MACI - modelo para autoria colaborativa de programas de intervenções para dispositivos móveis

MACI - model for collaborative authoring of intervention programs for mobile devices

DOI:10.34117/bjdv8n8-026

Recebimento dos originais: 21/06/2022
Aceitação para publicação: 29/07/2022

Laurentino Augusto Dantas
Mestre em Ciência da Computação
Instituição: Universidade de São Paulo (USP) - Campus São Carlos
Endereço: Av. Trab. São Carlense, 400, Parque Arnold Schimidt, São Carlos - SP, CEP: 13566-590
E-mail: laurentino.dantas@usp.br

Joab Cavalcante da Silva
Doutor em Ciência da Computação
Instituição: Universidade de São Paulo (USP) - Campus São Carlos
Endereço: Av. Trab. São Carlense, 400, Parque Arnold Schimidt, São Carlos - SP, CEP: 13566-590
E-mail: joabms@usp.br

Raphael Christian dos Santos Oliveira
Graduado em Ciência da Computação
Instituição: Universidade de São Paulo (USP) - Campus São Carlos
Endereço: Av. Trab. São Carlense, 400, Parque Arnold Schimidt, São Carlos - SP, CEP: 13566-590
E-mail: rcsoliveria@usp.br

Maria da Graça Campos Pimentel
Doutora em Ciência da Computação
Instituição: Universidade de São Paulo (USP) - Campus São Carlos
Endereço: Av. Trab. São Carlense, 400, Parque Arnold Schimidt, São Carlos - SP, CEP: 13566-590
E-mail: mgp@icmc.usp.br

RESUMO
Neste artigo apresentamos o modelo para a autoria colaborativa de programas de intervenções para dispositivos móveis (MACI). O modelo proposto divide o processo de monitoramento de participantes por intervenções programadas em quatro etapas sequenciais e cíclicas. Como principais características do modelo pode ser citado a sua simplicidade e a definição precisa das tarefas e objetivos de cada etapa, o que permite que sejam definidos os atores, seus papéis, as tarefas a serem realizadas e os resultados esperados. Como prova de conceito do modelo foi desenvolvida a ferramenta de autoria colaborativa para a criação de programas de intervenções programadas fundamentados no ESM (FACEE), que aplica técnicas de Desenvolvimento de Software pelo Usuário Final (EUD) para ofertar a especialistas, com pouca ou nenhuma experiência em

desenvolvimento de software, a possibilidade de criarem programas de intervenções programadas seguindo o método de amostragem de experiência (ESM).

Palavras-chave: dispositivos móveis, tecnologia, notificação, intervenções programadas.

ABSTRACT
In this paper we present the model for collaborative authoring of intervention programs for mobile devices (MACI). The proposed model divides the process of participant monitoring by programmed interventions into four sequential and cyclical steps. The main characteristics of the model are its simplicity and the precise definition of the tasks and objectives of each stage, which allows the definition of the actors, their roles, the tasks to be performed, and the expected results. As a proof of concept of the model, a collaborative authoring tool was developed for the creation of programmed intervention programs based on the ESM (FACEE), which applies End User Software Development (EUD) techniques to offer experts, with little or no experience in software development, the possibility of creating programmed intervention programs following the experience sampling method (ESM).

Keywords: mobile devices, technology, notification, scheduled interventions.

1 INTRODUÇÃO

O método de amostragem de experiência, em inglês *'Experience Sampling Method'* (ESM), é um procedimento de pesquisa para estudar o que as pessoas fazem, sentem e pensam durante sua vida diária. Consiste em pedir aos indivíduos que forneçam auto-relatos sistemáticos em ocasiões aleatórias durante sua rotina normal [17]. Conjuntos desses auto relatos de uma amostra de indivíduos criam um arquivo de experiência diária. O ESM tenta minimizar o viés de memória, nos estudos ESM os participantes descrevem detalhes sobre seus atuais estados físicos ou mentais, em vários momentos durante o dia, sendo possível desta forma, capturar os contextos naturais de emoções ou sintomas.

Anteriormente Estudos ESM eram realizados através de diários, onde os participantes faziam anotações no decorrer do dia, normalmente havia algum dispositivo que informava aos participantes os momentos em que deveriam ser feitas as anotações. Atualmente os dispositivos móveis substituíram os cadernos de notas, ofertando uma grande quantidade de formas de interação com usuários, possibilitando o envio e recebimento de diversos tipos de mídias, além de permitir a comunicação com outros dispositivos e sensores.

Especialistas de diversas áreas como: medicina, educação, psicologia, dentre

outras, podem se beneficiar da criação de aplicativos ESM para o monitoramento e acompanhamento de grupos de pessoas, entretanto, o modelo tradicional de desenvolvimento de software pode se mostrar uma barreira difícil de ser transposta, seja pelo custo ou tempo de desenvolvimento, mas principalmente pela dificuldade de um especialista conseguir detalhar corretamente os requisitos do software para o desenvolvedor.

Uma forma de prover os especialistas da capacidade de criarem suas próprias aplicações ESM é através de técnicas de desenvolvimento de software pelo usuário final (EUD) [19], que é uma área de estudo que visa permitir que usuários que não sejam desenvolvedores de software, consigam criar suas próprias aplicações ou fazer configurações em dispositivos. O desenvolvimento de aplicações pelo usuário final permite que ele crie as aplicações conforme as suas necessidades. Outro ponto relevante é que na medida que a aplicação necessitar de modificações, o próprio usuário poderá realizá-las.

Apesar da literatura já apresentar um número relevante de trabalhos que abordem o desenvolvimento de plataformas que aplicam o ESM e se utilizam de técnicas de EUD, ainda é carente de modelos que definem o processo de criação de programas, coleta e análise de dados de dados pelos especialistas, conforme demonstrado por Silva [44].

Devido a essa carência da literatura, propomos o modelo para a autoria colaborativa de programas de intervenções para dispositivos móveis (MACI). O objetivo do MACI é definir todo o processo de criação e execução dos programas de intervenções para dispositivos móveis baseados no ESM.

Como prova de conceito do MACI foi implementada uma ferramenta de autoria colaborativa para a criação de programas de intervenções programadas fundamentados no ESM utilizando técnicas de EUD (FACEE). A FACEE foi desenvolvida com base no Método de Amostragem de Experiências e Intervenção Programada (ESPIM), que é disponibilizado através de uma plataforma que visa apoiar pesquisadores ou profissionais na criação e acompanhamento de programas de intervenção interativa que lhes permitam interagir de forma assíncrona com seu público-alvo remotamente, em seus ambientes naturais[32].

O restante deste artigo está organizado da seguinte forma. Na Seção 2, apresentamos trabalhos relacionados a EUD no domínio ESM, desenvolvimento de ferramentas de autoria, colaboração e controle de versões. Na seção 3, fazemos uma contextualização do ESM e do ESPIM. Na Seção 4 descrevemos o MACI. Na Seção 5

descrevemos o FACCE, nossa prova de conceito para validação do MACI . Na Seção 6 descrevemos as avaliações já realizadas juntamente com os resultados obtidos. Na Seção 7 apresentamos nossas considerações finais e apontamos para trabalhos futuros.

2 TRABALHOS RELACIONADOS

Para o desenvolvimento da ferramenta de autoria apresentada neste artigo, foi necessário um estudo multidisciplinar que envolveu as áreas de: ESM, EUD; colaboração; editores gráficos; e controles de versões. Nesta seção descrevemos trabalhos das áreas citadas que estão relacionados com esta pesquisa.

