UNIVERSIDADE DO EXTREMO SUL CATARINENSE - UNESC

CURSO DE CIÊNCIA DA COMPUTAÇÃO

THALLES JACOBS VIEIRA

PROTÓTIPO DE APLICATIVO COM A UTILIZAÇÃO DA METODOLOGIA ÁGIL SCRUM E DO CONCEITO KANBAN PARA ACOMPANHAMENTO DO DISPÊNDIO ENERGÉTICO EM ATIVIDADES FÍSICAS

CRICIÚMA
2015

THALLES JACOBS VIEIRA

PROTÓTIPO DE APLICATIVO COM A UTILIZAÇÃO DA METODOLOGIA ÁGIL SCRUM E DO CONCEITO KANBAN PARA ACOMPANHAMENTO DO DISPÊNDIO ENERGÉTICO EM ATIVIDADES FÍSICAS

Trabalho de Conclusão de Curso, apresentado para obtenção do grau de Bacharel no curso de Ciência da Computação da Universidade do Extremo Sul Catarinense, UNESC.

Orientador: Prof. MSc. Luciano Antunes
Coorientador: Prof. Dr. Joni Marcio de Farias

CRICIÚMA
2015

THALLES JACOBS VIEIRA

PROTÓTIPO DE APLICATIVO COM A UTILIZAÇÃO DA METODOLOGIA ÁGIL SCRUM E DO CONCEITO KANBAN PARA O ACOMPANHAMENTO DO DISPÊNDIO ENERGÉTICO EM ATIVIDADES FÍSICAS

Trabalho de Conclusão de Curso aprovado pela Banca Examinadora para obtenção do Grau de Bacharel, no Curso de Ciência da Computação da Universidade do Extremo Sul Catarinense, UNESC, com Linha de Pesquisa em Engenharia de software

Criciúma, 25 de junho de 2015.

BANCA EXAMINADORA

Prof. - Luciano Antunes – MSc. - (UNESC) - Orientador

Prof. - Joni Marcio de Farias - Dr. - (UNESC) - Coorientador

Prof. Paracelso de Oliveira Caldas – MSc. - (UNESC)

Profa. Ana Claudia Garcia Barbosa - MSc. - (UNESC)

A todos que me apoiaram e me auxiliaram durante todo o meu caminho.

AGRADECIMENTOS

Em primeiro lugar agradeço a Deus, por me dar forças para percorrer todo o meu caminho, a minha namorada, Thairini Claudino Zavistanovicz, por me motivar, entender minha ausência em alguns momentos e por me ajudar em tudo que foi preciso.

Gostaria de agradecer também a toda a minha família por me apoiar e acreditar em mim, ao meu orientador Luciano Antunes e coorientador Joni Marcio de Farias pelo conhecimento passado a mim e por toda a atenção dedicada ao meu trabalho.

Por fim, agradeço a todos os meus amigos, em especial ao Leonardo Alves Neuwald por disponibilizar algumas dicas para o desenvolvimento de todo o TCC, e a todos que torceram por mim

.

RESUMO

O mercado de produções de aplicativos para dispositivos móveis vem avançando constantemente nos últimos anos, junto a este avanço evidencia-se a necessidade de métodos e ferramentas que facilitem a organização e proporcionem qualidade no desenvolvimento das aplicações. Desse modo, o presente trabalho objetivou aplicar a metodologia ágil *Scrum* em conjunto com o conceito *Kanban* a fim de desenvolver um protótipo de aplicativo de monitoramento do dispêndio energético em atividades físicas, baseado no sistema operacional Android. Foram estudados os métodos *Kanban* e *Scrum*, mesclando-os e aplicando as regras de cada um conforme as necessidades para a criação do projeto. Deste modo, tornou-se necessário a formação de uma pequena equipe para aplicar o *Scrum*, composta por dois docentes e um acadêmico. Iniciou-se a aplicabilidade das regras do *Scrum*, utilizando o *Kanban* em conjunto quando necessário, possibilitando melhor organização, qualidade e agilidade no desenvolvimento das etapas. Sendo assim foi realizada uma reunião inicial para definir todas as funcionalidades e pontos a serem produzidos, posteriormente, efetuaram-se *Sprints* para produzir todas as fases do projeto de forma separada e organizada, modelando o fluxo de informações, o banco de dados, as telas, funcionalidades e demais necessidades encontradas. Ao final de cada Sprint foram realizados testes e feitas reuniões com o intuito de analisar se a etapa concluída foi produzida da forma com que foi estabelecida inicialmente. A utilização destes métodos permitiu controlar todos os passos da criação de um aplicativo, proporcionando ao desenvolvedor implementar cada etapa com um tempo de demanda suportado pelo mesmo, além de contribuir para que não ocorressem interrupções, possibilitando obter um foco maior na qualidade e assim produzir um protótipo, que monitore o dispêndio energético, de forma organizada. Com isto, pode ser concluído que a utilização conjunta dos métodos obteve sucesso, permitindo uma visibilidade física das etapas, um desenvolvimento ágil, qualitativo e consequentemente, organizado.

Palavras-chave: Dispêndio energético. Metodologias ágeis. Android. Kanban. Scrum.

ABSTRACT

The market applications productions for mobile devices has been advancing steadily in recent years, along with this advance highlights the need for methods and tools that facilitate the organization and provide quality in application development. Thus, this study aimed to apply the Scrum agile methodology in conjunction with the Kanban concept to develop a monitoring application prototype of energy expenditure in physical activities, based on the Android operating system. The Kanban and Scrum methods were studied, mixing them and applying the rules of each as needed to create the project. Thus, it has become necessary to form a small team to apply Scrum, comprising two teachers and an academic. Began the applicability of the rules of Scrum, using the Kanban together when needed, enabling better organization, quality and agility in the development of the steps. Thus it was held an initial meeting to set all the features and points to be produced later, if effected-Sprints to produce all phases of separate and organized fashion design, modeling the flow of information, the database, screens , functionality and other needs met. At the end of each sprint tests and made meetings were held in order to analyze whether the complete stage was produced the way it was originally established. The use of these methods allowed to control every step of creating an application, providing the developer to implement each step with a demand supported by the same time, and contribute so that no interruptions occur, making it possible to obtain a greater focus on quality and thus produce a prototype, which monitor energy expenditure, in an organized manner. With this, it can be concluded that the combined use of the methods succeeded, allowing visibility of a physical steps, developing a fast, qualitative and thus organized.

Keywords: Dispêndio energético. Metodologias ágeis. Android. Kanban. Scrum.

LISTA DE FIGURAS

Figura 1 - Aumento do uso dos dispositivos móveis no mundo. 17
Figura 2 - Dispositivos móveis. 19
Figura 3 - iPad Air 20
Figura 4 - Iphone Apple 21
Figura 5 - Versões do Android. 24
Figura 6 - Gráfico da distribuição das versões. 25
Figura 7 - Arquitetura do Android. 26
Figura 8 - Estrutura da camada de Framework do Andoird. 27
Figura 9 - Arquitetura da camada de biblioteca do Android. 28
Figura 10 - Emulador do Android. 32
Figura 11 - Engrenagem do Scrum. 40
Figura 12 - Quadro visual Kanban. 42
Figura 13 - Modelagem do fluxo de informações do aplicativo. 56
Figura 14 - Diagrama de Caso de Uso da aplicação. 59
Figura 15 - Wireframe da tela principal. 61
Figura 16 - Wireframe da tela de cadastro de dados. 61
Figura 17 - Wireframe da tela de cálculo das calorias. 62
Figura 18 - Wriframe das atividades. 63
Figura 19 - Modelagem do banco de dados. 64
Figura 20 - ADT plugin no Eclipse. 67
Figura 21 - Classes com as tabelas do banco. 68
Figura 22 - Quadro visual Kanban inicial da segunda Sprint. 69
Figura 23 - Quadro visual Kanban com layouts finalizados. 70
Figura 24 - Quadro visual Kanban inicial da terceira Sprint. 74

LISTA DE TABELAS

Tabela 1 - Classificação do IMC.....47
Tabela 2 - Calorias dos alimentos por quantidade. 48
Tabela 3 - MET de cada atividade física. 50

LISTA DE ABREVIATURAS E SIGLAS

AVD	Android *Virtual Devices*
CSS	*Cascading Style Sheets*
DVM	Máquina Virtual *Dalvik*
IDC	*International Data Corporation*
IDE	*Integrated Development Environment*
IMC	Índice de Massa Corporal
JVM	Java *Virtual Machine*
Kcal	Quilocaloria
MET	Equivalentes Metabólicos
OHA	*Open Handset Alliance*
SDK	*Software Development Kit*
SGBD	Sistema de Gerenciamento de Banco de Dados
SGML	*Standard Generalized Markup Language*
SO	Sistema Operacional
XML	*Extensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO 11
1.1 OBJETIVO GERAL 13
1.2 OBJETIVOS ESPECÍFICOS 13
1.3 JUSTIFICATIVA 13
1.4 ESTRUTURA DO TRABALHO 16
2 DISPOSITIVOS MÓVEIS 17
2.1 TIPOS DE DISPOSITIVOS MÓVEIS 18
2.1.1 Tablets 19
2.1.2 Smartphones 20
3 SISTEMA OPERACIONAL ANDROID 22
3.1 ARQUITETURA DO SISTEMA ANDROID 25
3.1.1 Aplicações 26
3.1.2 Framework 26
3.1.3 Bibliotecas 28
3.1.4 Android Runtime 29
3.1.5 Linux Kernel 30
3.2 SDK ANDROID 30
3.3 EMULADOR DO ANDROID 31
3.3 IDE ECLIPSE 32
3.3 JAVA 33
3.3 XML 33
3.4 BANCO DE DADOS SQLITE 34
4 ENGENHARIA DE SOFTWARE 36
4.1 METODOLOGIASE CONCEITOS ágeis 36
4.1.1 Método ágil *Scrum* 37
4.1.2 Kanban 40
4.1.3 Scrum com Kanban 45
5 ATIVIDADES FÍSICA 46
5.1 ÍNDICE DE MASSA CORPORAL 46
5.2 QUILOCALORIAS 47
5.3 DISPÊNDIO ENERGÉTICO 48
6 TRABALHOS CORRELATOS 51

6.1 SCRUM EM APLICAÇÕES MÓVEIS 51

6.2 MÉTODO ÁGIL APLICADO AO DESENVOLVIMENTO DE SOFTWARE CONFIÁVEL BASEADO EM COMPONENTES..... 51

6.3 DIABETES CONTROL: UMA APLICAÇÃO MOBILE APLICADA AO GERENCIAMENTO DE INFORMAÇÕES MÉDICAS REFERENTES AO CONTROLE DO DIABETES 52

6.4 IMPACTO DAS ATIVIDADES FISICAS NOS INDICADORES DE SAÚDE DE SUJEITOS ADULTOS: PROGRAMA MEXA-SE 52

6.5 UM MODELO DE GERENCIAMENTO DE PROJETOS BASEADO NAS METODOLOGIAS ÁGEIS DE DESENVOLVIMENTO DE SOFTWARE E NOS PRINCÍPIOS DA PRODUÇÃO ENXUTA..... 53

7 CALORIAS CONTROL – PROTÓTIPO DE APLICATIVO PARA ACOMPANHAMENTO DO DISPÊNDIO ENERGÉTICO COM UTILIZAÇÃO DAS TÉCNICAS SCRUM E KANBAN 54

7.1 METODOLOGIA..... 54

7.1.1 Aplicabilidade do *Scrum* e *Kanban*..... 55

7.1.2 Definição dos papéis no *Scrum* 55

7.1.3 Reunião de definição do *Product Backlog* 56

7.1.4 Desenvolvimento da primeira *Sprint* 57

7.1.5 Desenvolvimento da segunda *Sprint*..... 66

7.1.6 Desenvolvimento da terceira *Sprint*..... 72

7.2 RESULTADOS OBTIDOS 76

7.3 DIFICULDADES ENCONTRADAS 78

CONCLUSÃO 79

REFERÊNCIAS 80

1 INTRODUÇÃO

O crescimento do mercado de dispositivos móveis vem se mostrando surpreendente nos últimos anos, graças à rápida evolução da tecnologia neste setor. Deste modo existe a necessidade de explorar ao máximo os recursos da tecnologia móvel, para que seja possível alimentar as exigências do crescente número de usuários (VETTORAZZI; SCHUTZ, 2013).

Junto a este significativo aumento no mercado consumidor eclode a evolução dos sistemas operacionais, em especial o Android, com um vasto número de recursos já integrados o qual tem se mostrado muito eficiente.

A plataforma de desenvolvimento Android está ganhando atenção no mercado consumidor, pois é uma opção de código aberto que oferece solução completa com sistema operacional, *middleware* e ferramentas de desenvolvimento para dar suporte aos desenvolvedores de aplicações móveis. Além disso, segundo o mesmo autor, o Android utiliza uma máquina virtual customizada para uso eficiente dos recursos físicos do dispositivo móvel (SALABERRIA, 2011).

Além destas ferramentas existem outros métodos que podem ser utilizados para aperfeiçoar o desenvolvimento destas aplicações, são as metodologias ágeis. Dentre diversos tipos procura-se uma que proporcione maior identificação com o desenvolvimento da aplicação, no caso deste o *Scrum.* Possibilitando um melhor desempenho e gerenciamento do projeto, permitindo a criação de um software em menos tempo e com maiores chances de sucesso (ETTINGER, 2012).

Com a utilização do *Scrum* permitiu-se aplicar outro conceito para obter um gerenciamento visual de cada passo dado na criação da aplicação, o método *Kanban.* Com este é possível verificar o projeto em todas as etapas e poder alocar uma nova tarefa no desenvolvimento da aplicação (GHISI, 2012).

A junção destes dois métodos se da de forma fácil, quando seguida as regras de cada, além de proporcionarem um aprimoramento no desenvolvimento dos processos, diminuindo os riscos no desenvolvimento em curtos períodos e possibilitando métodos mais eficientes e rápidos na entrega da aplicação.

Com a utilização desta metodologia híbrida é possível tornar o desenvolvimento de uma aplicação mais confiante, com maior qualidade. Tendo em

vista que podem ser retirados os pontos mais interessantes dos dois métodos para a implementação de uma aplicação é maior a garantia de solucionar problemas encontrados logo no início e produzir um software com segurança.

Aplicativos móveis, como softwares que são desenvolvidos para *smatphones* e *tablets,* possuem diversas finalidades como lazer, comunicação e entretenimento. Essas aplicações tornaram-se ferramentas importantes da mHealth, a medida que oferecem apoio remoto a pacientes ou promovem cuidados na saúde. Entende-se por mHealth práticas médicas e de saúde pública auxiliadas por aparatos portáteis, como celulares, aparelhos de monitoramento dos pacientes e outros aparelhos sem fio. De certo modo a sua principal função tem sido auxiliar políticas públicas de combate a doenças, como obesidade, tabagismo e dengue, além de estimular o usuário a manter ou iniciar práticas benéficas à sua saúde e bem-estar (BONOME et al, 2012).

Atualmente existem diversos aplicativos que auxiliam de alguma forma na vida das pessoas, sendo que muitos destes são voltados diretamente para a área da saúde e lazer, atendendo as necessidades de cada individuo (BONOME et al, 2012).

Hoje em dia o número de pessoas que pratica algum tipo de atividade física está aumentando (BRASIL, 2014). Este número vem crescendo principalmente devido à busca pela perda de peso e ganho de massa muscular, consequentemente procurando estar próximo do seu peso ideal e assim enfatizar um corpo esteticamente bonito e saudável (ARAUJO et al, 2007; Castro, 2007).

O peso adequado de um individuo pode ser calculado através da formula do índice de massa corporal, que tem como objetivo informar qual o peso mais apropriado de tal pessoa e assim auxiliar na prevenção e surgimento de doenças que podem ocorrer devido à obesidade ou sedentarismo (PIERIN, 2004; SOARES, 2008).

Para atingir ou chegar próximo desta massa corpórea as pessoas procuram fazer uma dieta regrada, baixando a quantidade de calorias ingeridas no seu dia, e ainda estarem aptas a realização de atividades físicas no tempo livre. Estas ações colaboram juntas para atingir um dispêndio energético maior que o normal, por conseguinte esta queima elevada de calorias resulta na perda de peso (SENAC, 2005).

No entanto, as pessoas praticam estes exercícios físicos sem saber qual o gasto calórico exercido. Com isto surgi à necessidade de uma tecnologia de fácil portabilidade, simples manuseio, rápido acesso e que possibilite informar aos usuários o dispêndio energético resultado em cada atividade desempenhada. Isto auxiliará no que diz respeito a motivar as pessoas a continuarem a prática de atividades físicas.

Para atender a todos os requisitos do trabalho teve-se a utilização de um conjunto de dois métodos, citados a cima, desta forma podem ser minimizados os riscos no desenvolvimento, agilizando os processos com menos foco em documentação e mais em implementação, além de permitir mudanças no decorrer do projeto e possibilitar a entrega mais frequente das atividades desenvolvidas, assim pode-se atingir um elo de confiança maior com o usuário.

1.1 OBJETIVO GERAL

Aplicar a metodologia ágil *Scrum* em conjunto com o conceito *Kanban* a fim de desenvolver um protótipo de aplicativo de monitoramento do dispêndio energético em atividades físicas, baseado no sistema operacional Android.

