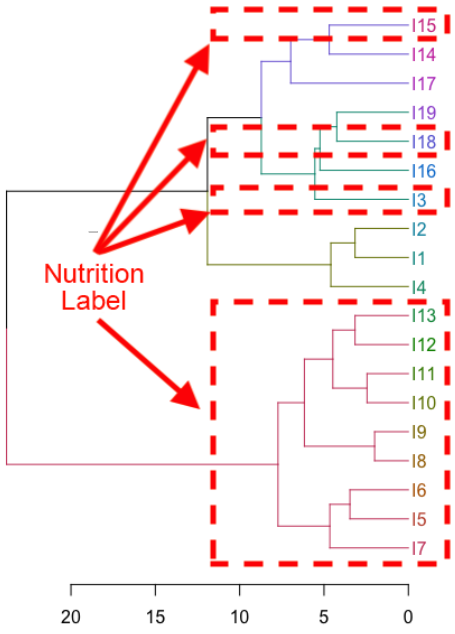


Figure 5: Comparing the dendrogram clusters with the information grouped at the nutrition label model.

To sum up, we used Google's *Material Design*[8] guidelines and assets from the Marvel App[9] prototyping tool. All prototyping steps were done in the Marvel App tool, considering the interface elements offered by the tool and which are in agreement with the *Material Design* guidelines by Google.

[8] material.io/design/

[9] We thank for the kind permission of the Marvel App to display images of their assets in our study.

As result, some screens of our prototype are very similar to the screens of Rafferty *et al.*'s interfaces, despite the use of menus (which in our proposal is hidden waiting for the user's event). Examples are the parent profile screens (see Figure 6) and the children (see Figure 7). On these screens, the navigation buttons have been replaced by the default navigation bar of *Material Design* and by creating a "Continue" button that also follows the standards of *Material Design*. This fact indicates that such screens from Rafferty *et al.*'s interface may have an appropriate usability for its purposes.

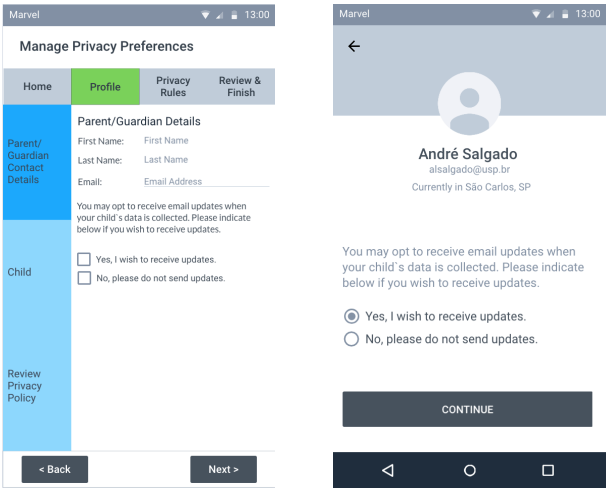


Figure 6: Subcategory of "Details of the Parent / Guardian". Rafferty *et al.*'s model on the left, and our new model on the right.

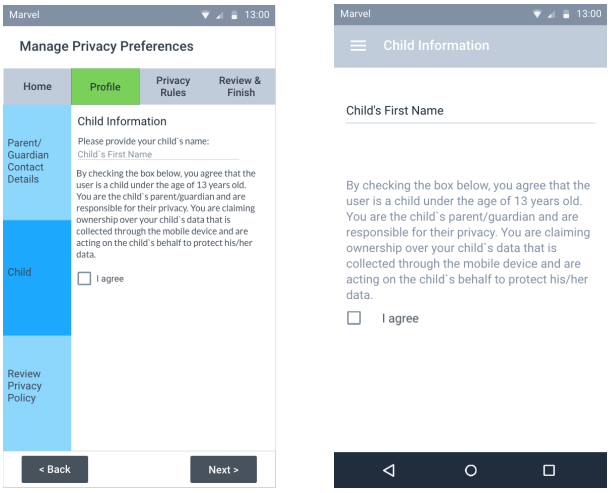


Figure 7: Subcategory of "Child Information". Rafferty *et al.*'s model on the left, and our new model on the right.

