UNIVERSIDADE DE PASSO FUNDO
Programa de Pós-Graduação em
Computação Aplicada

Dissertação de Mestrado

PROPOSIÇÃO DE UM FRAMEWORK PARA O DESENVOLVIMENTO DE APLICAÇÕES MÓVEIS SENSÍVEIS AO CONTEXTO

CLAYTON MAGALHÃES

UPF

UNIVERSIDADE DE PASSO FUNDO
INSTITUTO DE CIÊNCIAS EXATAS E GEOCIÊNCIAS
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO APLICADA

PROPOSIÇÃO DE UM FRAMEWORK PARA O DESENVOLVIMENTO DE APLICAÇÕES MÓVEIS SENSÍVEIS AO CONTEXTO

Clayton Magalhães

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre em Computação Aplicada na Universidade de Passo Fundo.

Orientador: Ana Carolina Bertoletti De Marchi

Passo Fundo
2020

CIP – Catalogação na Publicação

M188p Magalhães, Clayton
Proposição de um *framework* para o desenvolvimento de aplicações móveis sensíveis ao contexto [recurso eletrônico] / Clayton Magalhães. – 2020.
2.5 MB ; PDF.

Orientadora: Ana Carolina Bertoletti De Marchi.
Dissertação (Mestrado em Computação Aplicada) – Universidade de Passo Fundo, 2020.

1. Framework (Arquivo de computador). 2. Computação ubíqua. 3. Engenharia de software. 4. Aplicativos móveis. I. De Marchi, Ana Carolina Bertoletti, orientadora. II. Título.

CDU: 004.42

Catalogação: Bibliotecária Juliana Langaro Silveira – CRB 10/2427

UPF | PPGCA
Programa de Pós-Graduação
em Computação Aplicada
Instituto de Ciências Exatas e Geociências | ICEG

ATA DE DEFESA DO TRABALHO DE CONCLUSÃO DE CURSO DO ACADÊMICO

CLAYTON MAGALHÃES

Aos vinte e seis dias do mês de março do ano de dois mil e vinte, às quatorze horas, realizou-se, no prédio D01 sala 01, da Universidade de Passo Fundo (UPF), a sessão pública de defesa do Trabalho de Conclusão de Curso "Proposição de um framework para o desenvolvimento de aplicações móveis sensíveis ao contexto", de autoria do **Clayton Magalhães**, acadêmico do Curso de Mestrado em Computação Aplicada do Programa de Pós-Graduação em Computação Aplicada – PPGCA. Segundo as informações prestadas pelo Conselho de Pós-Graduação e constantes nos arquivos da Secretaria do PPGCA, o aluno preencheu os requisitos necessários para submeter seu trabalho à avaliação. A banca examinadora foi composta pelos doutores Ana Carolina Bertoletti De Marchi, Alexandre Lazaretti Zanatta, e Vaninha Vieira dos Santos. Concluídos os trabalhos de apresentação e arguição, a banca examinadora considerou o candidato Aprovado. Foi concedido o prazo de até quarenta e cinco (45) dias, conforme Regimento do PPGCA, para o acadêmico apresentar ao Conselho de Pós-Graduação o trabalho em sua redação definitiva, a fim de que sejam feitos os encaminhamentos necessários à emissão do Diploma de Mestre em Computação Aplicada. Para constar, foi lavrada a presente ata, que vai assinada pelos membros da banca examinadora e pela Coordenação do PPGCA.

Prof. Dra. Ana Carolina Bertoletti De Marchi – UPF
Presidente da Banca Examinadora
(Orientadora)

Prof. Dr. Alexandre Lazaretti Zanatta – UPF
(Coorientador)

Prof. Dra. Vaninha Vieira dos Santos – (UFBA)
(Avaliadora Externa)

Prof. Dr. Rafael Rieder
Coordenador do PPGCA

AGRADECIMENTOS

Este trabalho é dedicado primeiramente à Deus, pelo dom da vida, pela saúde e pela luz que orienta todos os dias, que dá forças para enfrentar e superar os momentos difíceis. À minha família que me dá suporte, a minha ilha segura, sempre acreditando e incentivando os meus objetivos.

À minha orientadora, Profa. Ana Carolina, por ter me acolhido tão bem e ter sido uma grande amiga, além de Mestre dessa caminhada, e por sempre acreditar neste trabalho. És uma referência profissional e humana, o qual sempre serei muito grato pela oportunidade de compartilhar todos os momentos.

Aos colegas parceiros do PPGEH, Profa. Ana Luisa e Luísa por darem todo o apoio necessário na demanda deste projeto no qual acreditamos. Aos colegas do PPGCA por compartilharem dessa caminhada e se tornarem grandes amigos também.

Aos profissionais nutricionistas que prontamente ajudaram na avaliação do aplicativo, dispondo de tempo e dando seu importante feedback. À Universidade de Passo Fundo que tem sido minha segunda casa há tantos anos e responsável pela minha formação.

À todas as pessoas queridas que, de uma forma ou de outra, contribuíram para a realização desse trabalho, o meu mais sincero muito obrigado.

PROPOSIÇÃO DE UM FRAMEWORK PARA O DESENVOLVIMENTO DE APLICAÇÕES MÓVEIS SENSÍVEIS AO CONTEXTO

RESUMO

Os dispositivos móveis atuais possuem capacidade de coletar informações de contexto do usuário de forma não intrusiva e integrada, por meio de sensores internos e externos. Esse avanço tecnológico abre muitas possibilidades para o desenvolvimento de aplicações, principalmente as sensíveis ao contexto. Por apresentarem muitos desafios e especificidades em relação aos sistemas tradicionais, algumas iniciativas vêm sendo propostas para o processo de desenvolvimento desse tipo de aplicação. Essas iniciativas concentram-se em diferentes etapas de projeto, bem como propõem diferentes modelos de representação do contexto. Contudo, nenhuma delas aborda requisitos funcionais e arquitetônicos, fornecendo ferramentas e guias de instruções práticas para implementação. Este trabalho tem como objetivo propor um framework para o desenvolvimento de aplicações móveis sensíveis ao contexto, por meio de modelos de representação, arquitetura e diretrizes de implementação integrados, considerando princípios de modularidade, escalabilidade e compatibilidade. Além disso, reúne e adapta diferentes modelos da literatura, aplicando os mesmos conceitos de forma complementar. Também trata de questões atuais como as limitações de armazenamento e energia em dispositivos móveis, bem como a disponibilidade de conexão. Para verificar a assertividade do modelo conceitual e da arquitetura proposta, foi feita uma avaliação com três analistas de sistemas os quais puderam ler e praticar os conceitos propostos, respondendo um questionário de feedback, em geral positivo e com sugestões. Também foi desenvolvida uma aplicação móvel com funcionalidades sensíveis ao contexto, voltada para cuidados alimentares e testada por seis profissionais da nutrição. Após um período de uso de sete dias, os profissionais responderam a um questionário de percepção em relação as funcionalidades com sensibilidade ao contexto. Os dados foram analisados e, em geral, os usuários também deram um feedback positivo. Como conclusão, a aplicação construída a partir do framework verificou a viabilidade da estrutura proposta por meio do seu uso prático primeiramente com profissionais de software e em um exemplo de domínio. Espera-se, com este trabalho, contribuir para a área da computação, com a disponibilização de um framework para aplicações sensíveis ao contexto.

Palavras-chave: Sistemas sensíveis ao contexto, Framework, Aplicações Móveis.

PROPOSITION OF A FRAMEWORK FOR THE DEVELOPMENT OF CONTEXT-SENSITIVE MOBILE APPLICATIONS

ABSTRACT

Current mobile devices have the ability to collect context information from the user in a non-intrusive and integrated way, through internal and external sensors. This technological advance opens up many possibilities for the development of applications, especially those sensitive to the context. As they present many challenges and specificities in relation to traditional systems, some initiatives have been proposed for the process of developing this type of application. These initiatives are concentrated in different stages of the project, as well as proposing different models of context representation. However, none of them addresses functional and architectural requirements, providing practical tools and instructions for implementation. This work aims to propose a framework for the development of context-sensitive mobile applications, through representation models, architecture and integrated implementation guidelines, considering principles of modularity, scalability and compatibility. In addition, it gathers and adapts different models from the literature, applying the same concepts in a complementary way. It also addresses current issues such as storage and power limitations on mobile devices, as well as connection availability. To verify the assertiveness of the conceptual model and the proposed architecture, an assessment was made with three systems analysts who were able to read and practice the proposed concepts, answering a feedback questionnaire, generally positive and with suggestions. A mobile application with context-sensitive features was also developed, focused on food care and tested by six nutrition professionals. After a period of use of seven days, the professionals answered a perception questionnaire regarding the features with context sensitivity. The data was analyzed and, in general, users also gave positive feedback. As a conclusion, the application built from the framework verified the viability of the proposed structure through its practical use primarily with software professionals and in an example domain. It is expected, with this work, to contribute to the computing area, with the availability of a framework for context sensitive applications.

Keywords: Context sensitive systems, Framework, Mobile Applications.

LISTA DE FIGURAS

Figura 1. Propriedades de adaptação. 17
Figura 2. Exemplo utilizando a notação Context4BPMN. 23
Figura 3. Modelo estrutural e conceitual para representação de contexto 33
Figura 4. Modelo conceitual lógico instanciado para o exemplo. 34
Figura 5. Modelo instanciado por meio de diagrama de classe com dados. 36
Figura 6. Simulação de um banco de dados de refeições classificadas. 37
Figura 7. Modelo dinâmico genérico para a notação Context4BPMN. 37
Figura 8. Exemplo de modelo dinâmico utilizando a notação Context4BPMN. 38
Figura 9. Visão geral da arquitetura. 43
Figura 10. Modelagem de dados parcial da aplicação. 58
Figura 11. Implementação para armazenamento na base de dados. 66
Figura 12. Detalhamento do módulo de processamento e serviços. 68
Figura 13. Detalhamento da arquitetura no cliente. 69
Figura 14. Telas para acessar o diário de sugestões de refeição. 72
Figura 15. Telas da aplicação: conectividade e mapa de feiras. 73

SUMÁRIO

1. INTRODUÇÃO 11
2. REVISÃO DA LITERATURA 13
2.1. CONTEXTO 13
2.2. SISTEMAS ADAPTATIVOS 16
2.3. SISTEMAS SENSÍVEIS AO CONTEXTO 18
2.4. SISTEMAS ADAPTATIVOS SENSÍVEIS AO CONTEXTO 19
2.5. MODELAGEM DE SISTEMAS SENSÍVEIS AO CONTEXTO 20
2.6. TRABALHOS RELACIONADOS 24
3. FAM4CSS: UM FRAMEWORK PARA APLICAÇÕES MÓVEIS SENSÍVEIS AO CONTEXTO 29
3.1. REQUISITOS PARA O DESENVOLVIMENTO DE APLICAÇÕES SENSÍVEIS AO CONTEXTO 29
3.2. DETALHAMENTO DO FRAMEWORK FAM4CSS 30
3.2.1. Modelo Estrutural 31
3.2.2. Modelo Dinâmico 37
3.2.3. Uma arquitetura para sistemas sensíveis ao contexto 39
3.2.3.1. Requisitos e aspectos técnicos 39
3.2.3.2. Proposta de arquitetura baseada em estrutura na nuvem 42
3.2.3.3. Detalhamento dos elementos da estrutura 44
3.2.4. Implementação da estrutura 48
4. RESULTADOS 53
4.1. AVALIAÇÃO DO FRAMEWORK POR ESPECIALISTAS DE SOFTWARE 53
4.1.1. Elaboração da avaliação 53
4.1.2. Procedimento de coleta de dados 54
4.1.3. Análise das respostas 54
4.2. INSTANCIANDO O FRAMEWORK EM UMA APLICAÇÃO PARA CUIDADOS DE SAÚDE ALIMENTAR 55
4.2.1. Domínio da aplicação escolhida 55
4.2.2. Requisitos para a aplicação escolhida 56
4.2.3. Modelagem da aplicação 57
4.2.4. Escolha das tecnologias envolvidas para implementação 60
4.2.5. Análise de requisitos arquitetônicos 62
4.2.6. Implementação da arquitetura na nuvem 64

4.2.7. Implementação da arquitetura no cliente 68
4.2.8. Aplicativo Guia Alimentar 71
4.3. AVALIAÇÃO DOS ESPECIALISTAS DE DOMÍNIO 73
4.3.1. Elaboração do questionário de avaliação 73
4.3.2. Procedimento de coleta de dados 74
4.3.3. Análise das respostas 74
5. CONCLUSÃO 77
REFERÊNCIAS 79
APÊNDICE A - APLICAÇÃO DE PESQUISA 83
APÊNDICE B - QUESTIONÁRIO DE FEEDBACK FAM4CSS 84
APÊNDICE C - QUESTIONARIO AVALIAÇÃO NUTRICIONISTAS 85
APÊNDICE D - MANUAL DE USO DO APLICATIVO 86
APÊNDICE E - TEXTO SOBRE OBJETIVO DA PESQUISA 88

1. INTRODUÇÃO

A expansão no uso de dispositivos móveis, aliado a outros avanços tecnológicos como a comunicação sem fio e sensores mais sofisticados embutidos em dispositivos cada vez mais poderosos e portáteis, abre possibilidades para a concretização do sonho de Weiser [1]. Em 1991 o autor descrevia a computação ubíqua como a área que tornaria possível disponibilizar informações e recursos computacionais “a qualquer hora, em qualquer lugar e em qualquer dispositivo”. Esses avanços proporcionam o uso de aplicações móveis nos mais diversos ambientes, em contextos distintos e cada vez mais complexos. Esta realidade emergente levou a comunidade de engenharia de software a investigar novas formas de desenvolvimento, implantação e gerenciamento de sistemas e serviços de software que se adaptam a mudanças que possam ocorrer em seus contextos operacionais, ambientes e requisitos [2].

Os dispositivos móveis possuem cada vez mais a capacidade de coletar informações de contexto do usuário de forma não invasiva. O ideal é que as aplicações percebam a intenção do utilizador, minimizando suas tarefas, especialmente a necessidade de especificar tudo o que deseja que o sistema faça por ele. Para tanto, é preciso considerar o contexto, que se refere ao “conhecimento que está por trás da habilidade de discriminar o que é ou não importante em um dado momento” [3] e, com isso, oferecer uma interação mais rica e atrativa aos usuários, dispondo de aplicações mais flexíveis, confiáveis, eficientes, personalizáveis, configuráveis e auto-otimizáveis.

Sendo assim, é possível guiar o desenvolvimento de aplicações que reconheçam informações de contexto relevantes dos usuários maximizando o potencial computacional destas soluções. Diferentes iniciativas visaram estabelecer abordagens adequadas de engenharia de software para viabilizar a autoadaptação e sensibilidade ao contexto em aplicações de software. Essas abordagens apresentam estruturas focadas em diferentes processos de software, bem como baseadas em modelos [4], arquitetura [5][6][7] e/ou métodos [8]. De um lado estão iniciativas com propostas de modelos e processos genéricos para apoiar a solução em vários domínios, e por outro, arquiteturas e diretrizes práticas para apoiar o desenvolvimento de aplicações dessa natureza. Além dessas diferenças, ao longo

dos anos novas tecnologias de hardware e software surgiram, alterando paradigmas de projeto e desenvolvimento, impondo maiores desafios e diversificando ainda mais as abordagens [9].

Apesar das iniciativas propostas, poucas apresentam métodos de engenharia baseados em arquiteturas genéricas, bem como um processo de desenvolvimento detalhado. Além disso, com exceção do trabalho de Cheik *et al*. [8], as demais abordagens propõem metamodelos de contexto e/ou arquiteturas, mas não fornecem guias ou instruções para o uso dos mesmos [4], a fim de facilitar a semântica, tratar dos requisitos funcionais e arquitetônicos, fornecendo também ferramentas e guias de instruções práticas, por meio de tecnologias e serviços adequados. Dessa forma, torna-se relevante investir esforços para entender e modelar projetos com reconhecimento de contexto, fornecendo uma abordagem robusta e integrada por meio da evolução de modelos, arquitetura e diretrizes práticas de forma sistemática. Uma abordagem também proativa em relação aos problemas de conectividade, processamento, variedade de dispositivos e recursos cada vez mais heterogêneos.

Diante desse cenário, o objetivo deste trabalho foi propor um framework conceitual, por meio de modelos, arquitetura e diretrizes atuais que possam apoiar a compreensão no processo de desenvolvimento de aplicações sensíveis ao contexto. A viabilidade do framework foi verificada a partir do seu instanciamento em uma aplicação para cuidados de saúde alimentar, e da avaliação da aplicação por profissionais de software, os quais puderam entender e praticar os conceitos da proposta em um caso de uso. Após essa etapa, foi desenvolvido uma aplicação com funcionalidades sensíveis ao contexto para o domínio da nutrição, avaliado por profissionais dessa área.

Essa dissertação está estruturada em 5 capítulos. No capítulo 2 será apresentada uma revisão da literatura. No capítulo 3 será apresentado o detalhamento do framework FAM4CSS. No capítulo 4 serão apresentados os resultados, e por fim, o capítulo 5 será apresentada a conclusão.

2. REVISÃO DA LITERATURA

Neste capítulo serão abordados conceitos sobre contexto de uso em sistemas adaptativos e sistemas sensíveis ao contexto, com foco em aplicações para dispositivos móveis, bem como a utilização de sensores e demais recursos para captação de informações de contexto.

2.1. CONTEXTO

O contexto ainda é um conceito vago para identificar aspectos que possam ser considerados úteis para modelar e descrever o ambiente onde um determinado serviço deve ser implantado e executado [10][11]. Muitas definições são encontradas na literatura, Dey and Abowd [12] descrevem o contexto como qualquer informação que possa ser usada para caracterizar a situação de uma entidade, sendo entidade uma pessoa, lugar ou objeto os quais possam ser considerados relevantes para a interação entre o usuário e uma aplicação, incluindo o próprio usuário e a aplicação. Schilit *et al*. [13] definem contexto como “onde você está, com quem está e que recursos estão nas proximidades”.

Mais especificamente em contexto de aplicações móveis, Dey [14] define como aplicações que exploram o contexto dinâmico dos seus usuários, provocado pela mobilidade e constante mudança no ambiente, onde essas aplicações venham a fazer uso de dispositivos portáteis multifuncionais, como os PDAs e telefones celulares (apud Sanches [15]). Chen e Kotz [11] definem contexto como o conjunto de estados e configurações ambientais que determinam o comportamento de uma aplicação ou em que um evento de aplicação ocorre e é interessante para o usuário.

Soylu *et al*. [16] discorrem sobre algumas dessas definições e concluem que a definição do escopo de contexto deve deixar um papel importante para o desenvolvimento de aplicações que utilizam o contexto, ao invés de somente fornecer uma definição exaustiva do contexto. Ainda, que a consciência de contexto está relacionada com a adaptabilidade e baseia-se na exploração das informações de contexto recrutadas, adaptando o seu comportamento em conformidade. E para que se possa considerar uma informação como contexto, é necessário garantir que essa informação permita a aplicação modificar seus comportamentos com relação a

essa informação, bem como sua relação com outras dimensões do contexto. Terry [17] também descreve que algo é considerado contexto por causa da maneira como é usado na interpretação e não devido às suas propriedades inerentes, onde as características do mundo tornam-se contexto somente por meio do seu uso, e, portanto, em caso contrário são apenas parte do ambiente.

Classificar o contexto é útil para gerenciar a qualidade e para a modelagem de contexto em fases conceituais, iniciais e posteriores. Também torna-se necessário para definir algumas especificidades de adaptabilidade e gerenciamento de contexto, como por exemplo a abstração [18]. Uma forma de classificar casos de contexto é a distinção de diferentes dimensões, chamadas por alguns autores de externas e internas [21] [22]. A dimensão externa (ou física) refere-se ao que pode ser medido por sensores de hardware, por exemplo, a localização, luz, som, toque, movimento, temperatura ou a pressão de ar, enquanto a dimensão interna (lógica) é na maior parte especificada pelo utilizador ou capturados pelo monitoramento das interações do usuário, como por exemplo, os objetivos do usuário, as tarefas de trabalho realizadas, os processos de negócios e o estado emocional do usuário. A maioria dos sistemas cientes de contexto faz uso de contexto de fatores externos [21].

Além desses, Sanchez *et al*. [22] explicam a distinção entre os dados brutos e a informação de contexto. Para os autores, dados brutos são recuperados diretamente da fonte, sem transformação, tais como dados de sensores e informações de contexto, geradas a partir do processamento dos dados brutos do sensor. Entretanto, é possível categorizar o contexto de vários outros aspectos, Schilit *et al*. [13] baseiam-se no agrupamento de dimensões de contexto semelhantes, sendo: contexto de computação, contexto do usuário e contexto físico. Posteriormente, Chen e Kotz [11] estendem essa categorização com um novo item chamado contexto de tempo. Han *et al*. [23] fornecem uma categorização de contexto similar: contexto físico, contexto social e contexto interno. Dix *et al*. [24] incluem contexto de infraestrutura, contexto de sistema, contexto de domínio e contexto geral.

Soylu *et al*. [16] propõem oito categorias buscando alcançar uma maior granularidade e desejando representar as principais entidades de uma configuração de computação difundida típica, orientada para o mundo real. Essa categorização fornece uma camada clara para o desenvolvimento do sistema sensível ao contexto,

servindo como direção a uma conceituação genérica. Sua categorização em camadas de contexto não considera qualquer relação taxonômica, sendo: (1) contexto do usuário (interno, externo), (2) contexto do dispositivo (hardware e software), (3) contexto da aplicação, (4) contexto da informação, (5) contexto ambiental (físico, digital), (6) contexto temporal, (7) contexto histórico, (8) contexto relacional.

A categorização dada por Dey *et al*. [25] define quatro características de contexto essenciais: identidade, localização, status (ou atividade) e tempo. Identidade refere-se à capacidade de atribuir um identificador exclusivo a uma entidade. A localização consiste não somente em informações de posição em um espaço bidimensional, mas também inclui orientação e elevação, bem como todas as informações que podem ser usadas para deduzir relações espaciais entre as entidades, como co-localização, proximidade ou contenção. O status (ou atividade) identifica características intrínsecas da entidade que podem ser detectadas. Para um determinado lugar, pode ser a temperatura atual ou a luz ambiente ou o nível de ruído. Para uma pessoa, pode se referir a fatores fisiológicos, como sinais vitais ou cansaço, ou a atividade em que a pessoa está envolvida, como ler ou falar. Já o tempo, refere-se às informações de contexto temporais que ajudam a caracterizar uma situação, e isso permite aproveitar o valor de uma informação histórica.

Para tratar o contexto computacionalmente, é importante compreender a diferença entre contexto e elemento contextual, uma vez que o contexto é algo abstrato e, portanto, deve-se considerá-lo um conjunto composto por pequenos segmentos gerenciáveis que caracterizam sua composição total. Além disso, é preciso diferenciar os conceitos entre dados, informações e conhecimento dentro de contexto: os dados são sinais sem significado ou relação, advindos de sensores virtuais ou físicos, como coordenadas de localização ou temperatura. Já as informações são um entendimento dos relacionamentos entre as partes de dados, enquanto o conhecimento é identificação de padrões e comportamentos dessas informações entre elas [26].

Sendo assim, especificar e modelar aplicações sensíveis ao contexto significa representar as informações e sua relação com o comportamento do sistema [4], ou em outras palavras, elencar entidades de um domínio e suas características, considerando que toda característica de uma entidade faz parte do seu contexto, independentemente de qual fonte será obtida essa informação. Pode-se categorizar

as atividades para essa etapa em: (i) Especificação do Contexto, sendo a modelagem das informações contextuais necessárias para a aplicação; (ii) Gerenciamento do Contexto, como o contexto será manipulado pelo sistema, por meio da aquisição, armazenamento, processamento e disseminação; (iii) Uso do Contexto, como o contexto influencia o comportamento do sistema e como será utilizado [3].

Em resumo, o conceito de contexto na computação vem sendo abordado por duas diferentes frentes de pesquisa. Aqueles que pretendem simplesmente aplicar o contexto na melhoria dos serviços e das funcionalidades das suas aplicações [27][28], e aqueles que tratam contexto como um conceito formal, que buscam tratá-lo computacionalmente por meio de formalizações, modelos, frameworks e metodologias [16][13][29][30].

Neste trabalho será abordado o contexto como melhoria das funcionalidades e interação do usuário com uma aplicação, por isso foi utilizada a definição de Vieira *et al*. [3]. Os autores abordam o contexto como a interação entre um agente e uma aplicação, sendo dinâmico e dependente da tarefa atual do agente, construído em tempo de execução de uma interação. Para a construção do contexto, um elemento de contexto é qualquer dado, informação ou conhecimento o qual possa caracterizar uma entidade em um domínio e consequentemente formar o contexto como um todo.

2.2. SISTEMAS ADAPTATIVOS

Os sistemas adaptativos fazem parte de um subconjunto da computação autônoma, conceito inicialmente proposto para descrever sistemas computacionais capazes de autogerenciamento e adaptação a mudanças imprevisíveis, permitindo a expansão dos sistemas complexos com uma melhor utilização dos recursos computacionais [31]. Nas definições mais recentes, os sistemas adaptativos devem ser capazes de adaptar seu comportamento ao usuário e contexto [32]. Em uma definição mais ampla, são um conjunto de entidades interagentes as quais são capazes de responder a mudanças, ou em outras palavras, quando o comportamento do usuário e os parâmetros mudam, é tarefa da adaptação funcional identificar os métodos para adaptar o comportamento da aplicação em resposta [33][34].