1.2 OBJETIVOS ESPECÍFICOS

Os objetivos desta pesquisa baseiam-se em:

a) aplicar a metodologia ágil Scrum;
b) utilizar conceito Kanban;
c) conhecer os métodos de cálculo para o dispêndio energético em atividades físicas;
d) conceber o funcionamento do sistema operacional móvel Android;
e) entender o ambiente de desenvolvimento Eclipse;
f) desenvolver um protótipo de aplicativo com a utilização da do Scrum; em conjunto com o Kanban para o sistema operacional Android;
g) simular testes com o objetivo de qualificar o presente trabalho.

1.3 JUSTIFICATIVA

De acordo com Lecheta (2013) o Android é uma plataforma de desenvolvimento para aplicativos móveis, onde possui um ambiente poderoso, ousado, flexível e com diversas aplicações já instaladas. Esta plataforma é uma resposta do Google que atinge as empresas, programadores que buscam uma plataforma moderna e ágil para desenvolver aplicações corporativas. Visando auxiliar os negócios e lucros, e os usuários, que procuram celulares modernos, com visuais elegantes, de fácil utilização e com uma diversidade de recursos.

De acordo com Macarini (2012) não existe restrição para o desenvolvimento de uma aplicação para esta plataforma, por ser *Open Source*, ou seja, código aberto. Conforme o mesmo autor, uma das melhores características do Android é que seus aplicativos podem ser desenvolvidos em Java e não estão presos a um hardware especifico, podendo estar em *smartphones*, tablets e outros dispositivos móveis.

Pelo fato do Android ser uma plataforma robusta, com muitas qualidades e vantagens, como as citadas, e ainda por possuir um layout agradável e de fácil utilização para os usuários é indispensável à criação de aplicativos nesse ambiente e da utilização de conceitos que buscam aperfeiçoar e melhorar a produção.

Um modo de melhorar essa produção foi adotando a metodologia ágil *Scrum*, onde esta proporciona agilizar e melhorar o gerenciamento de um projeto, aumentando a rapidez na entrega das atividades e possibilitando focar mais em qualidade do que na documentação. O *Scrum* teve uma identificação maior com a criação do projeto, por possuir regras que amplificam o desempenho e ser aplicado com uma equipe pequena. Para atender as necessidades da aplicação o *Scrum* foi modelado de acordo com requisitos do desenvolvimento (ETTINGER, 2012).

Outro conceito abordado para o aperfeiçoamento é o *Kanban,* que deixa explícito o andamento de um projeto, tornando visível para toda equipe. Com isto é possível gerenciar erros, atrasos e o progresso do mesmo, além de conceder a inserção tarefas no projeto sem danificar o andamento (MARIOTTI, 2012).

Para que possam ser criadas aplicações com uma percentagem maior de sucesso, com maior rapidez e melhor qualidade podem ser aplicados alguns métodos, que no caso do presente trabalho é a junção da metodologia ágil *Scrum* com o conceito *Kanban,* assim é possível visualizar e analisar todo o

desenvolvimento da aplicação, consequentemente obter um desempenho maior na implementação e produzir uma aplicação com melhor qualificação.

Desenvolvimentos de aplicações destinam-se na maioria das vezes acudir algum problema, ou seja, contribuir na ajuda de dificuldades cotidianas, auxiliando as necessidades pessoais.

Atualmente, há um denso e crescente corpo de conhecimento que consolida a atividade física como ferramenta crucial na promoção de saúde (GUALANO; TINUCCI, 2014).

Tendo em vista esta dinâmica fisiológica, a literatura científica aponta que o desenvolvimento de obesidade está fortemente associado ao “estilo de vida”, mais especificamente, no que se refere à prática de atividade física e alimentação. Nesse sentido, sabe-se que quanto mais ativo é o “estilo de vida” de uma pessoa, menor é a probabilidade desta se tornar um sujeito obeso. E quanto mais rica é a alimentação de um sujeito em açúcares, lipídeos e alimentos industrializados, maiores são as chances de este sujeito tornar-se obeso (BARBIERI; MELLO, 2012).

Como visto acima a falta de atividade física pode prejudicar rigorosamente a saúde e acarretar no surgimento de doenças. Deste modo as pessoas tendem a iniciar a prática de exercícios, buscando atingir um peso adequado ao seu conforto e sua saúde.

Conforme as informações pesquisadas, é de grande utilidade o desenvolvimento de um aplicativo, para dispositivos móveis, com a utilização de uma metodologia híbrida, assim pode-se monitorar todo o desenvolvimento da aplicação. Deste modo, as falhas podem ser encontradas no início e solucionadas logo. Com estes métodos a aplicação pode seguir seu papel com qualidade, no caso desta, informando o dispêndio energético ocorrido em cada atividade exercida pelo praticante, tornando as pessoas, que realizam atividades físicas, aptas a possuírem um controle e o conhecimento da quantidade de calorias gastas em determinadas atividades. Todavia, a utilização destes métodos proporciona um software seguro, que por consequência evidencia os usuários do mesmo a adquirem confiança no que estão usando.

1.4 ESTRUTURA DO TRABALHO

O presente estudo está dividido em sete capítulos. Sendo apresentado no primeiro capítulo a introdução, o trabalho proposto, o objetivo geral, os objetivos específicos e a justificativa da pesquisa realizada.

O segundo capítulo tem como objetivo mostrar informações sobre os dispositivos móveis e suas utilidades.

A plataforma Android, as tecnologias e ferramentas, bem como linguagens e ambiente de desenvolvimento, são apresentadas no capítulo 3.

O capítulo quatro descreve os conceitos e regras de utilização dos métodos *Scrum* e *Kanban*.

No capítulo cinco são apresentados os conceitos das atividades físicas, calorias e dispêndio energético que podem ser obtidos com atividades.

Os trabalhos correlatos são disponibilizados no sexto capítulo, com o objetivo de demonstrar alguns estudos semelhantes.

O desenvolvimento prático da pesquisa, com a utilização dos métodos, é descrito no capítulo sete.

2 DISPOSITIVOS MÓVEIS

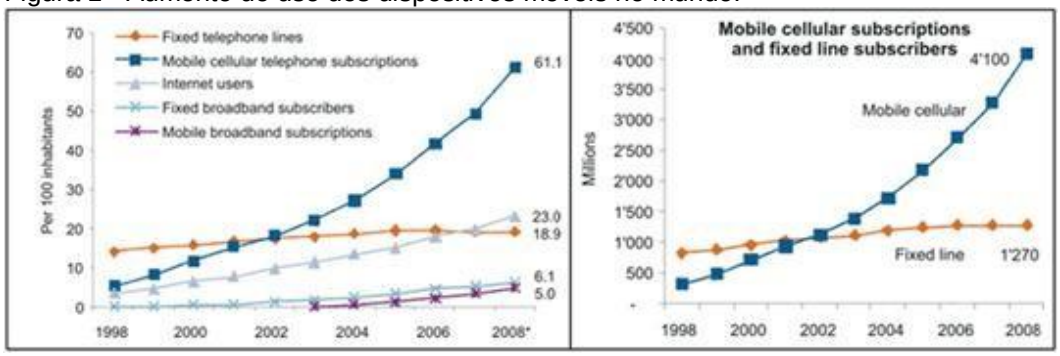
Dispositivos móveis podem ser definidos como aparelhos que tenham um tamanho reduzido, para serem transportados com maior facilidade, possuam uma capacidade de processamento e ainda permitam a troca de informações via rede. No entanto, outras características também são importantes, como apresentar uma bateria de longa duração, para que não seja necessário estar conectado a rede elétrica, interferindo na mobilidade do mesmo, e ainda ter acesso a tecnologia de rede sem fio (FIGUEIREDO; NAKAMURA, 2003).

A usabilidade destas tecnologias está aumentando devido à disponibilidade de tantos recursos. De acordo com pesquisas realizadas, com pessoas de 10 anos ou mais, em 2011, existiam 115,4 milhões de pessoas (69,1%) que tinham telefone celular para uso pessoal (IBGE, 2013).

Atualmente o celular é um dos dispositivos móveis mais requeridos e utilizados pelos usuários, tendo como origem o telefone fixo. Inventado por Alexandre Graham Bell e patenteado em 1876 o telefone era capaz de transmitir áudio entre dois pontos distintos, após esta conquista Graham Bell e seu auxiliar exploraram mais a fundo seus recursos e aperfeiçoaram cada vez mais seu invento. Consequentemente enfatizaram a evolução desta tecnologia que contribuiu com a criação dos celulares e demais dispositivos eletrônicos (COMUNICAÇÕES, 2012).

De acordo com a União Internacional de Telecomunicações (ITU, 2009) no ano de 2008 houve uma considerável mudança de telefone fixo para o uso dos dispositivos móveis, conforme pode ser visualizado na figura 1.

Figura 1 - Aumento do uso dos dispositivos móveis no mundo.



Fonte: ITU (2009).

Devido a tantas características positivas como a de possibilitar o envio de dados em movimento, não precisar estar conectado a nenhum tipo de cabo para

fornecer energia, ter acesso a redes sem fio, entre outros recursos, estes aparelhos estão se espalhando de forma acelerada.

A partir dos anos 90 foi possível identificar um grande crescimento no desenvolvimento da tecnologia móvel. Diante desta evolução obteve-se uma grande popularização destes dispositivos que passaram a permitir o acesso remoto as informações onde quer que se esteja, colaborando assim com um vasto campo de facilidades, aplicações e serviços aos usuários (FIGUEIREDO; NAKAMURA, 2003).

A área de dispositivos móveis esta ficando cada vez mais ampla, e com este crescente progresso na tecnologia os profissionais tendem aprimorar seus conhecimentos para ter maior domínio sobre as ferramentas (DARIVA, 2011).

Estes aparelhos foram criados com o intuito de fornecer algum auxilio na vida das pessoas, pois atualmente são utilizados para muitas tarefas, como despertadores, meios de acesso rápido para se ler noticias, emails entre outras facilidades disponibilizadas (DARIVA, 2011). Desta forma é importante o conhecimento sobre alguns dispositivos para verificar qual poderá suprir melhor as necessidades dos usuários.

2.1 TIPOS DE DISPOSITIVOS MÓVEIS

Esta tecnologia vem avançando de forma rápida e tornando os usuários dependentes destes aparelhos, deste modo o mercado tecnológico está sempre de portas abertas para novos inventos, procurando sempre estar com as tecnologias de ultima geração (DARIVA, 2011).

Atualmente existem vários tipos de dispositivos móveis tendo entre eles telefones celulares, *smartphones*, *Personal Digital Assistant* (PDA) e *tablets* como podem ser visualizados na figura 2, com esta diversidade de opções é sempre interessante escolher algo que possa acrescentar, de forma positiva, alguma ajuda no dia a dia, seja nos estudos, trabalho ou nas horas de lazer. Dentre estes podem ser citados alguns com maior importância e usabilidade como: *tablets e samrtphones*.

Figura 2 - Dispositivos móveis.

Fonte: Lee (2011).

2.1.1 Tablets

São considerados aparelhos digitais de forma de prancheta, maiores que os celulares, finos e que possuem uma tela sensível ao toque (*touchscreen*), além de oferecer uma ampla capacidade de armazenamento, tendo um processamento rápido, para um dispositivo de tamanho bem menor que um notebook, e de fácil portabilidade (GRUMAN, 2011). São muito utilizados para lazer e para a área profissional, onde podem ser armazenados arquivos para leitura ou para outras finalidades que o usuário ache interessante.

Entretanto, os *tablets* também passaram por uma série de modificações para serem o que são hoje, rápidos, leves e com um estilo agradável. O primeiro *tablet* lançado possuía um processador de 20 MHz e pesava aproximadamente 2 kg, criado pela empresa Grid Systems em 1989 chamava-se GRiDpad Pen Computer (DARIVA, 2011).

Os *tablets* já estavam no mercado brasileiro á algum tempo, no entanto começaram a se expandir entre a população há alguns anos. No ano de 2012 o mercado brasileiro teve um grande aumento no número de vendas de tablets, fazendo com que o país passasse de 17º lugar para 10 º lugar no ranking mundial de consumo, de acordo com a pesquisa do International Data Corporation (IDC), ainda que vendidos 3,1 milhões de aparelhos durante todo o ano o país possui um mercado de *tablets* 12 vezes menor que o norte-americano, o maior consumidor do mundo (OLIVEIRA; PAULINO, 2013).

Segundo David (2011) os *tablets* apresentam inúmeras aplicações, com isso, estão sendo muito procurados no mercado e uma das marcas mais solicitadas é a *Apple*, com o seu *iPad,* representado na figura 3, que sempre disponibiliza tecnologias inovadoras.

Figura 3 - iPad Air

Fonte: Apple (2014).

2.1.2 Smartphones

Os telefones celulares podem ser descritos como dispositivos criados com o objetivo de transmitir voz entre dois pontos, assim como os telefones fixos, porém com a vantagem de serem transportados para qualquer lugar. Todavia estes equipamentos não ficaram apenas neste pequeno invento, com o passar dos anos eles foram evoluindo, assim como outras tecnologias, mudando sua estética e suas funções internas, o que era antes apenas para se comunicar passou a abranger modernidade em seu layout e varias funcionalidades.

Os *smartphones* são dispositivos móveis formados pela combinação de celulares e PDAs, permitindo conectarem-se a web através de conexões 3G ou Wi-fi (MORIMOTO, 2009). Além destes, eles possuem outros recursos incluídos como tirar fotos, gravar vídeos, enviar mensagens de texto e reproduzir músicas, mas o que torna os mesmos telefones inteligentes (*smatphones*) é possuir um Sistema

Operacional (SO) que possibilita o próprio usuário criar aplicativos novos para usufruir em seu dispositivo (ILYAS; AHSON, 2006).

Existem hoje diversas empresas fabricantes de *smartphones* no mercado, algumas como: LG, Samsung, Lenovo, Motorola, Apple entre outras. No entanto a Apple lançou no mercado, em 2007, o seu *Iphone,* representado na figura 4, onde o mesmo possuía um teclado virtual, apenas com toque na tela e uma interface com conceitos que são utilizados atualmente pela grande maioria das empresas (MORIMOTO, 2011).

Figura 4 - Iphone Apple

Fonte: Morimoto (2011).

3 SISTEMA OPERACIONAL ANDROID

O Android é primariamente um esforço da Google, em colaboração com a Open Handset Alliance (OHA), uma aliança de dezenas de organizações, mais precisamente 84 empresas, que se comprometeram em disponibilizar no mercado um telefone celular *open-source* e com um maior número de vantagens. Em poucos anos o Android cresceu e foi capaz de mudar o mercado, ganhando o respeito como escárnio dos colegas de indústria (ABLESON et al, 2012).

O Android é um SO móvel baseado em uma versão modificada do Linux. Foi originalmente desenvolvido por uma equipe com mesmo nome, Android, Inc. Em 2005 a Google comprou o Android e assumiu o seu trabalho de desenvolvimento (bem como sua equipe de desenvolvimento), isso fazia parte de uma estratégia para entrar no espaço de dispositivos móveis (LEE, 2011).

A Google queria que o Android fosse *open-source* e software livre, portanto, a maioria do código do Android foi liberado sob a licença Apache *opensource*, deste modo qualquer um que tivesse interesse em utiliza-lo teria fácil acesso ao *download* do código fonte completo, além disso as empresas podem adicionar suas próprias extensões e customiza-lo, diferenciando o seu produto dos demais. Este recurso despertou o interesse de muitas empresas que foram afetadas pelo lançamento do *Iphone* da *Apple*, uma criação que revolucionou a indústria dos *smartphones.* Empresas como Motorola e Sony Ericsson, que desenvolviam seu próprio SO móvel, viram no Android a solução para revitalizar seus produtos, continuando com seu estilo de modelo de hardware, mas utilizando o Android como seu SO (LEE, 2011).

Por ser a primeira plataforma de *software* que possui aplicativos para telefones celulares *open-source,* o Android vem revolucionando o mercado mundial de aplicativos para *smartphones* e sendo demasiadamente notado pelos maiores mercados de telefonia celular do globo (ABLESON et al, 2012).

Considerado uma plataforma para tecnologia móvel completa, o Android, desenvolvido com base no Linux, possui um SO *middleware*, com já citado, aplicativos instalados, uma interface com o usuário e envolve um pacote de programas para celulares. Foi construído com a intenção de permitir os desenvolvedores tirarem total proveito do aparelho, desta forma pode-se criar aplicações que acessem qualquer funcionalidade do núcleo do sistema, tais como

efetuar chamadas, enviar mensagens ou qualquer possível atividade que o dispositivo venha a exercer. Com isto a tecnologia do Android sempre estará evoluindo, pois disponibiliza total controle do seu código fonte aos desenvolvedores, de tal modo que os mesmos sempre estarão trabalhando em cima do sistema incorporando novos recursos e construindo aplicações móveis inovadoras (PEREIRA; SILVA, 2009).

Com o grande número de vantagens concedidas por esta plataforma é de suma importância conhecer algumas partes essenciais que a compõem.