Barricelli e Valtolina [1] apresentaram uma linguagem visual e sua implementação e um sistema visual para o gerenciamento colaborativo de sensores da Internet das Coisas (IoT). O sistema, denominado SmartFit Rule Editor, foi projetado para ser usado por treinadores para monitorar e analisar a rotina de atletas. Serja et al. [35] descreveram o BEESM, um Aplicativo de programação de usuário final baseado em blocos que permitia a usuários inexperientes e programadores novatos desenvolver de forma rápida aplicativos para dispositivos e ambientes inteligentes.

Rough e Quigle [34] partindo do princípio que a adoção da EUD por usuários finais depende de vários fatores contextuais que não são bem compreendidos. Realizaram uma pesquisa visando determinar quais os fatores que influenciariam pesquisadores a adotar o desenvolvimento de aplicações do método ESM a partir de aplicativos para dispositivos móveis. Como resultado da pesquisa estabeleceram uma série de recomendações para o design de ferramentas EUD, de modo a permitir que não-programadores desenvolvam aplicativos para coletar dados de participantes em suas vidas cotidianas. As recomendações listadas por Rough e Quigle São um dos norteadores do presente trabalho.

Diversas tarefas exigem que os usuários criem gráficos com semântica e complexidade variadas. Gráficos sem semântica utilizam formas e conexões genéricas, como é o caso de um editor de formas básicas como Shaped Tools [37], ou com semântica associada, como no caso da criação de diagramas UML e fluxogramas - ambos suportados por draw.io, uma pilha de tecnologia de código aberto para a construção de aplicativos de diagramação [6], e por Lucidchart [21]. Ferramentas especializadas suportam edição e interação com elementos com semântica e comportamentos pré-definidos, como ferramentas de geometria dinâmica [12] ou simulações [24, 25]. Também existem ferramentas que empregam linguagem de marcação em vez de primitivas gráficas (por

exemplo, [22,29]).

A edição de gráficos colaborativa quase em tempo real é suportada, entre outros, quando os usuários empregam linguagem de marcação (por exemplo, [22, 29]), primitivos baseados em forma (por exemplo, [10, 11]), bem como diagramas associados a simulações ([21, 24]).

O suporte para edição colaborativa em tempo quase real síncrona exige tanto micro versão quanto macro versão, conforme destacado por Kuryazov et al. [15]. O primeiro identifica pequenas mudanças que, por sua vez, são trocadas entre colaboradores remotos e usadas na edição do histórico para suportar desfazer / refazer; no entanto, eles não são armazenados para fazer parte do histórico de versões. O último, por outro lado, registra explicitamente as mudanças para fornecer controle de versão.

Fraser [8] apresentou o método de Sincronização Diferencial (DS) para manter documentos sincronizados - é um algoritmo simétrico que emprega um ciclo interminável de diferença de fundo (diff) e operação de patch (difusa). O autor argumenta que o DS pode lidar com qualquer conteúdo, incluindo texto simples, rich text, bitmaps, gráficos vetoriais: o requisito é a disponibilidade de um algoritmo de diferença e um algoritmo de patch difuso para o conteúdo específico. A escalabilidade e a capacidade de resposta dependem da eficiência e precisão desses algoritmos.

No contexto da edição colaborativa de modelos, Kuryazov et al. [14–16], detalham sua arquitetura para suportar a modelagem colaborativa, que inclui componentes para macro versão (controle de versão) e micro versão. Este último é um componente sincronizador que recebe os *deltas* computados pelos clientes e os transmite aos colaboradores remotos.

3 ESM E ESPIM

Os especialistas podem contar com a ampla popularidade dos dispositivos móveis na sociedade para realizar o atendimento remoto que complementa o atendimento presencial [40]. Isso pode ser útil para apoiar, por exemplo, ações voltadas para a solidão, uma vez que é crescente o número de indivíduos que sofrem de solidão [2, 4], principalmente entre os idosos [28] . Porém, é importante que o atendimento remoto seja personalizado para o paciente e, ao mesmo tempo, não resulte em sobrecarga para o profissional.

A aplicação de programas de intervenção é ilustrada em um cenário hipotético em que um profissional de saúde monitora um paciente durante um período de tempo. O

profissional solicita um conjunto de tarefas a serem realizadas em horários pré-definidos do dia. Em uma situação hipotética que considera um cenário programado pelo terapeuta, para quando o paciente está ou não sozinho, como parte de um programa que contém um conjunto de perguntas diferentes para o caso do paciente estar ou não sozinho.

Estando o paciente está sozinho quando o programa de intervenção é acionado: o terapeuta pede uma mensagem de áudio e os sentimentos atuais do paciente; a seguir, o paciente é convidado a jogar um jogo (que é um programa externo); uma vez que o jogo terminar, pergunta novamente como o paciente se sente.

Não estando o paciente sozinho quando o programa de intervenção é acionado: neste caso, o programa de intervenção pede ao participante que envie uma *selfie* e informe quem é a outra pessoa; a última pergunta pede ao participante que informe como ele está se sentindo.

Os programas de intervenção ESPIM são projetados de forma colaborativa por especialistas no domínio e reproduzidos como um documento multimídia interativo em dispositivos móveis usados por seus usuários-alvo. Neste contexto, os especialistas de domínio são os autores do documento multimídia interativo e seus usuários-alvo são os consumidores.

De agora em diante, usaremos de forma intercambiável os termos especialistas de domínio e observadores. Da mesma forma, usamos os termos usuários-alvo e participantes de forma intercambiável.

Do ponto de vista da engenharia de documentos, os programas de intervenção ESPIM são documentos declarativos que são interpretados e processados por um aplicativo baseado em reprodutor quando acessados pelos usuários. Os programas de intervenção são documentos não textuais, uma vez que podem apresentar arquivos de texto, imagens, vídeos e áudio. Os programas de intervenção podem também conter formulários preenchidos e usados pelos especialistas do domínio para fazer perguntas e coletar informações dos usuários.

Os programas de intervenção também são documentos dinâmicos, pois sua apresentação pode depender da seleção de alternativas fornecidas pelo usuário. Além disso, os programas de intervenção são documentos centrais: eles podem iniciar aplicativos externos e receber dados correspondentes à interação do usuário com tais aplicativos. As intervenções são construídas na forma de fluxos não lineares: um determinado fluxo é executado como resultado da interação do usuário com o programa de intervenção, o que é possível porque diferentes respostas a uma questão de múltipla

escolha de alternativa pode levar a diferentes fluxos subsequentes da intervenção.

Os programas de intervenção ESPIM são construídos por meio de uma interface baseada na web. Ao criar um programa de intervenção, os especialistas de domínio informam quem são os participantes-alvo e quando os programas devem ser apresentados aos usuários (por exemplo, por meio de gatilhos baseados em tempo).

Para definir o que será apresentado ao usuário e como o usuário irá interagir com os conteúdos, os especialistas do domínio criam fluxos não lineares de mensagens e perguntas multimídia. Uma vez concluídos, os programas de intervenção são enviados aos dispositivos móveis dos usuários-alvo como documentos JSON junto com os conteúdos multimídia associados. Nos momentos especificados pelos autores, uma notificação interativa é enviada ao usuário: assim que o usuário responde à notificação, o conteúdo correspondente é reproduzido pelo aplicativo do receptor [42].