Os sistemas de software podem assumir o conceito de autoadaptáveis, onde seu comportamento consiste em um feedback constante em *loop* com o usuário, com vistas a detectar mudanças na operação e responder em tempo de execução gerando auto adaptação. Para isso, o software deve observar seu próprio contexto, identificar mudanças relevantes, tomar decisões sobre as reações apropriadas e implementar essas decisões tomando medidas [35]. Para Oreizy *et al*. [36], o software auto adaptativo modifica seu próprio comportamento, por meio do ajuste de atributos, parâmetros ou artefatos do sistema, em resposta a mudanças em seu ambiente operacional, sendo ambiente operacional qualquer coisa observável pelo sistema de software, como a entrada do usuário, dispositivos de hardware externos e sensores, ou instrumentação de programas. A autoadaptação é mapeada como evolução fundamental no ciclo de vida de todo sistema de software, especialmente os de domínios de negócios altamente voláteis [37].

As adaptações podem derivar do próprio sistema de software como causas internas, ou a partir do contexto, em eventos externos. Esses processos dependem de propriedades de adaptação, características de domínio (informações ou modelos de contexto) e preferências das partes interessadas. As propriedades de adaptação estão classificadas na Figura 1, onde o nível geral de auto adaptação é o topo da pirâmide, seguida de um subnível, em que aparecem as propriedades de autoconfiguração, autocura, auto-otimização e autoproteção. Na base da pirâmide de classificação aparecem as propriedades que provêm mecanismos de implementação para o apoio das propriedades descritas anteriormente, sendo a ciência de contexto interno e externo, respectivamente [35].

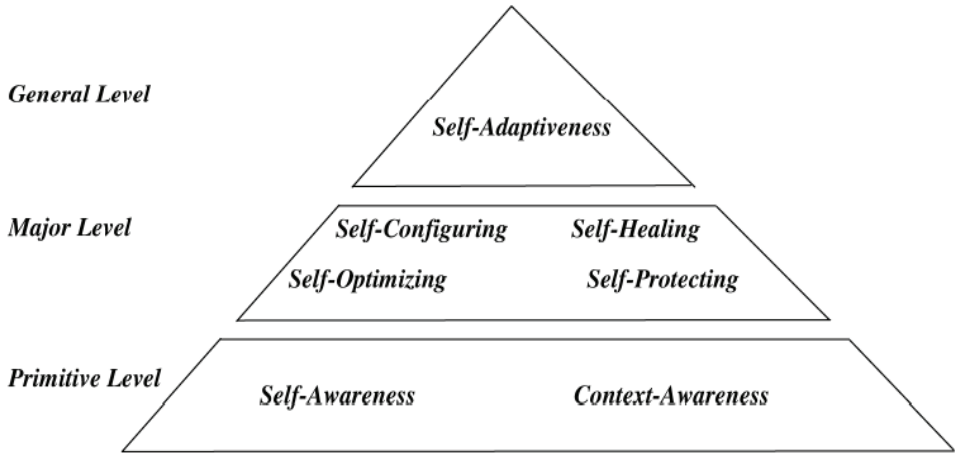


Figura 1. Propriedades de adaptação [35].

Salehie e Tahvildari [35] apresentam as questões 5W + 1H para obter requisitos de adaptação, as quais são: (when) Quando se adaptar? (why) Por que precisamos nos adaptar? (where) Onde precisamos implementar as mudanças? (what) Que tipo de mudança é necessária? (who) Quem tem que realizar a adaptação? (how) Como é realizada a adaptação? Outros autores formularam perguntas semelhantes [37] [38], da mesma forma como Krupitzer *et al*. [39] que respondem essas perguntas sob aspectos de sua própria taxonomia, nas dimensões de tempo, razão, técnica, nível e adaptação ao controle. Eles afirmam que na maioria das abordagens o foco é na adaptação reativa, sem a integração da adaptação ao contexto, por isso propõe sua própria abordagem incluindo o que chamam de adaptação ao contexto.

Basicamente, um sistema auto adaptativo mantém três modelos atualizados em tempo de execução: ambiente, arquitetura do sistema e os requisitos. O ambiente é considerado como tudo que o sistema não tem qualquer controle, mas pode ter impacto sobre sua funcionalidade, por exemplo um componente externo, largura de banda de rede ou a entrada do usuário. Por sua vez, a arquitetura é o principal artefato de representação abstrata de alto nível do sistema real, ou seja, representa os componentes de software e os recursos dos quais os componentes são implantados. Já os requisitos são a ligação entre o sistema e o ambiente. O sistema fornece funcionalidades para satisfazer os requisitos, sendo que mudanças no ambiente podem levar a mudanças significativas nos requisitos [40]

2.3. SISTEMAS SENSÍVEIS AO CONTEXTO

Os sistemas sensíveis ao contexto foram propostos inicialmente por Schilit e Theimer [41] em 1994. Os autores os definiram como softwares que se adaptam de acordo com a localização do usuário, a coleção de pessoas próximas, os hosts e os dispositivos acessíveis, bem como as alterações nessas características ao longo do tempo. Um sistema com esses recursos pode examinar o ambiente de computação e reagir às alterações no ambiente. Os autores restringiram a definição em: aplicações que são simplesmente informadas sobre o contexto e aplicações que se adaptam ao contexto. Posteriormente, Dey and Abowd [12] definiram o termo consciência de contexto da seguinte forma: “Um sistema é

sensível ao contexto se usa o contexto para fornecer informações relevantes e/ou serviços para o usuário, onde a relevância depende da tarefa do usuário".

Além dessas definições, o reconhecimento do contexto em aplicações tornou-se sinônimo de outros termos ao longo do tempo: sistemas adaptativos [42], reativos [43], responsivos [44], situados [45] e orientados ao ambiente [46]. Essas definições mais específicas, como "adaptativo", exigem que o comportamento de uma aplicação seja modificado para que seja considerado sensível ao contexto [12]. Outra definição específica em uma visão mais estreita é a de Brown [47], que define aplicações sensíveis ao contexto como aplicações que "fornecem automaticamente informações e/ou tomar ações de acordo com o contexto atual do usuário, conforme detectado pelos sensores", afirmando que essas ações podem assumir a forma de apresentar informações ao usuário, executando um programa de acordo com contexto, ou configurando um layout gráfico de acordo com o contexto. Este trabalho adotou a definição de Dey and Abowd [12] por ser mais ampla e completa.

Quanto aos recursos sensíveis ao contexto, depois de analisar e comparar os dois esforços anteriores realizados por Schilit *et al*. [13] e Pascoe [48], Dey e Abowd [12] identificaram três recursos que uma aplicação com reconhecimento de contexto pode suportar: apresentação, execução e marcação. Como apresentação, o contexto pode ser usado para decidir quais informações e serviços precisam ser apresentados ao usuário, enquanto em execução, ações precisam ser tomadas automaticamente com base no contexto, e finalmente na marcação, o contexto precisa ser anexado aos dados do sensor para serem processados e compreendidos posteriormente.

2.4. SISTEMAS ADAPTATIVOS SENSÍVEIS AO CONTEXTO

Esta seção abordará a mescla entre os conceitos apresentados anteriormente, proposição principal deste trabalho. Como observado nas seções anteriores, os sistemas adaptativos e os sistemas sensíveis ao contexto possuem muitas similaridades, entretanto não podem ser considerados totalmente idênticos.

Segundo o trabalho de Colman *et al*. [49], ambos detectam mudanças em seus ambientes e respondem alterando seu comportamento e/ou sua estrutura de forma adequada. Contudo, essas abordagens tendem a diferir em alguns aspectos, por exemplo: sistemas sensíveis ao contexto concentram-se na modelagem e no

raciocínio sobre o contexto ambiental relevante, utilizando modelos baseados em regras ou até mesmo, muitas vezes, ontologias formais, possuindo um determinado número de modos pré-estabelecidos. Os sistemas adaptativos, em contraste, concentram-se em como o sistema responde a mudanças ambientais imprevistas. No entanto, os sistemas adaptativos normalmente não possuem modelos sofisticados de contexto.

A auto adaptabilidade está mais preocupada em como adaptar o sistema, enquanto a consciência de contexto é mais preocupada com a forma de modelar, processar e gerenciar as informações do ambiente do sistema. Portanto, os requisitos de modelagem de contexto precisam ser considerados a partir da perspectiva de relacionamento entre sistema e contexto. Por outro lado, auto adaptabilidade precisa ter um sistema que possa lidar com as mudanças de contexto/requisitos antecipada e imprevista, devendo ser projetado objetivando a adaptação especificamente [50].

As aplicações adaptativas sensíveis ao contexto precisam considerar ambos os aspectos, ter a capacidade de perceber se o contexto muda, e se adaptar a eles. Sendo assim, o presente trabalho adotou ambos conceitos de forma conjunta, utilizando o termo simplificado “sistemas sensíveis ao contexto”, com a justificativa de que a capacidade de adaptabilidade estará intrínseca à modelagem ciente do contexto explicito, ou seja, usando uma infraestrutura de gerenciamento de contexto, separando a gestão do contexto e aplicação onde podem ser modificados de forma independente.

2.5. MODELAGEM DE SISTEMAS SENSÍVEIS AO CONTEXTO

Ao obter-se os requisitos de uma aplicação definidos e validados com o usuário, é necessário representar uma visão geral do domínio do sistema, bem como as entidades envolvidas, seus atributos e seus relacionamentos, facilitando o entendimento mútuo dos demais envolvidos em um projeto. Na etapa de modelagem é necessária uma representação formal dos dados de contexto em um modelo para verificação de consistência, bem como para garantir que um bom raciocínio seja executado sobre os dados.

Para esta etapa, duas questões tornam-se relevantes para uma estrutura de representação e raciocínio de contexto: a abstração do contexto em alto nível e a

incerteza das informações de contexto. Para isso, técnicas diferentes são propostas na literatura para apoiar a modelagem de dados: par chave-valor, linguagem de marcação, mapa de tópicos, ontologias e modelos gráficos, sendo que cada método possui suas próprias vantagens e desvantagens [51]. As abordagens existentes para modelagem de informações de contexto diferem na facilidade com que os conceitos do mundo real podem ser capturados pelos engenheiros de software por meio dos modelos de informações de contexto, no suporte que eles podem fornecer o raciocínio sobre informações de contexto, no desempenho computacional do raciocínio e na escalabilidade do gerenciamento de informações de contexto [52].

Neste trabalho foi adotada a técnica de modelagem gráfica, que apresenta vantagens em termos de heterogeneidade e interoperabilidade, além do fácil entendimento para desenvolvedores que não estão particularmente familiarizados com lógicas de descrição [52]. Para o modelo de dados, dois trabalhos da literatura destacam-se por utilizar diagramas UML para a atividade de especificação [4][8]. Em ambos os trabalhos são apresentadas propostas de modelos genéricos e abstratos para serem aplicados em domínios diferentes de sistemas. No primeiro trabalho são diferenciadas a modelagem de contexto estrutural do modelo de contexto dinâmico, orientado à eventos. Já no segundo, os autores denominam a dinâmica dos estados de uma entidade contextual como o foco, sendo a associação de um agente a à execução de uma tarefa, tornando o modelo sistemático e dinâmico.

Por considerar a UML uma linguagem atual e muito flexível/extensível, a qual explicita graficamente os conceitos e seus relacionamentos, neste trabalho também foi adotada essa notação, entretanto foi criado um modelo próprio a partir da adaptação dos conceitos de ambos os trabalhos. As adaptações visam agregar maior abstração conceitual ao modelo, uma vez que duas principais revoluções no gerenciamento de dados ocorreram recentemente: a análise de Big Data e os bancos de dados NoSQL (comumente referido como “Not Only SQL”)[53]. Os bancos de dados NoSQL surgem em resposta à demanda do desenvolvimento de aplicações móveis, em que cada vez mais os desenvolvedores trabalham com aplicativos que criam grandes volumes e novos tipos de dados, os quais mudam rapidamente, sejam estes dados estruturados, semiestruturados ou até mesmo não estruturados. Com o NoSQL são abordados problemas como dificuldades de expansão horizontal do banco de dados, baixo desempenho de consulta e baixa

capacidade de armazenamento, tentando reduzir diretamente os custos de infraestrutura, mantendo a escalabilidade e oferecendo soluções para distribuição e particionamento sem esforço de modelos de dados [53][54][55].

Considerar esse novo paradigma em uma modelagem conceitual torna-se importante, pois diferentemente dos bancos de dados relacionais, em NoSQL segue-se o paradigma BASE (Basicamente disponível, “Soft-State” e Eventualmente consistente), ou seja, itens primordiais de soluções relacionais como cardinalidade e normalização tornam-se secundários ou até mesmo inexistentes. O modelo conceitual deste trabalho, com as adaptações realizadas, propõe como diferencial abstrair esse novo paradigma, de forma que possa ser interpretado independentemente de qual será a escolha para os modelos lógicos e físicos de dados, no que diz respeito às entidades e seus relacionamentos. Acredita-se que para o entendimento dos elementos de contexto e sua relação com as entidades domínio, seja possível um esquema simplificado de design, com informações gráficas mais próximas possíveis do mundo real e menor complexidade técnica.

Além da UML, outra notação utilizada neste trabalho é o BPMN, uma linguagem de modelagem padrão para processos de negócios, a qual fornece notação gráfica para especificar processos de negócios em diagramas. O objetivo da notação é que seja compreensível para os usuários de negócios, mas que represente também uma semântica de processos complexos para usuários técnicos [56]. Além disso, possui um mecanismo interno de extensão [6] utilizado para criar a extensão Context4BPMN [57], a qual foi reaproveitada para o modelo proposto neste trabalho e será descrita de forma instanciada na sequencia do texto.

A notação Context4BPMN (Figura 2) é composta por cinco elementos: os retângulos que simbolizam os eventos da aplicação e possuem um ícone interno para classificar esse elemento em eventos de contexto e eventos intermediários. Eventos contextuais são representados por um olho, indicando que naquele momento há obtenção e/ou processamento de contexto necessários, podendo ser eventos de interrupção (dois círculos) ou ininterruptos (pontilhados). A condição de ativação de um evento, ou como denominado anteriormente, regra, é declarada em um hexágono com gramática livre para formalizar as condições (associado com uma linha pontilhada). O último elemento são as linhas com setas entre os retângulos e hexágonos que simbolizam as condições e caminhos da aplicação conforme tomada de decisão baseada nas regras.

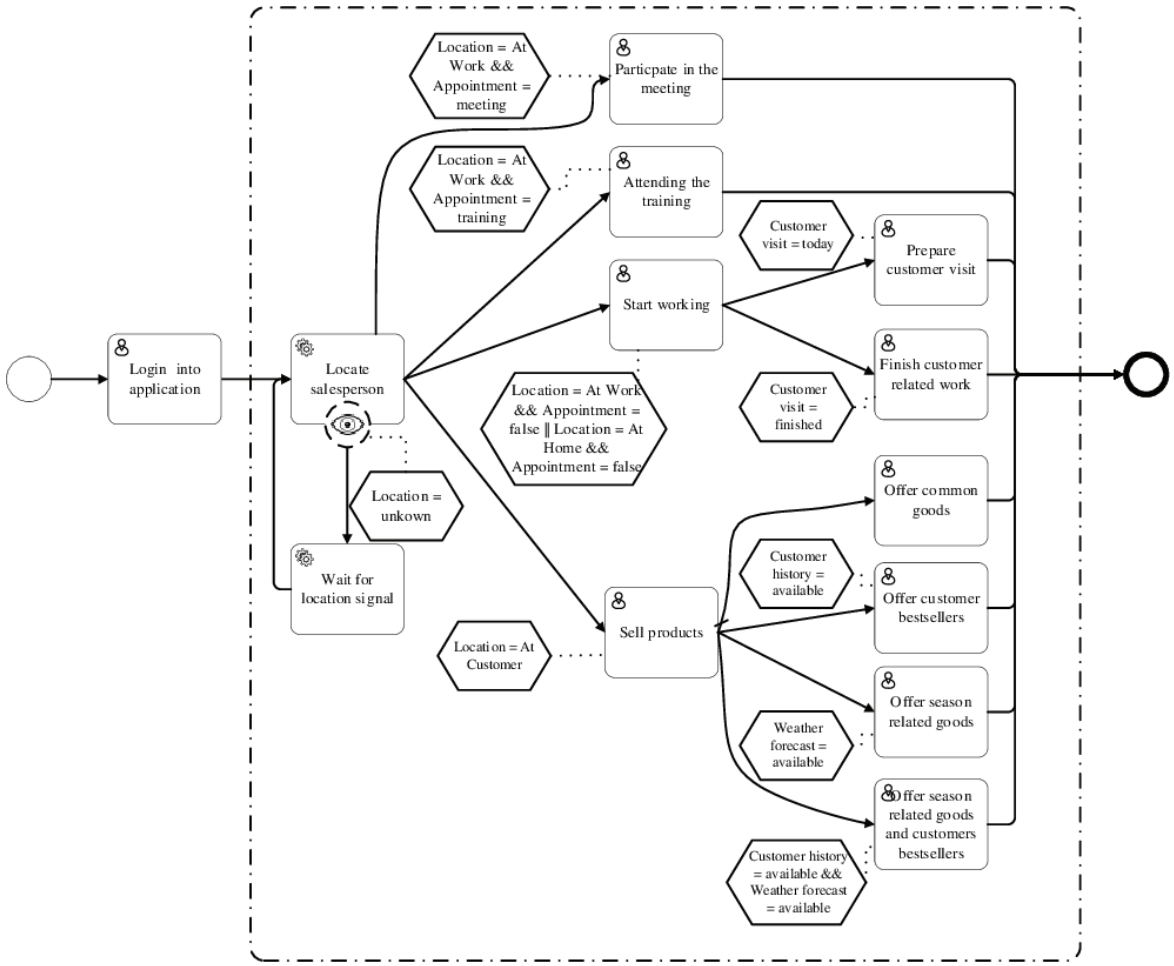


Figura 2. Exemplo utilizando a notação Context4BPMN [6].

A Figura 2 apresenta o exemplo utilizado pelos autores originais da notação, onde modelam os processos dinâmicos de uma aplicação para dispositivos móveis sobre vendedores externos, ou representantes comerciais como são popularmente denominados. Eles trabalham parcialmente locados no cliente, pois vendem produtos da empresa que representam, bem como realizam treinamentos e consultoria. Contudo, alguns documentos podem ser feitos em casa, e sendo assim, o primeiro parâmetro de contexto é o local, com três estados: no cliente, no escritório ou casa. Outros parâmetros a considerar são a estação e o clima, isso porque alguns produtos podem ser melhor vendidos em estações específicas ou em determinadas condições climáticas. O histórico do cliente é o último parâmetro de contexto, e com ele o aplicativo deve ser capaz de adaptar a lista de mercadorias às preferências do cliente. Por exemplo, produtos não alimentícios podem ser

desativados se o cliente só comprou produtos relacionados a alimentos até o momento.

A Figura 2 apresenta a solução para o problema, utilizando os conceitos do modelo proposto e exemplificados pelos autores. Após o *Login* na aplicação, um evento é acionado e simbolizado por um olho envolto por um círculo pontilhado, indicando evento de contexto sem interrupção. A condição de ativação é declarada no hexágono com uma gramática livre para formalizar as condições. Uma atividade paralela é iniciada, na qual o aplicativo tenta determinar a localização do vendedor, e se ela for encontrada, outras quatro tarefas serão possíveis na sequência, por exemplo, o vendedor participa do treinamento de vendas se estiver no trabalho e houver um treinamento anotado no calendário. Outro fluxo é quando o vendedor está no cliente, e nesse caso ele irá vender os produtos da empresa para o cliente. Os outros dois fluxos seguem a mesma lógica.

Após filtrar as quatro tarefas de acordo com a localização do vendedor, outras condições de ativação em uma segunda camada são apresentadas nos hexágonos suspensos que antecedem as demais tarefas. Um exemplo é o caso da sequência da tarefa de vender produtos, onde a aplicação pode oferecer suporte a esse processo, mostrando os mais vendidos do cliente e/ou os produtos que dependem da estação. A primeira ação precisa do histórico do cliente, podendo ser consultado por meio de uma conexão com o sistema de gerenciamento de relacionamento (CRM) do cliente. Outro caso são os produtos sazonais que podem ser mostrados quando a previsão do tempo para a área local estiver disponível, e assim por diante conforme a Figura 2

2.6. TRABALHOS RELACIONADOS

A literatura apresenta trabalhos descrevendo frameworks para o desenvolvimento de aplicações móveis sensíveis ao contexto. Além disso, há uma distinção entre diferentes tipos de suporte de software para criação de aplicações sensíveis ao contexto, classificados em bibliotecas, estruturas conceituais, kits de ferramentas ou infraestruturas. Essas abordagens não são exclusivas, e em muitos casos pode ser útil utilizar todas elas.

Bibliotecas são conjuntos de algoritmos relacionados com o objetivo de fornecer códigos reutilizáveis. Por outro lado, as estruturas se concentram mais na

reutilização do design básico para uma determinada classe de aplicativos. Já os kits de ferramentas baseiam-se nas estruturas, oferecendo também um grande número de componentes reutilizáveis para funcionalidades comuns. Uma infraestrutura é um conjunto de tecnologias bem estabelecido, difundido, confiável e acessível, a qual atua como base para outros sistemas, onde qualquer tipo de dispositivo ou aplicativo pode usar esses serviços aderindo a formatos de dados e protocolos de rede predefinidos. Um framework deve suportar essas diferentes abordagens de forma mútua [58].

O campo de pesquisa é muito abrangente e ao longo dos anos, diferentes paradigmas de desenvolvimento, tecnologias e infraestrutura de hardware estiveram em rápida evolução. Os modelos, arquiteturas e processos consequentemente acompanharam essa evolução, com trabalhos voltados ao paradigma relativo de cada época. Um dos primeiros trabalhos, ainda na década de 90, Schilit el al. [13] concentraram-se em uma arquitetura geral para suportar a computação móvel sensível ao contexto. A arquitetura visava apoiar a coleta de informações de contexto sobre dispositivos e usuários.

Com o crescimento no uso de dispositivos móveis, trabalhos foram desenvolvidos detalhando estruturas. Coppola *et al*. [59] apresentam um framework conceitual com reconhecimento de contexto, com o objetivo de baixar aplicativos conforme o contexto do usuário em um ambiente cliente-servidor. Posteriormente, La e Kim [7] apresentaram sua estrutura conceitual para provisionamento de contexto baseado em serviços na nuvem. Os autores citam a crescente tendência em Cloud Computing sendo amplamente aceita como um paradigma de reutilização eficaz para novas aplicações voltadas aos dispositivos móveis. O principal diferencial desse trabalho é a utilização do contexto para customizar a invocação de serviços na nuvem, ou seja, quando não há informações de contexto a aplicação está ligada diretamente com o serviço, caso contrário as informações fornecem uma pista chave para adaptar serviço de forma dinâmica em termos de personalização de serviços e reação a falhas.

Milic [5] por sua vez aborda um framework para aplicações móveis baseado na nuvem objetivando reduzir o volume de dados enviados ao servidor, propondo o processamento local de informações de contexto, no próprio dispositivo do usuário, e só posteriormente transmitidas ao servidor. O autor cita em sua proposta a previsão de serviços executados em segundo plano, ou seja, obtenção e

processamento de informações de contexto mesmo quando a aplicação estiver em segundo plano no sistema operacional do dispositivo. Este trabalho aborda especificamente o domínio de aplicações voltadas aos cuidados de saúde, assim como Mahmud e Sattar [60], que apresentam uma infraestrutura para unificar as informações de contexto e de domínio obtidas por diferentes dispositivos, não somente smartphones, mas utilizando o conceito de Internet das Coisas (IoT) para a área de cuidados de saúde. A infraestrutura apresentada abstrai as camadas de aplicações ou serviços, concentrando-se apenas em uma estrutura que permita unificar e conectar as informações obtidas de forma sistemática.

Os trabalhos relacionados apresentados acima mencionam arquiteturas e estruturas de implementação, por vezes genéricas, em outras, direcionadas a um domínio específico. Entretanto, na literatura também encontram-se trabalhos que possuem ênfase em conceber e modelar aplicações por meio de modelos e processos, de forma teórica, e descrevendo o processo de modelagem e desenvolvimento. É o caso do trabalho de Seel e Dörndorfer [6] que propõe um framework para modelar e implementar aplicativos para dispositivos móveis sensíveis ao contexto. O modelo consiste em três camadas principais: a camada de modelagem de negócios sensível ao contexto, descrevendo as diferenças de uma abordagem tradicional e utilizando uma notação estendida do BPMN, incluindo o contexto nos fluxos e tomadas de decisão. Também descrevem a camada de modelagem dos dados de sensores, com o objetivo de abstrair as entidades, atributos e seus relacionamentos. Por fim, a camada de arquitetura que descreve um modelo cliente-servidor, sendo a aplicação cliente a responsável por coletar os dados dos sensores e demais fontes, e o servidor para armazenamento de processamento das informações tanto da aplicação em geral quanto de contexto.

A principal contribuição de Seel e Dörndorfer [6] é fornecer um modelo integrado com processos e notações atuais, estendidas e específicas para cada uma das camadas, agregando o contexto como parte dos processos de concepção de design de aplicações sensíveis ao contexto. Semelhante a este trabalho, Cheikh *et al*. [8] utilizam UML para fornecer metamodelos de contexto em uma abordagem integrada para projetar aplicações sensíveis ao contexto. Os autores separam a modelagem de contexto em estrutural, ou seja, aquilo que são atributos de entidades, suas origens e links, enquanto os modelos de contexto dinâmico apresentam uma visão de eventos do contexto. Apresentam uma lógica orientada a

eventos. O método é usado para criar um aplicativo de assistente de passageiro no domínio do transporte inteligente.