Segundo Meier (2010) as partes necessárias e dependentes que o Android é composto incluem o seguinte:

a) um projeto de referência hardware, que descreve as capacidades necessárias para um dispositivo móvel apoiar a pilha de software;
b) o kernel do sistema operacional Linux, que fornece interface de baixo nível com o hardware, gestão de memória e controle de processos, todos otimizados para dispositivos móveis;
c) bibliotecas de código aberto para desenvolvimento de aplicações, incluindo SQLite, WebKit, OpenGL, e um gerenciador de mídia;
d) um tempo de execução usado para executar e hospedar aplicativos do Android, incluindo uma máquina virtual chamada *Dalvik* e as bibliotecas centrais que fornecem funcionalidades específicas para Android;
e) uma estrutura de aplicativo que expõe serviços do sistema para a camada de aplicação, incluindo o gerenciador de janelas e locação, provedores de conteúdo, telefonia e sensores;
f) um framework de interface de usuário usado para hospedar e lançar aplicações;
g) aplicativos pré-instalados enviados como parte da pilha;
h) um kit de desenvolvimento de software usado para criar aplicativos, incluindo ferramentas, plug-ins, e documentação.

Estes componentes fundamentais da plataforma foram evoluindo com cada versão aprimorada do Android, tornando-se mais eficientes na criação de aplicativos (GOOGLE, 2014).

Atualmente existem diversas versões do Android no mercado, desde as mais simples até as mais modernas. A plataforma esta sempre em constante evolução, fornecendo aos usuários novos benefícios a cada aprimoramento ou lançamento de uma nova versão (GOOGLE, 2014). Na figura 5 é representada a versão, o apelido (*Codename*), o nível da *Application Programming Interfac* (API) e o percentual de distribuição.

Figura 5 - Versões do Android.

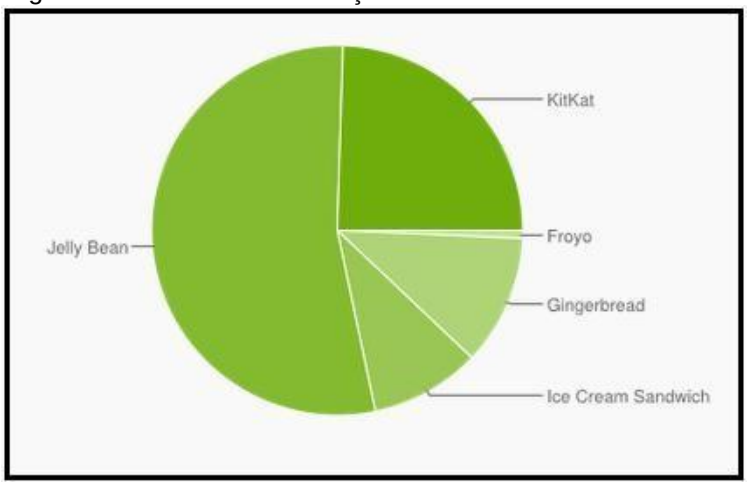
Versão	Codename	API	Distribuição
2.2	Froyo	8	0,7%
2.3.3 - 2.3.7	Gingerbread	10	11,4%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	9,6%
4.1.x	Jujuba	16	25,1%
4.2.x		17	20,7%
4.3		18	8,0%
4.4	KitKat	19	24,5%

Fonte: Google (2014).

A figura 5 representa o percentual de um número de dispositivos Android que acessaram a loja de aplicativos da Google (Google Play Store) durante um período de sete dias, que foi encerrado 9 de setembro de 2014. Como a Google *Play Store* suporta apenas a versão 2.2 e superiores as inferiores não estão incluídas (GOOGLE, 2014).

Na figura 6 é possível visualizar a distribuição de cada versão do Android de acordo com o seu apelido. Estes dados foram coletados no mesmo tempo que os da figura 5.

Figura 6 - Gráfico da distribuição das versões.



Fonte: Google (2014).

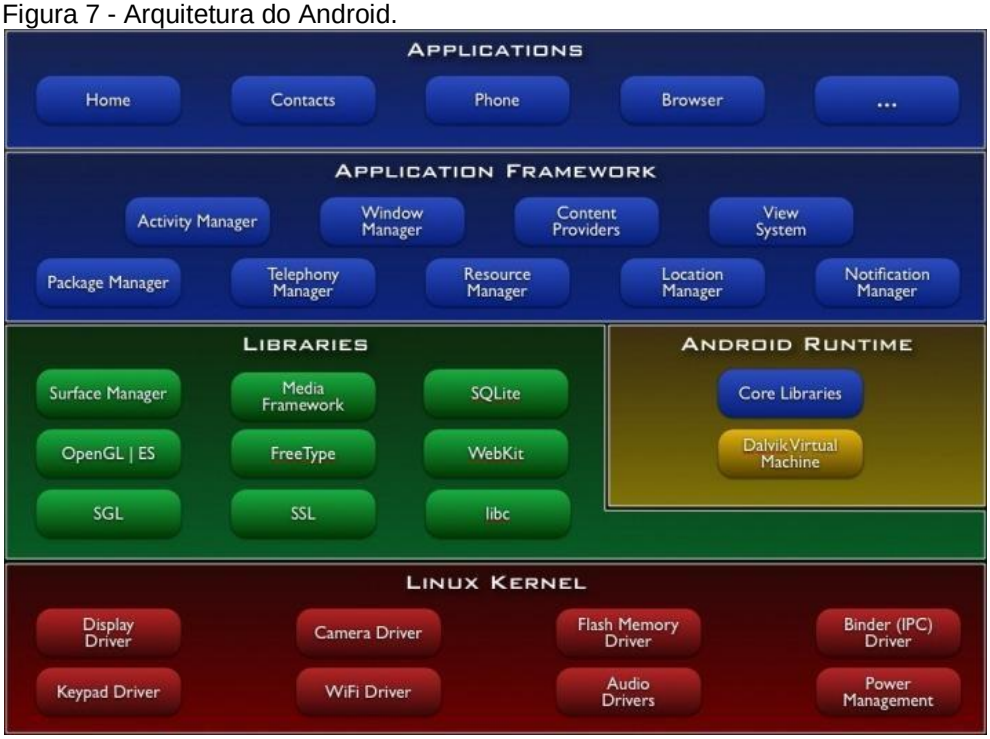
Como pode ser visto a versão com maior distribuição é a *Jelly Bean* seguido pela versão *KitKat.*

3.1 ARQUITETURA DO SISTEMA ANDROID

O Android foi desenvolvido inicialmente para *smartphones*, no entanto hoje é utilizado em diversos dispositivos. Seu SO foi baseado no *Kernel* 2.6 do Linux, responsável por realizar todo o controle da memória. Desta forma o *Kernel* consegue gerenciar processos e aplicativos, que podem ser executados simultaneamente, *threads* dos processos e a segurança dos arquivos e pastas (LECHETA, 2013).

A arquitetura do Android é formada basicamente por cinco camadas, sendo elas as seguintes: aplicações, *framework,* bibliotecas, Android runtime e linux kernel, conforme mostra a figura 7 (GOOGLE, 2014).

Figura 7 - Arquitetura do Android.



Fonte: Google (2014).

3.1.1 Aplicações

A camada de aplicações está acima de todas as outras. Nesta camada se encontram todos os aplicativos essências do Android, tais como navegadores, mapas, calendários, agendas e ainda os outros aplicativos desenvolvidos pelos programadores (PEREIRA; SILVA, 2009).

3.1.2 Framework

Frameworks são estruturas que servem tipicamente para a implementação de software de larga escala. Seu principal objetivo é possibilitar a reutilização do código e assim aumentar a produtividade durante o desenvolvimento de softwares (PEREIRA; SILVA, 2009).

Nesta camada, conhecida como *Application Framework*, são encontradas todas as APIs e recursos que são utilizados pelos aplicativos, como as classes visuais que englobam listas, grades, caixas de texto, gerenciadores de recursos, *View system,* que são componentes utilizados na construção de aplicativos, provedor de conteúdo (*Content Provider*), que possibilita que aplicações possam

compartilhar suas informações, acessar e utilizar as de outra aplicação e trocar informação entre aplicativos, entre outras vantagens (PEREIRA; SILVA, 2009).

A camada de *framework* trabalha em conjunto com a de aplicações, desta forma, fica mais fácil à reutilização dos componentes, ou seja, a troca de informação das aplicações. Com esta reutilização de código os desenvolvedores passam a ter recursos prontos para serem aplicados e consequentemente menos códigos para escreverem (JOBSTRAIBIZER, 2009). Na figura 8 é possível ver os elementos da camada de *framework.*

Figura 8 - Estrutura da camada de Framework do Andoird.

Fonte: Bray (2010).

Dentre estes *Frameworks*, podem ser citados alguns mais importantes como:

a) *activity manager*: é responsável por gerenciar o ciclo de vida de todas as *activities,* quando iniciar e quando termina-las, permitindo assim que uma *activity* consiga se deslocar para outra;
b) *package manager*: é utilizado pelo *activity manager* para se comunicar com o sistema informado sobre quais os pacotes estão sendo utilizados e sobre suas capacidades;
c) *window manager:* tem como função gerenciar a apresentação de janelas, mostrando quais janelas estarão ativas;

d) *content providers*: possibilita o compartilhamento de dados e a troca de informações entre os aplicativos;
e) *view system*: concede todo o tratamento gráfico para a aplicação, como botões, *layouts* e *frames;*

Outros elementos encontrados também nesta camada são o *Location Service, Bluetooth Service, Wi-Fi Service, USB Service* e *Sensor Service* (PEREIRA; SILVA, 2009).

3.1.3 Bibliotecas

Esta camada carrega consigo um conjunto de bibliotecas C/C++ utilizadas pelo sistema, incluídas nesse conjunto de bibliotecas C padrão. Possui também bibliotecas das áreas de multimídia, visualização de camadas 2D e 3D, funções para navegadores web, funções para gráficos, funções de aceleração de hardware, renderização 3D, fontes bitmap e vetorizadas e funções de acesso ao banco SQLite (PEREIRA; SILVA, 2009).

Estas bibliotecas, representadas na figura 9 quais são utilizadas por vários recursos do sistema, podem ser acessadas por *frameworks* disponibilizados para os desenvolvedores (PEREIRA; SILVA, 2009).

Figura 9 - Arquitetura da camada de biblioteca do Android.

Fonte: Bray (2010).

Dentre as principais bibliotecas podem ser citadas as seguintes:

a) *freetype:* para renderização de fontes e bitmaps;

b) *system C library:* sintonizada para dispositivos rodando Linux, a biblioteca padrão C *Bionic* é desenvolvida, customizada e otimizada para uso embutido no Android;

c) *webkit:* responsável por renderizar as páginas para navegadores, com suporte para *Cascading Style Sheets* (CSS), *javascript* e outros;

d) SQLite: um banco de dados poderoso relacional, leve e embutido para as aplicações do Android;

e) SGL: responsável pelos gráficos 2D;

f) *surface Manager:* permite o acesso aos subsistemas de exibição como as aplicações 2D e 3D;

g) *media libraries:* bibliotecas que suportam os formatos de áudio e vídeo;

h) *libewebcore*: web *browser* utilizado para exibições web;

i) *3D libraries*: bibliotecas que utilizam, quando disponível, aceleração 3D via hardware ou software de renderização 3D;

Além destas bibliotecas ainda existem diversas outras com mesmo nível de importância (PEREIRA; SILVA, 2009).

3.1.4 Android Runtime

Esta camada é uma instância da Máquina Virtual *Dalvik* (DVM), que é criada para cada aplicação executada no Android. A DVM possui um melhor desempenho comparada a outras, além deste dispõem de alguns outros recursos importantes, como permitir a execução simultânea de máquinas virtuais paralelamente e ter maior integração com os novos hardwares. Foi projetada para rodar em sistemas com um baixo consumo de memória, bateria e CPU, com o intuito de funcionar em sistemas mais “fracos”, ou seja, com uma CPU de baixa frequência, pouca memória RAM e SO com pouco espaço de *Swap* (PEREIRA; SILVA, 2009).

A DVM suporta um conjunto de bibliotecas do java e APIs que são completamente documentadas, abertas, e que disponibilizam o desenvolvimento de softwares. Apesar de o código Java ser escrito para a *Dalvik* usando estas bibliotecas, isto não faz a DVM uma verdadeira maquina virtual Java, apenas

suporta uma grande parte dos argumentos padrões Java por bibliotecas e APIs que são especificas do Android (JORDAN; GREYLING, 2011).

3.1.5 Linux Kernel

O Linux Kernel é encontrado no fundo da pilha do Android. O código do Linux Kernel foi tomado e modificado pelo Android para rodar embutido no seu ambiente. Porém, ele não tem todas as características e utilitários de uma distribuição Linux tradicional, deixando faltar alguns recursos. Alterações no Linux Kernel são incorporadas para uso em futuras versões do Android, estas mudanças são importantes porque muitas modificações de segurança e melhorias são feitas para serem utilizados em uma base contínua, assim os usuários obtêm o que o SO Linux tem de melhor para oferecer (DUBEY; MISRA, 2013; GOOGLE, 2014).

Esta camada utiliza o Linux Kernel para serviços centrais do sistema, como segurança, gestão de memória e processos, pilhas de protocolos de redes, modelo de drives e ainda atua como abstração entre o hardware e a pilha de software. Outra funcionalidade desta camada é o gerenciamento de energia, verificando quais dispositivos não estão sendo utilizados pelas aplicações e desligando os mesmos (PEREIRA; SILVA, 2009).

3.2 SDK ANDROID

O Software *Development Kit (*SDK) do Android é utilizado para o desenvolvimento de aplicações. Para rodar as mesmas o SDK desfruta de um emulador similar a um celular, que dispõem de ferramentas utilitárias e uma API completa para a linguagem Java. O emulador do SDK pode ser executado como um aplicativo comum, porém existe um *plug-in* para o Eclipse que possibilita a integração do ambiente de desenvolvimento com o emulador, assim é possível executa-lo diretamente de dentro do Eclipse, outro modo de testar as aplicações é conectando o celular na máquina e rodando-os dentro do próprio dispositivo, tornando o desenvolvimento mais produtivo (LECHETA, 2013).

O SDK do Android é composto por uma plataforma, ferramentas, exemplos de códigos e documentação necessária para desenvolver aplicações Android. É construído como um *add-on,* ou seja, uma extensão para o Kit de

desenvolvimento Java e tem um *plug-in* integrado para o ambiente de desenvolvimento do Eclipse (SCHWARZ et al, 2013).

3.3 EMULADOR DO ANDROID

A partir da versão 1.6 do Android SDK os desenvolvedores obtiveram um melhor controle sobre o emulador do Android, podendo fazer *downloads* das plataformas que despertam um maior interesse, o que não era possível nas versões anteriores (ABLESON, 2012).

O Android *Virtual Devices (*AVD) é a ferramenta perfeita para testar e depurar aplicativos. Uma implementação da DVM que simula os softwares construídos em emuladores de diferentes dispositivos, assim é possível testar um aplicativo em uma variedade de plataformas de hardware, sem a necessidade de comprar tais dispositivos (GOOGLE, 2014; MEIER, 2010).

Para facilitar a inicialização do emulador existe o Android *Developer Tool* (ADT) *plug-in,* que integra o emulador no ambiente de desenvolvimento Eclipse, com isto é possível inicia-lo automaticamente de dentro do Eclipse. Cada AVD criado é configurado com um nome e com a versão SDK requerida, onde é determinada a capacidade de cartão SD, resolução de tela ou qualquer característica da versão selecionada. Como o emulador simula um telefone com uma ótima realidade, conforme mostra à figura 10, ele além de possuir o poder de testar as aplicações criadas pode fazer simulações com os próprios aplicativos integrados na versão escolhida, como mensagens de texto, chamadas de voz, entre outros (MEIER, 2010).

Figura 10 - Emulador do Android.

Fonte: Google (2014).

3.3 IDE ECLIPSE

O Eclipse é uma *Integrated Development Environment* (IDE) de código aberto com o objetivo de construir programas. Sua primeira versão foi construída pela IBM, que o disponibilizou depois como software livre para a sociedade. Utiliza a linguagem Java e possui um forte desenvolvimento baseado em *plug-ins,* fornecendo funcionalidades de desenvolvimento de uma forma integrada*,* conseguindo assim atender as necessidades dos programadores. Com amplas características positivas é uma das IDEs mais utilizadas no mundo (SERSON, 2007).

A maioria dos desenvolvedores utiliza a IDE do Eclipse, para a criação de aplicativos para Android, pelo fato do SDK possuir *plug-ins* que são integrados com o Eclipse e disponíveis por SO como windowns, Mac e Linux (CINAR, 2012; DARCEY; CONDER, 2012).

Plug-ins são as menores unidades da plataforma Eclipse. Eles possuem um código estruturado que contribui para um conjunto de funcionalidades, podem ser desenvolvidos, distribuídos e implantados individualmente. A plataforma Eclipse permite que os *plug-ins* possam ser extensíveis, por meio de APIs, assim outros *plug-ins* podem expandir seus recursos (CINAR, 2012).

3.3 JAVA

Java é uma linguagem de programação desenvolvida pela empresa americana Sun Microsytems. Sua primeira versão foi lançada em 1995, com o passar do tempo foi evoluindo e aperfeiçoando sua estrutura e recursos, tornando-se uma plataforma completa, apta a resolver os problemas da programação da atualidade, sendo utiliza em diversas produções de programas corporativos, alguns como: processamento científico, jogos on-line, programas de educação, entre outros (COSTA, 2008).