Uma intervenção é uma tarefa que o observador solicita ao participante para realizar. As intervenções são organizadas em um fluxo de trabalho de intervenção não linear (um gráfico direcionado) em que (a) uma e única intervenção foi definida como a intervenção inicial; (b) cada intervenção aponta para outra (próxima) intervenção ou é definida como uma intervenção final; e (c) pelo menos uma intervenção é definida como uma intervenção final. As tarefas são solicitadas aos usuários por meio de perguntas ou mensagens.

As tarefas de intervenção podem ser de diferentes tipos: questões abertas, questões de múltipla escolha, questões de múltipla opção, solicitação de mídia, ativação de uma atividade externa e mensagem. Um tipo particular de pergunta - múltipla escolha com uma única resposta - permite que cada alternativa (escolha) leve a uma intervenção diferente: isso resulta na criação de fluxos não lineares com vários caminhos (ou seja, um gráfico direcionado).

Os eventos podem ser programados para ocorrerem de forma regular, em determinado dia e horário, ou podem ser disparados quando determinada situação específica ocorrer, como por exemplo o participante ir a um determinado local, ou realizar determinada atividade que possa ser detectada por algum tipo de sensor. Como exemplo podemos citar se um participante for a academia, pela localização do smartphone isso pode ser detectado e pode gerar uma intervenção perguntando se ele gostou da atividade.

A interface de autoria do ESPIM é uma aplicação web que permite que vários especialistas acessem e editem o mesmo programa, porém adota um modelo pessimista, onde apenas um autor pode editar o programa por vez. Além de não possuir recursos para

a colaboração síncrona, a ferramenta de autoria do ESPIM carece de um histórico e um controle de versões. A ferramenta de autoria do ESPIM permite a cópia de programas, eventos e intervenções, o que permite um tipo de reuso restrito e simplificado dos componentes.

Para incentivar a implementação de plataformas que utilizam o método ESM e realizem a coleta de dados por dispositivos móveis, Cunha [45] apresentou todo o modelo de implementação do ESPIM.

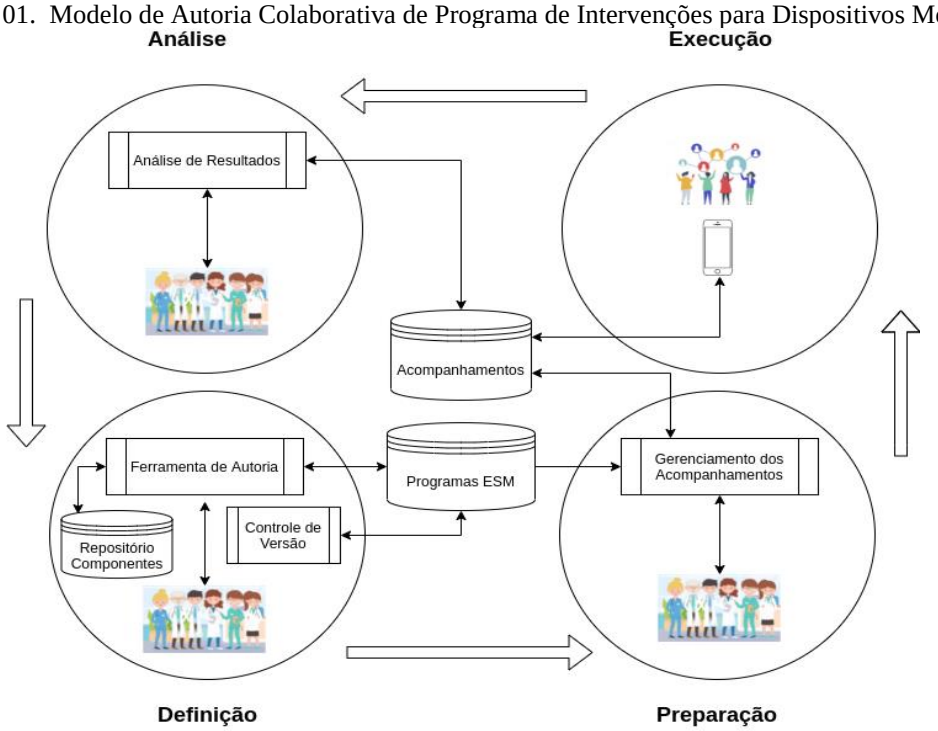
4 MACI - MODELO DE AUTORIA COLABORATIVA DE PROGRAMAS DE INTERVENÇÕES PARA DISPOSITIVOS MÓVEIS

O MACI tem por objetivo organizar em fases o processo de autoria de programas de intervenções em sistemas que aplicam o ESM com a utilização de recursos de EUD.

Com a elaboração do MACI pretendemos sistematizar o processo e definir as tarefas que são feitas em cada etapa e os resultados que se espera obter em cada uma delas.

Entendemos que desta forma, iremos apoiar o desenvolvimento de novas ferramentas e permitir que os desenvolvedores se concentrem em criar soluções específicas para uma determinada fase, sem ter que se preocupar em desenvolver plataformas completas como as que são apresentadas nos trabalhos publicados.

Figura 01. Modelo de Autoria Colaborativa de Programa de Intervenções para Dispositivos Móveis.



4.1 ETAPAS

Na nossa proposta elaboramos um modelo baseado em quatro etapas bem definidas, sequenciais, complementares e cíclicas, como ilustrado na Figura 01:

1. **Definição**: Autoria/Edição pelo Especialista. Na etapa de definição, os especialistas colaboram na definição e autoria dos programas de intervenção. Os programas criados serão gerenciados por um controle de versão, quando existir uma versão finalizada ela será disponibilizada no repositório de programas. No repositório poderão co-existir várias versões de um mesmo programa.
2. **Preparação**: Configuração Programa-Participante pelos Especialistas. Na etapa de preparação os especialistas a partir de um programa de intervenção disponível no repositório, cria uma observação, na observação são definidos os participantes e os observadores. Em uma definição sintética, uma observação é a aplicação de um programa de intervenções em um grupo de participantes com a supervisão de especialistas observadores. Os especialistas observadores não precisam ser os mesmos que criaram o programa de intervenções e podem ocorrer de forma simultânea várias observações com base em um mesmo programa de intervenções.
3. **Execução**: Entrega do programa e envio de respostas pelos usuários. Na etapa de execução, os participantes, definidos na etapa de preparação, irão receber as intervenções nos dispositivos móveis ou serão monitorados, conforme o que estiver definido no programa de intervenções. Esta é a fase onde a observação é efetivamente realizada, é nesta fase que os dados mais significativos para os especialistas são gerados.
4. **Análise**: Análise das Respostas pelos Especialistas. Na última fase, com o próprio nome sugere, é realizada a análise dos dados coletados. É implícito que os especialistas utilizarão os dados gerados como subsídio para o seu trabalho de pesquisa ou diagnóstico. Por outro lado, a análise também serve como referência para que os programas de intervenções sofram modificações, quando percebido que os dados gerados pelas intervenções não conseguem ilustrar de forma precisa aquilo que eles deveriam revelar. Desta forma o processo se torna cíclico, visto que o resultado da última fase serve como subsídio para o processo executado na primeira fase.