Um framework para projetar e desenvolver aplicações sensíveis ao contexto pode conter diferentes atividades dentro do processo de software. A maioria deles concentra-se nas atividades de modelagem, design e arquitetura, como citados anteriormente. Entretanto, outras atividades, como o levantamento de requisitos e testes, são muito importantes e possuem trabalhos encontrados na literatura [61] [62] [63] [64]. O framework proposto neste trabalho enquadra-se na primeira categoria, e acredita-se que posteriormente o mesmo possa ser estendido adicionando as atividades específicas de levantamento de requisitos e testes.

Com exceção do trabalho de Cheikh *et al*. [8], as demais abordagens apresentadas nesta seção propõem metamodelos de contexto, mas não fornecem guias ou instruções para o uso dos mesmos, a fim de facilitar a tarefa dos analistas e desenvolvedores. Considerando estes fatores, o diferencial do framework proposto neste trabalho é oferecer modelos de representação, bem como tratar dos requisitos funcionais e arquitetônicos, fornecendo também ferramentas e guias de instruções para desenvolver sistemas sensíveis ao contexto com tecnologias e serviços adequados.

3. FAM4CSS: UM FRAMEWORK PARA APLICAÇÕES MÓVEIS SENSÍVEIS AO CONTEXTO

Neste capítulo será descrito o framework proposto, denominado como “FAM4CSS”. Inicialmente serão abordadas as premissas e referências a serem utilizadas, bem como os requisitos para desenvolver uma aplicação sensível ao contexto. A decisão em gerar este tipo de artefato teórico com guias práticos surgiu a partir da revisão da literatura, aliada ao contexto atual tecnológico e novas ferramentas de mercado disponíveis para desenvolvedores, principalmente os de aplicativos móveis. Com isso, identificou-se que uma descrição integrada entre modelagem e arquitetura, com linguagens simplificadas e exemplos práticos poderia elucidar itens complexos relativos a atividade de projetar aplicações sensíveis ao contexto. Além disso, um guia teórico permitirá que o mesmo seja interpretado e evoluído de diferentes formas, enquanto outras contribuições como códigos, plug-ins e *widgets* reutilizáveis podem ser resultados consequentes a esse modelo generalizado.

3.1. REQUISITOS PARA O DESENVOLVIMENTO DE APLICAÇÕES SENSÍVEIS AO CONTEXTO

Projetar e desenvolver aplicações sensíveis ao contexto envolve muitos desafios. Vieira *et al*. [3] elencaram a caracterização dos elementos contextuais para uso na aplicação e a sua representação em um modelo semântico; a aquisição dos elementos contextuais a partir de fontes heterogêneas (por exemplo sensores físicos, bases de dados, agentes e aplicações); o processamento e interpretação das informações adquiridas a partir dessas fontes; a distribuição e o reaproveitamento dos elementos de contexto entre diferentes aplicações e a adaptação da aplicação às variações de contexto. Além desses, Dey e Abowd [25] também citam a separação de preocupações no desenvolvimento, a disponibilidade constante de aquisição de contexto, o armazenamento de contexto e história, bem como a descoberta de recursos e a transparência dos mesmos.

Adicionalmente a esses requisitos, ao desenvolver aplicações especificamente projetadas para dispositivos móveis, é preciso considerar fatores

como a variedade de dispositivos e plataformas mantendo compatibilidade e interoperabilidade, a indisponibilidade de conexão, a capacidade de armazenamento, a energia finita do dispositivo e principalmente a relação dessa energia com o processamento delegado ao dispositivo por meio de uma aplicação. Outra característica importante a considerar são os estados das aplicações nos sistemas operacionais móveis, em que dispõem as aplicações em segundo plano, ou seja, ainda que uma aplicação não esteja em uso ativo na tela do usuário, deve prover disponibilidade dos seus serviços conforme se propõe.

3.2. DETALHAMENTO DO FRAMEWORK FAM4CSS

Um framework conceitual pode ser definido como um esquema conceitual ou um modelo de dados, para um ou mais domínios, bem como uma representação com alto nível de abstração a qual modela os fatos do mundo real, suas propriedades e seus relacionamentos, de forma a preservar a semântica da aplicação [65]. O FAM4CSS é um framework conceitual, composto por um conjunto de metamodelos e diretrizes, com o objetivo de auxiliar o desenvolvimento de aplicações que se adaptem e reconheçam o contexto em suas funcionalidades.

A proposta é agrupada em três conjuntos: i) a modelagem e especificação de sistemas sensíveis ao contexto, que conta com dois modelos conceituais e a orientações de como utilizá-los; ii) uma arquitetura proposta a partir de um modelo baseado na nuvem, que agrega diretrizes e tecnologias para implementação da aplicação; e iii) desenvolvimento, reunindo os modelos propostos e as diretrizes, tecnologias e boas práticas para o desenvolvimento de aplicações sensíveis ao contexto por meio de exemplos reais em uma aplicação.

Para tanto, foram desenvolvidos dois artefatos voltados à representação do domínio, de forma clara e mais simples possível para que possam ser usados na fase de projeto e desenvolvimento. O primeiro artefato é o modelo de dados da aplicação, em que entidades e seus relacionamentos são representados de forma gráfica, e o segundo é um modelo de processos para apoiar a dinâmica do contexto por meio de fluxos, complementando a representação de dados. É importante ressaltar que ambos modelos são independentes e que podem ser usados isoladamente ou de forma complementar. A escolha de como utilizar é estabelecida

conforme a metodologia e/ou filosofia de desenvolvimento adotada pela equipe, sendo priorizado mais ou menos documentações nos seus processos.

3.2.1. Modelo Estrutural

Nesta seção será apresentado o modelo estrutural para representação das entidades e dos elementos contextuais envolvidos em uma modelagem para sistemas sensíveis ao contexto. O modelo proposto torna-se útil para ilustrar graficamente os conceitos e seus relacionamentos, em um nível de modelo conceitual, podendo posteriormente ser instanciado nos modelos lógico e físico independente da tecnologia escolhida. As principais adaptações em relação aos trabalhos de Cheikh *et al*. [8] e Vieira *et al*. [4] são a notação de relacionamento sem cardinalidade: (i) um para um → 1: 1, (ii) um para muitos → 1: * e (iii) muitos para muitos → *: * transcritas de forma descritiva, por exemplo “é de um tipo” define uma relação um para um, enquanto “disparado por” ou “pode possuir” indicam a obrigatoriedade e relação um para muitos/muitos para muitos (descrito pelos símbolos maior e menor <<Entidade>>). Essa adaptação busca agregar maior abstração e independência entre o modelo conceitual e o lógico. Outras adaptações são a utilização do conceito de eventos e ações para representar a dinamicidade dos dados no modelo.

O modelo baseia-se em cinco conceitos principais, conforme ilustra a Figura 3, a saber: Entidade, Elemento, Evento, Regra e Ação; outros dois conceitos que caracterizam um elemento contextual: fonte e tipo; e outro conceito que caracteriza a frequência de um evento. Os conceitos principais são o cerne do modelo, onde uma entidade é qualquer objeto do mundo real o qual possa servir para distinguir e descrever algo relativo ao domínio de uma aplicação (por exemplo: pessoa, empresa, função, etc). Uma entidade necessariamente é formada por pelo menos um elemento, conceito referente aos atributos ou características que descrevam uma entidade (por exemplo, para entidade Pessoa temos nome, idade, data de nascimento, localização, dentre outros), sendo importante destacar que todo e qualquer elemento deve ser considerado contextual, pois compõe um cenário de contexto como um todo, independentemente de quais formas serão obtidas essas informações posteriormente.

Em sistemas tradicionais, normalmente essas informações são obtidas de forma explicita por parte do usuário, enquanto em sistemas sensíveis ao contexto, algumas delas poderão ser obtidas de forma implícita e sistemática. No modelo deste trabalho, um elemento contextual quando obtido de forma implícita, é tratado em separado e associado externamente a uma entidade, possuindo relacionamentos com suas próprias entidades que o descrevem. Essa modelagem a parte não é necessária para os demais elementos obtidos de forma explícita, pois serão informados manualmente pelo usuário. Por consequência disso, o conceito Tipo (pré-definido em cinco possibilidades) caracteriza um elemento contextual de acordo com a definição 5W+1H, ou seja, o elemento corresponde a pelo menos uma das perguntas: quem? (um agente), onde? (um local), quando? (um momento), o que? (uma atividade ou evento), por quê? (uma motivação) e como? (de que forma). Outro conceito é o de Fonte, que define obrigatoriamente uma ou mais fontes responsáveis por obter as informações para o elemento em questão. Essas fontes podem ser das mais variadas, como por exemplo a utilização dos sensores de um dispositivo, sensores externos integrados à aplicação via dispositivo ou middleware de sistema, servidores na nuvem, informações de interação do usuário com a aplicação, dentre outros.

O terceiro conceito relacionado ao elemento de contexto é o Evento, o qual caracteriza o momento e a frequência (opcional) em que os dados serão atualizados, e por consequência, o modelo será instanciado pela aplicação. Eventos podem ser obtidos diretamente das fontes, sem composição ou processamento, ou de forma composta a partir do processamento de informações adquiridas e (ou) armazenadas. Um evento simples poderia ser um acesso a tela do sistema, um evento complexo poderia ser uma consulta e filtro de dados ou quando o usuário chega a uma determinada localização, enquanto um evento temporal poderia ser, programaticamente, todos os dias ao meio dia, por exemplo. Além do momento e frequência de um evento, o conceito de Regras no modelo é necessário para produzir informações a partir de dados obtidos pelos elementos contextuais, assim como filtrar e priorizar ações por meio dos eventos. Os conceitos de regras e ações devem permitir que o projetista/analista de sistemas indique explicitamente as ações de adaptação sensíveis ao contexto e como essas ações são disparadas sistematicamente.

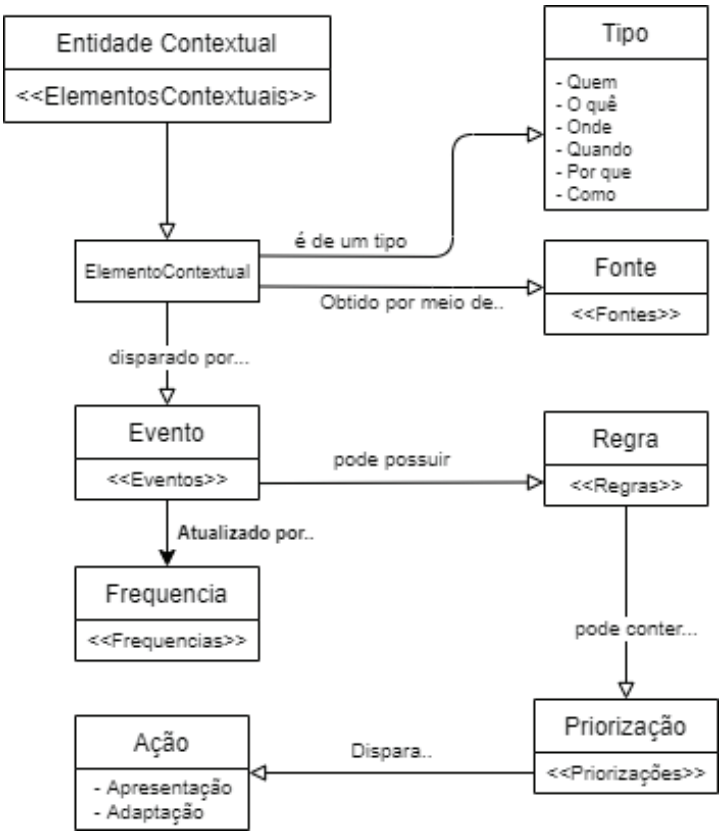


Figura 3. Modelo estrutural e conceitual para representação de contexto

Para ilustrar o modelo, considera-se uma aplicação que sugere as refeições de café, almoço e janta, conforme a região do usuário e a estação do ano atual. Conforme o horário mais próximo, é possível saber em qual das três refeições será a sugestão, assim como pela data é possível consultar qual a estação do ano atual, e por fim, pela localização do usuário, sabe-se a região do país onde ele se encontra. Como premissa, as refeições estarão classificadas e armazenadas em um banco de dados, prontas para serem selecionadas conforme o contexto do usuário (Figura 6). As refeições são classificadas podendo ser indicadas em uma ou mais estações e uma ou mais regiões, ficando de responsabilidade da aplicação resolver duplicidades de refeições. Neste exemplo, são oito refeições de cada tipo (café, almoço e janta), onde cada uma delas se enquadra bem em determinadas regiões e estações. Caso a sugestão obtenha duas ou mais refeições para uma estação e região, não há prioridade, seja, qualquer uma delas pode ser apresentada, sendo recomendado o uso de uma priorização aleatória – ou somente, sem priorização.

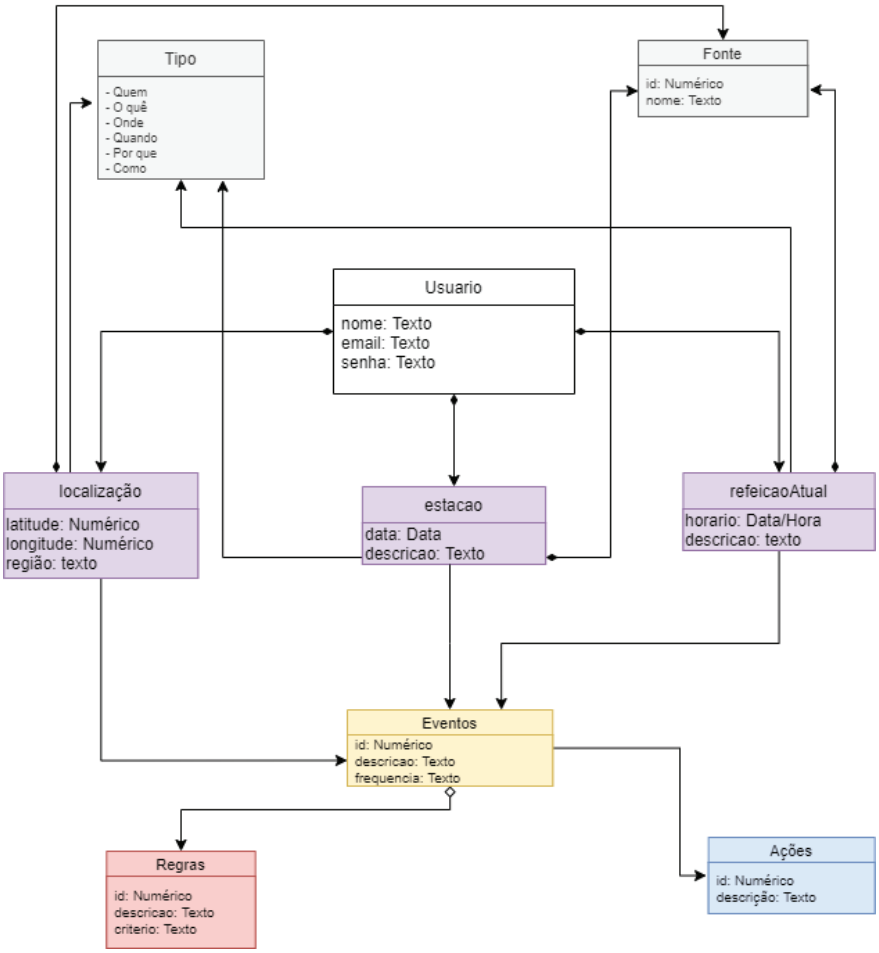


Figura 4. Modelo conceitual lógico instanciado para o exemplo.

Aplicando os conceitos do modelo e instanciando-o nas Figuras 4 e 5, pode-se dizer que: um Usuário é uma entidade contextual a qual possui os atributos nome, Email, senha os quais serão obtidos de forma explícita, por meio de um formulário de cadastro. Além disso, o usuário encontra-se em uma localização, em uma estação do ano e em um determinado momento (data e hora) onde o momento definirá qual a refeição atual mais próxima. Esses elementos devem ser obtidos de forma implícita, por meio de sensores, por isso devem estar associados externamente à entidade principal na modelagem e cada elemento corresponde a um tipo, por exemplo: a localização refere-se a “onde? (where) ” enquanto a data, hora e estação refere-se a “Quando? (when). Cabe salientar que os demais tipos como “Quem” refere-se ao usuário, e assim por diante, mas não irá modelar-se os elementos estáticos da aplicação, pois o foco são os elementos de contexto implícito. De acordo com o modelo, os dados para os elementos contextuais

implícitos podem vir de uma ou várias fontes, e elas devem estar relacionadas com o elemento declarado, por exemplo, a localização pode-se obter por meio do GPS, enquanto a estação do ano pode ser obtida com uma consulta a um banco de dados servidor conforme a data atual, enquanto a refeição atual pode ser obtida por meio do horário do sistema operacional do dispositivo.

Outro conceito aplicado é o de evento, onde nesse exemplo pode-se considerar que cada elemento contextual terá um evento de disparo, onde irá ocorrer a obtenção dos dados. Nesse exemplo serão três eventos: 1) obter a localização, 2) obter estação do ano e 3) obter próxima refeição cada qual vinculado e caracterizando o seu elemento respectivo. Contudo, eventos podem ocorrer em consequência de outros eventos, por exemplo, após obter os três parâmetros de caracterização do contexto do usuário, um evento de busca personalizada da refeição sugerida na base de dados deve ser disparado.

Um ou mais eventos podem precisar de regras para processar os dados adquiridos e filtrar os resultados, sendo assim, deve-se declarar também qual será a forma de priorização caso haja ambiguidade de contexto. Considerando o exemplo, se duas ou mais refeições cadastradas e classificadas na base de dados se encaixarem na busca com os parâmetros de contexto obtidos, deve-se indicar qual o critério para resolver a duplicidade. Para domínios mais simples, pode-se utilizar a priorização manual, que consiste em indicar explicitamente o resultado escolhido, por exemplo a primeira feira acessada pelo usuário, ou até mesmo em outros casos, enumerar as possibilidades com um valor único e ordenado de prioridade (1, 2, 3, etc..). Por outro lado, domínios complexos exigem motores de inferência anexados ou implementados no sistema. O modelo deste trabalho não entra em detalhes sobre motores de inferência, mas abstrai e contempla o uso dos mesmos para resolução de conflitos. Os trabalhos de Vieira *et al*. [4] e Cheikh *et al*. [8] descrevem essas duas estratégias.

Uma terceira forma de resolver conflitos também pode ser usada e é adicionada no modelo desse trabalho em relação aos demais, a qual será chamada de critério randomizado (sem critério de escolha), ou seja, mesmo que haja ambiguidade em dados de contexto resultantes, o sistema deve sortear aleatoriamente qual informação será apresentada. Indica-se essa solução para casos onde todas as opções em ambiguidade podem ser utilizadas em resposta, porém apenas uma deve ser escolhida.

Por fim, após descrever as possíveis regras aplicadas aos parâmetros de contexto obtidos e resolução de duplicidade, serão disparadas ações. Estas podem ser uma apresentação de dados ou uma adaptação do sistema. Em resumo, os elementos contextuais devem ser tratados como entidades relacionadas, sendo possível modelar a estrutura desses elementos em separado, formatando o contexto de uma forma ampla para toda uma funcionalidade do sistema. É importante salientar que os exemplos utilizados neste trabalho são de domínio simples para facilitar o entendimento do modelo. Entretanto, em aplicações mais complexas, essa modelagem deve considerar o conceito de reaproveitamento dos elementos contextuais entre diferentes entidades de uma aplicação, e assim compartilhando o contexto entre diversas funcionalidades.

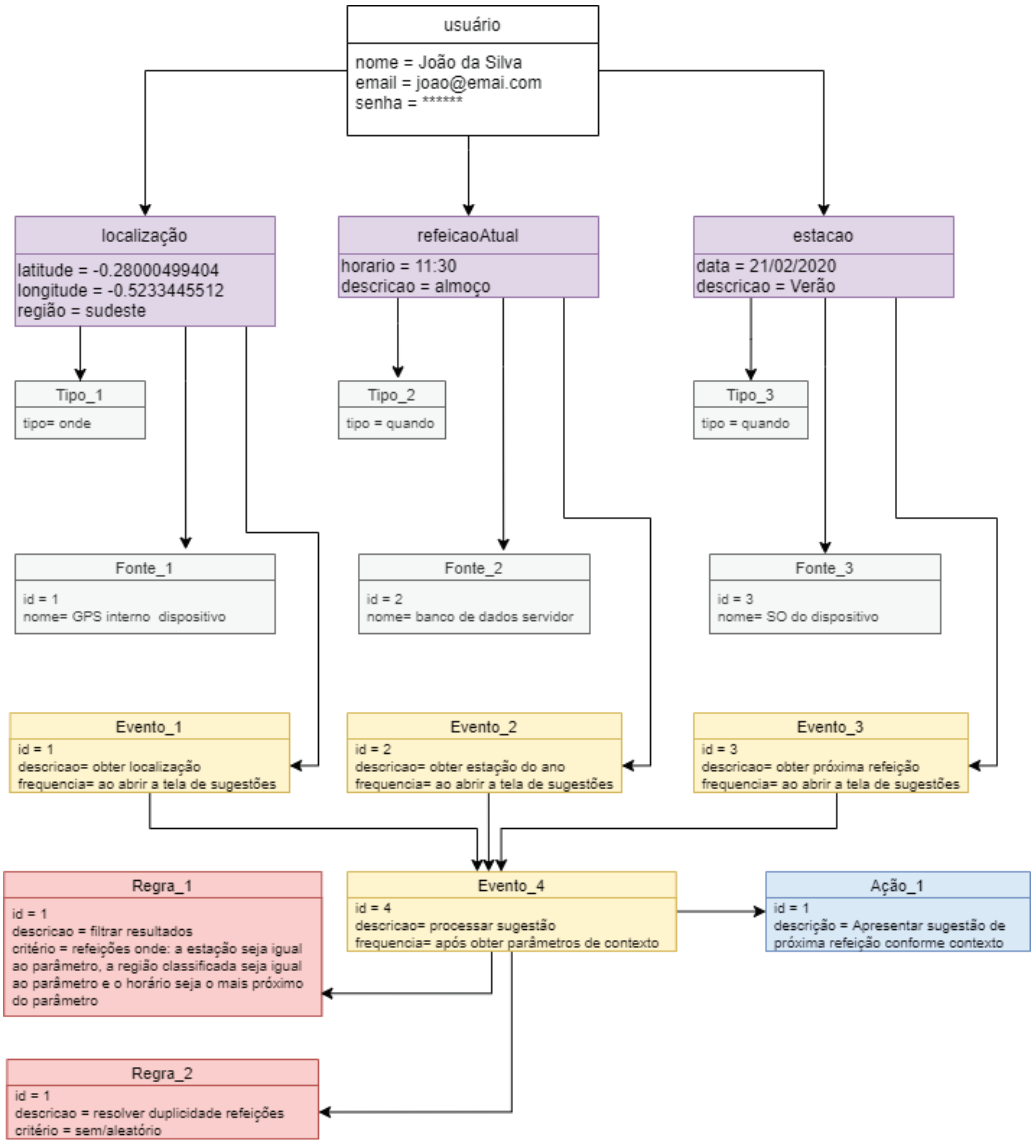


Figura 5. Modelo instanciado por meio de diagrama de classe com dados.

Horário	Nome	Descrição	Estações				Regiões				
08:00	Café da manhã	Café com leite, bolo de milho e melão	Verão	Inverno	Outono	Primavera	Norte	Nordeste			
08:00	Café da manhã	Leite, cuscuz, ovo de galinha e banana	Verão	Inverno	Outono	Primavera	Norte	Nordeste			
08:00	Café da manhã	Café, pão integral com queijo e ameixa.	Inverno				Sul	Sudeste	Centro-Oeste		
08:00	Café da manhã	Café com leite, tapioca e banana.	Verão	Inverno	Outono	Primavera	Sul	Sudeste	Centro-Oeste	Norte	Nordeste
08:00	Café da manhã	Café com leite, pão de queijo e mamão.	Verão	Primavera			Sudeste	Centro-Oeste			
08:00	Café da manhã	Café com leite, bolo de mandioca, queijo e mamão	Verão	Primavera			Sul	Sudeste	Centro-Oeste		
08:00	Café da manhã	Suco de laranja natural, pão francês com manteiga	Verão	Inverno	Primavera		Sul	Sudeste	Centro-Oeste	Norte	Nordeste
08:00	Café da manhã	Café com leite, cuscuz e manga.	Verão	Primavera			Norte	Nordeste			
12:00	Almoço	Arroz, feijão, coxa de frango assada, beterraba e p	Verão	Inverno	Outono	Primavera	Sul	Sudeste	Centro-Oeste		
12:00	Almoço	Arroz, feijão, omelete e jiló refogado.	Verão	Inverno	Outono	Primavera	Sul	Sudeste	Centro-Oeste		
12:00	Almoço	Alface, arroz com lentilha, pernil, batata, repolho e	Verão	Inverno	Outono	Primavera	Sul	Sudeste	Centro-Oeste		
12:00	Almoço	Alface, tomate, feijão, farinha de mandioca, peixe	Verão	Inverno	Outono	Primavera	Norte	Nordeste			
12:00	Almoço	Feijoada, arroz, vinagrete de cebola com tomate, fa	Verão	Inverno	Outono	Primavera	Sudeste	Centro-Oeste			
12:00	Almoço	Salada de tomate, arroz, feijão, bife grelhado e sal	Verão	Inverno	Outono	Primavera	Sul	Sudeste	Centro-Oeste	Norte	Nordeste
12:00	Almoço	Arroz, feijão, angu, abóbora, quiabo e mamão.	Inverno	Outono			Norte	Nordeste			
12:00	Almoço	Alface e tomate, arroz, feijão, beringela e suco de	Verão	Primavera			Norte	Nordeste			
20:00	Janta	Arroz, feijão, figado bovino e abobrinha refogada.	Verão	Inverno	Outono	Primavera	Sudeste				
20:00	Janta	Salada de folhas, arroz, feijão, ovo e maçã.	Verão	Inverno	Outono	Primavera	Sul	Sudeste	Centro-Oeste	Norte	Nordeste
20:00	Janta	Sopa de legumes, farinha de mandioca e açaí.	Verão	Inverno	Outono	Primavera	Norte	Nordeste			
20:00	Janta	Salada de folhas, macarrão e galeto.	Inverno	Outono			Sul	Sudeste			
20:00	Janta	Arroz, feijão, coxa de frango, repolho, moranga e la	Inverno	Outono			Sul	Sudeste	Centro-Oeste	Norte	Nordeste
20:00	Janta	Alface, tomate, arroz, feijão, omelete, mandioca de	Verão	Inverno	Outono	Primavera	Sul	Sudeste	Centro-Oeste	Norte	Nordeste
20:00	Janta	Arroz, feijão, peito de frango, abóbora com quiabo	Inverno	Outono			Norte	Nordeste			
20:00	Janta	Arroz, feijão, carne moída com legumes	Verão	Inverno	Outono	Primavera	Sul	Sudeste	Centro-Oeste	Norte	Nordeste

Figura 6. Simulação de um banco de dados de refeições classificadas.