Esta linguagem, simples e de fácil utilização, proporciona inúmeros pacotes adicionais para diversas funcionalidades exercidas, como pacotes para janelas gráficas. Uma das suas principais vantagens é a portabilidade, podendo ser executada em diferentes plataformas, necessitando apenas de pouca ou nenhuma alteração no seu código, deste modo para disponibilizar essa portabilidade e executar os programas o Java aplica os processos de compilação e interpretação, desfrutando do seu interpretador, embutido em um software chamado Java *Virtual Machine* (JVM), que é específico para cada plataforma, assim um programa compilado em Java pode ser executado em outro SO, como Windowns e Linux (COSTA, 2008).

O Java é uma das plataformas mais importantes do mundo, estando presente em vários ramos da tecnologia, dispondo de amplos pontos positivos é uma das principais linguagens de desenvolvimento para dispositivos móveis, incluindo para o Android (SILVEIRA et al, 2012).

3.3 XML

O *Extensible Markup Language* (XML) é uma linguagem de marcação, foi desenvolvida para fornecer uma alternativa para a *Standard Generalized Markup Language* (SGML). Antes do XML, SGML era a única meta linguagem disponível para o desenvolvimento de outras linguagens de marcação. Com esta precariedade o *World Wide Web* desenvolveu o XML que preserva as melhores características do SGML, mas filtra todos os aspectos que não são necessárias em um ambiente de publicação na web (HOLZNER, 2001; LADD et al, 2001).

No Android a linguagem XML é principalmente utilizada no *layout* dos aplicativos, possibilitando a criação de desenhos simples com arquivos XML, enfatizando a tela com os devidos objetos necessários, criando botões, campos e o que for essencial para a aplicação. Uma das razões de usar o XML é a simplicidade de manipulação, onde elementos gráficos desenvolvidos podem ser modificados futuramente por meio de programação, assim não tornam o projeto inicial relativamente estático (ABLESON, 2012).

3.4 BANCO DE DADOS SQLITE

A utilização de banco de dados em aplicações é consideravelmente importante hoje em dia, quando há a necessidade de salvar informações tanto do usuário quanto da própria aplicação este tipo de armazenamento entra em ação, disponibilizando uma maior segurança e facilidade no que diz respeito à manipulação destes registros (JOBSTRAIBIZER, 2009).

SQLite é um banco de dados leve e poderoso que possui integração com o Android, permitindo a utilização do mesmo no desenvolvimento de softwares O*pen source*, com um banco de dados relacional incorporado, o SQLite, lançado em 2000, foi projetado para gerenciamento de dados das aplicações sem uma sobrecarga, que muitas vezes vem com sistemas dedicados de gerenciamento de banco de dados relacionais. Possui uma alta e merecida reputação por ser altamente portátil, fácil de usar, compacto, eficiente e confiável (ALLEN; OWENS, 2010; LECHETA, 2013).

Criado em C, o SQLite é uma biblioteca recomendada onde haja programas de simples implementação, administração e manutenção. Quando o SQLite é usado não se da necessária a execução do processo Sistema de Gerenciamento de Banco de Dados (SGBD) (BESSA, 2014).

Diferentemente da maioria dos outros bancos de dados SQL, o SQLite não tem um processo servidor separado. Lê e escreve diretamente para arquivos de disco comuns. Além de ser um banco de dados SQL completo com várias tabelas, índices, *triggers* e *views* o SQLite possui diversas características como: transações atômicas e consistentes, não necessita de configuração ou administração, possui implementado a maioria das vantagens SQL92, suporta banco de dados de

terabytes, é uma API simples e fácil de ser usada, pode ser o único arquivo de código fonte escrito em ANSI-C, que tem facilidade na compilação e consequentemente uma simplicidade ao ser adicionado a um projeto maior, não possui dependências externas, portabilidade acessível para outros sistemas, por ser *open source* pode ser utilizado para qualquer finalidade, entre outros vários recursos (SQLITE, 2010).

4 ENGENHARIA DE SOFTWARE

Segundo Pressman (2011) a engenharia de software consiste em diversos princípios, métodos, conceitos e ferramentas que são recorridos no planejamento e desenvolvimento de um software. A utilização desta prática permite que a criação de programas computacionais seja mais organizada, proporcionando seguir um roteiro que pode detalhar cada etapa e assim obter uma chance maior de sucesso.

Engenharia de software pode ser definida como uma disciplina atingida no desenvolvimento de software. Esta tem como objetivo buscar uma solução para o despreparo no planejamento de projetos. Para ser atingida esta disciplina são utilizadas diversas combinações como: controle, administração, métodos que abrangem todas as fases do desenvolvimento de um software, ferramentas para melhorar os processos e técnicas que garantem a qualidade do software (PRESSMAN, 2011).

A Engenharia de Software tem como visão criar produtos de software utilizando conhecimentos científicos, técnicos, gerenciais e empíricos. Deste modo procuram atender as necessidades e requisitos de pessoas e instituições, produzindo softwares com uma variedade de características já citadas, quando praticada por profissionais treinados e bem informados (PRESSMAN, 2011).

4.1 METODOLOGIAS E CONCEITOS ÁGEIS

Atualmente é crescente o número de fabricantes de dispositivos móveis e do avanço tecnológico, com isto eclode uma exigência de aplicações que suportem esses dispositivos, aumentando as fabricações destas aplicações, que se tornam cada vez mais inteligentes. No entanto, o avanço destas proporciona uma busca por técnicas e métodos que melhorem as aplicações, onde podem ser citadas as metodologias ágeis, capazes de melhorar e aperfeiçoar os processos no desenvolvimento destes softwares. Estes métodos buscam implementações em curtos períodos, minimizando os riscos e tornando o desenvolvimento eficiente e rápido (JANDOZA; PRADO, 2013).

4.1.1 Método ágil *Scrum*

Scrum é considerado um *framework* utilizado para gerenciar o desenvolvimento ágil de softwares, no qual se podem aplicar diversas técnicas ou processos, podendo ser usado para identificar o que é necessário ser feito para desenvolver software de qualidade em pouco tempo. Desta forma pode-se melhorar a eficácia das práticas de gerenciamento e desenvolvimento destes softwares (PEREIRA; TORREÃO; MARCAL, 2007). Além destas características o Scrum possui ótimas práticas de gerenciamento, estas, por sua vez, aceitam ajustes rápidos, acompanhamento e visibilidade constantes (FONSECA; CAMPOS, 2008).

Projetos que utilizam o método Scrum são compostos por equipes pequenas que desenvolvem suas atividades a partir de requisitos ou do *Product Backlog,* explicado mais a baixo*.* O desenvolvimento do projeto é dividido em ciclos chamados de *Sprint*, este organiza o desenvolvimento de software em iterações de 2 a 4 semanas (ETTINGER, 2012).

Existem alguns conceitos do *Scrum* que possibilitam um melhor entendimento sobre o funcionamento do mesmo, além de papéis importantes dentro do projeto.

Papéis e responsabilidades do *Scrum:*

a) *scrum team*: é composto por *Product Owner*, equipe de desenvolvimento e o *Scrum Master*. Buscam determinar as funcionalidades do produto, analisando, desenvolvendo e testando os itens de cada *Sprint.* Por ser auto-organizável e gerenciável não depende de gestores para tomada de decisões;
b) *product owner*: tem como responsabilidade gerenciar o *Backlog* do projeto, ou seja, definir os requisitos iniciais e gerais, objetivos e planos de entregas. Solicita a priorização dos itens e funcionalidades com maior importância para o projeto em cada *Sprint.* Respeitar ao *Product Owner* é fundamental para o sucesso no desenvolvimento;
c) *scrum master*: responsável por coordenar a equipe de desenvolvimento e garantir que a rotina imposta pelo *Product owner* está sendo seguida, além de repassar o conhecimento sobre o *Scrum* para a equipe envolvida e assegurar que as práticas do mesmo estão sendo

exercidas. Assim permiti uma chance maior de sucesso no final do desenvolvimento (PEREIRA; TORREÃO; MARCAL, 2007).

Após o conhecimentos de papéis dentro do *Scrum* podem ser vistos os conceitos, artefatos e fases do *Scrum*:

a) *product backlog*: é uma lista de tudo o que foi especificado para ser desenvolvidas no projeto inteiro, prioridades de itens, requisitos e tudo que envolve o projeto como um todo, deste modo deve ser definido o mais detalhado possível;
b) *sprint backlog*: são itens do *Product Backlog* que serão desenvolvidos em uma sprint. Assim podem ser definidas as funcionalidades que serão implementadas de forma separada, sendo desenvolvida incrementalmente, relacionando ao *backlog* anterior;
c) eventos *scrum*: são reuniões pré–definidas, com a finalidade de impossibilitar que ocorram reuniões desorganizadas e sem prazos;
d) *sprint*: é uma repetição onde novas funcionalidades são desenvolvidas e acrescentadas ao projeto. Este é o diferencial do *Scrum*. A duração do *Sprint* é de até 30 dias, cada vez que um *Sprint* é finalizado outro começa automaticamente com novas funções para serem implementadas. Ao decorrer do período do *Sprint* não são feitas alterações que afetem o propósito final do mesmo, desta forma os riscos de complexidade e entrega final diminuem. O *Sprint* pode ser antecipado ou cancelado, porém apenas o *Product owner* tem este poder;
e) *daily meeting*: reunião diária feita com duração de no máximo 15 minutos, com o objetivo de avaliar o progresso do projeto, organizar as tarefas do dia e verificar as que foram feitas no dia anterior, mencionando problemas encontrados no desenvolvimento;
f) revisão do *sprint*: esta é feita no final dos *Sprints*, assim pode-se analisar se o projeto final saiu como proposto no início, os pontos fundamentais para a conclusão do mesmo e os pontos negativos ocorridos, tendo como responsável pela avaliação o *Product Owner* (ETTINGER, 2012).

Depois de conhecer os conceitos e responsabilidades do *Scrum* de uma forma mais geral podem ser vistas as regras de trabalho funcionando de uma forma conjunta, respeitando os conceitos do *Scrum* para que o projeto seja feito de forma correta.

O *Scrum* funciona de uma forma que proporciona respostas rápidas a mudanças. Primeiramente é feita uma análise do problema existente, após isso é construído o *Product BackLog* ou *Backlog* do produto, onde serão listadas todas as funcionalidades, requisitos e itens a serem desenvolvidos, priorizando as partes mais importantes, o *Backlog* tende sempre *a* evoluir junto com o projeto e com o ambiente. O responsável por esta construção é o *Product Owner* (ETTINGER, 2012).

O próximo passo é o planejamento da *Sprint,* onde são feitas reuniões para definir uma lista de prioridades que serão desenvolvidas na *Sprint*, esta lista é conhecida como *Sprint Backlog*. Desta forma o time deve verificar quais itens do *Product Backlog* podem ser feitos durante a *Sprint* e como serão desenvolvidas estas funcionalidades, o time tem de ser capaz de conhecer o quanto pode produzir durante a *Sprint* não ultrapassando os prazos definidos. As prioridades a serem implementadas são de responsabilidade do *Product Owner*. A *Sprint* tem duração de 2 a 4 semanas, onde os prazos devem ser obedecidos (ETTINGER, 2012).

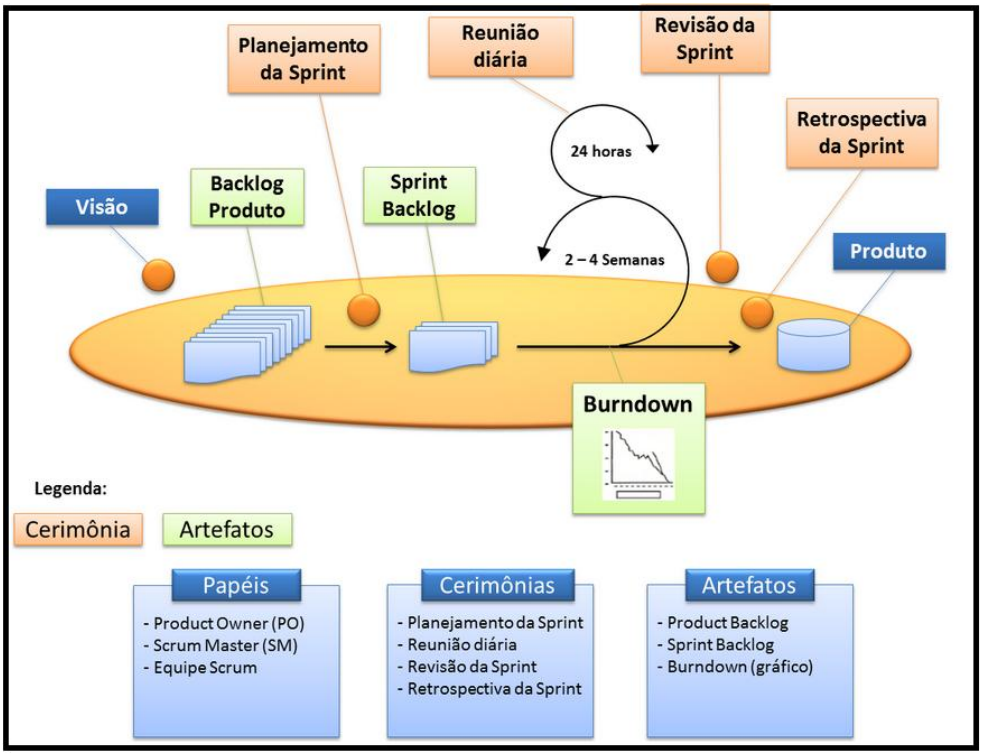
Para acompanhar o planejamento da *Sprint* existem as reuniões diárias (*Daily Meeting),* estas são feitas com duração máxima de 15 minutos onde os membros do time devem explicar ao *Scrum Master* quais tarefas foram feitas até o momento, quais tarefas serão realizadas após e quais as dificuldades encontradas, procuram esclarecer também o progresso do desenvolvimento. Desta forma o *Scrum Master* obtém o conhecimento da evolução do projeto e pode verificar como melhorar os pontos negativos (ETTINGER, 2012).

Ao final de cada *Sprint*, acontece a revisão do *Sprint*. Nesta fase o projeto realizado é entregue, testado e revisado. O *Product Owner* busca analisar e conferir se este foi feito como proposto no início e se está tudo conforme previsto. Após esta reunião é feita a reunião de retrospectiva, onde são levantados todos os pontos positivos e negativos que aconteceram no decorrer do desenvolvimento da *Sprint*, com isto é possível estabelecer pontos de melhorias e aprendizados para que a próxima *Sprint* tenha um processo melhor e mais ágil. Este ciclo *Scrum* é repetido

até que todo o *Product Backlog* proposto tenha sido finalizado (PEREIRA; TORREÃO; MARCAL, 2007).

A engrenagem do Scrum descrita pode ser observada na Figura 11.

Figura 11 - Engrenagem do Scrum.



Fonte: Ettinger (2012).

Muitas vezes os prazos determinados para o desenvolvimento de um projeto são curtos, onde existe muito tempo perdido com documentação e funções que podem ser inutilizáveis,desta forma a utilização do *Scrum* procura agilizar estes processos, focando nas funcionalidades com maior prioridade e proporcionando conhecimento do progresso do projeto diariamente, assim a equipe inteira consegue visualizar a melhor maneira de progredir.

4.1.2 Kanban

O nome *Kanban* vem de origem japonesa que tem como significado “sinal” ou “cartão”. O *Kanban* não é um processo e nem distribui papéis para serem seguidos, ele pode ser definido como um conceito ou abordagem que tem como objetivo transformar o trabalho em desenvolvimento visível para toda equipe. Deste

modo é possível sinalizar as atividades em andamento, o progresso do projeto, os prazos de entregas, as dificuldades encontradas e erros cometidos (MARIOTTI, 2012).

O *Kanban* é composto por um painel de cartões, onde estes representam a capacidade limite acordada em cada fase de um projeto que são colocadas em circulação. Estes cartões funcionam como um método de sinalização, com isto é permitido iniciar uma nova tarefa no projeto apenas quando existe um cartão disponível, ou seja, uma nova tarefa é iniciada apenas quando a outra for finalizada. Uma característica importante deste modelo é o conceito de “puxar tarefa”, ou seja, podem ser introduzidas alterações em um ciclo de desenvolvimento do projeto quando há capacidade de processa-las. Desta forma as atividades são adicionadas a lista de *Backlog*, assim estas tarefas esperam na fila até um membro da equipe liberar suas atividades e ter disponibilidade para iniciar uma nova, isso acarreta em manter um melhor desempenho na equipe (MARIOTTI, 2012).

O modelo *Kanban* pode ser resumido em três etapas:

a) visualizar os processos;
b) limitar o trabalho em processo do inglês WIP (*work in progress*);
c) gerenciar do *lead-time*, ou seja, verificar o tempo que a atividade leva para passar por todas as fases até a sua entrega (MARIOTTI, 2012).