4.2 REPOSITÓRIOS

Com o objetivo de armazenar de forma organizada o resultado do trabalhos de cada etapa, o modelo propõe o uso de quatro repositórios:

1. **Repositório de Programas**: Os programas representam o conjunto de eventos e intervenções que serão enviadas para coletar dados dos participantes. Este repositório pode ser descrito como o local onde ficam os programas que estão em edição.
2. **Repositório de Componentes**: Um dos objetivos do modelo é proporcionar um ambiente onde os componentes criados por um especialista possam ser reutilizados, tanto por ele ou por outros especialistas quando ele permitir.
3. **Repositório de Históricos**: Os Históricos de alterações são importantes em dois aspectos: primeiro para garantir o sincronismo no processo de edição colaborativa do programas, e também para o controle de versão de programas ESM.
4. **Repositório de Programas ESM**: O repositório de Programa ESM é onde ficam armazenados os Programas ESM que serão utilizados para coletar dados. A partir dos Programas ESM se inicia o processo de Preparação.

5 FACEE - FERRAMENTA DE AUTORIA COLABORATIVA EUD NO DOMÍNIO ESM

Como prova de conceito do modelo proposto foi desenvolvida uma ferramenta de autoria colaborativa para que especialistas, com pouco ou nenhum conhecimento em desenvolvimento de software, possam desenvolver programas de intervenções programadas, de forma que tais programas possibilitem ao especialistas efetuarem pesquisas ESM com grupos de participantes através de intervenções programadas executadas nos *dispositivos móveis* dos participantes. Nesta seção iremos apresentar o modelo que guiará o desenvolvimento da ferramenta de autoria e seus componentes.

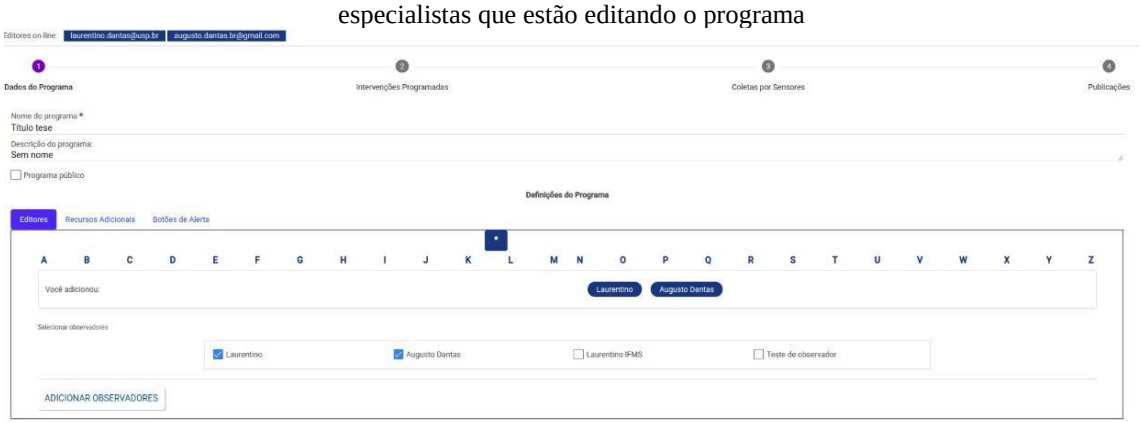
5.1 SISTEMAS

A ferramenta de autoria tem dois sistemas para dar suporte ao trabalho do especialista que são:

1. **Editor Colaborativo** - O Editor Colaborativo deve permitir aos especialistas criarem programas, eventos e intervenções. O editor mantém um

repositório de componentes, o que irá permitir que os componentes criados pelos especialistas possam ser reutilizados por quem os criou, e possa ser compartilhado com outros especialistas. Na figura 02 é possível observar a tela de edição de programas, nela é mostrada a aba de compartilhamento, na parte superior esquerda é mostrado os especialistas que estão editando o programa.

Figura 02. Tela do Editor colaborativo que mostra a tela de compartilhamento de programas e os especialistas que estão editando o programa



2. Sistema de Controle de Versões - Por sua vez, o controle de versões, será um sistema com função crucial dentro do modelo proposto. Pelo fato do do editor ser colaborativo, existe a necessidade de se manter um histórico de alterações, além disso, frente ao modelo cíclico proposto na seção 3.2 é muito importante compreender as diferenças entre as diversas versões que podem ser disponibilizadas de um mesmo programa. É importante esclarecer que toda alteração que é realizada em um programa gera uma nova versão, entretanto o especialista deve definir quando uma versão está pronta para ser utilizada na coleta de dados. Na Figura 03 é mostrada uma lista de programas, nela é mostrado o programa 140 que possui uma versão para coleta de dados e um o programa 141 que não possui nenhuma versão ainda pronta para realizar coleta de dados.

Figura 03. Lista de programas que mostra um programa que possui uma versão para coleta de dados e um programa que não possui nenhuma versão para coleta de dados.

A ferramenta FACEE foi desenvolvida como um sistema WEB cliente servidor de 3 camadas [27], seguindo os padrões atuais para desenvolvimento de sistemas de internet, possuirá um frontend desenvolvido em Angular [43], e um backend desenvolvido em Django [18], Django Rest [39] , com um banco de dados Postgres [5]. Os sistemas baseados na arquitetura cliente-servidor dividem o processamento das aplicações em várias máquinas [13], permite um compartilhamento mais fácil de recursos e reduz a replicação de dados [27].

5.2 EDITOR COLABORATIVO

Atualmente a colaboração é um recurso imprescindível em qualquer ferramenta de edição WEB, desta forma este recurso ganha grande destaque no desenvolvimento da FACEE. É de extrema importância que os especialistas possam criar programas de intervenções em conjunto, e que essa colaboração seja síncrona.

Para apoiar a abordagem baseada em edição colaborativa, foi criado um modelo de sincronização com base nos trabalho de Kuryazov et al. [14] [15] [16]. Foram desenvolvidos componentes de sincronização para: (a) coletar alterações (adicionar, excluir, atualizar) feitas pelos usuários, (b) transmitir essas alterações entre usuários remotos , (c) aplicar as mudanças a cada editor cliente.

Da mesma forma que Kuryazov et al. [15] [14] [16], que também apoiam a edição de modelos gráficos com semântica associada, estamos trabalhando para oferecer suporte a uma abordagem baseada em edição para a sincronização das mudanças feitas por usuários remotos. Para oferecer suporte a micro-versioning e macro-versioning, primeiro precisamos desenhar nosso sistema para permitir que todas as operações de edição possam ser desfeitas, para que possamos empregar a codificação delta com base nas operações de adição, exclusão e atualização.

Devido ao fato de ser um sistema web baseado em protocolo HTTP e sistema de banco de dados relacional, as operações realizadas pelos usuários são enviadas ao servidor constantemente. O servidor armazena os dados e cria identificadores para cada objeto armazenado. Os sistemas baseados em delta precisam manter o histórico de mudanças central, denominado macro-versão, este registro de mudanças central permite que todos os usuários fiquem em sincronia.

Quando a diferença entre versões é armazenada em um banco de dados, o banco de dados pode se tornar muito grande e complexo [16], fazendo com que os deltas de modelagem podem não sejam identificados e reutilizados. Visando evitar o problema que

pode ocorrer com o histórico de alterações por ser armazenado em um banco de dados, foi criado um histórico onde são registradas todas as operações em ordem de execução, cada registro possui data, hora, dados do objeto, editor e um número sequencial único .