3.2.2. Modelo Dinâmico

Para apoiar o entendimento da modelagem estrutural, pode-se agregar um modelo dinâmico baseado em notação de processos de negócio. Analisando os elementos gráficos caracterizados na notação Context4BPMN, percebeu-se que os mesmos podem ser associados aos conceitos do Modelo Estrutural, apoiando a sincronia dos modelos em relação aos conceitos e sua utilização.

A Figura 7 apresenta o diferencial deste trabalho em relação ao de Seel e Dörndorfer [6], por propor um metamodelo dos conceitos aplicados, ou seja, uma forma abstrata e sistemática de pensar na modelagem dinâmica do contexto em um paradigma de processos.

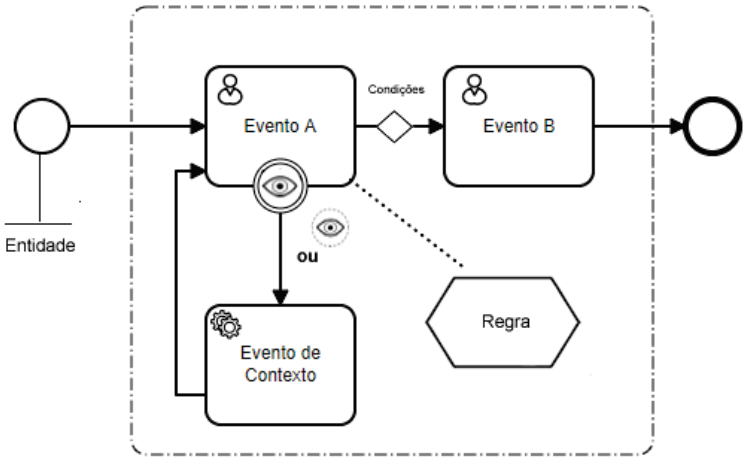


Figura 7. Modelo dinâmico genérico para a notação Context4BPMN.

É possível especificar e instanciar de forma segmentada os fluxos da aplicação, ou seja, para cada processo da aplicação, o modelo é instanciado de forma individual com seu fluxo e condições. Dessa forma, uma ou mais entidades contextuais são responsáveis por um processo, enquanto os elementos contextuais aparecem nas regras e nos eventos, uma vez que a fonte de dados contextuais torna-se opcional ser descrita nesse modelo e notação.

Para exemplificar o uso desta notação, utilizou-se o mesmo exemplo citado no modelo estrutural aplicando os conceitos como mostra a Figura 8.

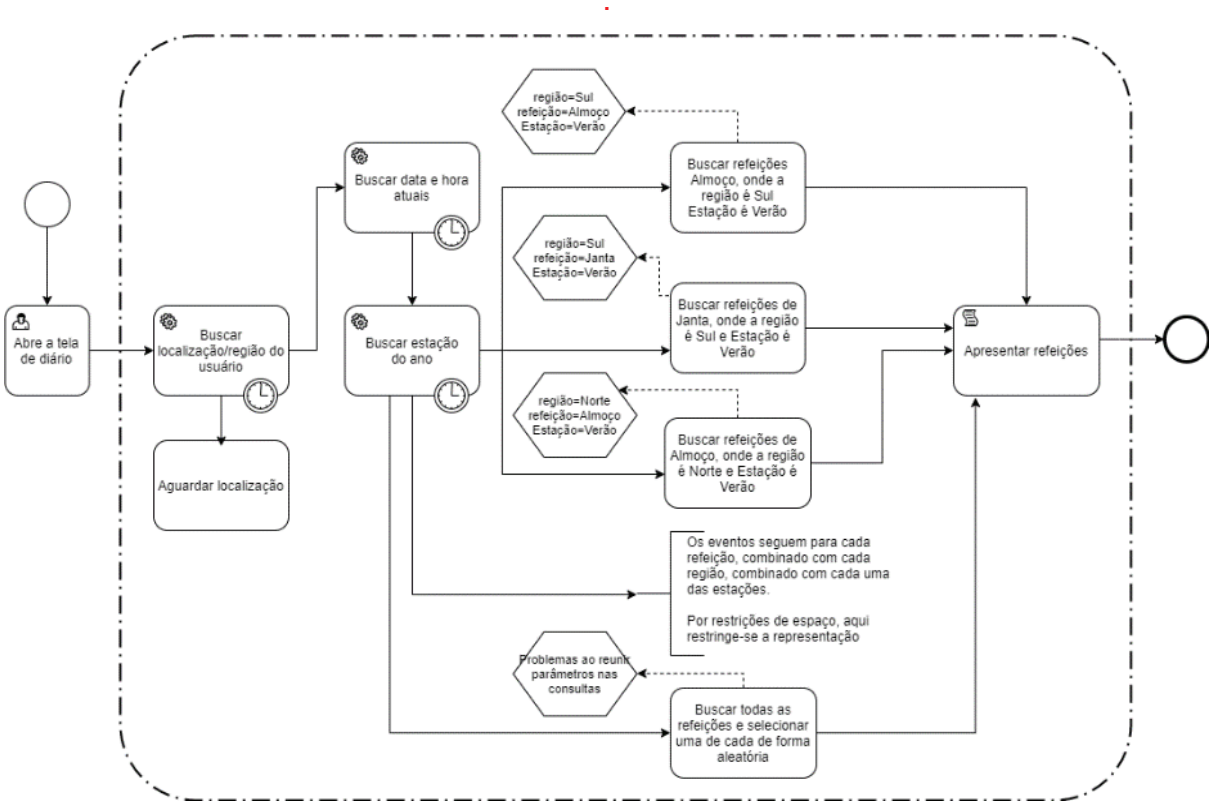


Figura 8. Exemplo de modelo dinâmico utilizando a notação Context4BPMN.

No exemplo é possível perceber os eventos de sistema responsáveis por buscar as informações de localização (região), data e hora (refeição) e estação do ano. Esses três eventos formam os parâmetros necessários para, posteriormente, ser possível disparar um novo evento, o de consulta às refeições que se enquadram nesses parâmetros de contexto, de acordo com a classificação da Figura 6. Para essa etapa, existem inúmeras possibilidades, de forma fatorial pois, para cada região, haverá a combinação de todas as estações do ano, para as três refeições, e assim sucessivamente. Para fins de viabilidade do modelo de representação, foi

adicionado um comentário após apresentar três possibilidades de eventos como exemplo.

3.2.3. Uma arquitetura para sistemas sensíveis ao contexto

A seguir é descrita a proposta de uma arquitetura para o desenvolvimento de aplicações móveis sensíveis ao contexto. Os principais objetivos da arquitetura são a escalabilidade, descentralização do processamento, a compatibilidade para implementação em diversas plataformas, dispositivos e tecnologias, bem como boas práticas e soluções para os desafios em desenvolver aplicações desse tipo.

O diferencial dessa arquitetura é detalhar não somente questões de projeto como de implementação, tecnologias nos atuais paradigmas de desenvolvimento e problemas os quais podem ser previstos e trabalhados previamente na implementação.

3.2.3.1. Requisitos e aspectos técnicos

Uma arquitetura para aplicações móveis sensíveis ao contexto deve prover componentes do sistema, bem como as relações entre eles e possíveis elementos externos. As principais plataformas de desenvolvimento atuais (iOS, Android e Microsoft) fornecem suas próprias arquiteturas e documentação específicas [66] [67] [68]. Apesar de cada uma delas ter suas especificidades, em comum abordam o padrão MVC (*Model-View Controller*) e algumas variações do mesmo. Essas diretrizes focam no desenvolvimento de aplicações cliente (local), entretanto, normalmente na prática, essa arquitetura torna-se necessária ser estendida com uma visão distribuída entre cliente e servidor.

Entretanto, uma arquitetura para aplicações desse tipo vai além de uma simples arquitetura cliente/servidor. Deve-se considerar elementos de software para obter, processar e disseminar dados e informações providas por sensores, sejam eles internos ao dispositivo, ou externos, os quais comunicam-se com a aplicação. Além disso, deve abstrair especificidades como quais tecnologias de desenvolvimento serão usadas, plataformas, variedade de dispositivos e de recursos desses dispositivos. Aspectos técnicos como a disponibilidade e largura de banda, energia finita dos dispositivos, impactam diretamente na arquitetura, pois a mesma

deve prever soluções para esses aspectos, como a separação de responsabilidades, distribuição de processamento e armazenamento.

Outro aspecto de grande importância em uma arquitetura desse porte é a computação na nuvem, termo que representa “um paradigma de computação econômica e de negócios em larga escala, com a virtualização como sua principal tecnologia” [69]. A virtualização traduz-se em um conjunto de serviços virtuais como hardware, infraestrutura, plataforma, software e armazenamento para diferentes aplicações, além de processamento paralelo, distribuído e computação em grade na Internet. A computação na nuvem tornou-se um forte paradigma de desenvolvimento para aplicações móveis nos últimos anos, devido à possibilidade de disponibilizar processamento, armazenamento e outros serviços para aplicativos móveis, minimizando esse trabalho diretamente no dispositivo cliente dos usuários. Em consequência disso, outro fator técnico a se considerar em uma arquitetura que utilize a computação na nuvem é o volume de dados gerados e armazenados por aplicações móveis na nuvem a todo instante. Ao considerar aplicações que obtém dados de sensores, como as sensíveis ao contexto, esse volume pode ser ainda maior. Demais aspectos técnicos, a saber:

- interoperabilidade técnica: visando flexibilidade na resolução de problemas, é preciso considerar o uso de serviços ou até mesmo sistemas terceiros em uma arquitetura. Para isso, o sistema deve oferecer um padrão de comunicação via interfaces tipo API’s e linguagens comuns.
- compatibilidade: considerando as inúmeras tecnologias e frameworks distintos para o desenvolvimento de software móvel, uma arquitetura precisa definir e descrever padrões que possam ser usados pela maior quantidade possível dessas tecnologias e sistemas operacionais, inclusive simultaneamente. Também refere-se a compatibilidade entre os mais variados modelos de dispositivos móveis disponíveis.
- disponibilidade: as informações e dados coletados, processados e disseminados por sensores da aplicação precisam estar disponíveis conforme a necessidade da mesma. Em algumas situações, pode tornar-se necessário utilizar sensores externos ao dispositivo, comunicando diretamente com a aplicação, seja no cliente ou servidor, buscando garantir a disponibilidade maior possível.

- conectividade: independente de largura de banda, disponibilidade de conexão com a internet ou qualquer outro problema relacionado, devem ser previstos pela aplicação, que precisa possuir formas de lidar com esse requisito atual, o qual apresenta-se como um dos grandes desafios no desenvolvimento móvel atual. Tratando-se de aplicações totalmente disponíveis, torna-se ainda mais importante e desafiador.
- modularidade e processamento distribuído: um dos itens essenciais de uma arquitetura deve ser o tratamento do processamento dos dados de sensores, assim como o volume desses dados e a incerteza da capacidade de processamento dos inúmeros dispositivos distintos. Quanto menos processamento local houver, melhor. Entretanto, em alguns casos pode ser necessário um pré-processamento local no dispositivo. Para isso, pode-se considerar o processamento na nuvem e no dispositivo cliente, prevendo de forma modular a arquitetura e separando claramente as responsabilidades, bem como implementá-las.
- descoberta de recursos: refere-se à incerteza de que um recurso esteja disponível no dispositivo cliente. Esses dispositivos podem ou não dispor de sensores necessários para captar os dados, bem como em alguns casos pode haver falha nesses sensores dependendo da qualidade de hardware. Isso acarreta a necessidade de prever alternativas para a adaptação de uma aplicação conforme a disponibilidade desses recursos.
- energia finita: dispositivos móveis possuem energia finita, e isso deve ser considerado tanto na distribuição de processamento quanto em outros itens citados anteriormente, como a disponibilidade de recursos. Há casos onde o uso de sensores continuamente pode consumir consideravelmente a energia dos dispositivos e consequentemente tornar inviável o uso de uma funcionalidade da aplicação. Exemplo disso é o monitoramento de localização via GPS, em que o uso contínuo, mesmo com a aplicação em segundo plano, pode consumir a energia do dispositivo rapidamente. A arquitetura deve prever e oferecer soluções para contornar esse problema.

Os itens citados acima descrevem as principais preocupações a serem abordadas pela arquitetura proposta neste trabalho. Contudo, nem sempre será necessário ou viável que todos esses itens sejam contemplados em sua plenitude por uma aplicação, sendo o objetivo deste modelo oferecer soluções por meio de

uma representação sistemática e semântica da arquitetura para um sistema sensível ao contexto.

3.2.3.2. Proposta de arquitetura baseada em estrutura na nuvem

Considerando os itens anteriores, foi proposta uma arquitetura genérica. O paradigma abordado apresenta uma estrutura cliente/servidor baseada em serviços na nuvem. Os principais diferenciais dessa estrutura são a escalabilidade, flexibilidade e alta compatibilidade, pois podem ser implementadas por diferentes linguagens e tecnologias. Além disso, as necessidades de cada aplicação individualmente podem ser adaptadas, sendo possível utilizar em sua completude ou parcialmente, com diferentes formas de implementar a mesma arquitetura. A seguir são detalhados os elementos que compõe a arquitetura. Uma visão geral da estrutura é apresentada na Figura 9.

A estrutura é dividida em dois módulos: Servidor e cliente.

- Servidor: Representa a estrutura de serviços disponível na nuvem para a aplicação. Os elementos contidos nesse módulo são o banco de dados, representado por um cilindro, o módulo de processamento e serviços, representado por um hexágono, os serviços oriundos do módulo anterior, representados por retângulos e o elemento adicional denominado HTTP que representa a interface de entrada das requisições.
- Cliente: Representa a aplicação e os recursos do seu dispositivo ou externos, integrados com a aplicação. O contêiner que envolve a maioria dos elementos representa a aplicação e os recursos utilizados pelo dispositivo, representados por retângulos arredondados. Os sensores internos do dispositivo estão posicionados dentro desse contêiner, enquanto os sensores externos estão posicionados na parte inferior, externo. Descritivamente estão enumerados, entretanto essa representação é meramente ilustrativa para indicar a possibilidade de um ou mais sensores no modelo. Exemplos de sensores internos: Acelerômetro, Giroscópio, Magnetômetro, GPS, sensor de proximidade, luminosidade, dentre outros. Incluem-se no termo sensores internos qualquer dado ou informação obtida localmente no dispositivo, como por exemplo o horário, preferências, histórico de navegação, dentre outras. Já os sensores externos são todos aqueles dados ou informações obtidas por

hardware ou software externos ao dispositivo, sejam eles servidores na nuvem, bancos de dados (cilindro externo), serviços e API's terceiros, bem como relógios inteligentes, monitores de sinais vitais, dentre outros dispositivos físicos que possam comunicar-se com a arquitetura. Por fim, a ação de apresentação e adaptação resultantes são representados pela forma de fitas perfuradas.

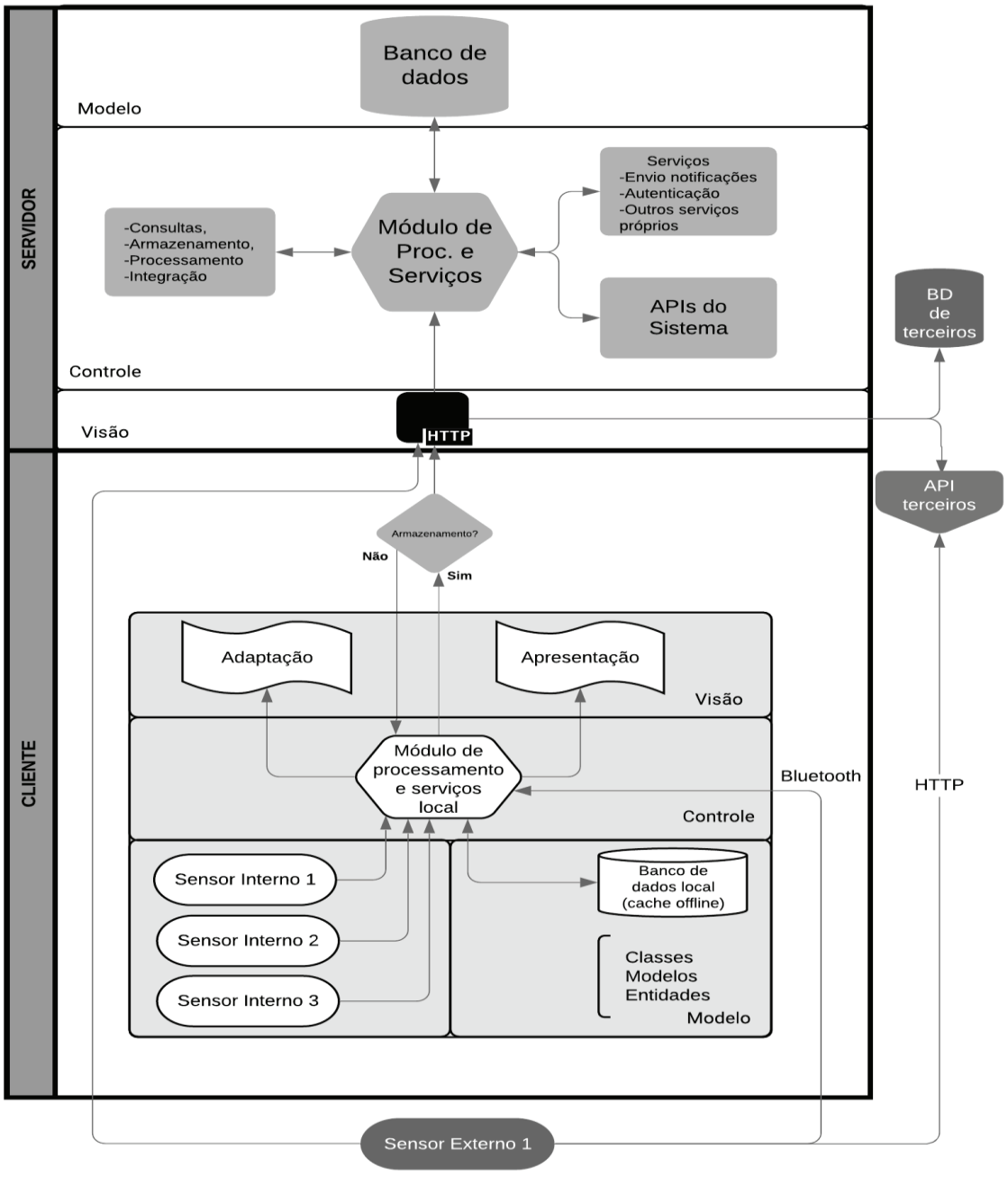


Figura 9. Visão geral da arquitetura.

3.2.3.3. Detalhamento dos elementos da estrutura

Nessa seção serão detalhados os elementos da estrutura e como funciona a sistemática para implementação dos mesmos na arquitetura proposta. As três atividades macro de uma estrutura sensível ao contexto, como já citados anteriormente, são a obtenção, o processamento e a disseminação das informações de contexto do sistema. Considerando essa ordem, a primeira atividade abordada é a de obtenção dos dados de contexto.

Para obter dados e/ou informações de contexto na aplicação cliente, ou seja, na aplicação em um dispositivo móvel, os elementos da arquitetura envolvidos são os sensores, representados por retângulos arredondados na arquitetura. Eles podem ser sensores externos ou internos:

- Sensores internos: utilizam o hardware e software do dispositivo. Sensores de hardware podem ser o acelerômetro, giroscópio, magnetômetro, GPS, sensor de proximidade, luminosidade, dentre outros conforme os modelos de smartphones. As fontes de software internas podem ser informações do sistema operacional do dispositivo, bem como de outras aplicações presentes, como por exemplo data e hora, versão do sistema operacional, configurações de preferências, histórico de navegação, dentre outros. Aplicações como o Google Chrome fornecem Informações sobre sua navegação e outras atividades em sites, aplicativos e dispositivos que usam os serviços do Google, sendo um exemplo de fonte de obtenção de dados via aplicações no sistema operacional.
- Sensores externos: utilizam tanto hardware quanto software externos ao dispositivo como fonte para obtenção de dados. Sensores físicos externos compreende-se por todo e qualquer dispositivo de hardware que obtenha informações e possa comunicar-se com o dispositivo ou com o servidor. Muitos dos sensores internos ao smartphone também são oferecidos como dispositivos externos, como por exemplo o GPS. Outros dispositivos como relógios inteligentes, sensores de monitoramento cardíaco ou até mesmo aparelhos como drones e dispositivos vestíveis. Apesar de não ser o foco deste trabalho, é importante citar o termo Internet das Coisas, conceito que se refere à interconexão digital de objetos cotidianos

com a internet, e este torna-se pertinente quando fala-se em sensores externos, afinal qualquer objeto que possua essa interconexão e forneça dados pode ser utilizado na arquitetura. Os sensores externos via software compreendem toda e qualquer informação adquirida por meio de sistemas, serviços ou repositórios de terceiros (representado por um cilindro na arquitetura lado cliente), normalmente da nuvem. Exemplos disso são a obtenção de dados climáticos, temperatura em qualquer região do mundo, notícias, bancos de dados de alimentos, medicamentos, dados geossociais, históricos, ou qualquer outro repositório. Um exemplo de destaque são os serviços de monitoramento geográfico na nuvem, onde uma aplicação no servidor comunica-se com o GPS de um dispositivo local e, obtendo as coordenadas, efetua o processamento das mesmas, devolvendo informações como: usuário está em casa, entrou em uma cerca geográfica demarcada, está em movimento, está no trabalho, o trânsito em volta e suas características, dentre outros. Nesse caso, apesar de haver uma obtenção local por meio do sensor interno de hardware, a base de conhecimento está em um repositório na nuvem, como um sensor externo de software.

- Bases de dados: outra fonte de dados de contexto são as bases de dados próprias, ou seja, desenvolvidas para o sistema em questão. Essas bases podem ser locais ou na nuvem, dependendo das necessidades identificadas, sendo que na arquitetura, ambos estão representados pela forma de cilindros. A base local do dispositivo está descrita como “cache off-line”, de modo a considerar toda e qualquer base de dados local no dispositivo, seja uma memória temporária, quanto para o armazenamento permanente de dados. Por outro lado, o cilindro representando a base na nuvem significa utilizar-se do armazenamento de dados permanentes em um banco de dados no servidor, sendo necessária a disponibilidade de conexão para sincronização desses dados.

Para o processamento dos dados obtidos, um módulo chamado de Serviços e Processamento, representado por um hexágono no cliente e na nuvem, é proposto na arquitetura. Sua função é transformar os dados obtidos por sensores em informações, ou até mesmo processar conjuntos de informações as quais formem um contexto da informação. Ao mesmo tempo, o módulo em questão está presente

tanto do lado cliente quanto servidor, pois a arquitetura proposta considera o processamento local como uma forma de divisão de responsabilidades e pré-processamento, ou seja, descentralização do processamento.

Outra justificativa é a incerteza na disponibilidade de conexão, o que torna essencial o processamento local, este a ser avaliado para cada funcionalidade. Nessa configuração, o processamento dos dados de contexto pode ser feito local ou no servidor, mas também parcialmente nos dois lados, tornando a arquitetura o mais flexível e adaptável possível. Os módulos são:

1. Módulo de Serviços e Processamento (servidor): compreende o conjunto de serviços da aplicação hospedados no lado servidor. O módulo de Serviços e Processamento deve implementar o armazenamento e consultas de dados, webservices, rotinas de processamento, inferência de dados, inteligência artificial, bibliotecas, envios de notificação, interfaces e integrações com outras API's e serviços terceiros, bem como todo e qualquer serviço necessário para a aplicação. O objetivo do módulo é utilizar-se da computação na nuvem e suas vantagens como o desempenho, escalabilidade, segurança e confiabilidade, por meio das possibilidades que a tecnologia oferece.