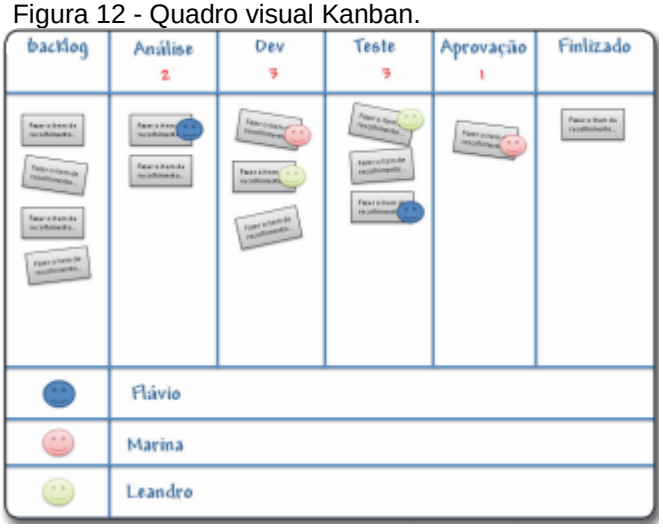
O sinalizador visual *Kanban* funciona como um quadro de sinalização de processos, deixando visível o andamento do projeto. Este quadro de sinalização é composto pelas seguintes etapas: análise, desenvolvimento, aceitação e implementação, podendo estas ser adaptadas conforme as necessidades do projeto, obtendo em vista que a maior importância é seguir as regras do *Kanban*. Estas etapas são inseridas como colunas no quadro de visualização, assim cada coluna representa uma fase do desenvolvimento da tarefa, cada vez que a atividade é concluída em uma fase ela passa para a próxima, até estar pronta para a entrega (MARIOTTI, 2012).

O sistema *Kanban* pode lembrar o modelo cascata de engenharia de software, porém ele não atua como tal, buscando evitar erros que poderiam ocorrer neste modelo com a regra de limitar os processos em andamento, WIP. Esta regra é utilizada em cada coluna do quadro determinando um número máximo de cartões em cada fase. Cada cartão possuiu uma descrição resumida dos seus requisitos.

Quando as atividades referentes a um cartão são finalizadas em uma determinada coluna, o mesmo passa para a próxima coluna liberando espaço para outro cartão que estará na fila de *Backlog* aguardando capacidade para entrar em andamento (MARIOTTI, 2012).

Na figura 12 pode ser observado um exemplo de quadro visual *Kanban*, onde o número em cada coluna representa o máximo de cartões e as cores em cada cartão o responsável pela tarefa.

Figura 12 - Quadro visual Kanban.



Fonte: MARIOTTI (2012).

Para que a utilização do *Kanban* em um projeto ou organização tenha sucesso e resultados rápidos é preciso seguir algumas etapas:

a) focar na qualidade;
b) limitar a quantidade de trabalho em progresso;
c) entregar frequentemente;
d) balancear demanda com a capacidade máxima;
e) priorizar tarefas;
f) atacar fontes de variedade (GHISI, 2012).

Estas devem ser obedecidas na ordem em que estão.

4.1.2.1 Focando na qualidade

De acordo com Anderson (2010) a qualidade do software é muito importante no andamento do projeto, pois os defeitos nesta etapa podem ocasionar

retrabalho no desenvolvimento, atraso na entrega e consequentemente desmotivar a equipe e deixar o cliente inseguro. Ainda segundo o mesmo autor, se existir um foco na qualidade, mesmo com uma equipe ruim, a produtividade pode aumentar em até dez vezes. Metodologias ágeis e tradicionais já possuem alguns métodos na melhoria da qualidade, no entanto o autor ressalta algumas práticas para a melhoria:

a) escrever testes automatizados, preferencialmente antes: existem vários casos de sucesso com escritas de testes após a codificação, porém há uma vantagem psicológica com em pedir para os desenvolvedores fazerem esta escrita antes;
b) revisar código (Verificação): esta etapa proporciona uma melhoria tanto interna como externa do software e são melhores quando aplicadas várias vezes em pequenas quantidades de código;
c) fazer atividades de análise e design do software de forma colaborativa: com a equipe trabalhando em conjunto para analisar problemas para soluções de design a qualidade tende a ser melhor, esta prática pode ser aplicada como a anterior, fazendo pequenas quantidades de modelagens;
d) usar Design *Patterns*: o uso desta prática possibilita soluções para problemas conhecidos e frequentes no desenvolvimento de software, onde os padrões garantem a eliminação de defeitos de design;
e) usar ferramentas modernas de desenvolvimento: esta dispõe de funções estáticas e dinâmicas para uma melhor qualidade, impedindo o desenvolvedor de introduzir alguma vulnerabilidade no software.

4.1.2.2 Limitando a quantidade de trabalho em andamento

Limitando a quantidade de tarefas para uma capacidade suportada pela equipe é possível controlar seu rendimento e aumentar sua produção, pois deste modo à equipe não trabalha de uma forma congestionada e sim apenas com a demanda suportada. Como o *Kanban* fornece visibilidade nos processos, deixando os problemas explícitos e focando a equipe em qualidade torna-se mais fácil identificar as dificuldades, defeitos, sobrecargas e rendimento, com isto é possível ter uma concentração maior para resolver as falhas encontradas, prevenindo

problemas futuros. Com a identificação de sobrecargas pode-se aplicar a regra de limitar os trabalhos em andamento, contribuindo para que a equipe não trabalhe com uma demanda não suportada ou estresses, isto proporciona uma qualidade melhor e um maior desempenho no desenvolvimento das tarefas (GHISI, 2012).

Este equilíbrio no fluxo de atividades auxilia na previsibilidade dos prazos de entrega das atividades. Com o cumprimento destes prazos e um rendimento na produção do projeto a confiança das entidades relacionadas, como clientes, se torna mais forte (GHISI, 2012).

4.1.2.3 Entregando frequentemente

Realizar entregas frequentes de pequenas tarefas com qualidades proporcionam uma confiança maior do cliente e da equipe do que entregas grandes com um tempo maior. Deste modo, estar sempre com este contato de entregas frequentes passa ao cliente uma segurança maior do projeto que esta sendo desenvolvido (ANDERSON, 2010).

4.1.2.4 Balanceando a demanda com a capacidade máxima

Esta etapa procura dispor um equilíbrio entre a demanda de atividades e capacidade da equipe. Com isto, procura-se analisar a taxa de aceitação de novos requisitos no processo de desenvolvimento do projeto para igualar com a capacidade de entregar de atividades com qualidade. Isto torna possível fixar a quantidade de trabalho em andamento, proporcionando que a equipe viabilize a demanda de acordo com sua capacidade (ANDERSON, 2010).

4.1.2.5 Priorizando tarefas

Esta priorização de tarefas tem o objetivo de qualificar o código entregue, ou seja, busca-se entregar um código com valor otimizado, com maior prioridade, dispondo de uma melhor qualidade do que quantidade (GHISI, 2012).

4.1.2.6 Atacando fontes de variedade

Fontes de variedade podem ser definidas como tudo que pode prejudicar o desempenho no processo de desenvolvimento. Algumas fontes de variedade requerem mudanças maiores de comportamento, por conseguinte uma persistência maior da equipe. Com isto, procura-se atacar fontes de variedade que exigem mudanças de comportamento menores, assim podem ser aceitas com maior facilidade pela equipe. Para implementar esta última técnica as outras cinco citadas anteriormente devem estar dominadas, assim a equipe torna-se mais madura (ANDERSON, 2010).

4.1.3 Scrum com Kanban

Por ser um método de simples utilização e proporcionar visualizar o andamento de um projeto o *Knaban* pode ser facilmente aplicado junto com uma metodologia ágil, como o *Scrum*. Deste modo, o monitoramento feito pelo *Scrum Master* se torna mais fácil e ágil, pelo simples fato de toda equipe poder observar algo em atraso e buscar a solução antes de virar um problema maior (JANDOZA; PRADO, 2013).

Para realizar o desenvolvimento de um software utilizando estes dois métodos em conjunto é de suma importância encontrar os pontos positivos de cada técnica e molda-las conforme o projeto a ser implementado. Desta forma um software tem maiores chances de ser entregue rápido e com qualidade, além de proporcionar um conhecimento maior em cima de uma metodologia híbrida (JANDOZA; PRADO, 2013).

5 ATIVIDADES FÍSICAS

Movimentos corporais produzidos pelos músculos, que proporcionam um aumento no gasto de energia acima do normal são considerados atividades físicas. A prática regular de atividades é um dos principais componentes na prevenção do crescimento mundial de doenças crônicas. Atualmente existem informações suficientes, sobre como a realização destes exercícios são benéficos, para que todo individuo exerça alguma atividade e melhore sua condição, tanto física como de saúde (MAREGA; MALUF, 2012).

Segundo Castro (2007) os maiores motivos para a prática de exercícios físicos implicam na busca pela estética do corpo, onde pode ser incluído o dispêndio energético e, por conseguinte, a perda de gordura, outros fatores relacionados a este motivo são o condicionamento físico, desfrutado para suportar os esforços realizados no dia a dia e uma saúde bem estruturada, onde seja alto o índice de prevenção do surgimento de doenças crônicas.

Com tudo, uma pessoa pode verificar se está no seu peso saudável calculando seu índice de massa corporal.

5.1 ÍNDICE DE MASSA CORPORAL

De acordo com Pierin (2004) o Índice de Massa Corporal (IMC) é a uma das melhores medidas de relação entre altura e peso, o objetivo deste é informar se determinada pessoa está ou não acima do peso correto. Para que seja possível verificar se tal indivíduo possui uma massa corporal elevada existem pontos de cortes fixos que informam qual o IMC ideal, conforme representada na tabela 1. O IMC de uma pessoa é calculado através da divisão do peso, em quilos, pela altura, em metros, ao quadrado, que pode ser mais bem representado da seguinte forma:

$$IMC = peso/altura^2$$

Tabela 1 - Classificação do IMC.

IMC(kg/m²)	Classificação
< 18	Baixo peso
≥ 18 e < 25	Peso saudável
≥ 25 e < 30	Pré-obesidade
≥ 30 e < 35	Obesidade grau I
≥ 35 e < 40	Obesidade grau II
≥ 40	Obesidade grau III

Fonte: Pierin (2004).

Muitas pessoas tem dificuldade em conseguir alcançar o IMC adequado apenas com dieta, no entanto a atividade física regular é um auxiliar importante no quesito de gasto de calorias, possibilitando alcançar os resultados com maior facilidade e rapidez (PIERIN, 2004; PRENTICE, 2012).

5.2 QUILOCALORIAS

O corpo humano está constantemente queimando quilocalorias para manter ativas as atividades e funções realizadas diariamente, desta forma necessita sempre repor a energia gasta para manter sua sobrevivência, buscando a mesma em proteínas, gorduras, carboidratos e álcool disponibilizados na digestão de alimentos (CAMPOS, 2002).

Visto que a sustentabilidade e saúde do corpo dependem principalmente da alimentação é de suma importância o conhecimento sobre as calorias encontradas nos alimentos, a tabela 2 mostra as calorias de alguns alimentos por quantidade (SENAC, 2005).

Tabela 2 - Calorias dos alimentos por quantidade.

Alimento	Porção	Calorias
Goiaba	1 (100g)	57,4
Goiabada	1 fatia (60g)	164,07
Grão-de-bico cozido	100g	230,00
Laranja	1 (110g)	45,5
Leite integral	1 copo (200g)	127,0
Leite desnatado	1 copo (200g)	72,2
Lentilha cozida	150g	190,5
Lombinho assado	100g	331,8
Macarrão	160g	153,6
Maça vermelha	1 (180g)	103,8
Maionese	25g	180,0
Mamão	100g	36,1

Fonte: Senac (2005).

Conforme a tabela alguns alimentos, de mesma quantidade, possuem um número de calorias maior que outros, variando para cada tipo.

Cada indivíduo dispõe de uma necessidade energética diferente dos outros, por possuir hábitos e características distintas, como o sexo, peso, altura, massa muscular e a realização de atividades físicas. Porém, o organismo das pessoas age de forma semelhante aos demais, retirando a energia dos alimentos e estocando as calorias ingeridas nas refeições, assim quando o corpo necessitar realizar alguma funcionalidade ele buscará energia dos alimentos estocados (CAMPOS, 2002; PRENTICE, 2012).

5.3 DISPÊNDIO ENERGÉTICO

O dispêndio energético nada mais é que o gasto das quilocalorias ingeridas. O corpo necessita de uma quantia de quilocalorias para seu funcionamento diário, todavia este consumo é diferente entre homens e mulheres, onde os homens carecem em média de 2500 a 3000 quilocalorias e as mulheres precisam em média de 1800 a 2300 por dia, excluindo os praticantes frequentes de atividades físicas, que requerem uma quantidade maior (CAMPOS, 2002).

Uma pessoa possui um dispêndio energético que pode variar de 1500 a 2500 calorias por dia, apenas por estar viva, sem ter hábitos que ajudem neste quesito. O gasto calórico para alguém que pratica atividade física, ou que possui

uma massa muscular mais elevada, é gradativamente maior, dependendo da frequência que exerce alguma atividade, do peso e dos tipos de atividades realizadas, com tudo, também é preciso uma absorção maior de calorias para manter estas funções ativas (CLAYTON; VANDERKAM, 2006).

As atividades físicas são fundamentais para se alcançar um dispêndio energético maior que o normal, assim como a alimentação, que também está diretamente ligada a este fato (SOARES, 2008).

Com esta equivalência de calorias decorrentes dos suprimentos e do dispêndio energético podem ser apresentadas três situações de equilíbrio energético, sendo estas o positivo, negativo e isoenergético (GUEDES, 2006).

O equilíbrio energético positivo ocorre quando existe uma ingestão de calorias que exceda o dispêndio energético, onde o indivíduo ingeri mais calorias do que gasta nas suas atividades do dia. O equilíbrio energético negativo ocorre de forma contrária, onde a quantidade de calorias gastas ultrapassa a quantidade consumida, assim o corpo procura retirar energia das proteínas, gorduras e até mesmo da massa muscular, a fim de efetuar suas funções, procedendo a uma perda de peso (GUEDES, 2006; PRENTICE, 2012).

O isoenergético acontece quando a absorção de calorias se iguala ao dispêndio energético, sem contribuir com aumento ou perda de peso (GUEDES, 2006).

Para que seja possível converter as atividades físicas em dispêndio energético pode-se recorrer a um compêndio, apresentado por pesquisadores da área, que oferece informações sobre o custo energético, em Equivalentes Metabólicos, de diversas atividades (WILMORE; COSTILL, 2007).

O Equivalente Metabólico (MET) é o valor relativo ao Oxigênio Consumido (VO2) por kg de massa por minuto. De modo que, 1 MET iguala-se a 3,5 $ml.kg^{-1}.min^{1}$, convertendo este valor pelo coeficiente calórico do oxigênio obtêm-se o valor de 17,5 calorias ou 0,0175 Quilocaloria (Kcal). Esta unidade permite estimar uma quantidade de oxigênio consumida por um individuo em uma determinada quantia de tempo (WILMORE; COSTILL, 2007).

O MET referente a algumas atividades físicas é representado na tabela 3, onde é possível visualizar o tipo de atividade praticada, o detalhe ou descrição da mesma, conforme foi realizada, e o MET relativo.

Tabela 3 - MET de cada atividade física.

Tipo de atividade	Detalhe	MET
Atividades de caminhada	Marcha atlética	6,5
Atividades de caminhada	Terreno plano e firme 6-7 km/h	5,0
Atividades aquáticas	Surf e *body-board*	3,0
Corrida	Correr subindo escadas	15,0
Dança	Dança aeróbia, alto impacto	7,0
Dança	*Step:* 15-30 cm	10,0
Esportes	Basquetebol: jogo	8,0
Esportes	Boxe no ringue	12,0
Esportes	Boxe no saco	6,0
Esportes	Futebol: competição	10,0
Esportes	Pular corda: vigoroso	12,0

Fonte: Guedes (2006).

Segundo Guedes (2006) para calcular o dispêndio energético em uma atividade utiliza-se o MET da mesma, o peso em quilos do praticante, o tempo referente ao exercício. O calculo pode ser definido da seguinte forma:

$$Dispêndio\ energético = MET\ *\ peso *\ tempo/60$$

Cada atividade apresenta um dispêndio energético diferente, dependendo da situação em que foi realizada e do peso do praticante.

De acordo com Darido e Souza Junior (2007) para atingir um equilíbrio energético negativo com mais facilidade é preciso praticar atividades com um tempo de duração maior do que trinta minutos, além de diminuir a quantidade de calorias consumidas no dia. Desta forma o corpo atinge um dispêndio energético maior, proporcionando a diminuição de massa corpórea.

6 TRABALHOS CORRELATOS

Neste capitulo são apresentados trabalhos que possuam conteúdos semelhantes ou que tenham alguma relação com este projeto de pesquisa, disponibilizando maiores informações sobre a proposta apresentada.

6.1 SCRUM EM APLICAÇÕES MÓVEIS

Diante do avanço tecnológico e do crescente número de fabricantes de dispositivos móveis surgi à exigência de aplicações melhores que suportem estes dispositivos. Com isto obteve-se um estudo em métodos para obter maior eficiência e qualidade, onde foram encontradas as metodologias ágeis.

Com a diversidade destas metodologias foi escolhida uma, a metodologia *Scrum.* Esta foi criada para gerenciar projetos de desenvolvimento de software com equipes pequenas, desenvolvimentos em curtos espaços de tempos e com a minimização de erros. Para ter maior chance de sucesso o *Scrum* dispõem de alguns pontos interessantes como: redução de documentação, comunicação constante com o cliente e equipe de desenvolvimento, entrega de funcionalidades do projeto já desenvolvidas e testadas, entre outras.