At the end of the re-design process, we reviewed the symbols used at the nutrition label interface. Because the nutrition label

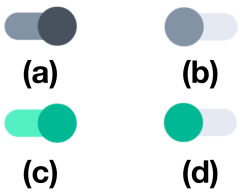


Figure 8: Icons'labels adapted from Kelley *et al.* [15]: (a) "*We will use the information in this way*"; (b) "*We will not collect or we will not use the information in this way*" ; (c) "*We will use the information in this way unless you opt-out*"; and (d) "*We will not use the information in this way unless you opt-in*" .

employs detailed symbols, which may be difficult to read on mobile screens, we suggested some updates by employing Google Material Design symbols. Since the use of *switches* in other Android configuration apps, we used variations of the *switch* to represent the four symbols of nutrition label ('!', 'OUT', '-' e 'IN'). Figure 8 shows how the definition suggested by Kelley *et al.* related to the icons chosen by us.

The adoption of the nutrition label model indicated that the creation of privacy rules for each mobile service (as required in the previous parental control model) may be complex for users. The nutrition label model concentrates all the rules in the same interface, and different services must comply with it. This may enhance the efficiency of the parental control by saving users' time and effort. Because efficiency is a requirement for usability, this finding may also enhance the usability of parental controls.

5 CONCLUSIONS

In this study, we aimed to identify and describe: *(i) usability improvements for smart toy privacy control interfaces, (ii) adaptations for the nutrition label approach for the domain of smart toys and (iii) indications for more usable privacy policy tools.*

Our findings provide a linear interface (non tabular) for the nutrition label model, aiming parental control for smart toys. We also suggest the use of material design icons to replace symbols from the original nutrition label interface. In addition, we reduced the amount of information from Rafferty *et al.'s* parental control model. Although privacy policies require long and complex information, there is a need to reduce to what is more relevant to the task. Detailed information can be place at secondary screens, as suggested by Kelley *et al.* in the nutritional label model. These modifications assisted us to compose a more efficient parental control in what regards the tasks: *reviewing, reading and creating privacy policies*; and *profile information of parents/guardians and children.* We understand that these were the main improvements that this study has proposed to the parental control model of Rafferty *et al.*.

In this paper, we created a six steps process (see Section 3) to perform Card Sorting with cluster analysis in evaluation of IA of mobile privacy policy tools. The results of our method corroborated the IA of the original nutrition label interface, and its groups (clusters) of information. We understand that the main advantage of employing method is that resulted clusters can evolve over time, as new Card Sorting answers may be collected. Thereafter, clusters can be updated by inserting the new answers an executing the cluster analysis again. This is not dependent on discussions with users to define the final IA, but only on data analysis and specialist opinions, which makes the Card Sorting faster to apply. We believe that adopting our process may help praticitioners to design more usable privacy controls. Nevertheless, the effectiveness of our process still needs validation. Future studies can investigate this topic and, also, provided deeper activities for the process. We also suggest, as future studies, to investigate the employment of this process in the design of privacy nudges [1].

Future studies may diagnose usability problems with our prototype and, then, enhance its quality. Further, we suggest exploring the benefits of using our prototype to generate prototypes for other domains of IoT privacy controls (e.g. connected and autonomous vehicles). Iconography studies are also suggested to better understand the use of icons to replace the symbols suggested by Kelley *et al.* [15] in their nutrition label. Also, future studies are suggested to investigate the differences on users' performance using our and Rafferty *et al.*'s model.

ACKNOWLEDGMENTS

We are also grateful to Anderson Garcia and all our colleagues from Intermidia Lab at ICMC/USP.

This study was supported by the grants 2015/24525-0, 2017/15239-0, 2018/19323-8 and 2018/26038-8 , São Paulo Research Foundation (FAPESP). This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001. This study was supported by The Brazilian National Council for Scientific and Technological Development (CNPq–MCTIC).