5.3 SINCRONIZAÇÃO DE USUÁRIOS

A fim de fornecer um canal de comunicação que permitisse a sincronização entre diferentes usuários, optou-se pelo uso de um websocket. O protocolo WebSocket fornece um canal de comunicação bidirecional full-duplex que opera através de um único soquete na Web [20]. Na FACEE um WebSocket foi implementado para notificar todos os usuários quando ocorrerem alterações no banco de dados.

Conforme Ellis and Gibbs [7] em um editor colaborativo é muito importante que as alterações feitas por um usuário, sejam replicadas para os outros usuários que estão editando o mesmo documento da forma mais rápida e transparente possível, fazendo com que o editor colaborativo pareça tão responsivo quanto um editor de usuário único. Quando uma determinada operação ocorre em um editor colaborativo, esta operação deve ser propagada para outros usuários através do canal de comunicação. Um algoritmo de controle de simultaneidade deve ser usado para manter a consistência entre os diferentes usuários que editam um documento simultaneamente [9].

Para atender às necessidades do editor de intervenções, foi desenvolvida uma solução baseada nos algoritmos de controle de concorrência Jupiter [26] e SOCT4 [41]. Cada operação realizada recebe um número sequencial, um tíquete, que identifica aquela operação. Com base nos números dos tíquetes, o servidor é capaz de identificar as versões dos documentos de intervenção.

Quando um usuário vai editar um programa de intervenção, no momento em que solicita a edição do programa recebe a versão mais atual que se encontra no servidor, junto com o tíquete que identifica essa versão. Quando o usuário faz uma alteração, a alteração é enviada ao servidor junto com o tíquete, o servidor baseado no tíquete é capaz de realizar a operação mantendo a consistência do documento que está sendo editado. Após fazer a alteração, o servidor notifica todos os usuários que estão editando aquele documento, transmitindo a operação por meio do websocket. Deve-se notar que toda operação enviada em broadcast pelo servidor leva consigo o número do tíquete.

A figura 04 ilustra o processo de criação das operações e geração do tíquete, quando Ana abre o editor de intervenção ele está vazio e ela recebe o tíquete 0 (zero) em resposta. Ana cria a intervenção 1 e recebe como resposta o tíquete 1, quem define o id

para todas as intervenções é também o servidor. Ana cria a intervenção 2, recebe o tíquete 2 do servidor, finalmente Ana altera a intervenção 2 e recebe o tíquete 3 em resposta.

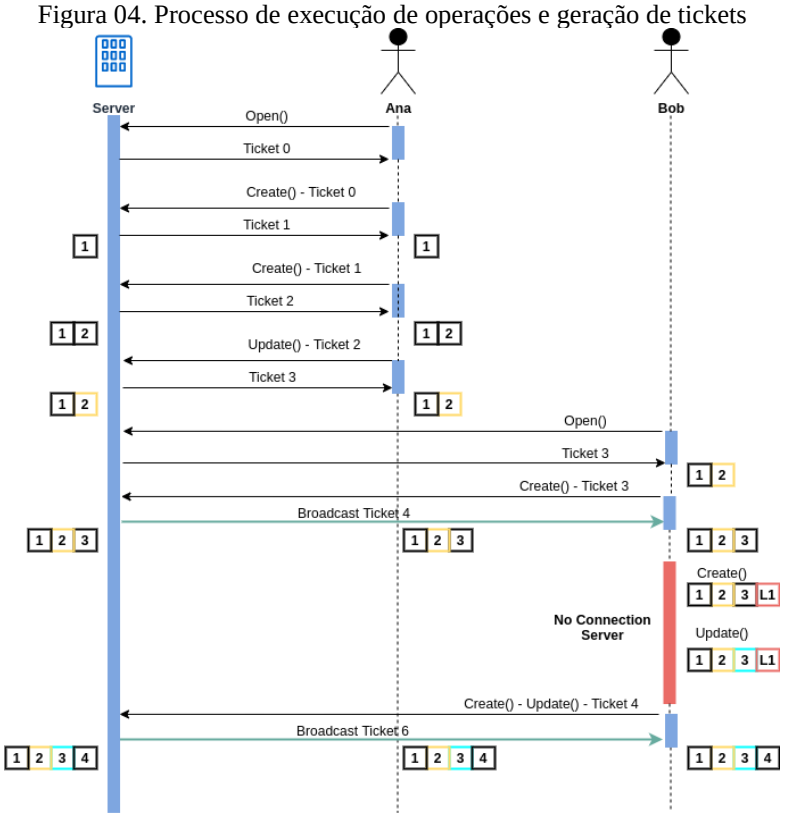
Quando Bob abre o editor de intervenção, ele recebe uma cópia do modelo de intervenção e o tíquete 3. Bob cria uma intervenção, o servidor envia por broadcast a todos os usuários (Ana e Bob) as informações da nova intervenção criada e o valor da nova bilhete. Todos os usuários são atualizados e sincronizados.

Em algum ponto, Bob perde sua conexão com o servidor, ele continua a trabalhar no modelo e cria uma nova intervenção e faz uma atualização. A intervenção criada receberá apenas uma identificação local (L1). Quando Bob retoma a conexão com o servidor, as alterações são enviadas ao servidor, a intervenção criada recebe um identificador do servidor e as operações realizadas são transmitidas por todos os usuários. Assim, todos os usuários e o servidor terão o mesmo modelo e tíquete.

5.4 CONTROLE DE CONCORRÊNCIA E MERGE

No exemplo da figura 04 enquanto Bob estava desconectado do servidor, Ana não executou nenhuma operação, no entanto, se Ana tivesse feito alguma alteração no modelo, o servidor deveria agir de forma que não houvesse perda de dados ou sobreposição de dados.

Figura 04. Processo de execução de operações e geração de tickets



Por meio de regras é possível implementar o controle de simultaneidade, tais regras definirão o que o servidor deve fazer ao receber uma lista de operações de um usuário que realizou operações sem estar conectado ao servidor. Em nossa implementação, quando o servidor receber uma lista de alterações e perceber que o tíquete está desatualizado, ele obedecerá às seguintes regras:

1) Se a intervenção foi atualizada após o tíquete, ela não pode ser excluída. A operação de exclusão será descartada e um tíquete não será gerado;

2) As atualizações recebidas serão realizadas e um tíquete será gerado;

3) Se a intervenção foi deletada e outra operação vem para deletá-la, a operação será descartada e não será gerado um tíquete;

4) Se for recebida uma atualização para uma intervenção que foi deletada, ela será recriada com o conteúdo que tinha na deleção e a atualização será aplicada, serão gerados dois tíquetes, um para a criação e outro para a atualização;

5) Todas as criações serão realizadas no final do arquivo de intervenção.

O algoritmo apresentado na figura 05 ilustra o processo de *merge* utilizado pelo editor colaborativo.