   Para efetuar o processamento dos dados de contexto, pode-se tornar necessário executar diversos serviços, como por exemplo: uma consulta ao banco de dados, seguida de uma consulta a um serviço terceiro, a utilização de uma biblioteca, um processamento com motores de inferência, inteligência artificial, dentre outros. Todos estes podem estar sequenciados em um mesmo processamento e interpretação de contexto, por isso o módulo é descrito como o conjunto desses serviços que trabalham em sincronia na estrutura do servidor. Os serviços devem ser implementados mantendo princípios de reutilização de código, ou seja, funções claras e separadas cada qual com sua função específica e que possam ser utilizadas por qualquer outro módulo do sistema ou até mesmo de outros sistemas, mantendo a interoperabilidade. Detalhes da implementação e sistemática estarão descritos nas próximas seções.
2. Módulo de Serviços e Processamento (local): é necessário para o processamento onde a incerteza da disponibilidade de conexão é um fator chave, bem como o desempenho do processamento dos dados e sua

complexidade são considerados. Nem todas funcionalidades ou dados de contexto precisarão de um processamento complexo, armazenamento ou até mesmo envio para a nuvem. Nesses casos, o módulo local é uma forma de descentralização e (ou) pré-processamento, mas também um fator essencial para o desempenho da aplicação. A sua implementação deve considerar três questões principais: o processamento desses dados é viável e adequado sendo no dispositivo? É necessário o uso do servidor para esse processamento? É necessário armazenar definitivamente esses dados e recuperá-los posteriormente? Essas questões ajudam a elucidar quanto do processamento deve ficar no módulo local. Por exemplo, para identificar se um usuário está conectado à rede, os sensores retornam um valor binário onde não há necessidade de processamento, e dependendo da aplicação, também não precisa armazenado.

Em resumo, dados de contexto adquiridos por sensores internos ou externos muitas vezes são pontuais e não necessitam processamento complexos. Em determinados casos, torna-se um desperdício consultar parâmetros na nuvem quando a informação está pré-formatada ou totalmente pronta. Para casos onde o processamento necessita do servidor, é de responsabilidade do módulo local apenas enfileirar requisições, tratar possíveis erros e interpretar os resultados. Outras funções do módulo são: implementar o armazenamento e sincronização dos dados temporários, em cache, para uso na aplicação, assim como delegar as ações de adaptação e apresentação.

O módulo local possui limitações em relação ao módulo da nuvem, isso porque determinadas funções e serviços dependem da comunicação com o servidor como o envio de notificações, e-mails, dentre outros. Outra limitação a ser considerada é o fato de precisar ser implementada em uma linguagem específica para dispositivos móveis, o que dificulta manter a interoperabilidade.

Os elementos de adaptação e apresentação (forma de fitas perfuradas) representam a implementação das ações resultantes do módulo de processamento e serviços, recomendando-se que sigam o mesmo conceito dos módulos, ou seja, a

reutilização sempre que possível de códigos, bem como a separação total da funcionalidade em relação ao processamento dos dados.

O último elemento da arquitetura a ser descrito são as API's de terceiros, onde não é indicada nenhuma implementação, mas sim um elemento que represente a utilização, por meio de interfaces, de serviços terceiros pela aplicação tanto no lado servidor quanto do lado cliente. O elemento está posicionado no lado servidor, pois essas interfaces estarão necessariamente hospedadas em servidores de seus proprietários, e por isso será necessária conexão com a internet para trabalhar em conjunto com as mesmas.

3.2.4. Implementação da estrutura

Após detalhados os elementos, será descrito a forma como eles interagem entre si, bem como as diretrizes para implementação em uma forma sistemática. Este passo a passo será divido em aquisição dos dados, para onde devem ir estes dados e processamento dos mesmos por meio dos módulos detalhados.

1. **Aquisição de dados de sensores;**
   - Internos: Utilizando-se das API's ou Plugins de cada tecnologia, implementar a comunicação da aplicação com os sensores do dispositivo. Dados de sensores internos necessariamente passam pelo módulo de processamento local antes de serem enviados ao módulo na nuvem.
   - Externos: Utilizando API's do fabricante do sensor externo, é possível comunicar com o módulo de processamento local ou diretamente com o módulo de processamento no servidor, ambos via protocolos web comuns desses dispositivos. Essa diretriz de relacionamento justifica-se pois ao utilizar-se de dispositivos externos para obtenção de dados de contexto, pode-se tornar necessário o envio diretamente para a nuvem, sem nenhum processamento prévio, seja pelo volume de dados, frequência da obtenção desses dados ou até mesmo a complexidade do processamento.

A definição de **quando** se deve optar por sensores internos ou externos deve ser analisada com as seguintes questões:

- A frequência a qual será preciso requisitar um sensor para obter dados é esporádica ou a todo instante?

Sensores que precisam ser requisitados a todo instante como monitoramento cardíaco, contador de passos ou movimento, dentre outros, normalmente consomem energia do dispositivo em grande quantidade. Por isso, uma alternativa é o uso de dispositivos externos como relógios inteligentes, GPS, cintas de monitoramento cardíaco, dentre outros os quais possuem sua própria energia e conectividade com várias plataformas por meio do protocolo padrão de comunicação Bluetooth.

- Os sensores internos dos dispositivos e público alvo possuem a precisão suficiente para obter os dados necessários?

Aqui o foco é na precisão dos sensores, onde, caso os dados a serem obtidos sejam críticos e necessitem maior precisão, a alternativa prefereincial também deverá ser o uso de dispositivos externos de hardware específicos para o fim desejado.

Ainda que a arquitetura e as diretrizes permitam e indiquem a flexibilidade, em alguns casos pode ser inviável o uso de alguma das possibilidades. Nos casos onde será necessário o uso de sensores internos do dispositivo para monitorar dados de contexto o tempo todo, indica-se o planejamento de uma modelagem de energia para a aplicação, tópico discutido e apresentado na literatura por Mizouni *et al*. [70], em que discutem como minimizar e adaptar recursos da aplicação de acordo com a energia disponível no dispositivo móvel, estendendo e sincronizando com o lado servidor.

2. Envio de dados de sensores ao módulo de processamento;

A definição de quando deve-se enviar dados ao módulo local, ou ao módulo servidor, deve ser analisada de acordo com as seguintes questões:

- Qual o volume de dados e a frequência de obtenção dos mesmos?
- É importante o processamento desses dados localmente?
- É preciso armazenar o histórico desses dados?

Os módulos de serviço e processamento são o intermédio entre os sensores e as ações de adaptação e apresentação que a aplicação deve realizar. Sendo assim, as questões apresentadas acima ajudam a elucidar qual o melhor caminho a seguir na hora de projetar uma funcionalidade. Por exemplo, quando o volume de dados e a frequência em que esses dados fluem dos sensores para o módulo é alta, a indicação é retirar a responsabilidade desse controle do dispositivo local, o qual possui capacidade e desempenho muito inferiores à de um servidor. A segunda

questão endossa a primeira, entretanto foca na conectividade, afinal em alguns casos, mesmo que o volume e frequencia dos dados seja alta, a aplicação precisa responder de forma rápida em situações críticas onde não deve ser dependente de conexão com o servidor.

Finalmente a terceira questão apresentada refere-se a tomada de decisão em relação ao armazenamento e histórico dos dados dos sensores, na base de dados no servidor. Podem haver casos onde será preciso armazenar, processar e recuperar posteriormente esses dados, ou após a obtenção dos dados de sensores, os mesmos sejam processados e descartados. Um exemplo prático para as questões acima são aplicações de monitoramento de localização, onde ao entrar em uma cerca geográfica demarcada previamente, notificam seus usuários. Esse é um exemplo de aplicação onde o volume e a frequência dos dados é alta, pois o monitoramento é contínuo e os parâmetros de localização devem estar mantidos no servidor, portanto o mais indicado seria o envio dos dados dos sensores diretamente para o módulo de serviços e processamento no servidor. Em contraponto, se uma aplicação avisa seus usuários de possíveis riscos ao atingir níveis de batimentos cardíacos, a partir do monitoramento com sensores internos ou externos, depender da conectividade pode não ser a melhor alternativa, e sendo assim, processar esses dados localmente independentemente do armazenamento de histórico ou sincronização com o servidor, também é possível de acordo com a arquitetura. A Tabela 1 apresenta um esquema de resumo dessas questões.

3. Implementação do módulo de serviços e processamento;

- Módulo Local: de acordo com a linguagem escolhida para o desenvolvimento, conceitos como provedores, serviços, controladores ou módulos podem ser utilizados para implementação. A reutilização de funções é essencial para otimização e desempenho do código, o qual terá o objetivo de implementar algoritmos para processamento, direcionar e sincronizar o armazenamento de dados, enfileirar e sincronizar requisições, bem como tratar e processar os resultados das requisições as quais serão direcionadas para as ações de adaptação ou apresentação.
- Módulo Servidor: linguagem única. A estrutura do servidor permite mais possibilidades de integrações e processamento. O conceito a ser utilizado é o de serviços, onde implementam-se funções com única responsabilidade e que possam

ser requisitadas por outras funções do servidor, ou até mesmo de terceiros. Além de tratar do armazenamento e manutenção dos dados, algoritmos de processamento em rotinas, webservices e respostas para a aplicação cliente, o módulo de serviços e processamento na nuvem deve permitir incorporar demais possibilidades como motores de inferência, inteligência artificial, envio de notificações, emails, autenticação segura, consultas à bases de terceiros, integração com outras API's, dentre outros.

Em resumo, o módulo de serviços e processamento na nuvem deve ser uma implementação central orientada à serviços, no servidor. Esses serviços devem:

- manter e recuperar dados na base;
- prover rotinas e funções úteis para processar e disseminar as informações de contexto;
- permitir a integrações com API's, serviços e sistemas terceiros.

Por outro lado, o módulo local funciona como um pré-processamento dos dados de sensores, em que deve implementar rotinas pontuais ou críticas, as quais não dependam de conectividade, armazenamento ou desempenho. Para auxiliar o módulo local, pode-se utilizar de bases de dados embarcadas no dispositivo, tanto para memória temporária, quanto definitiva (e/ou posteriormente sincronizada com a nuvem). O módulo local também deve implementar o retorno das informações para os elementos de adaptação e apresentação, onde irão conter somente a implementação da camada de visão, sendo uma adaptação e/ou apresentação de dados conforme o contexto processado.

4. RESULTADOS

A avaliação do framework proposto foi realizada de três formas: (i) avaliação de especialistas da área de software (ii) a partir do seu instanciamento em uma aplicação com vistas a verificar sua assertividade (modelos, arquitetura e diretrizes); e (iii) a partir da avaliação de profissionais da área de nutrição, para verificar se a aplicação desenvolvida a partir do framework efetuava corretamente as adaptações e apresentações sensíveis ao contexto.

4.1. AVALIAÇÃO DO FRAMEWORK POR ESPECIALISTAS DE SOFTWARE

Nesta seção serão descritas as etapas realizadas para a avaliação da aplicação com profissionais da área do desenvolvimento de software, que teve como objetivo verificar se a proposta de framework é viável, bem como o nível de entendimento e feedback. Para tanto, a avaliação se concentrou em obter a opinião de três Analistas de Sistema por meio da leitura do Framework e uma atividade de modelagem utilizando os conceitos deste trabalho. Após um período de leitura e modelagem, os participantes foram convidados a responderem a um questionário online para coleta de dados.

4.1.1. Elaboração da avaliação

Para coleta de dados dos participantes foi utilizado um exemplo de caso de uso (APENDICE A)., pensado a partir de funcionalidades sensíveis ao contexto, onde os participantes deveriam aplicar os conceitos do framework, desenvolvendo uma modelagem do sistema de exemplo. Além disso, após concluída a modelagem, foi enviado um questionário online para responderem (APENDICE B). O objetivo foi formular questões descritivas para obter a opinião dos profissionais em relação ao entendimento e utilização do framework.

Em um primeiro momento foi enviado o texto de descrição e proposta do framework FAM4CSS para que os participantes pudessem ler. Somado a isso, o caso de uso para formular uma modelagem de sistema sensível ao contexto. Os

participantes tiveram 15 dias para ler e elaborar a modelagem do caso de uso. Após isso, foi enviado o questionário para feedback.

O questionário foi dividido em três questões descritivas, sendo as dificuldades encontradas para utilizar o framework, as facilidades e por fim, sugestões em relação ao mesmo. O instrumento foi desenvolvido na plataforma de Formulários Google.

4.1.2. Procedimento de coleta de dados

O estudo foi realizado com 3 participantes, selecionados por amostragem não probabilística - conveniência. Os objetivos da pesquisa foram esclarecidos previamente em reuniões online via comunicadores de mensagem instantânea. Todos os participantes possuem cargo de Analistas de Sistemas e trabalham com modelagem de sistemas na sua rotina.

O estudo foi realizado nos meses de abril e maio de 2020, sendo que cada participante deveria efetuar a leitura do framework, em seguida modelando um caso de uso de software sensível ao contexto e por fim respondendo o questionário de feedback.

4.1.3. Análise das respostas

O tempo de profissão dos participantes foi de 7, 9 e 10 anos. Em relação aos pontos positivos do framework, em comum, todos eles citaram o fácil entendimento da proposta para modelar e projetar. Para isso, dois deles citaram que a simplificação dos termos e representações gráficas ajudaram para esse entendimento, enquanto outro citou a facilidade em representar ações e eventos em um modelo estático como essencial para modelar esse tipo de aplicações. Outro ponto comum entre todos participantes foi a citação de que projetar sistemas sensíveis ao contexto pode se tornar complexo se não houver uma sistemática teórica, porém apoiada por uma arquitetura que permita pensar os elementos do projeto previamente: “A arquitetura proposta é de boa valia para analistas e desenvolvedores por trazer um modelo base de arquitetura de software. Nem sempre a preparação de todo ambiente é pensada antes do desenvolvimento, com soluções sendo construídas conforme necessidades surgem, o que não é o ideal.

Um modelo arquitetônico torna mais fácil o planejamento de tecnologias e de que forma elas irão interagir. ”

Um dos usuários destacou também que “O uso de diagramas e fluxos pode ser aplicado em testes de aceitação, servir de apoio para setor de suporte e até mesmo para o preparo de um manual ao usuário final”. Outro participante destacou a abrangência da proposta: “A abrangência deste modelo também me chamou atenção podendo ser aplicado com algumas adaptações outros tipos de domínios”. Por fim, um dos usuários destacou que a utilização de os modelos gráficos escolhidos e as nomenclaturas dos elementos foram compreendidas por serem de uso comum no dia a dia do desenvolvimento de sistemas tradicionais os quais trabalham.

Sobre os pontos de dificuldade e sugestões de melhoria, dois usuários questionaram o uso de dados em modelos instanciados UML. Para eles poderia haver um modelo intermediário de instanciamento em nível lógico, como um Diagrama de classes comum, reaproveitando classes e objetos. O terceiro participante citou uma melhoria que poderia ser feita no Modelo de dados em relação às setas que ligam as entidades: “Na Figura 3, do modelo estrutural, as flechas indicam que um Evento pode possuir Regra, que por sua vez pode conter uma Priorização que irá disparar uma Ação. Isso pode gerar o entendimento de que só será possível haver uma Ação caso exista uma Priorização e uma Regra”. Além disso, citaram pequenos ajustes pontuais em relação a arquitetura e tecnologias, todos estes importantes para a evolução deste trabalho posteriormente.

4.2. INSTANCIANDO O FRAMEWORK EM UMA APLICAÇÃO PARA CUIDADOS DE SAÚDE ALIMENTAR

Nesta seção é descrita a modelagem e desenvolvimento de uma aplicação real, utilizando os modelos propostos por este trabalho, instanciando-os e exemplificando seu uso por meio de funcionalidades, tecnologias e diretrizes que buscam detalhar e verificar a assertividade do framework proposto. Inicialmente será apresentado o tema abordado, os requisitos levantados, e em seguida, aplicados os modelos, a arquitetura e implementação.

4.2.1. Domínio da aplicação escolhida

Em parceria com o Programa de Pós-Graduação em Envelhecimento Humano da Universidade de Passo Fundo (PPGEH), o tema escolhido foi a promoção da saúde perante a sociedade brasileira, por meio de um conjunto de informações e recomendações sobre alimentação, utilizando-se das diretrizes de um documento chamado Guia Alimentar para a População Brasileira.

Pensando em promover a alimentação saudável, com vistas à segurança alimentar e educação nutricional dos brasileiros, o Ministério da Saúde elaborou o Guia com vocabulário claro, objetivo e simples, visando a fácil compreensão da população geral. Leva em conta não apenas a ingestão de nutrientes, mas também outros aspectos como a comensalidade, cultura, fatores socioeconômicos, produção/distribuição dos alimentos e a sustentabilidade [71]. Todavia, esse documento não apenas extenso, é de pouca acessibilidade à população, tornando-se um potencial instrumento de uso nesse trabalho, aplicando o conceito de sistemas sensíveis ao contexto e disponibilizando informações do Guia de forma personalizada, considerando fatores temporais, sazonais e de localidade.

4.2.2. Requisitos para a aplicação escolhida

Considerando as diretrizes do Guia Alimentar, o qual propõe alguns passos para uma alimentação saudável, foram elencados requisitos que pudessem abordar esses itens em formato de funcionalidades, com o objetivo de desenvolver um aplicativo para dispositivos móveis multiplataforma. Abaixo são listadas as diretrizes abordadas pelo guia, e em seguida, os requisitos do sistema para cada item:

- Promover escolhas alimentares;
    - Desenvolver diário de sugestões de refeições;
    - Desenvolver quizes e tutoriais;
- Promover senso crítico;
    - Desenvolver feed para notícias, artigos e vídeos;
    - Permitir compartilhamento em redes sociais/email;
- Sugerir alimentação regular e ambiente adequado para refeições;
    - Enviar notificações com informações e sugestões;
- Sustentabilidade e comensalidade;
    - Disponibilizar mapa de feiras orgânicas e receitas;

- Planejamento e organização;
    - Enviar lembretes periodicamente;
    - Permitir configuração de novos lembretes.

Visando aumentar o engajamento e assertividade da aplicação, outros itens não referentes ao Guia foram adicionados, como seguem para conhecimento:

- Avaliação e pontuação;
- Permitir questionário de avaliação conforme o Guia;
- Exibir *Dashboard* de evolução para pontuações;
- Considerar informações de contexto para as funcionalidades.

O último item de requisitos refere-se a análise dos requisitos e onde poderia utilizar-se de informações de contexto para a aplicação, a qual poderia gerar adaptações, apresentações e interações personalizadas para seus usuários. Muitas possibilidades encontram-se disponíveis quando cada funcionalidade é pensada individualmente em relação às tecnologias e sensores disponíveis, entretanto, para fins de exemplificar o framework em questão, buscou-se variar situações que pudessem explorar os modelos e arquitetura. Sendo assim, três funcionalidades principais foram destacadas, a saber: Um diário de refeições, onde o conteúdo é sugerido para os usuários de acordo com o horário (café da manhã, almoço, lanches, janta), com a região onde o usuário encontra-se no país (norte, sul, sudeste, nordeste, centro-oeste) e a estação do ano (inverno, verão, primavera, outono); um mapa de feiras orgânicas com cercas geográficas demarcadas, onde ao entrar em uma dessas cercas, próximo a alguma feira, o usuário deve ser notificado desse evento; e por fim a adaptação da aplicação conforme a disponibilidade atual de conexão com a internet.

4.2.3. Modelagem da aplicação

Para a modelagem dos itens destacados na seção anterior, foram instanciados os dois modelos propostos no framework, estrutural e dinâmico, conforme apresentado a Figura 10. No primeiro modelo o foco são as três funcionalidades principais, portanto, as demais entidades e elementos não relevantes para exemplificar os modelos contidos na aplicação, não estão representados na figura. No segundo exemplo, também para facilitar o

entendimento, o modelo dinâmico está focado apenas no processo da funcionalidade principal.

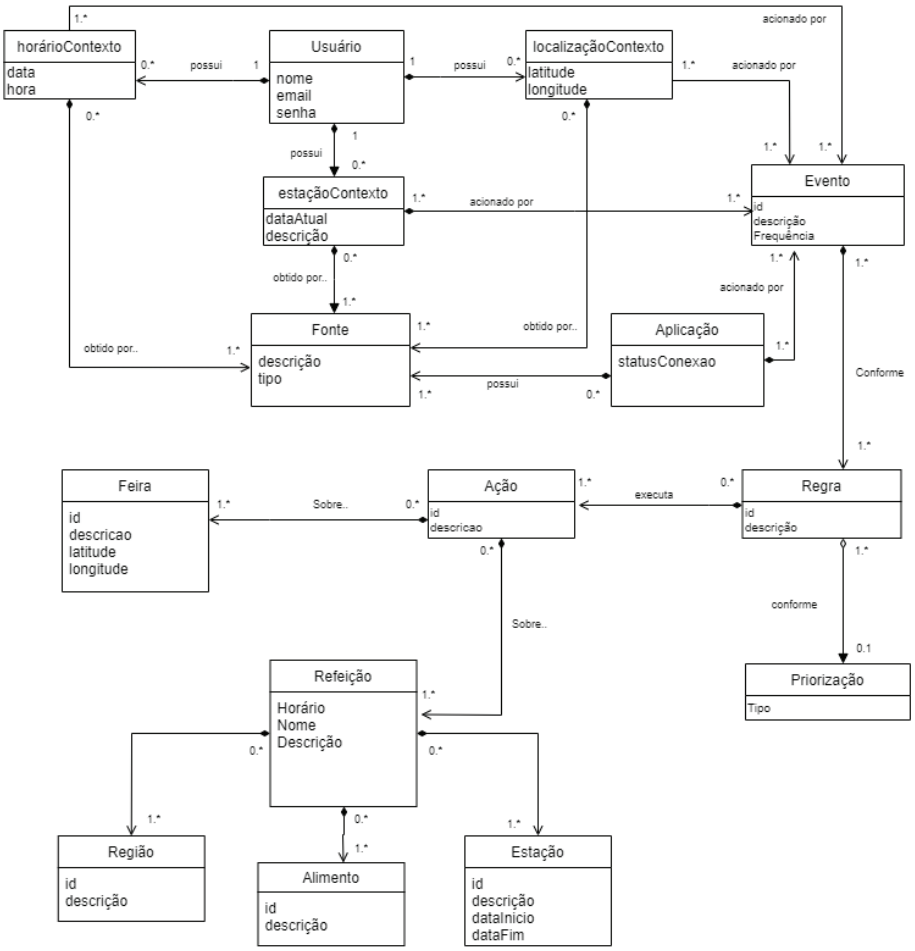


Figura 10. Modelagem de dados parcial da aplicação.

Primeiramente foram elencadas as entidades principais: Usuário e Aplicação. Posteriormente, elencados os seus elementos contextuais que serão obtidos de forma intrínseca, como localização, estação do ano e horário atual para o Usuário, enquanto a entidade Aplicação possuirá o status da conexão no momento. O relacionamento base entre esses elementos é de uma composição um para muitos, onde a entidade principal Usuário ou Aplicação, poderá ter muitas ocorrências dos atributos de contexto.

Os elementos de contexto estão sufixados com a palavra Contexto após seu nome para facilitar o entendimento e diferenciar das classes com o mesmo nome, as quais compõem outras entidades da aplicação. A relação entre esses elementos e os eventos é uma composição de muitos para muitos, ou seja, um

elemento contextual pode conter muitos eventos de aquisição, da mesma forma como um evento e frequência podem aparecer em mais de um elemento. As regras possuem a mesma relação com os eventos e poderão ou não, em uma relação de agregação, conter uma priorização para resolver ambiguidades. Por fim, uma ou muitas ações precisam estar associadas em forma de composição a pelo menos uma regra.

O modelo instanciado na Figura 10 pode ser implementado em qualquer banco de dados, seja ele relacional ou não, entretanto, a representação lógica do modelo de dados não compreende o armazenamento de dados, sendo o modelo apenas uma representação das entidades e seus relacionamentos, nesse caso com detalhes a mais de implementação como a cardinalidade dentro de um domínio. Para a aplicação desenvolvida, foi utilizado banco de dados baseados em documentos, porém detalhes da implementação e tecnologia serão apresentados posteriormente.

O modelo dinâmico dos processos da aplicação também foi criado, e um exemplo de processo instanciado para a aplicação foi apresentado na Figura 8 ainda no capítulo 3, seção 3.2.2. Nele é possível perceber como está disposta a estrutura dos dados e seus eventos de forma sistemática. O processo inicia quando o usuário manualmente acessa a tela de diário, em que serão carregadas sugestões de refeições conforme informações de contexto. A atividade está representada pelo ícone de uma pessoa, indicando ser manual. Em seguida, o primeiro evento automático (representado pelo ícone de uma engrenagem dentro da atividade) é a obtenção da localização do usuário, que é um evento de interrupção, representado pelo ícone de um relógio. Essa atividade deverá identificar qual região do país ele está no momento, por meio do processamento dos dados de latitude e longitude obtidos pelo GPS, especificados na modelagem estrutural.

Na sequência ao evento de obter localização, estão os eventos de obter a estação do ano e a data/horário atuais. Os três são eventos de interrupção, ou seja, dependentes do término do anterior. Se houver falha na obtenção dos dados de algum desses eventos, o hexágono apontado por uma linha pontilhada indica o tratamento do erro, seguindo um fluxo alternativo, em que a aplicação irá ignorar as informações de contexto e adaptar sua funcionalidade, trazendo a lista de sugestões de forma aleatória.

Caso os eventos de obtenção de dados de contexto sejam concluídos com sucesso, a aplicação segue seu fluxo para o evento de buscar refeições, conforme os parâmetros de contexto obtidos nos eventos anteriores (horário, estação e região), e subsequente ao evento de apresentar as informações filtradas na tela. No modelo instanciado foi adicionado o ícone de banco de dados, apontados por linhas pontilhadas, as quais significam o envio de mensagens, ambos para representar e facilitar o entendimento de que a atividade consistirá em uma consulta ao banco de dados, conforme os parâmetros encontrados nas atividades anteriores (hexágono associado).