REFERENCES

[1] Alessandro Acquisti, Idris Adjerid, Rebecca Balebako, Laura Brandimarte, Lorrie Faith Cranor, Saranga Komanduri, Pedro Giovanni Leon, Norman Sadeh, Florian Schaub, Manya Sleeper, Yang Wang, and Shomir Wilson. 2017. Nudges for Privacy and Security: Understanding and Assisting Users&Rsquo; Choices Online. *ACM Comput. Surv.* 50, 3, Article 44 (Aug. 2017), 41 pages. https://doi.org/10.1145/3054926

[2] Alessandro Acquisti, Curtis R. Taylor, and Liad Wagman. 2016. *The Economics of Privacy.* SSRN Scholarly Paper. Social Science Research Network, Rochester, NY. https://papers.ssrn.com/abstract=2580411

[3] M. Aljohani, J. Blustein, and K. Hawkey. 2017. Participatory Design Research to Understand the Legal and Technological Perspectives in Designing Health Information Technology. In *Proceedings of the 35th ACM International Conference on the Design of Communication (SIGDOC '17).* ACM, New York, NY, USA, Article 39, 3 pages. https://doi.org/10.1145/3121113.3121240

[4] E. Bertino. 2016. Data Security and Privacy: Concepts, Approaches, and Research Directions. In *2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC)*, Vol. 1. 401. https://doi.org/10.1109/COMPSAC.2016.89

[5] Kelly Caine. 2016. Local Standards for Sample Size at CHI. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16).* ACM, New York, NY, USA, 981–992. https://doi.org/10.1145/2858036.2858498

[6] Hichang Cho, Bart Knijnenburg, Alfred Kobsa, and Yao Li. 2018. Collective Privacy Management in Social Media: A Cross-Cultural Validation. *ACM Transactions on Computer-Human Interaction* 25, 3 (June 2018), 1–33. https://doi.org/10.1145/3193120

[7] Luca Alexander De and Emanuel von Zezschwitz. 2016. Usable privacy and security. *it - Information Technology* 58, 5 (2016), 215–216. https://doi.org/10.1515/itit-2016-0034

[8] André de Lima Salgado, Fabrício Horácio Sales Pereira, and André Pimenta Freire. 2016. User-Centred Design and Evaluation of Information Architecture for Information Systems. In *Handbook of Research on Information Architecture and Management in Modern Organizations.* IGI Global, 219–236.

[9] ISO: International Organization for Standardization. 2016. ISO/IEC 25066:2016(en), Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for Usability — Evaluation Report. https://www.iso.org/obp/ui/#iso:std:iso-iec:25066:ed-1:v1:en

[10] ISO: International Organization for Standardization. 2016. ISO/IEC 27000:2016(en), Information technology — Security techniques — Information security management systems - Overview and vocabulary. https://www.iso.org/obp/ui/#iso:std:iso-iec:27000:ed-4:v1:en

[11] Simson Garfinkel and Heather Richter Lipford. 2014. *Usable Security: History, Themes, and Challenges*. SYNTHESIS LECTURES ON INFORMATION SECURITY, PRIVACY, AND TRUST, Vol. 5. Morgan & Claypool Publishers.

[12] Patrick C. K. Hung, Jeff K. T. Tang, and Kamen Kanev. 2017. Introduction. In *Computing in Smart Toys*, Jeff K.T. Tang and Patrick C. K. Hung (Eds.). Springer International Publishing, Cham, 1–5. https://doi.org/10.1007/978-3-319-62072-5_1

[13] Jakob Nielsen. 2018. 10 Heuristics for User Interface Design. https://www.nngroup.com/articles/ten-usability-heuristics/

[14] Julian Jang-Jaccard and Surya Nepal. 2014. A survey of emerging threats in cybersecurity. *J. Comput. System Sci.* 80, 5 (Aug. 2014), 973–993. https://doi.org/10.1016/j.jcss.2014.02.005

[15] Patrick Gage Kelley, Joanna Bresee, Lorrie Faith Cranor, and Robert W. Reeder. 2009. A "Nutrition Label" for Privacy. In *Proceedings of the 5th Symposium on Usable Privacy and Security (SOUPS '09)*. ACM, New York, NY, USA, 4:1–4:12. https://doi.org/10.1145/1572532.1572538

[16] Patrick Gage Kelley, Lorrie Faith Cranor, and Norman Sadeh. 2013. Privacy as part of the app decision-making process. ACM Press, 3393. https://doi.org/10.1145/2470654.2466466

[17] Spyros Kokolakis. 2017. Privacy attitudes and privacy behaviour: A review of current research on the privacy paradox phenomenon. *Computers & Security* 64, Supplement C (Jan. 2017), 122–134. https://doi.org/10.1016/j.cose.2015.07.002