O *Scrum* é citado como uma das melhores metodologias ágeis no desenvolvimento de aplicações móveis em curto prazo e com funcionalidades simples. O estudo efetuado teve como objetivo mostrar uma metodologia de simples utilização, eficaz para aplicações móveis, onde a documentação feita é apenas para armazenar o que o produto possui de essencial e ainda possibilita alterações em qualquer fase do projeto (JANDOZA; PRADO, 2013).

6.2 MÉTODO ÁGIL APLICADO AO DESENVOLVIMENTO DE SOFTWARE CONFIÁVEL BASEADO EM COMPONENTES

Os métodos ágeis tem progredido muito nos últimos anos, por meio de métodos como Extreme Programming (XP) e Scrum, sendo aplicados no desenvolvimento de diversos sistemas computacionais. Esse fato apresenta necessidade de softwares com maior confiabilidade, mais rigorosos e que possuam

uma quantidade adequada de modelagem e documentação, proporcionando maior qualidade nos resultados.

A pesquisa propõe uma solução para guiar desenvolvimento de sistemas confiáveis, baseados em componentes por meio da adição do MDCE+ ao Scrum, resultando no método Scrum+CE (Scrum com Componente Excepcional. Para a realização do método foi realizado um experimento utilizando o Scrum e Scrum+CE, coletando métricas para comparar a eficiência do novo processo e o resultado obtido com a utilização do mesmo, produzindo melhor qualidade, porém com menos funcionalidade (MARTINS, MORAES, 2013).

6.3 DIABETES CONTROL: UMA APLICAÇÃO MOBILE APLICADA AO GERENCIAMENTO DE INFORMAÇÕES MÉDICAS REFERENTES AO CONTROLE DO DIABETES

A crescente onda de aplicativos móveis vem buscando solucionar problemas ou auxiliar profissionais de áreas especificas. Dentre estas pode ser citada a saúde, que vem se destacando muito pela utilização de computação móvel, contribuindo com os médicos para que possuam acesso rápido e fácil a informações de pacientes.

A pesquisa apresenta um aplicativo para computação móvel, desenvolvido em Android, tendo como objetivo gerenciar informações de diabetes em pacientes, possibilitando o controle da doença na população. Desta forma, foi obtida a construção de um modelo que agilize e facilite as informações entre médicos e paciente.

O dispositivo do paciente atua como um transmissor, enviando informações do seu estado de saúde para o celular do médico, permitindo um acompanhamento melhor dos médicos no tratamento da diabetes (NEUWALD, 2012).

6.4 IMPACTO DAS ATIVIDADES FISICAS NOS INDICADORES DE SAÚDE DE SUJEITOS ADULTOS: PROGRAMA MEXA-SE

A prática de atividade física tem trazido benefícios a saúde das pessoas, diversos estudos comprovam que estilos de vida mais ativos estão mais prevenidos

de doenças crônicas, já os estilo sedentário possui o dobro de chance de adquirir estas doenças.

O objetivo do trabalho foi analisar impactos das atividades físicas buscando melhoria na qualidade de vida e saúde de pessoas adultas, através do programa chamado Mexa-se Unicamp. Para realização do trabalho foram utilizadas técnicas como “causa” e “efeito” para melhor investigação de doenças.

Para realização da pesquisa foi utilizado um protocolo de perguntas e respostas, e ainda coletado dados sobre pressão arterial e de relação cintura quadril. Desta forma foram encontrados resultados indicativos, onde indivíduos que trabalham sentados possuem um dispêndio energético muito baixo comparado ao de pessoas que fazem uso de atividades no trabalho, além de terem chances maiores em obter doenças devido à falta de atividade física no dia. Com isto, foi visto que a prática de atividade física ajuda em uma vida com maior qualidade e saúde (SANTOS et al, 2009).

6.5 UM MODELO DE GERENCIAMENTO DE PROJETOS BASEADO NAS METODOLOGIAS ÁGEIS DE DESENVOLVIMENTO DE SOFTWARE E NOS PRINCÍPIOS DA PRODUÇÃO ENXUTA

Buscando ganhos sustentáveis de produtividade e qualidade no desenvolvimento de software foi realizado um estudo propondo gerenciamento de projetos baseado nas metodologias ágeis, *scrum* e *Extreme programming,* e nos princípios e valores do pensamento enxuto.

Com isto, foram avaliados os resultados da combinação das metodologias ágeis e nos princípios e valores *da produção* enxuta. Deste modo, foi feita uma análise qualitativa para determinar as vantagens na utilização do métodos e seu escopo de aplicação. Após isto se aplicou uma comparação entre o modelo proposto e a metodologia cascata, onde o modelo proposto contorna os problemas que ocorrem na utilização do modelo cascata (FRANCO, 2007).

7 CALORIAS CONTROL – PROTÓTIPO DE APLICATIVO PARA ACOMPANHAMENTO DO DISPÊNDIO ENERGÉTICO COM UTILIZAÇÃO DAS TÉCNICAS SCRUM E KANBAN

O presente estudo mostrou como resultado a eficiência da utilização da metodologia ágil *Scrum* em conjunto com o *Kanban*, aplicados na construção de um protótipo que possibilita um controle de dispêndio energético, dos usuários, em atividades físicas, tornando o desenvolvimento do projeto ágil e qualitativo.

Os métodos pesquisados e utilizados contribuíram de forma significativa em todo o processo da criação do software, proporcionando um acompanhamento das etapas do projeto, tornando-as visíveis e permitindo detectar erros logo no início, contribuindo assim com uma rapidez na evolução do mesmo.

O aplicativo criado recebe os dados inseridos pelos usuários e possibilita um monitoramento do dispêndio energético obtido em atividades praticadas, estes dados são retroalimentados, transformados em informações e disponibilizados visualmente para o usuário. Para a construção do protótipo foram utilizadas algumas tecnologias e ferramentas necessárias, como a IDE do eclipse, a máquina virtual Java entre outras.

7.1 METODOLOGIA

Devido à necessidade de se obter um conhecimento mais amplo sobre o trabalho proposto, foi realizado um levantamento bibliográfico, com o objetivo de pesquisar o maior número de informações, para disponibilizar um estudo de maior qualidade sobre o assunto. Primeiramente o estudo realizado consistiu, em sua maior parte, à tecnologia Android, buscando-se conteúdo de todos os requisitos necessários, como ferramentas e tecnologias, para o desenvolvimento do protótipo. Dentre estas ferramentas podem ser citadas o Eclipse IDE, utilizado para desenvolver o protótipo, juntamente com as linguagens Java e XML, a Máquina virtual Java, ADT plugin, banco de dados SQLite, entre outras utilizadas propriamente.

Para que fosse possível desenvolver o projeto de forma mais organizada, ágil e qualitativa, foram estudados os métodos *Kanban* e *Scrum*, mesclando-os e aplicando as regras de cada um conforme as necessidades para a criação do

projeto. Deste modo, tornou-se necessário a formação de uma pequena equipe para aplicar o *Scrum*, composta pelo professor coorientador, do curso de educação física, o professor orientador, do curso de ciência da computação, e acadêmico. Por fim, foram apresentadas neste trabalho, de forma mais breve, assuntos sobre o dispêndio energético, descrevendo os pontos mais importantes e utilizados na aplicação.

O projeto teve início com a aplicabilidade das regras do *Scrum*, utilizando o *Kanban* em conjunto quando preciso, possibilitando melhor organização nas etapas. Com isto, foi feita uma reunião inicial para definir todas as funcionalidade e pontos a serem produzidos, após esta foram feitas *Sprints* para produzir todas as fases do projeto de forma separada e organizada, modelando o fluxo de informações, o banco de dados, as telas, funcionalidades e demais necessidades encontradas. Ao final de cada Sprint foram realizados testes e feitas reuniões com o intuito de analisar se a etapa concluída foi produzida da forma com que foi estabelecida inicialmente.

7.1.1 Aplicabilidade do *Scrum* e *Kanban*

Para o desenvolvimento bem sucedido do protótipo foram estabelecidas regras a serem seguidas, estas convém dos métodos utilizados, visando proporcionar organização, qualidade no aplicativo, rapidez na construção e visibilidade das etapas, permitindo que o desenvolvimento seja eficiente e proporcionando um limite de tarefas suportável, em um determinado limite de tempo, para serem produzidos. Deste modo os erros e etapas demoradas são encontrados e analisados logo no inicio em que aparecem possibilitando uma solução rápida para não haver perda de tempo.

7.1.2 Definição dos papéis no *Scrum*

Inicialmente, com o objetivo de organização foram estabelecidos os papéis dentro do *Scrum* para a equipe, permitindo o correto funcionamento do método. Com isto, a equipe foi definida com o coorientador do presente trabalho como o *Product Owner*, proporcionando a ele a responsabilidade de gerenciar o

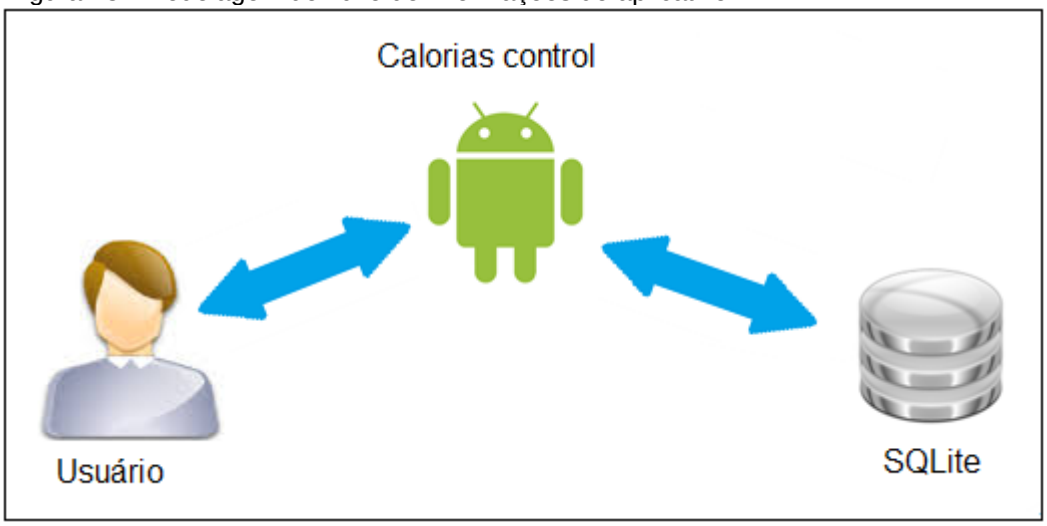
Backlog do projeto, o orientador como o *Scrum Master*, e por fim, o acadêmico na parte do desenvolvimento, sendo coordenado e orientado pelo *Scrum Master*. Com os papéis de cada membro da equipe já estabelecidos foi iniciada a produção da parte prática do projeto definindo o *Product Backlog*.

7.1.3 Reunião de definição do *Product Backlog*

Para obter uma análise completa do protótipo foi feita uma reunião com toda a equipe, esta teve como objetivo levantar todos os requisitos, funcionalidades, prioridades e etapas a serem desenvolvidas para a criação do protótipo.

Primeiramente foram verificadas as informações necessárias para o funcionamento do aplicativo, estabelecendo o fluxo de informações entre o usuário e o protótipo, como pode ser verificado na figura 13. Com isto, foram definidas as informações que seriam importantes retornar para o usuário, dentre estas podem ser citadas o IMC, onde tanto o número referente quanto a classificação são mostradas, a taxa metabólica, quantidade de calorias referente a sua massa corpórea, gasto calórico diário e principalmente o dispêndio energético obtido nas atividades praticadas. Estas informações são apresentadas em diferentes telas.

Figura 13 - Modelagem do fluxo de informações do aplicativo.



Como pode ser visto na figura 13 existe um envio e recebimento de informações entre o usuário, aplicação e o banco de dados, onde o usuário entra com os dados na aplicação e a mesma apresenta os retornos citados anteriormente,

entre outros. Estas informações são salvas e retroalimentadas no banco de dados SQLite e podem ser consultadas pelo usuário.

Com o fluxo de informações determinado foi possível analisar previamente os dados necessários do usuário para gerar os resultados requeridos, onde destes podem ser citados como: idade, altura em centímetros, massa corporal em quilos, o peso que o usuário deseja alcançar, quantidade de dias que o usuário deseja para chegar a seu peso requerido. Com isto, foi determinado para que o protótipo disponibilizasse atividades para o usuário praticar, estas apresentam o tempo ou distância de execução da atividade e as calorias a serem consumidas.

Assim como os retornos, estes dados são inseridos em telas diferentes, onde podem ser consultados quando necessário. Esta etapa teve maior influência do *Product Owner.*

Posteriormente, com uma gerência maior do *Scrum Master*, mas com o acompanhamento do desenvolvedor e do *Product Owner* foram analisadas as etapas, para a produção do protótipo, de uma forma mais prática. Estas consistiam em desenvolver um diagrama de casos de uso, possibilitando ser visualizado pelo *Product Owner* e verificado se estava conforme requerido, um modelo dos protótipos de *wireframes,* permitindo conhecer as telas onde são inseridos e retornados os dados, uma modelagem do banco de dados, onde ficam visíveis as informações que são armazenadas, obtendo assim um entendimento maior do circulo de dados e a fase de implementação do protótipo. Deste modo foi possível obter uma lista de etapas para serem desenvolvidas, possibilitando a organização da construção do protótipo e facilitando o encontro de novas necessidades que poderiam ser acrescentadas no decorrer do desenvolvimento.

Com o *Product Backlog* de todos os pontos da aplicação determinados foi preciso dividir esta lista em pequenas *Sprints Backlogs,* assim foi possível produzir cada etapa com melhor qualidade e com prazos a serem cumpridos.

7.1.4 Desenvolvimento da primeira *Sprint*

A primeira *Sprint* identificou como objetivo a divisão de itens prioritários e de grande necessidade para o inicio do desenvolvimento do projeto, permitindo que fossem desenvolvidas etapas interessantes para a obtenção de sucesso no final.

7.1.4.1 Reunião de planejamento da primeira *Sprint*

Para construir um software com rapidez visando qualidade foi feita uma reunião com o objetivo de planejar a primeira *Sprint.* Nesta fase foi reunida toda a equipe novamente para identificar itens importantes do *Product Backlog* e formar a primeira *Sprint Backlog*. Deste modo, conforme a equipe propôs foram selecionados para serem desenvolvidos inicialmente na primeira *Sprint* o caso de uso da aplicação, a modelagem das telas e a modelagem do banco de dados.

Para que as tarefas propostas na primeira *Sprint* fossem bem sucedidas foram utilizadas algumas regras do *Kanban,* assim foi possível determinar as tarefas produzidas na *Sprint* com um prazo de acordo com a capacidade do desenvolvedor. Desta forma foi possível ter foco na qualidade, procurando evitar erros no desenvolvimento da *Sprint.* O prazo definido foi de quinze dias, onde se procurou estabelecer uma demanda balanceada de tempo para que o desenvolvimento fosse qualitativo e aceitasse a entrada de outras tarefas caso precisasse.

7.1.4.2 Desenvolvimento da primeira *Sprint*: modelagem

Para construir um aplicativo de forma mais segura, buscando prever problemas e encontrar a melhor maneira para produzi-lo é de suma importância à utilização da modelagem, esta proporciona um entendimento melhor e facilita na implementação do aplicativo.

Atualmente existem diversas ferramentas que facilitam a modelagem. Neste trabalho foram utilizadas ferramentas gratuitas, aplicadas na modelagem de caso de uso, na modelagem do banco de dados e na criação de *wireframes*, com o intuito de apresentar algumas telas.

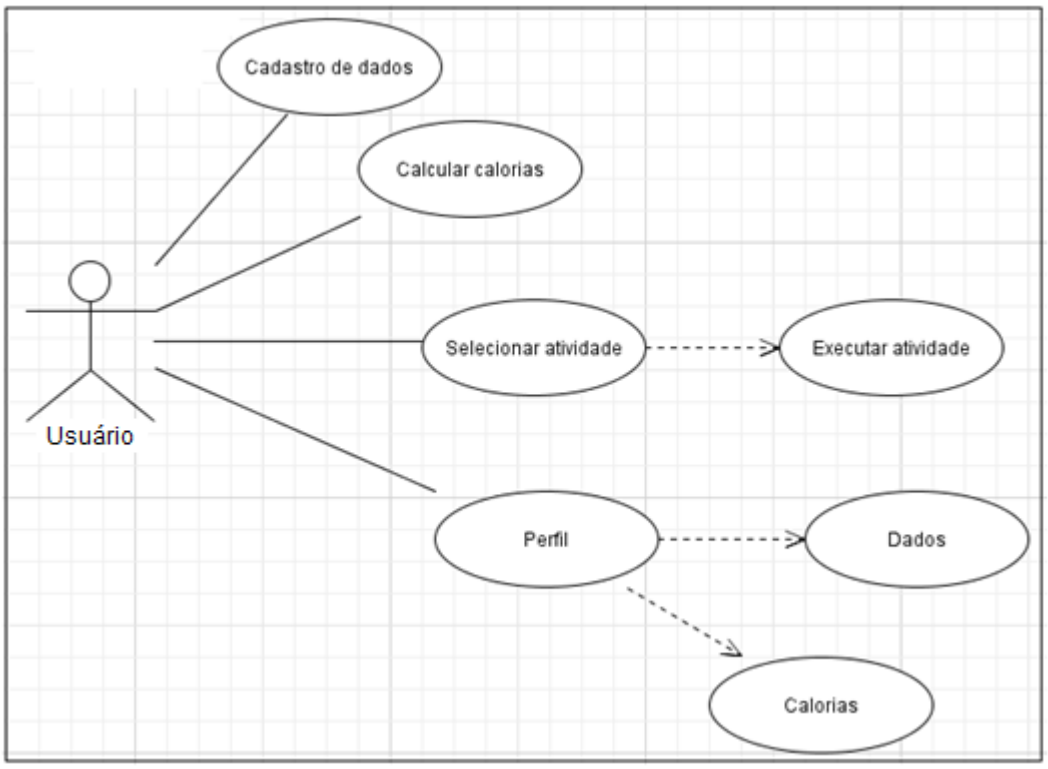
7.1.4.2.1 Modelagem do caso de uso

O desenvolvimento do caso de uso se deu pela manipulação da linguagem *Unified Modeling Language* (UML), esta modelagem apresenta de forma visual os artefatos da construção do aplicativo, auxiliando o conhecimento da aplicação.