O modelo dinâmico pode ser muito útil para auxiliar a implementação, pois ao observar separadamente seus eventos, é possível notar pequenas partes de código que precisarão ser implementadas e sequenciadas, principalmente as atividades onde há o ícone de uma engrenagem, que indica serviços que o sistema deverá conter. Além disso, ele complementa os conceitos do modelo estrutural de forma cíclica, ou seja, apresentando como os dados se comportam ao longo dos eventos descritos na modelagem de dados anterior.

4.2.4. Escolha das tecnologias envolvidas para implementação

A escolha das tecnologias adequadas para implementação da arquitetura da aplicação deve ser feita ponderando os fatores a seguir. O foco deste trabalho são aplicações móveis, portanto há duas frentes tecnológicas a serem escolhidas e integradas, uma para a aplicação no servidor e outra no dispositivo.

A arquitetura no lado servidor visa garantir flexibilidade, desempenho, modularidade, disponibilidade e compatibilidade. O desempenho significa receber, processar e disseminar informações vindas de requisições dos dispositivos cliente da forma rápida e eficiente. Interferem nisso os fatores de grandes quantidades de dados (Big Data), a escalabilidade para armazenamento e distribuição desses dados, bem como a alta concorrência nas inúmeras requisições ao servidor. Em um estudo recente, Lei *et al*. [72] analisaram e compararam o desempenho das principais tecnologias para o desenvolvimento em servidor, concluindo que o emergente Node.Js [73] possui um desempenho maior em relação aos tradicionais servidores Apache/PHP, Java e Python, pois trabalha bem com a alta concorrência de requisições, dados complexos e muitos usuários. O Node.Js é uma tecnologia de

desenvolvimento no servidor, assíncrona e orientada a eventos a qual trabalha com uma única *thread* de execução, ou seja, diferente das demais tecnologias tradicionais onde, para cada requisição, é criada uma *thread* de execução. Por assíncrona, significa que cada requisição não bloqueia o processo como um todo, atendendo a um volume alto de requisições simultâneas, mesmo sendo em *thread* única. Outro diferencial dessa tecnologia é ser orientada a eventos, onde a requisição apenas é repassada para um chamado “loop de eventos” não bloqueantes, ideal para a arquitetura do framework proposto neste trabalho.

Somado a isso, tecnologias “backend as a service” como Firebase [74] e Amazon AWS possuem suporte a Node.js e disponibilizam uma completa estrutura de armazenamento e serviços compatíveis com boa parte das tecnologias de desenvolvimento móvel do lado cliente. Isso significa utilizar-se de soluções padrão, configuráveis para cada aplicação e tecnologia local, sem a necessidade de preocupar-se em manter e configurar servidores físicos, controlar capacidade, escalabilidade, dentre outros benefícios.

Por outro lado, a escolha da tecnologia para implementar o lado cliente é muito mais ampla, pois dispõe-se de uma variedade maior de linguagens e frameworks para desenvolvimento de aplicativos móveis. Um dos requisitos do trabalho proposto é a alta compatibilidade entre os elementos da arquitetura, por isso, optar por tecnologias multiplataforma/híbridas pode tornar o desenvolvimento produtivo. Contudo, dependendo da complexidade e uso de sensores do dispositivo, também deve-se considerar o uso de linguagens nativas do sistema operacional, pois o desempenho é consideravelmente melhor quando usada esta alternativa. Em ambos casos é possível implementar a arquitetura proposta, pois a mesma é baseada em serviços, via protocolo de requisições e interfaces padronizadas e que podem ser feitas por qualquer tecnologia cliente. Esse detalhamento constará nas próximas seções.

Deverá ser optado pelo desenvolvimento nativo quando:

- A aplicação precisa de alto desempenho, pois é robusta e de domínio complexo;
- A aplicação precisa de integração de alto desempenho com os sensores do dispositivo;
- A aplicação precisa ser compatível com o maior número de dispositivos e versões do sistema operacional possíveis;

- Os responsáveis pelo projeto devem possuir conhecimento em linguagens nativas distintas para manter repositórios de códigos diferentes, bibliotecas e estrutura de hardware disponíveis.

Java Android e Swift são as linguagens indicadas para Android e iOS, respectivamente. Por outro lado, deve-se optar pelo desenvolvimento multiplataforma quando:

- O domínio da aplicação não exige alto desempenho e complexidade;
- Deseja-se produtividade, utilizar-se de frameworks e bibliotecas as quais facilitam e aceleram o desenvolvimento;
- Deseja-se compatibilidade com os principais dispositivos e sistemas operacionais atuais.

Ionic [75], React Native [76] e Flutter [77] são as principais tecnologias indicadas para o desenvolvimento multiplataforma da arquitetura deste trabalho. A grande vantagem das duas primeiras é utilizarem o JavaScript como linguagem base e HTML para a camada de apresentação. Possuem vasta documentação e *plug-ins* de comunicação com sensores, muitos deles pré-processando dados de sensores em informações. O Flutter, por sua vez, não usa como base o JavaScript, mas tem como grande vantagem o desempenho semelhante ao de uma aplicação nativa quando trata-se da integração com sensores de hardware, mesmo sendo multiplataforma.

Para o exemplo instanciado neste trabalho, escolheu-se desenvolver a aplicação cliente com o framework Ionic (versão três), enquanto o lado servidor, optou-se pelo Node.js, por meio da estrutura do Firebase em sua versão gratuita. Com ambos, foi possível utilizar JavaScript para desenvolver os serviços de processamento e integração, armazenar grandes volumes de dados escalonáveis e não-estruturados, com um banco de dados orientado a documentos, além de utilizar-se dos serviços de envio de notificações, autenticação, dentre outros os quais o Firebase dispõe em sua plataforma. Isso porque a aplicação exemplo deste framework não necessita de alto desempenho e possui poucas funcionalidades iniciais de integração com os sensores do dispositivo.

4.2.5. Análise de requisitos arquitetônicos

Antes de iniciar a implementação da aplicação, deve-se pensar nas soluções adequadas, de acordo com as tecnologias escolhidas. O modelo dinâmico já detalha o que o sistema deve fazer, mas não como deverá será feito, portanto cabe ao desenvolvedor/analista optar pela forma adequada. A seguir, utilizando-se das funcionalidades da aplicação instanciada como exemplo, os principais desafios e soluções abordadas:

1. Sugerir refeições de acordo com o contexto: Um exemplo de implementação baseado em consultas. Inicialmente o sistema deve obter as informações de localização para encontrar a região do país do usuário. Nesse caso, não é necessário nenhum armazenamento em base de dados, afinal o sensor GPS retorna informações que podem ser processadas localmente no dispositivo no momento da aquisição, sem muitos esforços de hardware, para traduzir as coordenadas em uma informação de região. Obter a data e hora também não necessita armazenamento, pois pode-se obter a partir do sistema operacional do dispositivo e a partir de um processamento local, parametrizar as possibilidades de café, almoço, lanche ou janta. Por outro lado, as estações do ano devem ser armazenadas na base de dados na nuvem, pois precisam ser mantidas conforme datas iniciais e finais de cada ano, sendo assim a aplicação irá consultar na base e identificar qual estação é a atual.

   Após obtenção desses dados de sensores, eles servirão como parâmetros para uma nova consulta, dessa vez na base de dados para comparar e processar quais refeições se encaixam nos parâmetros obtidos naquele momento.
2. Envio de notificação em feiras próximas: monitorar a localização do usuário em tempo real é um desafio complexo de implementação em várias aplicações e domínios distintos. A aplicação deve, mesmo em segundo plano, manter o monitoramento das coordenadas do usuário, ao mesmo tempo em que processa essas coordenadas, comparando-as com locais específicos armazenados (no exemplo, as feiras). O primeiro desafio é o processamento, o qual deve-se evitar que seja no dispositivo, portanto indica-se implementar uma rotina na nuvem que receba as coordenadas, processe e compare o raio (geocerca) determinado com os

locais armazenados na base. Quando houver localização dentro desse raio, enviar notificação a partir do servidor, para o dispositivo do usuário em questão. O processamento local de um sinal de sensor para obter informações detalhadas só deve ser implementado quando há necessidade de disponibilidade constante, ou seja, não pode haver indisponibilidade de conexão com o servidor. O segundo desafio é o consumo de energia do dispositivo, pois ao requisitar comunicação com qualquer sensor (por exemplo o GPS) constantemente, diminui a carga de energia do dispositivo consideravelmente, tornando inviável o monitoramento constante. O trabalho de Mizouni *et al*. [70] detalha esse desafio, propondo uma modelagem baseada em separação de responsabilidades entre cliente e servidor para economizar energia. Além disso, modelar a requisição dos dados ao sensor em um intervalo adequado, desconsiderar estados estacionários do usuário, dentre outros, pode levar a um bom resultado para esses desafios.

3. Adaptar a aplicação conforme disponibilidade de conexão: esse é um exemplo mais simples e comum a ser abordado. Isso porque os sensores de conexão retornam o próprio estado da mesma, sem necessidade de processamento na nuvem ou consultas. O dado é utilizado em tempo real para definir uma ação de adaptação e em seguida pode ou não ser descartado.

Outros sensores como acelerômetro, magnetômetro, sensores de luminosidade, de ruído, dentre outros, retornam valores que normalmente devem ser processados com parâmetros definidos previamente e suas regras armazenadas na nuvem. Para resolver questões de disparidade de precisão na aquisição dos dados conforme o dispositivo, e se a aplicação exigir dados críticos, indica-se usar sensores externos que enviem os dados diretamente ao servidor de processamento. Exemplo disso são monitores de batimentos cardíacos e relógios inteligentes, dentre outros os quais podem monitorar de forma precisa e sem usar recursos de energia do dispositivo, além de possuírem interface de integração com servidores web.

4.2.6. Implementação da arquitetura na nuvem

De acordo com a arquitetura proposta nas seções anteriores, pode-se dividir a implementação na nuvem em três grandes conjuntos de elementos: o banco de dados e suas operações, os serviços para processar e disponibilizar informações, e as integrações com sistemas, API's e outros módulos terceiros.

- Banco de dados: O banco de dados utilizado foi o Cloud Firestore, um banco de dados flexível e escalonável para desenvolvimento de aplicações móveis, oferecido de forma gratuita, com limitações, pelo Google Cloud Platform por meio do Firebase. É um banco de dados orientado a documentos, semelhante ao MongoDB, outra opção de bancos de dados NoSQL. O NoSQL é considerado o melhor uso para dados não estruturados, porque lida com documentos organizados em entidades, ao invés de tabelas com restrições de linha e coluna. Cada entidade pode ser descrita como uma única unidade de informação, codificada em dados de bytes para armazenamento, e considera-se que não possuem esquema, embora haja implicitamente. Segue-se o paradigma BASE (basicamente disponível, estado flexível, eventualmente consistente) e tornou-se o método preferido quando há necessidade de escalabilidade, disponibilidade e simplicidade [54].

Considerando esses fatores, os modelos estruturais, conceitual e lógico nas Figuras 3 e 8, contemplam essas abordagens. Em ambos, por independer de tecnologias de bancos de dados, apresenta-se os relacionamentos entre as entidades e dados, porém ao implementar com bancos de dados NoSQL, e consequentemente sem esquemas, é preciso transformar entidades em coleções, as quais irão conter documentos, onde cada documento conterá os dados. As colunas contidas em um documento não indicam integridade e nem relacionamentos com outras coleções de documentos, sendo esse o principal paradigma a ser quebrado, como grande diferencial para implementação da arquitetura. Mesmo assim, é possível representar um esquema ilustrando como foi instanciado o modelo físico no banco de dados, utilizando o exemplo de armazenamento das refeições, contendo a estação, região e horário, para serem consultadas pela aplicação após obtenção e processamento dos dados de contexto, os quais formam os parâmetros para a consulta. É imprescindível observar que o modelo de dados apresenta o usuário relacionado com seus elementos de contexto (localização, estação, região), entretanto esses dados não precisam ser armazenados na base, mas sim obtidos e

utilizados como parâmetro para a consulta na base sobre refeições sugeridas, portanto ao instanciar o modelo físico, será focado nos dados armazenados para consulta.

A Figura 11 mostra o exemplo aplicado ao Firebase Cloud Firestore, onde para implementação da funcionalidade “sugerir refeições conforme contexto”, foram criadas as coleções usuários, refeições e estações do ano. Para a notificação de feiras orgânicas próximas, foi criada a coleção Feiras, a qual contém as informações das mesmas, como latitude e longitude e que possam ser processadas posteriormente. Como citado anteriormente, o foco desse tipo de banco de dados não é a integridade dos dados e sim a facilidade nas consultas, ainda que haja redundância. Portanto, a aplicação consultará a estação do ano em uma coleção, e posteriormente consultará a coleção de refeições utilizando o campo estação contido nela. Em alguns casos será possível e indicado criar índices para determinados campos em coleções.

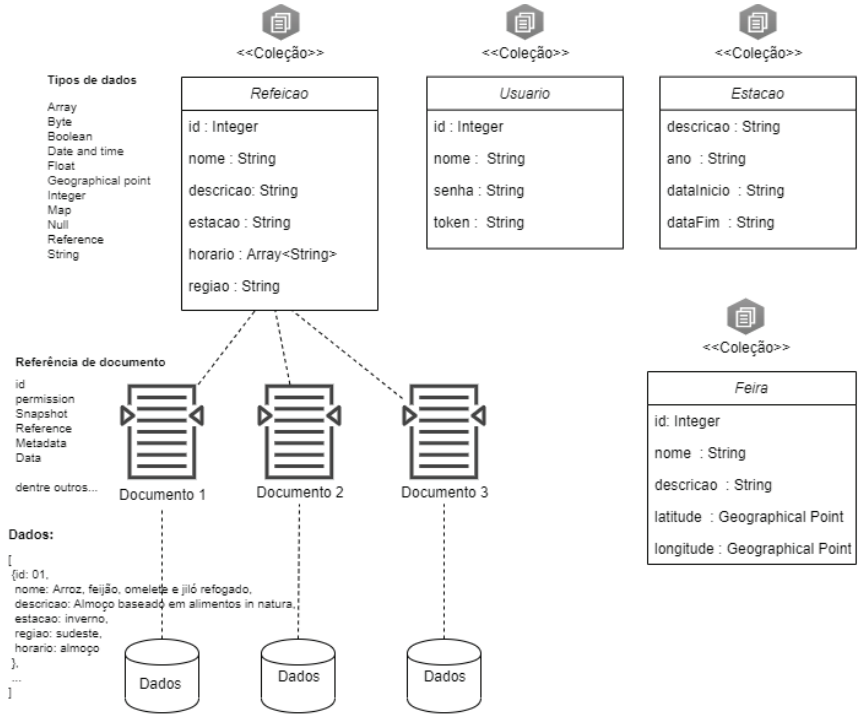


Figura 11. Implementação para armazenamento na base de dados.

- Módulo de processamento e serviços: baseia-se no conceito de Aplicações Orientadas a Serviços (SOA), onde a chave é o serviço, e cada

> serviço é uma função precisamente definida e bem encapsulada. SOA é indicado para sistemas distribuídos onde deve-se usar interfaces e padrões bem definidos entre as várias unidades (serviços) dos aplicativos em uma plataforma heterogênea. Também prioriza o baixo acoplamento, ou seja, integra vários aplicativos existentes através da rede, de forma independente e se comunicando por meio de padrões abertos [78].

Uma implementação SOA consiste em uma combinação de tecnologias, produtos, APIs, extensões de infra-estrutura e várias outras partes, por isso para implementação da arquitetura deste trabalho, o Node.Js é parte desse conceito de implementação. A Figura 12 detalha especificamente o módulo de processamento e serviços, ampliando a visão geral da arquitetura proposta no servidor. O módulo consiste basicamente em uma API Rest, fornecendo métodos via protocolo HTTP para acessar as funções da aplicação servidor, também devendo ser implementado em uma arquitetura MVC (Model-View-Controller ou Modelo, Apresentação e Controle) básica. O primeiro submódulo é o de apresentação, o qual recebe as requisições e as encaminha para o controlador respectivo, também chamado de API Gateway. O controlador consiste nas funções e toda a lógica de negócio do serviço, sendo que também pode necessitar importar módulos, API's e bibliotecas terceiras. A lógica é estendida para o submódulo de modelo, onde fica a integração com o banco de dados, seja para consulta, armazenamento de dados e validações.

A implementação utilizando essas tecnologias, dentro de uma arquitetura com conceitos orientados a serviços, permite que o módulo de processamentos e serviços, via protocolo HTTP em uma API Rest no servidor, tenha a possibilidade de receber requisições em alto volume, escalabilidade, sendo flexível. Com isso,pode importar e integrar com toda e qualquer biblioteca, serviços ou API's de terceiros, incorporando funcionalidades e soluções dentro da lógica de negócio, assim como disparar serviços próprios como envio de notificações, autenticações, dentre outros.

Para a aplicação instanciada neste trabalho, utilizou-se de uma solução disponível em seu plano gratuito, a qual abstrai boa parte dessa implementação, a fins de ganho de produtividade e melhores práticas adotadas pela Google, por meio do Firebase, classificado como um "back-end as a service". O Firebase possui suas próprias soluções em bibliotecas de envio de notificação, autenticação, gerenciamento de bancos de dados e vários outros serviços úteis para o desenvolvimento de aplicações móveis podendo ser usado parcialmente ou

integralmente dentro de uma implementação no servidor. Uma dessas soluções é o Firebase Cloud Functions que permite executar automaticamente o código de back-end em resposta a eventos acionados por recursos do Firebase e solicitações HTTPS, ou em outras palavras, abstrai as camadas de visão e modelo propostas na arquitetura do módulo em questão, sendo necessário apenas desenvolver a lógica dos serviços em um ambiente gerenciado [74].

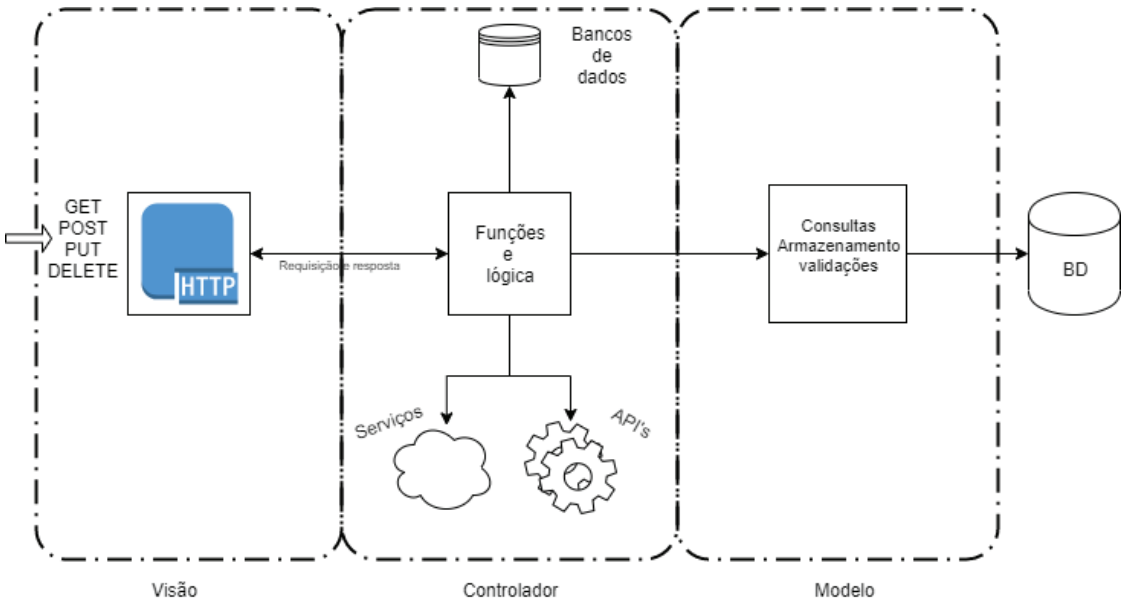


Figura 12. Detalhamento do módulo de processamento e serviços.

4.2.7. Implementação da arquitetura no cliente

Como citado anteriormente, as principais tecnologias em desenvolvimento móvel atuais, tanto nativas (Java Android e Swift) quanto multiplataforma (Ionic, React, Flutter) seguem o padrão MVC para implementação. Dentro desse padrão, sugere-se desenvolver os elementos da arquitetura proposta, sendo o módulo de serviços e processamento o principal deles.

O módulo central tem a responsabilidade de interagir com sensores internos e externos, manter e sincronizar dados com o banco de dados temporário local e comunicar-se com o módulo de processamento e serviços da nuvem. A Figura 13 apresenta uma visão geral detalhada da arquitetura no lado cliente,

ampliando os elementos e exemplificando a implementação das funcionalidades da aplicação em questão.

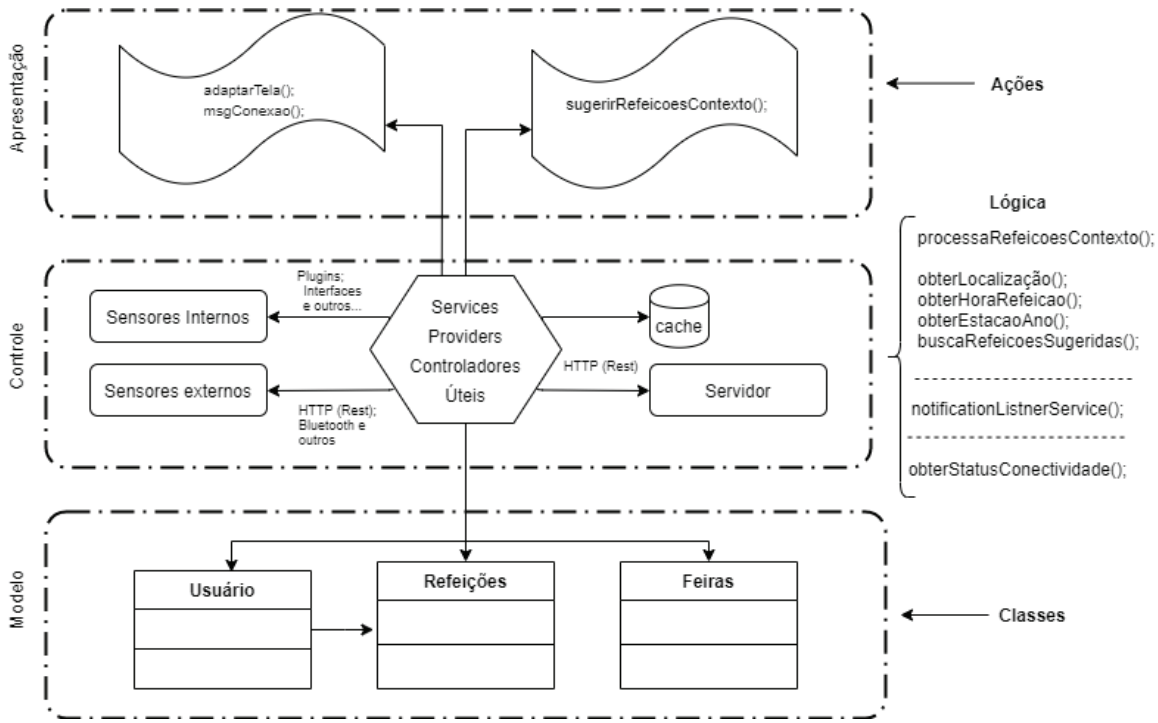


Figura 13. Detalhamento da arquitetura no cliente.

Em um padrão modelo-controle-apresentação, a última camada apenas deve ser responsável por delegar as ações que o sistema deve tomar, por exemplo uma adaptação conforme a conectividade ou uma apresentação de dados conforme o contexto selecionado. Nessa camada não há lógica, ao contrário da camada de controle, onde é implementado o código principal, por meio de serviços descentralizados e reutilizáveis, como Services, Providers, Controllers e utilitários, dependendo da linguagem escolhida. Essa codificação pode variar de acordo com a tecnologia, mas comumente representam classes públicas dentro da aplicação para uso comum, sendo que dentro delas devem ser codificadas e encadeadas as requisições externas, os dados e o processamento dessas informações. Também é nesse elemento da arquitetura onde deve ser implementada a integração com os sensores do dispositivo, seja por meio de bibliotecas nativas de uma linguagem, como plug-ins para linguagens multiplataforma. Sensores externos como relógios inteligentes, monitores de sinais vitais e outros possuem comunicação Bluetooth e podem ser recebidos comunicando-se com o dispositivo, assim como enviados

diretamente para o módulo da nuvem, dependendo da interface que o dispositivo prover. Utiliza-se também da camada de modelo onde estão declaradas as classes de domínio, as quais auxiliam na manipulação dos dados recebidos dos sensores e posteriormente no fluxo de dados entre as funções.

Em resumo, na prática, a camada de apresentação recebe os dados prontos para executar uma ação. Já a camada de controle implementa uma funcionalidade por meio de funções encadeadas, requisições aos sensores e processamento desses dados, quando esta responsabilidade ficar do lado cliente. No exemplo da Figura 13, duas funcionalidades estão descritas: a primeira é a sugestão de refeições conforme contexto, onde a camada de controle deve implementar a requisição da localização do usuário ao sensor de GPS, a requisição ao servidor próprio onde estão armazenadas as estações do ano, a requisição de data e hora ao próprio sistema operacional e, utilizando-se do processamento local dos resultados dessas funções, utiliza-os como parâmetro para consultar as refeições sugeridas com estes, comunicando-se com o módulo de serviços e processamento na nuvem, o qual será responsável por delegar essa requisição ao banco de dados e filtrar a resposta. Ainda utiliza-se de classes na camada de modelo para declarar e manipular os dados de resposta dentro da aplicação.