[18] James R. Lewis. 2014. Usability: Lessons Learned and Yet to Be Learned. *International Journal of Human-Computer Interaction* 30, 9 (2014), 663–684. https://doi.org/10.1080/10447318.2014.930311

[19] Peter Morville and Louis Rosenfeld. [n. d.]. *Information Architecture for the World Wide Web*. http://shop.oreilly.com/product/9780596527341.do

[20] Fionn Murtagh and Pierre Legendre. 2014. Ward's Hierarchical Agglomerative Clustering Method: Which Algorithms Implement Ward's Criterion? *Journal of Classification* 31, 3 (Oct. 2014), 274–295. https://doi.org/10.1007/s00357-014-9161-z

[21] Jakob Nielsen and Raluca Budiu. 2013. *Mobile usability*. MITP-Verlags GmbH & Co. KG.

[22] Maggie Oates, Yama Ahmadullah, Abigail Marsh, Chelse Swoopes, Shikun Zhang, Rebecca Balebako, and Lorrie Faith Cranor. 2018. Turtles, Locks, and Bathrooms: Understanding Mental Models of Privacy Through Illustration. *Proceedings on Privacy Enhancing Technologies* 2018, 4 (2018). https://content.sciendo.com/view/journals/popets/2018/4/article-p5.xml

[23] Federica Paci, Anna Squicciarini, and Nicola Zannone. 2018. Survey on Access Control for Community-Centered Collaborative Systems. *ACM Comput. Surv.* 51, 1 (Jan. 2018), 6:1–6:38. https://doi.org/10.1145/3146025

[24] Hannah Quay-de la Vallee, Paige Selby, and Shriram Krishnamurthi. 2016. On a (Per)Mission: Building Privacy Into the App Marketplace. ACM Press, 63–72. https://doi.org/10.1145/2994459.2994466

[25] Laura Rafferty, Marcelo Fantinato, and Patrick C. K. Hung. 2015. Privacy Requirements in Toy Computing. In *Mobile Services for Toy Computing*, Patrick C. K. Hung (Ed.). Springer International Publishing, 141–173. http://link.springer.com/chapter/10.1007/978-3-319-21323-1_8

[26] Laura Rafferty, Patrick C. K. Hung, Marcelo Fantinato, Sarajane Marques Peres, Farkhund Iqbal, Sy-Yen Kuo, and Shih-Chia Huang. 2017. Towards a Privacy Rule Conceptual Model for Smart Toys. In *Computing in Smart Toys*. Springer, Cham, 85–102. https://doi.org/10.1007/978-3-319-62072-5_6

[27] Robert W. Reeder. 2008. *Expandable Grids: A user interface visualization technique and a policy semantics to support fast, accurate security and privacy policy authoring*. PhD Thesis. Carnegie Mellon University.

[28] Robert W. Reeder, Lujo Bauer, Lorrie Faith Cranor, Michael K. Reiter, Kelli Bacon, Keisha How, and Heather Strong. 2008. Expandable Grids for Visualizing and Authoring Computer Security Policies. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*. ACM, New York, NY, USA, 1473–1482. https://doi.org/10.1145/1357054.1357285

[29] F. Schaub, R. Balebako, and L. F. Cranor. 2017. Designing Effective Privacy Notices and Controls. *IEEE Internet Computing* 21, 3 (May 2017), 70–77. https://doi.org/10.1109/MIC.2017.75

[30] A. C. Squicciarini, D. Lin, S. Sundareswaran, and J. Wede. 2015. Privacy Policy Inference of User-Uploaded Images on Content Sharing Sites. *IEEE Transactions on Knowledge and Data Engineering* 27, 1 (Jan. 2015), 193–206. https://doi.org/10.1109/TKDE.2014.2320729

[31] Jeremiah D. Still. 2016. Cybersecurity Needs You! *interactions* 23, 3 (April 2016), 54–58. https://doi.org/10.1145/2899383

[32] Alma Whitten and J. D. Tygar. 1999. Why Johnny Can'T Encrypt: A Usability Evaluation of PGP 5.0. In *Proceedings of the 8th Conference on USENIX Security Symposium - Volume 8 (SSYM'99)*. USENIX Association, Berkeley, CA, USA, 14–14. http://dl.acm.org/citation.cfm?id=1251421.1251435