Para realização da criação do diagrama de caso de uso permitiu-se a utilização da ferramenta Poseidon, por ser gratuita e proporcionar todos os requisitos necessários, além de possuir técnicas avançadas de ferramenta.

A figura 14 demonstra o diagrama de caso de uso da aplicação, onde o ator representa o usuário em comunicação com a aplicação.

Figura 14 - Diagrama de Caso de Uso da aplicação.



Conforme pode ser visualizado na figura 14 o usuário tem comunicação direta com a aplicação, podendo navegar entre as telas inserindo e consultando dados. Os casos de uso inseridos com ligação direta ao usuário são considerados da tela inicial do protótipo, proporcionando extensões para outras telas.

Os casos de usos *Cadastro de dados* e *Calcular calorias* são referentes a inserção de dados, possibilitando um cadastro de todas as informações necessárias para o protótipo e apresentando para o usuário informações importantes para a execução das atividades.

Com os dados cadastrados o aplicativo gera uma lista de atividades, onde esta é disponibilizada para o usuário no caso de uso *Selecionar atividade,* assim permite a escolha de uma, ativando a ação de redirecionamento para o caso de uso *Executa atividade*, este proporciona a funcionalidade para armazenar as calorias

gastas. No caso de uso *Perfil* apenas disponibiliza uma consulta dos dados do usuário, permitindo o mesmo visualizar suas informações, conforme é visto na figura 14 estas consultas são visibilizadas em extensões para outras duas telas, *dados* e *calorias.*

Com a criação do caso de uso da aplicação efetuado a etapa seguinte se refere à realização da modelagem das telas.

7.1.4.2.2 Modelagem das telas (wireframe)

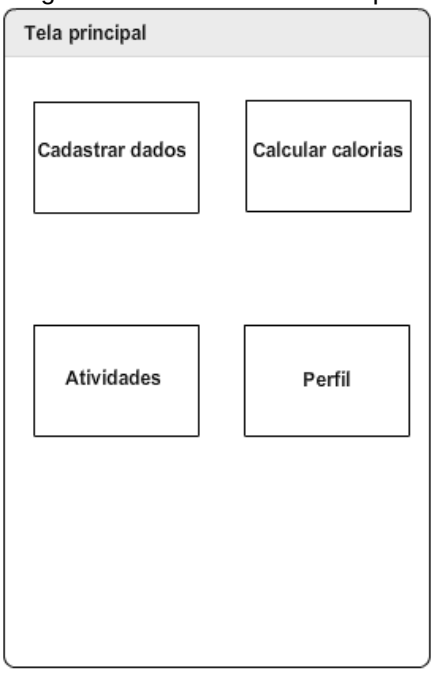
Com o objetivo de criar as telas inicialmente, sem que houvesse grandes ajustes futuramente, se deu o uso da modelagem das mesmas. Desta forma foram construídos *wireframes* utilizando uma ferramenta gratuita, sem que precisasse desenvolver diretamente no XML da aplicação, assim foi possível evitar atrasos e seguir os padrões de telas modelados.

Para esta modelagem foi utilizada a ferramenta MockFlow, pois esta pode ser executada diretamente no navegador, e não tem necessidade de instalações. Com esta ferramenta foram encontrados todos os requisitos para os layouts das telas.

Devido à necessidade de criação de *wireframes* para servirem como bases, foram modeladas três telas com a ferramenta citada, onde esta proporcionou análise e o conhecimento do desenvolvimento concreto na aplicação.

Na figura 15 pode ser visualizada a tela inicial do aplicativo, esta apresenta quatro botões, os quais possuem ações de redirecionamento para outras telas.

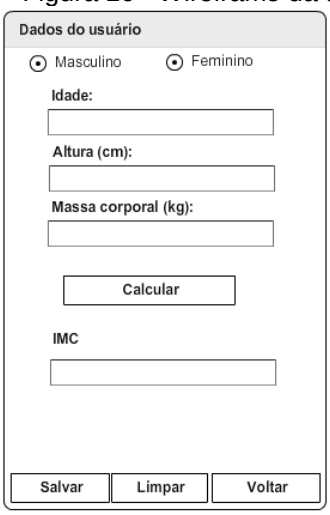
Figura 15 - Wireframe da tela principal.



A tela inicial da aplicação foi modelada com o intuito de disponibilizar ao usuário uma seção de escolhas. Estas são representadas pelos botões que permitem a navegação entre as opções da aplicação. Para que a tela seja usada de modo correta a mesma segue uma sequencia lógica, cadastrando seus dados, cadastrando e calculando os dados referentes às calorias, selecionando e executando as atividades e consultando suas informações no perfil.

A partir da modelagem da tela inicial se deu a construção das telas seguintes, sendo que o seguinte *wireframe* construído foi o de cadastro de dados, conforme mostra a figura 16.

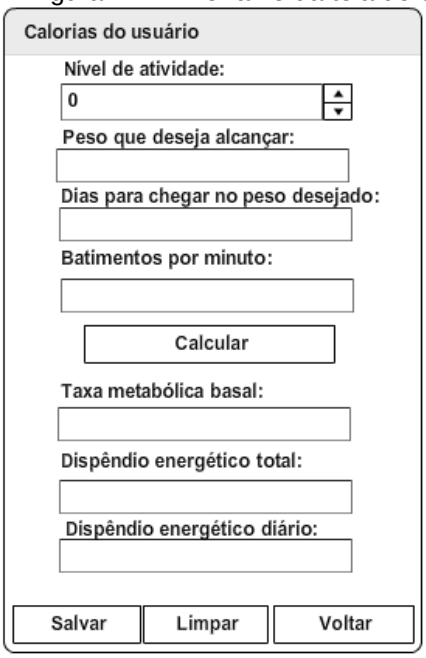
Figura 16 - Wireframe da tela de cadastro de dados.



A figura 16 apresenta campos para a inserção de alguns dados do usuário no aplicativo, estes dados são referentes ao sexo, idade, altura, e massa corporal. Além destas inserções a tela apresenta um botão *calcular,* com a funcionalidade de gerar e apresentar o IMC do usuário, um botão *Salvar,* com a ação de armazenar os dados inseridos e retornados, um botão *Limpar* para apagar as informações dos campos e um botão *Voltar* que permite o usuário retornar a tela principal.

O próximo passo foi à criação do *wireframe* das calorias, de acordo com a figura 17. Esta tela permite a inclusão de outros dados do usuário, onde são registrados: o peso objetivo, ou seja, o peso em que se deseja chegar, a quantidade de dias para chegar ao peso requerido e a quantidade de batimentos por minuto.

Figura 17 - Wireframe da tela de cálculo das calorias.

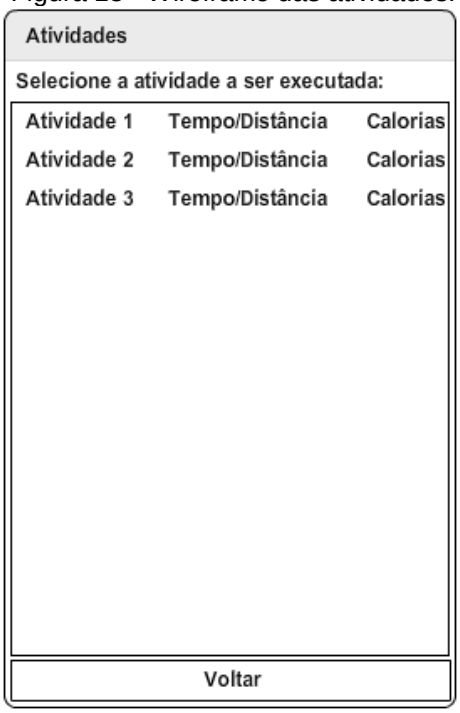


Como pode ser visualizada, esta também apresenta os botões *Calcular, Salvar, Limpar* e *Voltar,* todos realizando as mesmas ações do *wireframe* anterior, porém neste o botão *Calcular* preenche os campos restantes com informações da taxa metabólica basal, do dispêndio energético que deve ser obtido diariamente e do dispêndio energético total, que se refere à quantidade de calorias a serem queimadas.

Com o propósito de encontrar um layout de fácil utilização para o usuário foi modelada a tela onde são selecionadas as atividades. A figura 18 permite a identificação das informações repassadas na tela, onde é disponibilizada uma lista

de atividades a ser escolhida pelo usuário, contendo informações de cada atividade, como modo de praticar e a quantidade de calorias que podem ser gastas.

Figura 18 - Wireframe das atividades.



Posteriormente à modelagem de alguns *wireframes* que constituem a aplicação foi possível modelar o banco de dados, facilitando assim a criação do mesmo no software.

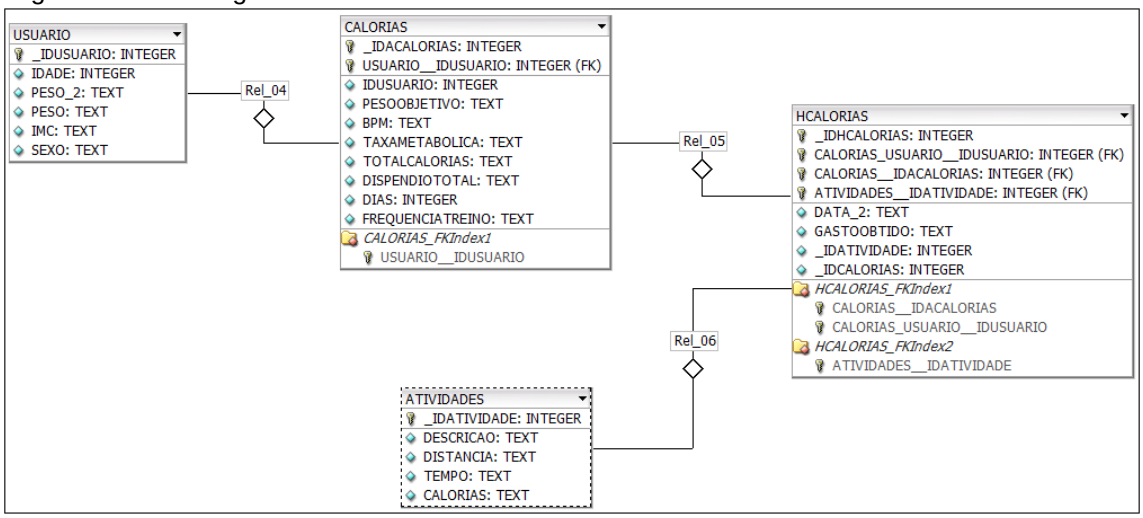
7.1.4.2.3 Modelagem do banco de dados

De modo a construir a base de dados da aplicação, com a possibilidade de visualizar as tabelas e analisar a forma coerente de inserção dos dados foi modelado o banco de dados. Desta forma foi possível ter uma visão do comportamento dos dados.

A ferramenta utilizada para esta modelagem foi a DBDesigner 4, por proporcionar todos os requisitos necessários para este tipo de modelagem.

A modelagem do banco de dados é apresentada na figura 19 no seu modelo físico.

Figura 19 - Modelagem do banco de dados.



Conforme mostra a figura é possível verificar todas as ligações entre as tabelas e os respectivos campos de cada. Com esta representação de forma física obteve-se uma estrutura padrão para o conhecimento de como os dados iriam se comportar.

Esta modelagem permitiu o entendimento e melhor organização dos campos. Como pode ser visto as tabelas USUARIO e CALORIAS apresentam dados fixos, inseridos pelo usuário, a tabela HCALORIAS disponibiliza um histórico das determinadas atividades praticadas pelo usuário em cada dia e seus respectivos gastos calóricos, já a tabela de atividades possui uma quantidade fixa de exercícios, as quais calculam as calorias, distância ou tempo, dependendo do tipo de atividade, de acordo com os dados do usuário e são disponibilizadas para que possam ser registradas como executadas.

7.1.4.3 Reuniões diárias da primeira *Sprint* (*Daily Meeting*)

Com a finalidade de promover as regras do Scrum de forma coerente foram realizadas reuniões quase diárias, conhecida como *Daily Meeting*. Como existiam dificuldades em reunir o *Scrum Master* com o desenvolvedor diariamente foram estabelecidos alguns pontos para executar as regras do *Scrum*, porém modelando conforme as necessidades do projeto. Deste modo foi percebido que não existia um demanda com necessidade de realizar reuniões diárias, mas reuniões com encontros de três dias semanais.

Estas reuniões foram executadas com encontros entre o desenvolvedor e *Scrum Master,* onde aconteciam de forma pessoal, *online,* via troca de emails, entre outros meios de comunicação. Com isto foi possível identificar erros que poderiam ocasionar o atraso do projeto e possibilitar a coordenação sobre o desenvolvedor, determinando passos para aperfeiçoamento das etapas.

Os encontros propuseram discussões para sanar dúvidas e continuar o desenvolvimento de modo correto, contribuindo para o ganho de tempo e deixando o *Scrum Master* ciente da evolução do projeto.

7.1.4.4 Revisão da primeira *Sprint*

Esta revisão foi realizada de modo a reunir toda a equipe em uma reunião e apresentar ao *Product Owner* as etapas produzidas. Todavia, possibilitou explicar ao mesmo, detalhes da aplicação, dentre estes como seria o funcionamento e as informações retratadas em cada tela.

Posteriormente a todos os detalhes expostos, o *Product Owner* verificou se todas as informações estavam de acordo com as definições estabelecidas na estrutura da *Sprint Backlog*. Deste modo foram obtidas de forma concreta as considerações do *Product Owner,* propondo pequenas mudanças no que diz respeito à forma de funcionamento e concordando com as etapas produzidas.

7.1.4.5 Retrospectiva da primeira *Sprint*

A retrospectiva da *Sprint* concedeu reunir o desenvolvedor e *Scrum Master* com o intuito de diagnosticar os pontos negativos e positivos da primeira *Sprint.* Nesta reunião foram discutidos os passos alcançados e analisado os pontos a serem melhorados para a segunda *Sprint*.

Alguns pontos negativos encontrados foram às definições de telas, onde era preciso obter uma base para posteriormente implementar de modo efetivo as mesmas. Com o termino da primeira *Sprint* foi possível utilizar como experiência toda a produção da mesma. Consequentemente, após a finalização da retrospectiva, foi determinada uma data próxima para a realização da segunda *Sprint.*

7.1.5 Desenvolvimento da segunda *Sprint*

A segunda *Sprint* deu início após o termino da primeira, buscando reunir toda a equipe novamente para uma nova seleção de itens do *Product Backlog*. Esta permitiu agregar a utilização de regras do *Kanban* com a finalidade de aumentar a qualidade e diminuir o tempo de produção, ou seja, proporcionar uma agilidade maior.

Esta *Sprint* procura utilizar as mesmas regras da primeira, apenas com novas prioridades a serem desenvolvidas. Deste modo é possível verificar que as *Sprints* passam a ser um ciclo, onde são repetidas até a finalização de todo o *Product Backlog.*

7.1.5.1 Reunião de planejamento da segunda *Sprint*

Primeiramente, como antes, foi estabelecida uma nova reunião, porém com novos itens e necessidades a serem desenvolvidos. Para determinar a lista de requisitos da segunda *Sprint* foram consultados os itens do *Backlog* e acrescentados os Itens propostos pelo *Product Owner* na reunião de revisão da primeira *Sprint.*

Com a modelagem do projeto criada a segunda *Sprint* deu inicio as etapas do desenvolvimento do protótipo. Com isto, foram determinados às prioridades a serem implementadas. De modo a seguir o gerenciamento do *Product Owner* foram averiguar as telas de maior importância para desenvolver, onde agregam o maior número de inserção de dados. Assim, foi determinada para a segunda *Sprint* a criação do banco de dados na aplicação, a criação das telas de cadastro de dados e calculo das calorias, no XML da aplicação, e as funcionalidades das mesmas.

Como propósito de qualificar e agilizar ainda mais os processos de desenvolvimento aplicou-se as regras do *Kanban*, estas foram utilizadas no modo de desenvolvimento e na determinação do tempo de realização da *Sprint,* onde se procurou estimar quinze dias para a finalização dos itens selecionados. Assim foi possível identificar um tempo padrão, que não interferisse na qualidade, mas também que não apresentasse tempos longos de folga, ou seja, um tempo balanceado da capacidade do desenvolvedor.