É importante destacar que as arquiteturas voltadas para servidores na nuvem e dependentes de requisições via protocolo necessitam de atenção no encadeamento das mesmas, ou seja, é preciso que a aplicação aguarde o resultado de uma requisição para que a próxima seja enviada e assim por diante. Outro ponto importante a ser destacado é a implementação, quando necessário, do banco de dados local para armazenamento em cache. Este é um recurso que os bancos de dados MongoDB e Firebase Firestore oferecem (assim como outros) e permite que mesmo sem conectividade, as requisições e os dados manipulados localmente na aplicação sejam persistidos de forma ordenada e mantendo em cache os dados da última requisição com conectividade. Essa abordagem possui limitações, pois o armazenamento em cache é rápido e eficiente, mas apenas permite que os dados fluam de uma maneira: do servidor para o dispositivo. A replicação manual oferece aos usuários capacidade de leitura e gravação, mas com o custo de introduzir possíveis conflitos e possíveis. Por isso recomenda-se este recurso apenas para funcionalidades onde a visualização dos dados ou até mesmo a manipulação dos

mesmos não seja imediata ou crítica, podendo haver a implementação de toda essa funcionalidade com contexto localmente e enviada quando houver conectividade.

O módulo de processamento e serviços local deve servir na maioria das vezes como um organizador e pré-processamento para a nuvem. Isso torna a resposta para o usuário mais rápida e eficiente, pois como citado anteriormente muitas informações não necessitam de processamentos robustos no dispositivo, portanto seria um desperdício enviá-las ao servidor. Por outro lado, deve-se ter o cuidado ao implementar essas funcionalidades locais, afinal é preciso analisar o quanto esse pré-processamento será custoso ao dispositivo – e a uma gama variada de dispositivos. Assim como implementar uma estratégia de cache torna-se preciso analisar qual a frequência que o usuário precisa das informações de contexto atualizadas e sincronizadas com o servidor, ou apenas de que forma esses dados disponíveis sem conectividade podem contribuir para o contexto da funcionalidade.

4.2.8. Aplicativo Guia Alimentar

O aplicativo de exemplo utilizando as diretrizes e a estrutura proposta neste trabalho foi disponibilizado para o sistema operacional Android, por meio da loja Play Store, no programa de testes Alfa Fechado. Além das tecnologias como o Firebase e o Ionic Framework, utilizou-se da ferramenta online Draw.io para criação dos modelos apresentados nas figuras, do Visual Studio Code para a codificação da aplicação e do Canva online para elaboração da aparência. O aplicativo foi desenvolvido e testado nas plataformas Android e Apple, utilizando-se de emuladores Genymotion e XCode. No entanto, em um primeiro momento foi disponibilizado gratuitamente apenas para a plataforma Android, por restrições de custo para publicação nas duas lojas de aplicativos.

A Figura 14 apresenta a sequência de telas para que o usuário acesse a aplicação, efetue o cadastro e acesse a tela de diário de refeições, onde a funcionalidade de sugerir refeições conforme o contexto estará disponível. A aplicação detecta qual é a próxima refeição de acordo com o horário atual, destacando a mesma e sugerindo uma refeição de acordo com a estação do ano e a região do país onde está no momento. Também notifica o usuário caso a aplicação esteja fechada ou em segundo plano.

Previamente as refeições foram marcadas e categorizadas por região e estação, por isso a aplicação usa os parâmetros de contexto obtidos para buscar a sugestão mais indicada ao usuário. Caso haja falha na obtenção do contexto, será selecionada toda a lista de refeições para determinado horário e sugerido aleatoriamente uma delas. Caso duas ou mais refeições estejam em critérios de empate, o mesmo critério é aplicado para elas. Essa forma de classificação foi escolhida pois busca-se encontrar uma sugestão mais próxima dos parâmetros do usuário. Contudo, caso não haja, o Guia Alimentar não deixa explícito que as demais refeições não possam ser sugeridas, pelo contrário, é um documento que visa a acessibilidade e facilidade para encontrar os alimentos em qualquer região.

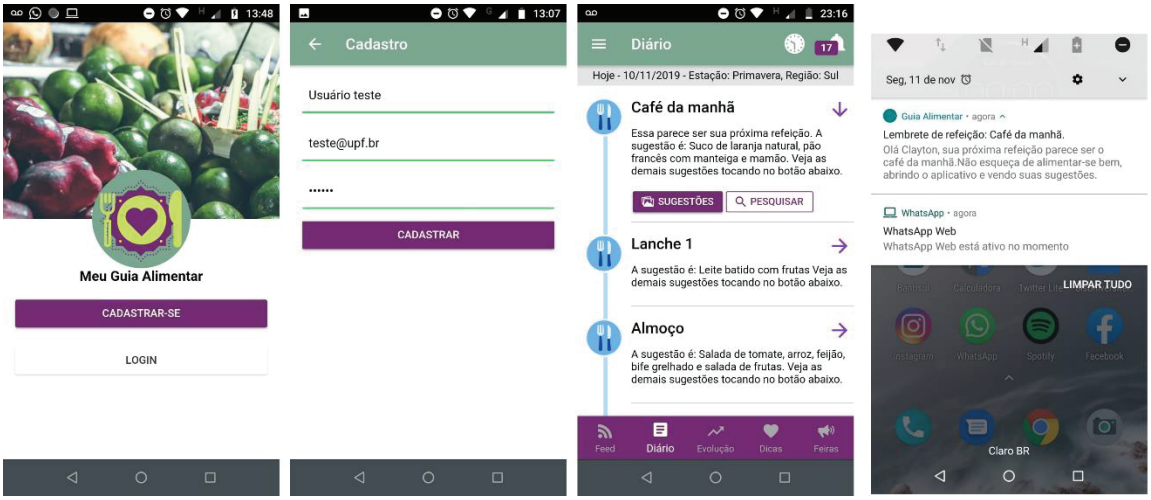
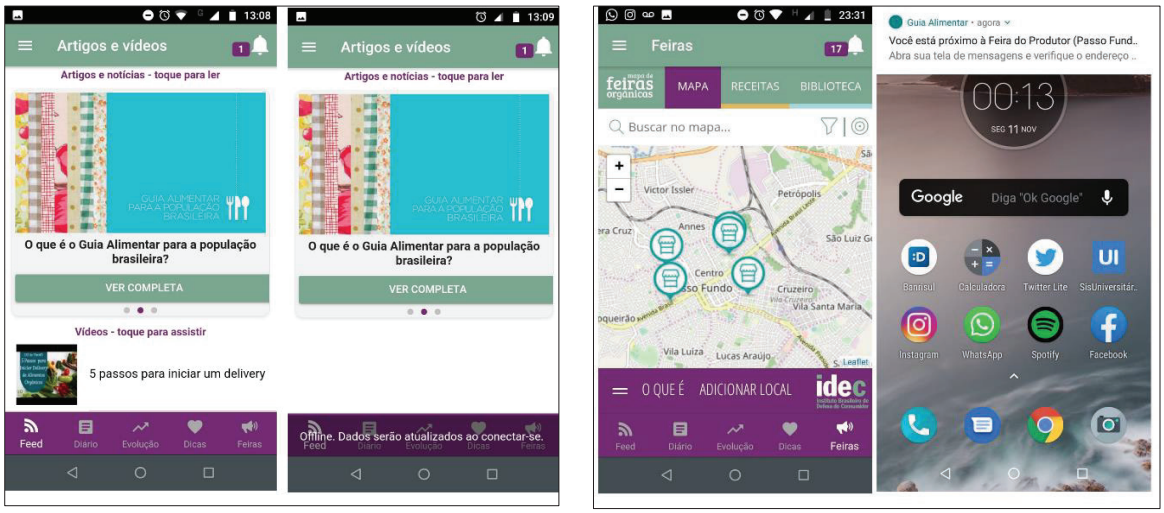


Figura 14. Telas para acessar o diário de sugestões de refeição.

A Figura 15(a) apresenta uma das adaptações da aplicação, na tela de *Feed*, onde as notícias e artigos possuem dados em cache, enquanto os vídeos são carregados diretamente do Youtube. Nesse exemplo, os artigos e notícias podem ser carregados mesmo sem conectividade, mas os vídeos não, sendo assim a aplicação adapta-se ocultando os vídeos e mostrando uma mensagem ao usuário que está sem conectividade. Na Figura 15(b) há o Mapa de Feiras orgânicas, onde os pontos estão replicados na base de dados própria da aplicação deste trabalho, sendo que o usuário é monitorado com sua localização e ao aproximar-se em um raio de duzentos metros de algum desses pontos de feiras, será notificado da mesma, recebendo também uma mensagem com a localização no mapa para que possa encontra-la e guiá-la.



(a) Adaptação. (b) Mapa de feiras.

Figura 15. Telas da aplicação: conectividade e mapa de feiras.

4.3. AVALIAÇÃO DOS ESPECIALISTAS DE DOMÍNIO

Nesta seção serão descritas as etapas realizadas para a avaliação da aplicação com especialistas, que teve como objetivo verificar se a aplicação desenvolvida, a partir do framework, realizava as adaptações e apresentações corretamente, cumprindo os requisitos para uma aplicação sensível ao contexto. Para tanto, a avaliação se concentrou nas três funcionalidades principais da aplicação, sendo elas: sugestões de refeição, notificação de feiras próximas e adaptação conforme conectividade disponível. Após um período de uso, os participantes foram convidados a responderem a um questionário online para coleta de dados.

4.3.1. Elaboração do questionário de avaliação

Para coleta de dados dos participantes foi utilizado um questionário online, elaborado e revisado por três especialistas, dois da área da computação e um da área da nutrição. O objetivo foi formular questões com valor semântico aos usuários, abordando as funcionalidades a serem avaliadas.

O questionário foi dividido em duas partes (Apêndice C). A primeira parte buscou coletar informações para caracterização da amostra, a saber: idade, tempo

de atuação profissional (em anos), se já utilizou algum aplicativo de hábitos alimentares e qual o modelo do smartphone/Sistema operacional utilizado. Em seguida, três questões objetivas relacionadas ao cumprimento ou não das funcionalidades. Para essas questões foi utilizada a escala Likert [79] de cinco pontos: Discordo totalmente, Discordo, Não concordo nem discordo, Concordo e Concordo totalmente. A segunda parte, composta por três questões descritivas, teve como objetivo avaliar a percepção do usuário para cada uma das três funcionalidades. Por fim, havia uma quarta questão descritiva solicitando possíveis sugestões em relação aos itens abordados anteriormente. O instrumento foi desenvolvido na plataforma de Formulários Google.

4.3.2. Procedimento de coleta de dados

O estudo foi realizado com 6 participantes, 2 do sexo masculino e 4 do sexo feminino, todos selecionados por amostragem não probabilística, por conveniência. Os objetivos da pesquisa foram esclarecidos previamente à instalação do aplicativo, com um manual de uso do aplicativo (Apêndice D), seguido de um texto explicativo (Apêndice E). Todos os participantes selecionados possuíam smartphones com sistema operacional Android e já haviam utilizado ou possuíam algum conhecimento sobre aplicações de qualquer categoria.

O estudo foi realizado no segundo semestre de 2019, sendo que cada usuário deveria instalar, registrar-se e utilizar o aplicativo por sete dias, tempo estabelecido pelos pesquisadores para que os usuários tivessem acesso e interagissem com as três funcionalidades sensíveis ao contexto do aplicativo. Após esse período, o link para o questionário de avaliação (Apêndice C) foi enviado por Email aos participantes.

4.3.3. Análise das respostas

A idade média dos participantes foi de 24,4 ± 2,7 anos. O tempo de profissão variou entre 2 a 72 meses. Apenas um participante nunca havia utilizado algum aplicativo de hábitos alimentares, e os demais citaram os aplicativos Dietbox, Nutrium e Desrotulando, sendo apenas este último também com foco em escolhas alimentares, sem priorizar dietas, quantidades e acompanhamento clínico.

A avaliação dos participantes em relação às sugestões de refeição foi de concordo plenamente (83,3%) e concordo (16,7%). Entre as respostas descritivas dessa questão, dois usuários destacaram como “Excelente! Trabalhando com variedades, lembrando sobre frequência e estimulando bons hábitos” e “Refeições adequadas nutricionalmente, de acordo com a região. Optando por escolhas corretas”. Os demais descreveram como adequada, boa ou muito boa a funcionalidade.

Para a questão sobre notificações de feiras próximas, três usuários concordaram plenamente, um usuário concordou, um usuário não concordou nem discordou e outro usuário discordou totalmente. Nas respostas descritivas dessa questão foi possível observar que os usuários que não concordaram ou não souberam opinar não haviam sido notificados de férias próximas, enquanto os demais foram notificados e destacaram como “Excelente” e “Interessante, principalmente para quem desconhece a presença/local das feiras”. Um usuário descreveu como “Regular” e sugeriu que o aplicativo não notificasse todas as vezes em que estivesse próximo ao local, e sim somente uma vez.

Em relação a questão sobre o aplicativo mostrar seu conteúdo independente da disponibilidade de conexão, apenas um usuário discordou, outros dois concordaram plenamente, dois concordaram e dois não concordaram nem discordaram. Analisando as respostas descritivas, os usuários que não concordaram nem discordaram escreveram que não puderam perceber diferença no conteúdo, pois acreditaram que não estiveram sem conexão durante o uso do mesmo. O usuário que discordou totalmente destacou que “Não apresenta todas as funcionalidades, porém alerta sobre a desconexão”.

Na última questão aberta, apenas três usuários deixaram sugestões, sendo uma delas em relação às funcionalidades sensíveis ao contexto, já citada anteriormente, notificar apenas uma vez o usuário quando estiver próximo a uma feira.

Considerando as proposições da literatura revisada, cada uma com suas especificidades, os processos de validação e verificação dos frameworks são tópicos pouco explorados. Em relação aos principais trabalhos elencados na literatura revisada, dois deles não utilizam alguma forma avaliação, focando apenas na proposição de suas estruturas e aplicando em um caso de uso [6][8]. Outro estudo avalia o desempenho da aplicação desenvolvida por meio do framework [5], e por

fim, um utiliza a avaliação do framework com especialistas em design [4], verificando a assertividade dos modelos e dos processos para o entendimento mútuo dos envolvidos em um projeto de software sensível ao contexto.

Este trabalho apresenta algumas limitações com relação a avaliação do framework proposto. Na avaliação dos profissionais de software, o caso de uso limitou-se a uma modelagem simples sem entrar no nível de desenvolvimento sequente a análise, ponto esse que poderia ser objeto de estudo futuro. Já na avaliação dos profissionais da nutrição, nem todos passaram por uma troca de estação ou de região devido ao curto tempo de teste, o que inviabilizou uma possível verificação de conteúdo. Outra limitação foi o tamanho reduzido da amostra, que não permite generalizações. O ideal é que em estudos futuros o aplicativo seja testado por uma amostra maior de profissionais, em regiões diferentes do país e em um período que o aplicativo possa ser utilizado em pelo menos duas estações do ano.

5. CONCLUSÃO

Este estudo teve como objetivo propor um framework visando apoiar o desenvolvimento de aplicações sensíveis ao contexto por meio de modelos de representação, arquitetura e diretrizes para guiar o desenvolvimento de aplicações sensíveis ao contexto. Adicionalmente a isso, foi desenvolvida uma aplicação exemplificando o uso do framework, instanciando os modelos e aplicando as diretrizes de desenvolvimento, tecnologias e arquitetura proposta, com o objetivo de verificar a assertividade do mesmo.

O framework mostrou-se viável no exemplo apresentado, a partir da aplicação dos conceitos nos modelos e arquitetura propostos. De acordo com o feedback dos profissionais especialistas em análise de software, o framework apresenta uma linguagem simples e detalhada, além de poder ser reutilizado em domínios diferentes, segundo os relatos dos participantes. Ainda sobre a avaliação, os profissionais relataram alguns itens que podem ser evoluídos na representação dos modelos, agregando ainda mais entendimento e praticidade. Em outro eixo, a avaliação da aplicação instanciada por especialistas de domínio, também se mostrou positiva, onde os mesmos relataram boa interação e assertividade nas funcionalidades propostas com sensibilidade ao contexto durante o período de utilização.

As principais contribuições deste trabalho foram: (i) reutilização e adaptação de modelos consolidados da literatura, buscando abstração e atualização, mas também aplicando os mesmos conceitos em duas notações complementares; (ii) proposição de uma arquitetura detalhada para tratar de requisitos funcionais e arquitetônicos; (iii) ferramentas e guias de instruções práticas, por meio de tecnologias e serviços atuais, descritos para implementação da arquitetura. Acredita-se que essa seja uma proposta original, pois aborda não somente o foco em modelos conceituais e genéricos para representação do contexto, mas também detalha boas práticas e os desafios para encontrar melhores soluções de implementação, buscando avançar a literatura ao entendimento de engenheiros e desenvolvedores que pretendem trabalhar com aplicações sensíveis ao contexto. Em linhas gerais, o framework aborda as fases de projeto e

desenvolvimento de forma vertical e horizontal, fornecendo instrumentos, guias de uso e ferramentas tecnológicas para implementação.

Como trabalhos futuros, sugere-se: (i) aprofundar avaliações de feedback da aplicação do framework com profissionais de tecnologia; (ii) aplicar os conceitos desta estrutura em outros domínios de aplicação, principalmente os que possuem maior complexidade e dados críticos, de forma a verificar a flexibilidade dos modelos e da arquitetura; (iii) o desenvolvimento de uma aplicação idêntica ao exemplo desse trabalho, porém sem o uso do contexto também pode ser objeto de estudo para comparar a avaliação dos usuários e sua percepção entre ambos.

REFERÊNCIAS

[1] WEISER, M. The computer for the 21st century. *Sci. Am. (International Ed.* vol. 265, no. 3. p. 66–75. 1991.

[2] DE LEMOS, R. *et al.* Software engineering for self-adaptive systems: A second research roadmap. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*. vol. 7475 LNCS. p. 1–32. 2013.

[3] VIEIRA, V.; TEDESCO, P.; SALGADO, C. Modelos e Processos para o Desenvolvimento de Sistemas Sensíveis ao Contexto. *Challenges*. vol. 20, no. 3. p. 381–431. 2009.

[4] VIEIRA, V.; TEDESCO, P.; SALGADO, A. C. Designing context-sensitive systems: An integrated approach. *Expert Syst. Appl.* vol. 38, no. 2. p. 1119–1138. 2011.

[5] MILIĆ, E. EgoSENSE : A Framework for Context-Aware Mobile Applications Development. vol. 7, no. 4. p. 1791–1796. 2017.

[6] SEEL, C. A Framework to Model and Implement Mobile Context- Aware Business Applications. p. 23–38. 2018.

[7] LA, H. J.; KIM, S. D. A Conceptual Framework for Provisioning Context-aware Mobile Cloud Services. *2010 IEEE 3rd Int. Conf. Cloud Comput.* p. 466–473. 2010.

[8] BEN CHEIKH, A. *et al.* An engineering method for context-aware and reactive systems. *Proc. - Int. Conf. Res. Challenges Inf. Sci.* 2012.

[9] ZAKAS, N. C. The evolution of web development for mobile devices. *Commun. ACM*. vol. 56, no. 4. p. 42. 2013.

[10] BELLAVISTA, P. *et al.* A survey of context data distribution for mobile ubiquitous systems. *ACM Comput. Surv.* vol. 44, no. 4. p. 1–45. 2012.

[11] CHEN, G.; KOTZ, D. A Survey of Context-Aware Mobile Computing Research. *Dartmouth Comput. Sci. Tech. Rep.* vol. 3755. p. 1–16. 2000.

[12] DEY, A. K.; ABOWD, G. D. Towards a Better Understanding of Context and Context-Awareness. *Comput. Syst.* vol. 40, no. 3. p. 304–307. 1999.

[13] SCHILIT, B. N.; ADAMS, N.; WANT, R. Context-aware computing applications. *IEEE Work. Mob. Comput. Syst. Appl.* p. 85–90. 1994.

[14] TO, P.; FULFILLMENT, I. P. Providing Architectural Support for Building Context-Aware Applications By Anind K . Dey In Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy in Computer Science College of Computing Georgia Institute of Technology. no. August. 1999.

[15] SANCHES, H. Aplicações e Sistemas Sensíveis ao Contexto. *06/02/2014*. 2014. Disponível em: <https://nuvemandroid.wordpress.com/2014/02/06/aplicacoes-e-sistemas-sensiveis-ao-contexto/>. Acesso em: 21 out. 2018.

[16] SOYLU, A.; DE CAUSMAECKER, P.; DESMET, P. Context and adaptivity in pervasive computing environments: Links with software engineering and ontological engineering. *J. Softw.* vol. 4, no. 9. p. 992–1013. 2009.

[17] WINOGRAD, T. Architectures for context. *Human-Computer Interact.* vol. 16, no. 2–4. p. 401–419. 2001.

[18] HENRICKSEN, K.; INDULSKA, J.; RAKOTONIRAINY, A. Modeling Context Information in Pervasive Computing Systems. *Pervasive '02 Proc. First Int. Conf. Pervasive Comput.* p. 167–180. 2002.

[19] PREKOP, P.; BURNETT, M.; PARK, F. H. Activities , Context and Ubiquitous Computing 2 . Existing Context-Awareness Approaches. *Spec. Issue Ubiquitous Comput. Comput. Commun.* vol. 26, no. 11. p. 1168–1176. 2003.

[20] GUSTAVSEN, R. M. Condor – an application framework for mobility- based context-aware applications. *Proc. Work. Concepts Model. Ubiquitous Comput.* vol. 39. p. 1–6. 2002.
[21] BALDAUF, M. A survey on context-aware systems Schahram Dustdar * and Florian Rosenberg. *Inf. Syst.* vol. 2, no. 4. 2007.
[22] SANCHEZ, L. *et al.* A generic context management framework for personal networking environments. *2006 3rd Annu. Int. Conf. Mob. Ubiquitous Syst. Netw. Serv. MobiQuitous*. 2006.
[23] HAN, L. *et al.* Research on Context-Aware Mobile Computing. *22nd Int. Conf. Adv. Inf. Netw. Appl. - Work. (aina Work. 2008)*. p. 24–30. 2008.
[24] DIX, A. *et al.* Exploiting space and location as a design framework for interactive mobile systems. *ACM Trans. Comput. Interact.* vol. 7, no. 3. p. 285–321. 2000.
[25] DEY, A. K.; ABOWD, G. D.; SALBER, D. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interact.* vol. 16, no. 2–4. p. 97–166. 2001.
[26] VIEIRA, V. *et al.* A context-oriented model for domain-independent context management. *Rev. d'Intelligence Artif.* vol. 22, no. 5. p. 609–627. 2008.
[27] TEDESCO, S. A. S. C. S. A. A contextualised Learning Interaction Memory. *J. Brazilian Comput. Soc.* vol. 13, no. 3. 2007.
[28] PARK, D. J. *et al.* A context-aware smart tourist guide application for an old palace. *2007 Int. Conf. Converg. Inf. Technol. ICCIT 2007*. p. 89–94. 2007.
[29] MACÍAS-ESCRIVÁ, F. D. *et al.* Self-adaptive systems: A survey of current approaches, research challenges and applications. *Expert Syst. Appl.* vol. 40, no. 18. p. 7267–7279. 2013.
[30] KEAYS, R.; RAKOTONIRAINY, A. Context-Oriented Programming. vol. 7, no. 3. p. 125–151. 2003.
[31] JEFFREY, O.; DAVID, M. The Vision of. *J. Ecol.* vol. 90, no. January. p. 936–946. 2002.
[32] WEIBELZAHL, S. Problems and Pitfalls in Evaluating Adaptive Systems 1. 2005.
[33] VAUGHAN, N.; GABRYS, B.; DUBEY, V. N. An overview of self-adaptive technologies within virtual reality training. *Comput. Sci. Rev.* vol. 22. p. 65–87. 2016.
[34] STENUDD, S. *Using machine learning in the adaptive control of a smart environment*. 2010.
[35] SALEHIE, M.; TAHVILDARI, L. Self-adaptive software. *ACM Trans. Auton. Adapt. Syst.* vol. 4, no. 2. p. 1–42. 2009.
[36] OREIZY, P.; MEDVIDOVIC, N.; TAYLOR, R. N. Architecture-based runtime software evolution. *Proc. 20th Int. Conf. Softw. Eng.* p. 177–186. 1998.
[37] BUCKLEY, J. *et al.* Towards a taxonomy of software change. *J. Softw. Maint. Evol.* vol. 17, no. 5. p. 309–332. 2005.
[38] SCHMIDT, A.; LAERHOVEN, K. Van. How to Build Smart Appliances? no. August. 2001.
[39] KRUPITZER, C. *et al.* A survey on engineering approaches for self-adaptive systems. *Pervasive Mob. Comput.* vol. 17, no. PB. p. 184–206. 2015.
[40] SHARIFLOO, A. M. Models for Self-Adaptive Systems. *Proc. 2015 Eur. Conf. Softw. Archit. Work. - ECSAW '15*. p. 1–5. 2015.
[41] SCHILIT, B. N.; THEIMER, M. M. Disseminating Active Map Information to Mobile Hosts. 1994.
[42] BROWN, M. G. Supporting User Mobility. *Mob. Commun.* p. 69–77. 1996.
[43] COOPERSTOCK, J. R. *et al.* Evolution of a Reactive Environment. *Proc. 1995 ACM Conf. Hum. Factors*. p. 170–177. 1995.