Figura 05. Algoritmo para *merge* de operações implementado no FACEE

```
while houver operações do
    if operação == create then
        create(Intervenção);
        Servidor.ticket ++;
        Intervenção.ticket = Servidor.Ticket;
        broadcast(create(Intervenção),Servidor.ticket);
    else
        if operação == update then
            if (Intervenção.status == deletada) then
                undoDelete(Intervenção);
                Servidor.ticket++ ;
                broadcast(undoDelete(Intervenção),Servidor.ticket);

                update(Intervenção);
                Servidor.ticket ++;
                Intervenção.ticket = Servidor.ticket;
                broadcast(update(Intervenção),Servidor.ticket);

            else
                update(intervenção);
                broadcast(update(Intervenção),Servidor.ticket);

            end
        else
            if operação == delete  User.ticket >=
            Intervenção.ticket then
                delete(Intervenção);
                Servidor.ticket ++;
                Intervenção.ticket = Servidor.ticket;
                broadcast(delete(Intervenção),Servidor.ticket);

            else
                cancel (Delete);
            end
        end
    end
end
```

No exemplo mostrado na figura 06, Ana abre o editor de intervenção e cria uma nova intervenção. Quando Bob abre o editor de intervenção, ele recebe o mesmo modelo e tíquete que Ana. Bob cria uma nova intervenção que é atualizada no servidor e distribuída pelo servidor. Bob perde a conexão com o servidor e executa 3 operações: uma atualização, uma criação e uma exclusão. Como ele não está conectado ao servidor, suas operações são realizadas apenas localmente.

Enquanto Bob não está conectado ao Servidor Ana, ele também executa três operações, uma exclusão, uma criação e uma atualização. Por estar em conexão com o servidor, as operações de Ana são sincronizadas com o Servidor, portanto, o servidor e Ana têm o mesmo modelo e valor de tíquete.

Quando Bob consegue se reconectar ao servidor, ele envia todas as operações que realizou. Ele verifica o valor do tíquete e é capaz de determinar como o modelo estava quando Bob realizou suas operações e o que aconteceu depois que Bob perdeu sua conexão. O servidor recebe as operações de Bob e as executa da seguinte maneira:

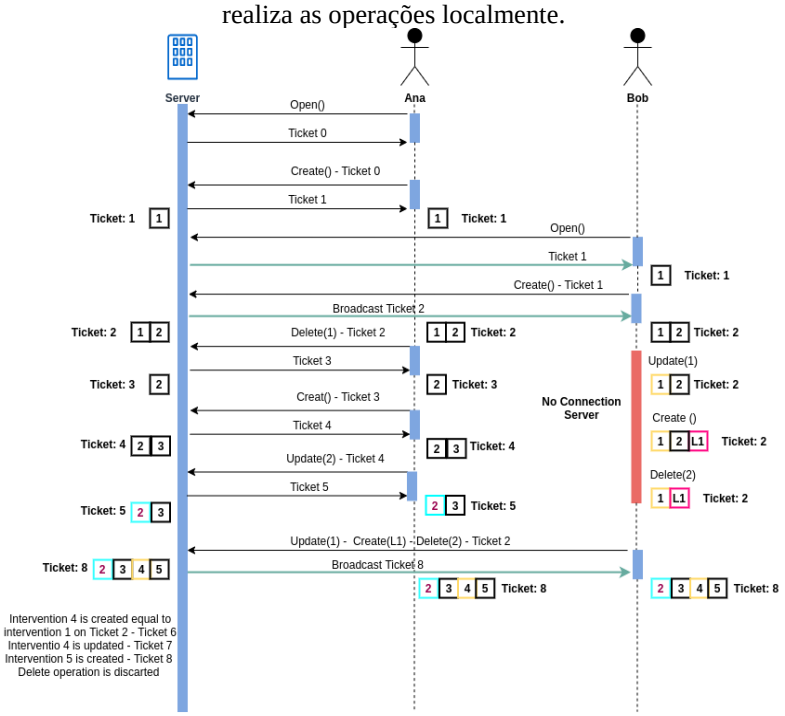
a) Atualização (1) - A intervenção foi eliminada por Ana no Bilhete 3, seguindo as regras do controlo da competição, o servidor irá recriar a intervenção 1 com os dados que ela tinha no Bilhete 2, a regra diz que todas as intervenções são criadas no final da fila, após recriar a intervenção, o servidor fará a atualização, conforme solicitado por Bob. Dois tíquetes serão gerados, um para a criação e outro para a atualização;

b) Criar (L1) - A intervenção L1 foi criada localmente por Bob, então ela precisa ser enviada ao servidor, a regra de criação de intervenção diz que todas as intervenções devem ser criadas no final da fila. Desta forma, é criada a intervenção 5 e gerado um novo tíquete;

c) Eliminar (2) - A intervenção 2 foi alterada por Ana no bilhete 5, a regra do controle da concorrência diz que quando uma intervenção foi alterada não pode ser eliminada por operações com um bilhete anterior. Portanto, a operação de exclusão da intervenção 2 será descartada sem gerar um tíquete.

Depois que todas as alterações enviadas por Bob forem realizadas, o servidor irá transmitir todas as operações realizadas. E todos os usuários serão sincronizados novamente.

Figura 06. Demonstração do processo de merge quando um usuário perde a conexão com o servidor e realiza as operações localmente.



5.5 REPOSITÓRIO DE COMPONENTES

O repositório de componentes tem por função armazenar e organizar os componentes que foram desenvolvidos de modo que eles possam ser reutilizados novamente. São considerados componentes um programa, conjuntos de eventos e conjuntos de intervenções. Todo programa, evento e intervenção criados automaticamente ficam disponibilizados no repositório de componentes. Para aumentar a flexibilidade e a possibilidades na composição de componentes, os autores poderão selecionar conjuntos de eventos ou intervenções e armazenar estes conjuntos como componentes no repositório.

Os componentes disponibilizados no repositório terão os seus níveis de acesso determinados pelos seus autores. Todo autor poderá determinar se os componentes criados por ele poderão ou não ser acessados por outros autores, além de poder determinar quais autores poderão ou não acessar os componentes criados por ele.

Um programa ESM que esteja em edição, também é tratado como um componente, permitindo assim que um novo programa se inicie a partir de um ou vários programas, o fato de tratar todos os elementos componentes facilita a composição e a criação de novos programas, e vai de encontro à metodologia modular adotada no projeto.

Por entendermos que o processo é cíclico, adotamos o fato de que um programa nunca está finalizado. Quando o especialista entende que um programa está apto a ser

utilizado em coletas de dados, ele deve criar um versão de coleta no repositório de programas ESM.

5.6 REPOSITÓRIO DE PROGRAMAS ESM E CONTROLE DE VERSÕES

Seguindo o modelo proposto neste artigo, os programas ESM são criados na fase de Definição, nesta fase os especialistas trabalham de forma colaborativa na criação dos programas ESM. Quando os especialistas entendem que chegaram ao ponto do programa atender todos os requisitos necessários, os especialistas disponibilizam uma versão de coleta para o programa no repositório de Programas ESM.

Apesar das de resguardadas a devidas diferenças entre o desenvolvimento de software tradicional e uma plataforma EUD, ainda é necessário ter em mente que este último também possui um ambiente de desenvolvimento, neste deve-se ter e mente a importância e a relevância de se efetuar um rigoroso controle de mudanças nos ambientes de desenvolvimentos [30].

Os programas ESM que são desenvolvidos pelos especialistas devem ser visualizados como componentes que carregam dentro de si um significado. Quando forem executados nos dispositivos móveis dos participantes irão coletar dados. Os dados coletados pelo programa trazem um significado para o especialista devido à forma que foram pré-definidos na criação do programa.