[44] ELROD, S. C. O. I. T.; HALL, G. Discount for designers and programmers at. vol. 2, no. July. p. 7–9. 1993.
[45] HULL, R.; NEAVES, P.; BEDFORD-ROBERTS, J. Towards situated computing. *Dig. Pap. First Int. Symp. Wearable Comput.* p. 146–153. 1997.
[46] FICKAS, S.; KORTUEM, G.; SEGALL, Z. Software organization for dynamic and adaptable wearable systems. *Dig. Pap. First Int. Symp. Wearable Comput.* p. 56–63. 1997.
[47] BROWN, P. J. Triggering information by context. *Pers. Ubiquitous Comput.* vol. 2, no. 1. p. 18–27. 1998.
[48] PASCOE, J. Adding generic contextual capabilities to wearable computers. *Int. Symp. Wearable Comput. Dig. Pap.* vol. 1998-Octob, no. 0. p. 92–99. 1998.
[49] COLMAN, A. W. *et al.* Context in Computing. no. December. 2014.
[50] HUSSEIN, M.; HAN, J.; COLMAN, A. Context-Aware Adaptive Software Systems: A System-Context Relationships Oriented Survey. *Ict.Swin.Edu.Au*. no. July. 2009.
[51] PERERA, C. *et al.* Context Aware Computing for The Internet of Things: A Survey. *IEEE Commun. Surv. Tutorials*. vol. 16, no. 1. p. 414–454. 2014.
[52] BETTINI, C. *et al.* A survey of context modelling and reasoning techniques. *Pervasive Mob. Comput.* vol. 6, no. 2. p. 161–180. 2010.
[53] VENKATRAMAN, S.; KASPI, K. F. S.; VENKATRAMAN, R. SQL Versus NoSQL Movement with Big Data Analytics. *Int. J. Inf. Technol. Comput. Sci.* vol. 8, no. 12. p. 59–66. 2016.
[54] HAMMES, D.; MEDERO, H.; MITCHELL, H. Comparison of NoSQL and SQL Databases in the Cloud. *SAIS 2014 Proc.* p. 1–8. 2014.
[55] MARIO, N.; DENIS, P.; MILAN, P. Primjena i prednosti NoSQL baza podataka Using the advantages of NoSQL databases Mario Novoselec , Denis Pavlović , Milan Pavlović GET user _ role.
[56] GROUP, O. M. BUSINESS PROCESS MODEL & NOTATION. Disponível em: <https://www.omg.org/bpmn/>. Acesso em: 17 dez. 2019.
[57] DÖRNDORFER, J.; SEEL, C. A Meta Model Based Extension of BPMN 2.0 for Mobile Context Sensitive Business Processes and Applications. *13. Int. Tagung Wirtschaftsinformatik*. p. 301–315. 2017.
[58] LANDAY, J. A.; HONG, J. I. An Infrastructure Approach to Context-Aware Computing. *Human-Computer Interact.* vol. 16, no. 2–5. p. 287–303. 2001.
[59] PAPER, C. MoBe : A Framework for Context-Aware Mobile Applications MoBe : A Framework for Context-Aware. no. April 2016. 2005.
[60] MAHMUD, A.; SATTAR, A. Deployment of contextual E-healthcare system: A prospective e-service based on context aware conceptual framework and ICTization framework model. *Proc. 2016 IEEE 11th Conf. Ind. Electron. Appl. ICIEA 2016*. p. 77–82. 2016.
[61] KIRSCH-PINHEIRO, M. *et al.* Requirements Analysis for Context-oriented Systems To cite this version : HAL Id : hal-01527345 ScienceDirect Requirements Analysis for Context-oriented Systems. 2017.
[62] HANSEL, J.; VOGEL, T.; GIESE, H. A testing scheme for self-adaptive software systems with architectural runtime models. *Proc. - 2015 IEEE 9th Int. Conf. Self-Adaptive Self-Organizing Syst. Work. SASOW 2015*. p. 134–139. 2015.
[63] SANTOS, R.; ROCHA, L. Análise do Contexto Móvel nos Testes de Usabilidade de Aplicações Móveis. *Researchgate.Net*. no. November 2011. 2011.
[64] AARAB, Z.; SAIDI, R.; RAHMANI, M. D. Towards a Framework for Context-Aware Mobile Information Systems. *2014 Tenth Int. Conf. Signal-Image Technol. Internet-Based Syst.* vol. 1, no. 1. p. 694–701. 2014.

[65] VARGAS DA ROCHA, L.; EDELWEISS, N.; IOCHPE, C. GeoFrame-T: A temporal conceptual framework for data modeling. *Proc. ACM Work. Adv. Geogr. Inf. Syst.* p. 124–129. 2001.
[66] GOOGLE DEVELOPPERS. Guia para arquitetura do app. Disponível em: <https://developer.android.com/jetpack/docs/guide>. Acesso em: 04 jan. 2020.
[67] APPLE INC. ULKit | Apple Developper Documentation. Disponível em: <https://developer.apple.com/documentation/uikit>. Acesso em: 04 jan. 2020.
[68] MICROSOFT. Mobile Application Architecture Guide. Disponível em: <http://robtiffany.com/wp-content/uploads/2012/08/Mobile_Architecture_Guide_v1.1.pdf>. Acesso em: 04 jan. 2020.
[69] QI, H.; GANI, A. Research on mobile cloud computing: Review, trend and perspectives. *2012 2nd Int. Conf. Digit. Inf. Commun. Technol. its Appl. DICTAP 2012*. p. 195–202. 2012.
[70] MIZOUNI, R. *et al.* Towards battery-aware self-adaptive mobile applications. *Proc. - 2012 IEEE 9th Int. Conf. Serv. Comput. SCC 2012*. p. 439–445. 2012.
[71] MELO, EDUARDO ALVES; JAIME, PATRÍCIA CONSTANTE; MONTEIRO, C. A. *Guia Alimentar para a População Brasileira Guia Alimentar para a População Brasileira*. 2014.
[72] LEI, K.; MA, Y.; TAN, Z. Performance comparison and evaluation of web development technologies in PHP, Python and Node.js. *Proc. - 17th IEEE Int. Conf. Comput. Sci. Eng. CSE 2014, Jointly with 13th IEEE Int. Conf. Ubiquitous Comput. Commun. IUCC 2014, 13th Int. Symp. Pervasive Syst.* . p. 661–668. 2015.
[73] FOUNDATION, O. Node.Js. Disponível em: <https://nodejs.org/en/about/>. Acesso em: 28 jan. 2020.
[74] GOOGLE DEVELOPPERS. Firebase. Disponível em: <https://firebase.google.com/>. Acesso em: 29 jan. 2020.
[75] FRAMEWORK, I. Build amazing apps in one codebase, for any platform, with the web. 2012. Disponível em: <https://ionicframework.com/>. Acesso em: 18 out. 2018.
[76] FACEBOOK. Use React Native. Disponível em: <http://www.reactnative.com/>. Acesso em: 09 fev. 2020.
[77] FLUTTER-DEV. Flutter. Disponível em: <https://flutter.dev/>. Acesso em: 09 fev. 2020.
[78] LI, H.; WU, Z. Research on distributed architecture based on SOA. *Proc. 2009 Int. Conf. Commun. Softw. Networks, ICCSN 2009*. p. 670–674. 2009.
[79] SALKIND, N. “Technique for the Measurement of Attitudes, A.” *Encycl. Res. Des.* 2012.

APÊNDICE A - APLICAÇÃO DE PESQUISA

Caso de uso aplicação

1. Efetuar a leitura da dissertação, onde apresenta-se um framework para desenvolvimento de aplicações móveis sensíveis ao contexto.
2. Tentar aplicar os conceitos no caso de uso abaixo:

“Uma prefeitura decide criar um aplicativo para ajudar os pequenos produtores rurais que expõe seus produtos em feiras que acontecem em alguns pontos da cidade. Eles trabalham com frutas e legumes da estação, produtos caseiros e bebidas naturais. Sendo assim, o aplicativo também incentiva a população a comprar produtos locais, de boa procedência e trazendo sustentabilidade ao comércio local. “

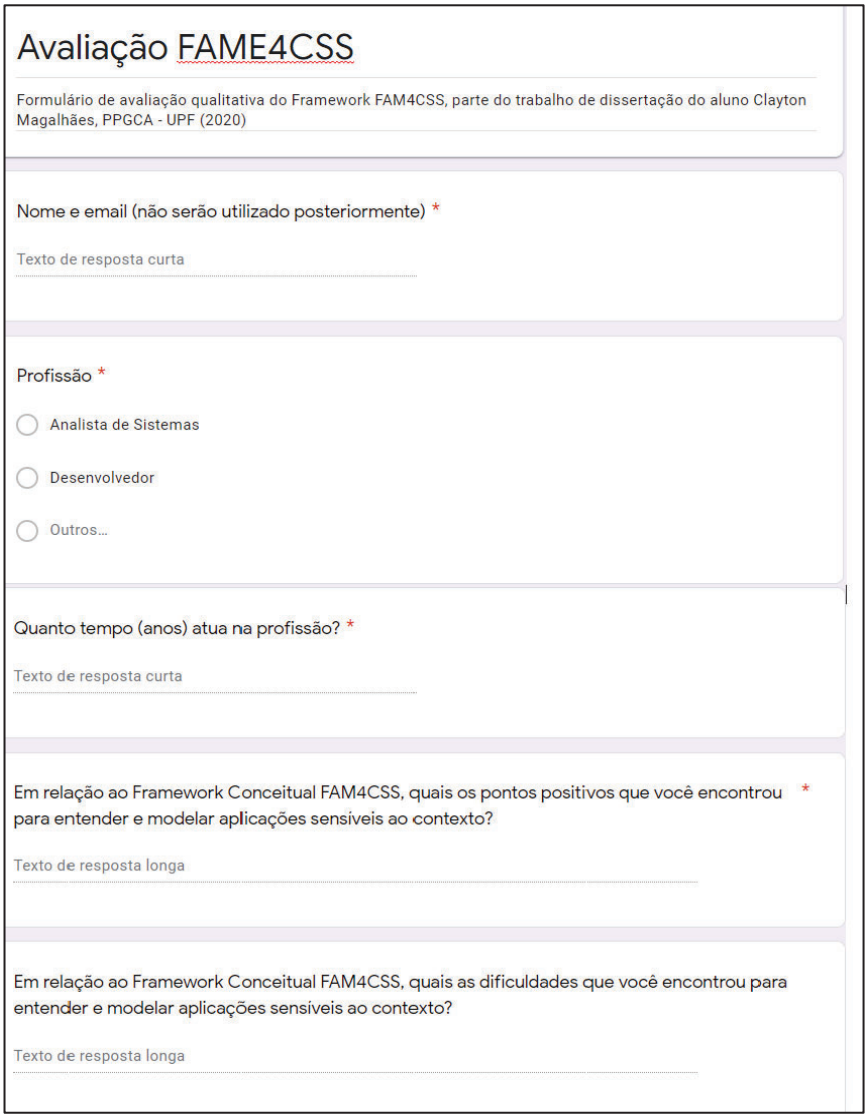
Você é o responsável pela equipe de desenvolvimento desse aplicativo, e desde já, identificou-se que pode usar o contexto como melhoria dos serviços do aplicativo, que deverá atender aos seguintes requisitos:

- O usuário baixa o aplicativo e se registra nele, informando seu nome, email, e telefone para contato.
- Após se registrar ou abrir o aplicativo posteriormente, uma tela com o mapa da cidade deve ser apresentada, sinalizando as feiras da cidade.
    - Listar as feiras em horário de funcionamento vigente mais próximas de onde ele está no momento em que abrir o aplicativo.
    - Se ele estiver em território fora da cidade, mostrar o mapa reduzido, com todas as feiras em horário de funcionamento naquele momento.
- Mesmo com o aplicativo fechado, todas as manhãs (07:00 hrs) o aplicativo deve verificar se o usuário está no local de sua casa, e caso esteja, enviar uma notificação com a seguinte mensagem: “Bom dia, lembre-se de fazer suas compras de produtos orgânicos nas feiras da cidade. Aproveite as frutas do (a).....” sinalizando no final da mensagem qual a estação atual do ano (Verão, inverno, outono e primavera). Caso o usuário não esteja em sua residência não deve enviar a mensagem via notificação.
    - Para este requisito, também deve-se verificar se a previsão do tempo é de chuva na cidade. Caso seja, as feiras não abrirão neste dia pois são ao ar livre, e, portanto, não deve enviar a notificação.
- As feiras deverão estar cadastradas em um banco de dados no servidor, com os seguintes dados: Descrição da feira, horário abertura, horário fechamento, dias de funcionamento (seg., ter., qua., etc.) e qual o endereço da feira.
- A aplicação deve verificar essas regras automaticamente e consultar as feiras, mostrando-as no mapa respectivo com seu endereço e (ou) enviando notificação aos seus usuários.

Atividades:

- Identificar entidades e seus atributos
- Modelagem de dados (estrutural e dinâmica)
- Proposta de arquitetura
- Protótipo de telas (opcional)

APÊNDICE B - QUESTIONÁRIO DE FEEDBACK FAM4CSS



Avaliação FAME4CSS

Formulário de avaliação qualitativa do Framework FAM4CSS, parte do trabalho de dissertação do aluno Clayton Magalhães, PPGCA - UPF (2020)

Nome e email (não serão utilizado posteriormente) *

Texto de resposta curta

Profissão *

- Analista de Sistemas
- Desenvolvedor
- Outros...

Quanto tempo (anos) atua na profissão? *

Texto de resposta curta

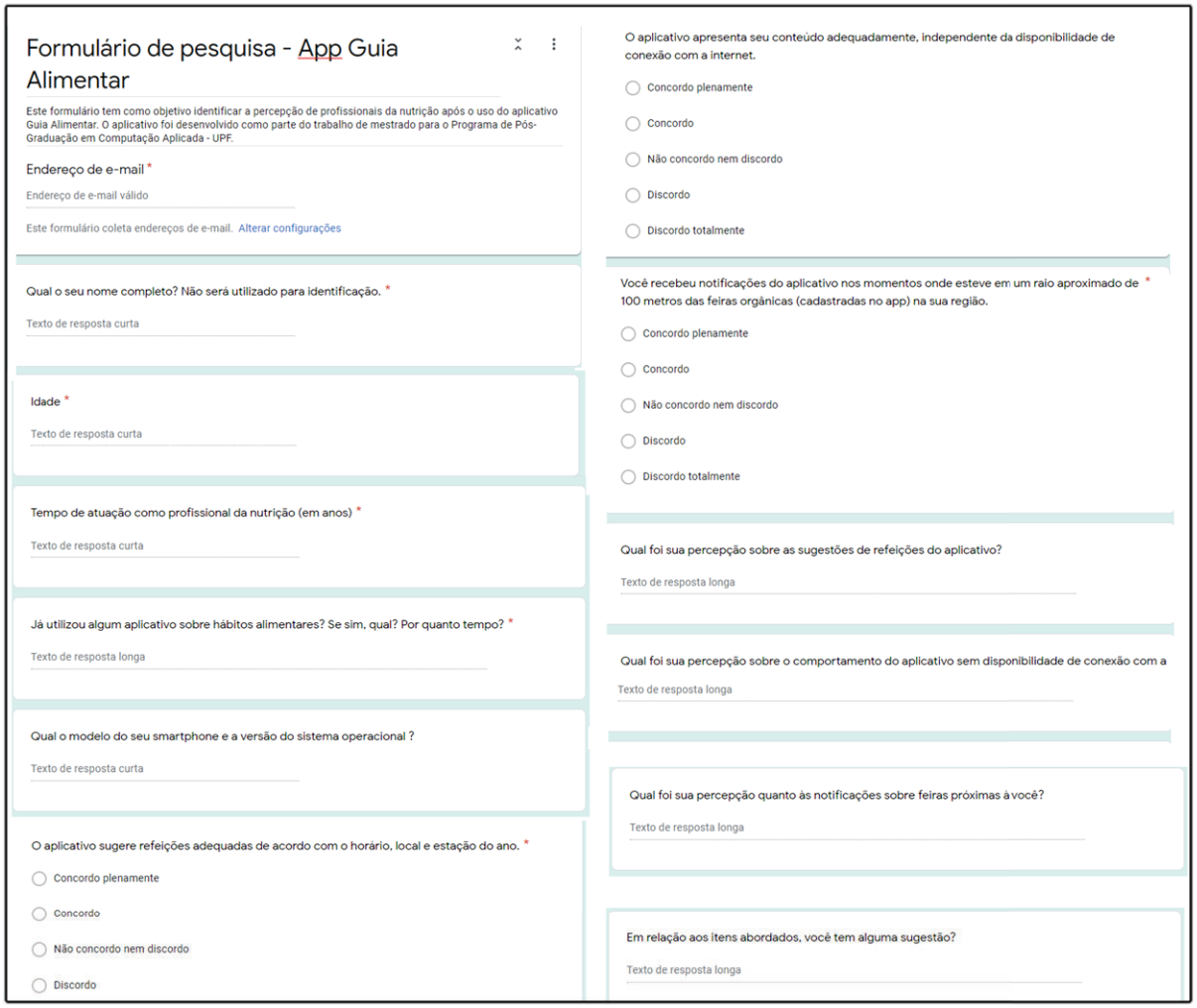
Em relação ao Framework Conceitual FAM4CSS, quais os pontos positivos que você encontrou para entender e modelar aplicações sensíveis ao contexto? *

Texto de resposta longa

Em relação ao Framework Conceitual FAM4CSS, quais as dificuldades que você encontrou para entender e modelar aplicações sensíveis ao contexto?

Texto de resposta longa

APÊNDICE C - QUESTIONARIO AVALIAÇÃO NUTRICIONISTAS



Formulário de pesquisa - App Guia Alimentar

Este formulário tem como objetivo identificar a percepção de profissionais da nutrição após o uso do aplicativo Guia Alimentar. O aplicativo foi desenvolvido como parte do trabalho de mestrado para o Programa de Pós-Graduação em Computação Aplicada - UPF.

Endereço de e-mail *

Endereço de e-mail válido

Este formulário coleta endereços de e-mail. Alterar configurações

Qual o seu nome completo? Não será utilizado para identificação. *

Texto de resposta curta

Idade *

Texto de resposta curta

Tempo de atuação como profissional da nutrição (em anos) *

Texto de resposta curta

Já utilizou algum aplicativo sobre hábitos alimentares? Se sim, qual? Por quanto tempo? *

Texto de resposta longa

Qual o modelo do seu smartphone e a versão do sistema operacional ?

Texto de resposta curta

O aplicativo sugere refeições adequadas de acordo com o horário, local e estação do ano. *

- Concordo plenamente
- Concordo
- Não concordo nem discordo
- Discordo

O aplicativo apresenta seu conteúdo adequadamente, independente da disponibilidade de conexão com a internet.

- Concordo plenamente
- Concordo
- Não concordo nem discordo
- Discordo
- Discordo totalmente

Você recebeu notificações do aplicativo nos momentos onde esteve em um raio aproximado de 100 metros das feiras orgânicas (cadastradas no app) na sua região. *

- Concordo plenamente
- Concordo
- Não concordo nem discordo
- Discordo
- Discordo totalmente

Qual foi sua percepção sobre as sugestões de refeições do aplicativo?

Texto de resposta longa

Qual foi sua percepção sobre o comportamento do aplicativo sem disponibilidade de conexão com a

Texto de resposta longa

Qual foi sua percepção quanto às notificações sobre feiras próximas à você?

Texto de resposta longa

Em relação aos itens abordados, você tem alguma sugestão?

Texto de resposta longa

APÊNDICE D - MANUAL DE USO DO APLICATIVO

Guia Alimentar

Aplicativo para uma alimentação saudável, baseado nas diretrizes do Guia Alimentar para a População Brasileira (2014) e em parceria com o Mapa de Feiras Orgânicas do IDEC.

Instalação

Acessar o link abaixo, no seu smartphone com Android.

https://play.google.com/apps/internaltest/4699160865476757655

Usuários e formas de acesso

1) Novos usuários devem efetuar o cadastro simples, informando nome, um email e uma senha (para esse app) (figura 1)
2) Se você já possui um usuário, basta acessar via botão de login, informar usuário e senha. (figura 1)
3) Após efetuar login uma vez, não será mais pedida essa informação, o aplicativo irá redirecioná-lo. diretamente à tela inicial sempre que for aberto
4) Para sair do aplicativo, usar o botão da figura. (figura 2)

Tela inicial - artigos e vídeos

1) É possível ler os artigos nos cartões da barra superior tocando neles. Os mesmos irão virar e mostrar uma prévia do artigo. Para ler o artigo por completo, tocar no botão "Ver Completo"
2) Também é possível compartilhar esse artigo em suas redes sociais, basta tocar no botão "compartilhar" como mostra a figura ao lado.
3) Na parte inferior da tela há uma playlist com muitos vídeos, onde tocando nos mesmos, irá abrir seu aplicativo do Youtube, no vídeo referido.

Tela de diário

- Nessa tela, o aplicativo irá reunir suas informações sobre localização, estação do ano, dia e horário.
- Com essas informações, será identificado e sugerido seu diário de refeições, indicando sua próxima refeição e o que o Guia Alimentar sugere para você.
- O aplicativo irá sugerir uma refeição que mais combina com você, contudo, tocando no botão SUGESTÕES você pode ver as demais refeições que você pode alternar em seu dia.
- É possível consultar alimentos e sua tabela nutricional, baseado na tabela TACO, usando o botão PESQUISAR
- A pesquisa pode ser feita por comando de voz.

Notificações sobre Refeições

- O aplicativo Guia Alimentar irá notificar seus usuários 3x ao dia lembretes de suas refeições:
  - Café da manhã
  - Almoço
  - Janta

Esses lembretes tem o intuito de salientar a importância em não pular refeições e manter a regularidade nos horários em alimentar-se.

Não será recebida mensagem, apenas uma notificação que permitirá abrir o aplicativo e visualizar seu diário.

Tela de dicas

- Possui dicas em formato de tutorial para ler e colocar em prática.
- Possui quizes com perguntas e respostas, mitos e verdades dentre outros para serem preenchidos e fixar o aprendizado do usuário.

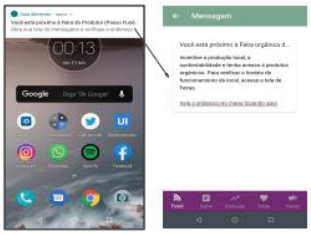
Mapa de feiras orgânicas - IDEC

- Disponibiliza o mapa de feiras orgânicas do IDEC. Nele é possível consultar as feiras cadastradas no site, além de receitas e uma biblioteca de livros sobre alimentação orgânica.
- Para cadastrar uma nova feira basta clicar em ADICIONAR LOCAL e então será pedido o formulário de preenchimento.
- Importante: Este é um recurso terceiro gentilmente cedido pelo IDEC e incorporado ao aplicativo.

Notificações sobre feiras orgânicas

- O aplicativo irá monitorar sua localização e quando o usuário estiver em um raio aproximado de 100-200 metros de uma feira orgânica, cadastrada no Mapa, irá notificar e inserir uma mensagem com o endereço, via Google Maps, para que o usuário visite essa feira.

Obs.: ainda não estão sendo controlados os dados de horário de funcionamento das feiras.

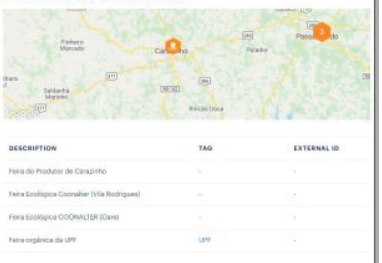


Feiras orgânicas cadastradas

As feiras cadastradas em Carazinho e Passo Fundo são as seguintes, mostradas na figura ao lado:

- Feira do Produtor de Carazinho (Praça Albino Hilebrand)
- Feira Ecológica Coonalter (Vila Rodrigues)
- Feira Ecológica Coonalter (na Gare)
- Feira Ecológica da UPF (No Campus I da UPF)

Além dessas há outras 746 em todo o Brasil, podendo serem consultadas no Mapa do App.



Tela de evolução

- Apresenta o resumo da pontuação de cada usuário, separando pelas 4 áreas de objetivo do Guia Alimentar. (planejamento, comensalidade, cozinhar e escolhas)
- Os conteúdos do Aplicativo são classificados para, quando consumidos, gerarem pontuação para o ranking.

Tabela de pontuação e locais de disparo:

- abrir/compartilhar notícia, vídeo, (feed 10 pts)
- acessar o diário (10 pts)
- finalizar tutorial (15 pts)
- quiz insere o score de acertos (30 ou 20 pts máx.)
- pesquisa alimento (10 pts)

Tela de mensagens e conversas

- Nessa tela estarão armazenadas todas mensagens recebidas, marcadas se já foram lidas ou não.
- É possível excluir as mensagens permanentemente, arrastando a mesma para o lado esquerdo, assim irá surgir o botão de apagar e confirmar exclusão como mostra a figura ao lado.
- As conversas serão em formato de chatbot, inicialmente projetados para avaliações e feedbacks pontuais com pacientes e nutricionistas.

APÊNDICE E - TEXTO SOBRE OBJETIVO DA PESQUISA

"As instruções e funcionalidades que o aplicativo possui estão em anexo em uma apresentação, bem como a forma de instalação. O download poderá ser feito em: https://play.google.com/apps/internaltest/4699160865476757655 que ficará disponível para o seu Email. O objetivo do período de testes é utilizar o aplicativo o máximo possível por um período de 7 dias, verificando a assertividade das suas funcionalidades descritas no manual (que está em anexo). Como o aplicativo está em fase de testes, seria importante verificar se há alguma inconformidade nessas funcionalidades ou no seu conteúdo apresentado. Ao final do período, será enviado um questionário para responder sobre sua percepção em relação ao aplicativo. ”

UPF
UNIVERSIDADE
DE PASSO FUNDO

UPF Campus I - BR 285, São José
Passo Fundo - RS - CEP: 99052-900
(54) 3316 7000 - www.upf.br