Quando o especialista recebe os dados das coletas para análise pode entender que os dados coletados por um programa não contemplam todas as suas necessidades, neste caso o especialista pode efetuar mudanças e gerar uma nova versão do programa ESM. A partir de um novo programa ESM iniciar uma nova coleta e o especialista recebe um novo conjunto de dados.

A função do sistema de controle de versões [36] é gerenciar as várias versões de um mesmo programa ESM, bem como manter a compatibilidade das diversas versões de um mesmo programa. A compatibilidade entre várias versões de um mesmo programa é particularmente importante frente ao fato que um programa estar sempre em edição na fase de Definição.

O controle de versões implementado faz uso do histórico de alterações descrito na seção 5.3 e com base nos tíquetes consegue determinar e controlar as diversas versões de um mesmo programa ESM.

6 AVALIAÇÃO

O grande objetivo do trabalho é verificar a validade do MACI, e para tanto, como prova de conceitos, implementamos o FACEE, desta forma o processo de avaliação será realizado no FACEE de modo que possa ser validado o modelo proposto.

O foco da avaliação recaíram sobre as novas características propostas pelo modelo, que são: a) edição colaborativa com histórico e colaboração síncrona; b) controle de versão; e c) sistemas de concorrência e de *merge*. Um objetivo dos testes foi verificar se as novas características se adequam ao ESPIM e mantêm as características dos programas de intervenções.

Nesta primeira versão da FACEE não foram testados nem validados os elementos e características implementados conforme o descrito por Cunhas [45], visto a literatura já apresentar diversos trabalhos que demonstram as capacidades do ESPIM.

A interface de criação de programas do ESPIM já foi avaliada por Rodrigues *et al* [32]. A estrutura de programas de intervenções baseada em grafos direcionados do Cunha [45], visto o ESPIM já foram utilizados em um grande número de trabalhos publicados com estudos de casos reais [3,32,33,38]. Além disso, o ESPIM já demonstrou as suas capacidades tanto em programas de intervenções na área de educação e saúde. Na área de saúde, Cunha et al [46] relata 6 estudos de casos realizados com o ESPIM.

Para avaliar a funcionalidade de edição colaborativa, o backend foi disponibilizado em um servidor, e seis instâncias do frontend foram executadas em três computadores simultaneamente, em cada computador dois navegadores foram abertos com o frontend. Os objetos foram criados, atualizados e excluídos em todas as instâncias abertas. Todas as operações realizadas em uma instância foram replicadas corretamente em todas as outras.

Para testar a validade do histórico, todas as operações foram realizadas em ordem, como uma fila, no final os registros existentes eram exatamente os mesmos que os criados no primeiro teste. Por fim, todas as operações foram desfeitas em ordem reversa, como uma pilha e, como resultado, todos os registros criados para o teste foram excluídos.

Com a utilização apenas das três operações delta, a construção de um histórico de operações e o uso da comunicação via websocket, implementamos recursos de colaboração na ferramenta de autoria ESPIM. Nos testes foi possível: (a) coletar alterações (adicionar, excluir, atualizar) feitas pelos usuários, (b) divulgar essas alterações entre os usuários remotos, (c) aplicar as alterações a cada editor cliente. Como resultado de nossos esforços para fornecer à ferramenta de autoria do ESPIM recursos de trabalho

colaborativo, desenvolvemos uma solução simples que usa apenas operações delta, um registro de histórico de operações e um websocket.

7 CONSIDERAÇÕES FINAIS

Neste artigo apresentamos o MACI, um modelo colaborativo para o desenvolvimento de ferramentas de autoria, voltadas para o desenvolvimento de programas de intervenções baseados no ESM utilizando o EUD.

Como prova de concneitos do modelo foi implementada a ferramenta de autoria FACEE. A ferramenta FACEE permite a colaboração síncrona entre especialistas, implementando conceitos de reusabilidade de componentes, através da disponibilização de um repositório de componentes. A ferramenta também possui um sistema de controle de versões dos programas de intervenções ESM. Além disso foram implementados algoritmos e funções de controle de concorrência e de *merge.*

Após a implementação da ferramenta foram realizados testes de modo a validar a edição colaborativa baseada em operações deltas. Os testes demonstraram a validade dos históricos criados e a capacidade da ferramenta de autoria em prover a colaboração síncrona através da utilização de operações delta.

A ferramenta de autoria foi desenvolvida com base no modelo ESPI . O método ESPIM é usado por especialistas em áreas que incluem a interação humano-computador e pesquisadores de computação ubíqua, educadores, psicólogos, terapeutas ocupacionais e fonoaudiólogos.

Um trabalho futuro está planejado para estender a plataforma para que ela permita a criptografia de conteúdo antes da troca e armazenamento [22] , e para suportar versões semânticas [31].

Também estão previstos como trabalhos futuros a extensão do modelo para detalhar de forma mais explícita as tarefas dos especialistas e a colaboração entre especialistas em cada uma das fases do modelo.

REFERÊNCIAS

[1] Barbara Rita Barricelli and Stefano Valtolina. 2017. A visual language and interactive system for end-user development of internet of things ecosystems. Journal of Visual Languages & Computing 40 (2017), 1–19.

[2] Manfred E Beutel, Eva M Klein, et al. 2017. Loneliness in the general population: prevalence, determinants and relations to mental health. BMC psychiatry 17, 1 (2017), 97.

[3] Meire Cachioni, Isabela Zaine, et al. 2019. Aprendizagem ao longo de toda a vida e letramento digital de idosos: um modelo multidisciplinar de intervenção com o apoio de um aplicativo. Revista Brasileira de Ciências do Envelhecimento Humano 16, 1 (2019), 18–24.

[4] Stephanie Cacioppo, Angela J Grippo, Sarah London, Luc Goossens, and John T Cacioppo. 2015. Loneliness: Clinical import and interventions. Perspectives on Psychological Science 10, 2 (2015), 238–249.

[5] Hudson Francis Ventura de Souza and Roberto Bendito de Oliveira Pereira. 2020. Análise de desempenho de banco de dados: Postgresql padrão e um cluster utilizando o Postgres-bdr. Profiscientia 14 (2020), 88–108.

[6] DrawIO. [n.d.]. DrawIO - Diagram Software and Flowchart Maker. https: //draw.io/

[7] Clarence A Ellis and Simon J Gibbs. 1989. Concurrency control in groupware systems. In Proceedings of the 1989 ACM SIGMOD international conference on Management of data. 399–407.

[8] Neil Fraser. 2009. Differential Synchronization. In Proceedings of the 9th ACM Symposium on Document Engineering (Munich, Germany) (DocEng '09). Association for Computing Machinery, New York, NY, USA, 13–20. https://doi.org/10.

[9] Cristian Gadea. 2021. Architectures and Algorithms for Real-Time Web-Based Collaboration. Ph.D. Dissertation. Université d'Ottawa/University of Ottawa.

[10] Google. [n.d.]. Google Drawings - create diagrams and charts, for free. https: //docs.google.com/drawings

[11] Google. [n.d.]. GoogleSlides: Free Online Presentations for Personal Use. https: //www.google.com/slides

[12] Seiji Isotani and Leônidas de Oliveira Brandão. 2008. An Algorithm for Automatic Checking of Exercises in a Dynamic Geometry System: IGeom. Comput. Educ. 51, 3 (Nov. 2008), 1283–1303. https://doi.org/10.1016/j.compedu.2007.12.004

[13] Channu Kambalyal. 2010. 3-tier architecture. Retrieved On 2 (2010), 34.

[14] Dilshodbek Kuryazov and Andreas Winter. 2014. Representing Model Differences by Delta Operations. In Proceedings of the 2014 IEEE 18th International Enterprise Distributed Object Computing Conference Workshops and Demonstrations (EDOCW '14). IEEE Computer Society, USA, 211–220. https://doi.org/10.1109/EDOCW. 2014.39

[15] Dilshodbek Kuryazov and Andreas Winter. 2015. Collaborative Modeling Empowered By Modeling Deltas. In Proceedings of the 3rd International Workshop on (Document) Changes: Modeling, Detection, Storage and Visualization (Lausanne, Switzerland) (DChanges 2015). Association for Computing Machinery, New York, NY, USA, 1–6. https://doi.org/10.1145/2881631.2881633

[16] Dilshodbek Kuryazov, Andreas Winter, and Ralf Reussner. 2018. Collaborative Modeling Enabled By Version Control. Modellierung 2018 (2018).

[17] Reed Larson and Mihaly Csikszentmihalyi. 2014. The experience sampling method. In Flow and the foundations of positive psychology. Springer, 21–34.

[18] Suryadiputra Liawatimena, Harco Leslie Hendric Spits Warnars, et al. 2018. Django web framework software metrics measurement using radon and pylint. In 2018 Indonesian Association for Pattern Recognition International Conference (INAPR). IEEE, 218–222.

[19] Henry Lieberman, Fabio Paternò, Markus Klann, and Volker Wulf. 2006. End-user development: An emerging paradigm. In End user development. Springer, 1–8.

[20] Peter Lubbers, Brian Albers, Ric Smith, and Frank Salim. 2010. Pro HTML5 Programming: Powerful APIs for Richer Internet Application Development (1st ed.). Apress, USA.

[21] Lucidchart. [n.d.]. Lucidchart. https://www.lucidchart.com/

[22] Aaron MacSween, Caleb James Delisle, Paul Libbrecht, and Yann Flory. 2018. Private Document Editing with Some Trust. In Proceedings of the ACM Symposium on Document Engineering 2018 (Halifax, NS, Canada) (DocEng '18). Association for Computing Machinery, New York, NY, USA, Article 29, 10 pages. https: //doi.org/10.1145/3209280.3209535

[23] Tom Marrs. 2017. JSON at work: practical data integration for the web. "O'Reilly Media, Inc.".

[24] MathWorks. [n.d.]. Simulink: Simulation and Model-Based Design. https: //www.mathworks.com/products/simulink.html

[25] MathWorks. [n.d.]. Stateflow: Model and simulate decision logic using state machines and flow charts. https://www.mathworks.com/products/stateflow.html

[26] David A Nichols, Pavel Curtis, Michael Dixon, and John Lamping. 1995. Highlatency, low-bandwidth windowing in the jupiter collaboration system. In Proceedings of the 8th annual ACM symposium on User interface and software technology. 111–120.

[27] Haroon Shakirat Oluwatosin. 2014. Client-server model. IOSRJ Comput. Eng 16, 1 (2014), 2278–8727.

[28] Anthony D Ong, Bert N Uchino, and Elaine Wethington. 2016. Loneliness and health in older adults: A mini-review and synthesis. Gerontology 62, 4 (2016), 443–449.

[29] Overleaf. [n.d.]. Overleaf, Online LaTeX Editor. https://www.overleaf.com/

[30] Roger S PRESSMAN. [n.d.]. Engenharia de software, uma abordagem profissional–8ª Ed–AMGH Editora Ltda. Porto Alegre–RS-2016 ([n. d.]).

[31] S. Raemaekers, A. van Deursen, and J. Visser. 2017. Semantic Versioning and Impact of Breaking Changes in the Maven Repository. J. Syst. Softw. 129, C (July 2017), 140–158. https://doi.org/10.1016/j.jss.2016.04.008

[32] Kamila RH Rodrigues, Bruna CR Cunha, et al. 2018. ESPIM System: Interface Evolution to Enable Authoring and Interaction with Multimedia Intervention Programs. In Proceedings of the 24th Brazilian Symposium on Multimedia and the Web. 125–132.

[33] Kamila RH Rodrigues, Caio C Viel, et al. 2017. Data collection and intervention personalized as interactive multimedia documents. In Proceedings of the 23rd Brazillian Symposium on Multimedia and the Web. 57–60.

[34] Daniel J Rough and Aaron Quigley. 2020. End-User Development of Experience Sampling Smartphone Apps-Recommendations and Requirements. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 4, 2 (2020), 1–19.

[35] Mazyar Seraj, Serge Autexier, and Jan Janssen. 2018. BEESM, a block-based educational programming tool for end users. In Proceedings of the 10th Nordic Conference on Human-Computer Interaction. 886–891.

[36] Diomidis Spinellis. 2005. Version control systems. IEEE Software 22, 5 (2005), 108–109.

[37] Scott Stringer. [n.d.]. Shapes Tool. https://www.getpaint.net/doc/latest/ShapeTools.html

[38] Isabela Ternero and Valéria Meirelles Carril Elui. 2020. O Uso Da Técnica Do Espelho Através Da Realidade Aumentada Com Acompanhamento Remoto. Brazilian Journal of Health Review 3, 3 (2020), 6697–6709.

[39] Joel Vainikka. 2018. Full-stack web development using Django REST framework and React. (2018).

[40] Niels Van Berkel, Denzil Ferreira, and Vassilis Kostakos. 2017. The experience sampling method on mobile devices. ACM Computing Surveys (CSUR) 50, 6 (2017), 1–40.

[41] Nicolas Vidot, Michelle Cart, Jean Ferrié, and Maher Suleiman. 2000. Copies convergence in a distributed real-time collaborative environment. In Proceedings of the 2000 ACM conference on Computer supported cooperative work. 171–180.

[42] Caio C. Viel, Kamila R.H. Rodrigues, et al. 2017. Personalized Ubiquitous Data Collection and Intervention as Interactive Multimedia Documents. In Proceedings of the 2017 ACM Symposium on Document Engineering (Valletta, Malta) (DocEng '17). Association for Computing Machinery, New York, NY, USA, Article 1, 4 pages. https://doi.org/10.1145/3103010.3121046

[43] Eric Wohlgethan. 2018. SupportingWeb Development Decisions by ComparingThree Major JavaScript Frameworks: Angular, React and Vue. js. Ph.D. Dissertation. Hochschule für Angewandte Wissenschaften Hamburg

[44] Silva, Joab Cavalcante da. MIAC: um modelo para autoria de intervenção apoiada por colaboração via dispositivos móveis. Diss. Universidade de São Paulo, 2021

[45] Cunha, Bruna Carolina Rodrigues da. ESPIM: um modelo para guiar o desenvolvimento de sistemas de intervenção a distância. Diss. Universidade de São Paulo, 2019.

[46] Cunha, Bruna Carolina Rodrigues, Kamila Rios Da Hora Rodrigues, Isabela Zaine, Elias Adriano Nogueira da Silva, Caio César Viel, and Maria Da Graça Campos Pimentel. "Experience sampling and programmed intervention method and system for planning, authoring, and deploying mobile health interventions: design and case reports." Journal of medical Internet research 23, no. 7 (2021): e24278.