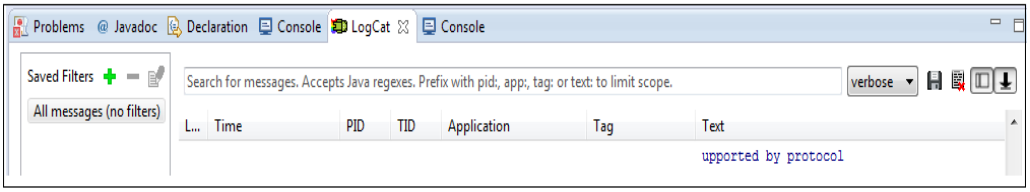
Além desta, outras regras do *Kanban* foram aplicadas, como a priorização de tarefas a serem realizadas, a escrita de testes antes da implementação do código, a inserção de novas tarefas no meio do desenvolvimento e a utilização do quadro visual.

7.1.5.2 Implementação da segunda *Sprint*

Para construir um software *mobile,* utilizando o sistema operacional Android, foi necessária a utilização de algumas ferramentas. Estas contribuiriam para que fosse possível produzir o software em um ambiente de desenvolvimento e formular breves testes em um emulador.

Conforme citado na fundamentação teórica o ambiente de desenvolvimento utilizado foi o Eclipse IDE, porém para que o mesmo proporcionasse suporte para a criação de aplicativos foi essencial à instalação e configuração do Android SDK e do AVD manager. Além destes foi possível fazer o uso do Android *Development Tools plugins,* onde eram exibidas as visualizações de erros, alocação de memória, entre outras características explicitas na figura 20.

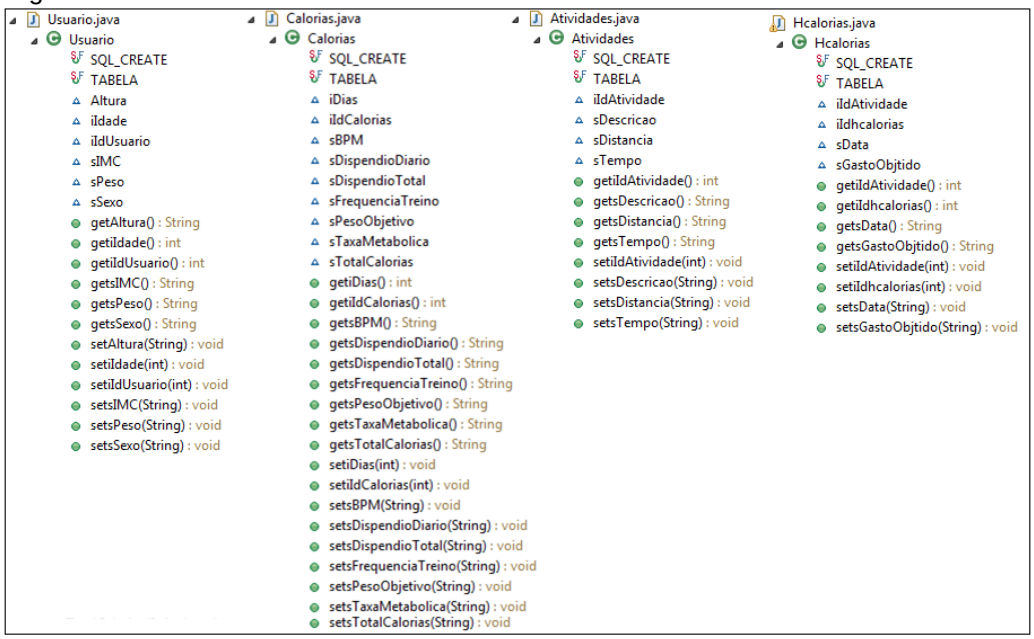
Figura 20 - ADT plugin no Eclipse.



Com as ferramentas citadas permitiu-se a criação da aplicação, onde sem elas não seria possível. Outra ferramenta importante adotada foi o SQLite, um banco de dados próprio do Android, que contribui para facilitar a movimentação dos dados.

A empregabilidade das ferramentas deu inicio a implementação do protótipo. Com isto, primeiramente, foi realizada a criação das tabelas do banco de dados, concebendo a criação de quatro classes em um pacote, onde cada uma destas apresenta o seu SQL de criação e respectivos *getters* e *setters* referentes a cada coluna, conforme mostra a figura 21.

Figura 21 - Classes com as tabelas do banco.



Para que fosse possível gerar as tabelas no Android foram implementadas duas classes para manipular estas criações, a classe ConfigDAO, que tem como objetivo armazenar a versão do banco de dados, o nome do mesmo e as classes que contém as tabelas a serem criadas, e a classe HelperDAO, esta estende a classe SQLiteOpenHelper, que busca as classes que estão contidas na ConfigDAO. Assim permite a criação do banco de dados caso não exista ou proporciona uma atualização do mesmo se a versão for alterada.

Após a criação do banco estabelecida foi iniciada a criação das telas definidas para esta *Sprint* e suas funcionalidades, utilizando um quadro visual *Kanban* para o desenvolvedor e *Scrum Master* acompanharem em conjunto os passos de cada tarefa a ser desenvolvida.

O quadro visual *Kanban* foi modelado conforme os requisitos do projeto e adicionadas tarefas determinadas na *Sprint Backlog*. Assim foi possível controlar as atividades que estavam sendo realizadas e adicionar novas sem ocasionar interrupções. O quadro visual *Kanban* inicial pode ser verificado na figura 22.

Figura 22 - Quadro visual Kanban inicial da segunda Sprint.

Sprint Backlog	Análise	Desenvolvimento	Testes	Aprovação	Finalização
Layout da tela de Cadastro de dados Layout da tela de Cálcular calorias Funcionalidades da tela principal Funcionalidades da tela de Cadastro de dados ... Funcionalidade da tela deCalcular cálorias	Layout da tela principal				
Desenvolvedor	X	X	X		X
Scrum Master				X	

Conforme mostra o quadro *Kanban* existem diversas etapas por onde cada tarefa passa, sendo que a aprovação é concedida pelo *Scrum Master,* pelo fato de produzir uma validação do que foi produzido.

Inicialmente, a primeira tarefa foi a de construir o *Layout* da tela principal, posteriormente às demais definidas no quadro *Kanban*. As elaborações das telas foram baseadas nas modelagens dos *wireframes,* assim, estas passaram pela etapa de análise e foram introduzidas na etapa de desenvolvimento de forma separadas, sendo produzida unitariamente, assim uma nova tarefa começa apenas quando a outra for finalizada, respeitando a regra do *Kanban* de limitar a quantidade de trabalho em progresso. O desenvolvimento das mesmas foi realizado em XML, onde as funcionalidades a serem desenvolvidas, posteriormente, em classes Java, adquirem acesso através da classe R do Android, gerada automaticamente.

Após o desenvolvimento dos *layouts*, as telas passaram direto para a etapa de aprovação, pelo fato de não apresentarem funcionalidades momentâneas, onde o *Scrum Master* verificou se as mesmas estavam de acordo com o esperado para progredir para finalização. Com o término dos *layouts* se tornou interessante aplicar a regra da escrita de testes antes da produção do código. Desta forma foi possível visualizar a aplicação como um todo e escrever testes para serem realizados em outras etapas.

As próximas tarefas a serem realizadas foram às implementações das funcionalidades de cada tela. Deste modo, como anteriormente, foram analisadas de forma separadas e passadas para a etapa de desenvolvimento, aplicando a mesma regra do *Kanban*, conforme apresenta a figura 23.

Figura 23 - Quadro visual Kanban com layouts finalizados.

Sprint Backlog	Análise	Desenvolvimento	Testes	Aprovação	Finalização
Funcionalidades da tela de Cálcular calorias Funcionalidades da tela de Cadastro de dados	Funcionalidades da tela principal				Layout da tela principal Layout da tela Cadastrar dados Layout da tela Calcular calorias
Desenvolvedor	X	X	X		X
Scrum Master				X	

Para realizar a implementação das funcionalidades foram criadas novas classes para cada tela, assim foi possível manipular os campos elaborados no XML acessando a classe R do Android. Com o intuito de organizar de forma correta foram criadas classes de controle referentes a cada classe de tela, estes controles são responsáveis por organizar os dados e enviar requisições de inserções ou consultas para a classe Banco. Esta tem finalidades como as de abrir e fechar as conexões com o banco de dados, realizar inserções e consultas, as quais são retornadas as classes de controles.

Na tela de calcular calorias, após a inserção dos dados, foi inserida uma funcionalidade para calcular as calorias de algumas atividades, definidas pelo desenvolvedor. O sistema basicamente busca o MET das atividades, sendo que existem algumas pré-definidas, multiplica este valor do MET da atividade pelo peso do usuário, multiplica o resultado obtido por um tempo, que foi determinado aleatoriamente pelo desenvolvedor, como por exemplo, cinquenta minutos e dividi tudo por sessenta, este resultado é referente ao dispêndio energético da atividade.

De modo a qualificar as implementações foram realizados os testes, apresentando as telas criadas no XML e as funcionalidades desenvolvidas. Com os testes escritos logo no início e a adição de outros no decorrer da implementação foi possível rever todos os possíveis equívocos e revisar o código fonte. Quando eram encontrados erros nas funcionalidades, estas voltavam para a etapa de desenvolvimento até estarem aptas para a aprovação e finalização.

No decorrer do desenvolvimento outra tarefa a ser realizada surgiu, com tudo, esta foi adicionada a lista da *Sprint Backlog* de modo que não ocasionasse

interrupção nas demais atividades, sendo implementada após a finalização das outras e passada por todas as etapas do quando *Kanban*.

Permitindo obter um foco maior melhor no projeto outra regra do *Kanban* foi utilizada, no que diz respeito a atacar fontes de variedades, isto possibilitou deixar o desenvolvedor focado no projeto, retirando alguns itens que poderiam afetar a concentração, tornando melhor e mais ágil o desenvolvimento.

7.1.5.3 Reuniões diárias da segunda *Sprint* (*Daily Meeting*)

As reuniões da segunda *Sprint* foram realizadas seguindo o mesmo modelo da primeira, onde não foram feitos encontros diários, mas sim três vezes por semana, podendo ser estes encontros pessoais ou com a utilização de meios de comunicação, para sanar dúvidas e repassar as etapas finalizadas.

Em algumas destas reuniões foram discutidas as formas de acrescentar uma tarefa a ser modificada repassada pelo *Product Owner,* citada anteriormente, e como estava o andamento do projeto. Estas reuniões proporcionavam ao *Scrum Master* conhecer as funcionalidades do projeto e orientar a continuidade do mesmo.

Um modo de apresentar a evolução do mesmo para o *Scrum Master* foi exibindo o quadro visual do *Kanban* utilizado, assim facilitou a identificação de atrasos e erros cometidos, permitindo resolver os mesmos sem perda de tempo. Estes quadros eram apresentados em todas as reuniões possibilitando um conhecimento visual do andamento do projeto.

7.1.5.4 Revisão da segunda *Sprint*

Ao final desta *Sprint* foi realizada uma nova reunião com toda a equipe, permitindo assim apresentar de forma detalhada as telas implementadas e suas respectivas funcionalidades para o *Product Owner*. Deste modo, possibilitou serem realizados alguns testes com dados fictícios para analisar e exemplificar o funcionamento da mesma, verificando se as telas criadas estavam coerentes com o estabelecido.

Com a aceitação do *Product Owner* foi dada continuidade nas etapas da *Sprint*, registrando algumas novas melhorias que poderiam ser efetuadas no decorrer da implementação da terceira *Sprint.*

7.1.5.5 Retrospectiva da segunda *Sprint*

Esta etapa ocorreu com o mesmo objetivo da primeira, procurando expor os pontos positivos e negativos encontrados no decorrer da *Sprint*. Desta forma foi possível explorar mais a fundo, pelo *Scrum Master* e o desenvolvedor, os pontos a serem melhorados.

Dentre estes pontos negativos encontrados podem ser citados os layouts criados no XML. Por serem produzidos para *móbile*, onde a tela de comunicação com o usuário é relativamente menor que um sistema para *desktop*, existiu uma demanda de tempo um pouco maior que o esperado para esta parte, porém com esta dificuldade se tornou possível adquirir experiência para implementar na terceira *Sprint* de forma mais ágil.

Os pontos positivos identificados podem ser citados como a linguagem Java, que disponibiliza uma fácil utilização, a lógica computacional e a organização das tarefas, proporcionadas pelas regras do *Scrum* e *Kanban,* onde a demanda de tempo diminui e os atrasos são identificados rapidamente. Outro fator importante que contribuiu foi à escrita dos testes antes da implementação, proporcionando testar a aplicação por inteiro e não focando apenas na parte desenvolvida.

Com o termino e a entregada segunda *Sprint* para o *Product Owner* foi possível iniciar a terceira *Sprint,* verificando os itens restantes do *Product Backlog*.

7.1.6 Desenvolvimento da terceira *Sprint*

Assim como as anteriores, a terceira *Sprint* procurou reunir toda a equipe novamente para analisar os itens prioritários a serem desenvolvidos. Porém, como a demanda restante era pequena possibilitou a finalização de todo o *Product Backlog* na terceira *Sprint*.

Esta *Sprint*, por ser parecida com a segunda, obteve a aplicação das mesmas regras, unindo *Scrum* e *Kanban* para realizar a implementação das telas faltantes.

7.1.6.1 Reunião de planejamento da terceira *Sprint*

Com a união de toda a equipe foi possível iniciar a reunião de planejamento da terceira *Sprint*, onde foram determinadas as tarefas a serem realizadas. Esta *Sprint* proporcionou a criação das telas de perfil e atividades, buscando implementar posteriormente as funcionalidades referente as mesmas e funções para unir o protótipo por completo.

A realização desta *Sprint* se deu com a utilização de regras do *Kanban* novamente e principalmente com a utilização do quadro visual, este que proporcionou uma organização importante e o conhecimento de etapas a serem percorridas, dentre outras regras aplicadas também na segunda *Sprint* que possibilitaram melhor entendimento do progresso do projeto.

Por fim, foi essencial fazer um teste completo do aplicativo, buscando avaliar se todas as funcionalidades aplicadas estavam coerentes e proporcionando ligar todas as ações implementadas anteriormente com as que foram desempenhadas nesta *Sprint.*

7.1.6.2 Implementação da terceira *Sprint*

A terceira *Sprint* iniciou com a produção do quadro visual *Kanban*, procurando deixar explicitas as tarefas desenvolvidas, assim o *Scrum Master* obtia um controle do crescimento do projeto, bem como seus atrasos.

A figura 24 apresenta o quadro visual *Kanban* do início da terceira *Sprint*, onde as tarefas foram modificadas conforme estabelecidas no planejamento desta.

Figura 24 - Quadro visual Kanban inicial da terceira Sprint.

Sprint Backlog	Análise	Desenvolvimento	Testes	Aprovação	Finalização
Layout da tela de execução das atividades Layout das telas de perfis Funcionalidade da tela de atividades Funcionalidade da tela de exec. de atividades Funcionalidade das telas de perfis	Layout da tela de atividades				
Desenvolvedor	X	X	X		X
Scrum Master				X	

Como poder ser visto na figura 24, primeiramente foram realizadas as análises e posteriormente elaborações dos *layouts* das telas no XML. Com isto possibilitou o enviou das mesmas para a aprovação do *Scrum Master* e por fim a finalização destas. Antes do desenvolvimento das funções foram escritos os testes referentes a cada tela e a aplicação por inteira, permitindo qualificar todos os itens produzidos.

As próximas tarefas efetuadas foram às implementações das funcionalidades de cada tela. Conforme a figura 24 mostra cada tarefa passou por todas as etapas do quadro *Kanban* de forma individual, assim o tempo de desenvolvimento de cada tarefa foi específico para ela, contribuindo desta forma com a qualidade do software.

Assim como explicado na segunda *Sprint* foram criadas novas classes Java para realizar a manipulação dos dados e classes de controle, com o objetivo de requisitar a classe Banco para efetuar inserções, consultas e apagar registros, caso necessário. Estas funções proporcionaram, além de manipulação de dados, a execução de cálculos essenciais para o funcionamento da aplicação.

Após o desenvolvimento de cada funcionalidade se deu a realização do teste de cada implementação de forma separada, primeiramente revisando o código e as funções que a própria tela executa, com o objetivo de enviar para aprovação sem erros.

A figura 24 mostra como última tarefa o desenvolvimento das funcionalidades das telas de perfil dos dados do usuário e perfil das calorias do usuário, isto aconteceu pelo fato das duas telas formarem basicamente as mesmas consultas, mesmo por serem apresentadas em telas separadas as funções pertencentes a elas proporcionaram esta implementação.

Com a finalização de todas as tarefas da *Sprint* foi possível realizar um teste do protótipo por inteiro a fim de qualificar o mesmo. Deste modo, os testes contribuíram para revisar os possíveis equívocos que poderiam ocorrer.

7.1.6.3 Reuniões diárias da terceira *Sprint* (*Daily Meeting*)

Estas reuniões proporcionaram ao *Scrum Master* visualização da evolução da *Sprint* e, por conseguinte, do projeto. Com estas foram analisadas as tarefa no quadro *Kanban,* permitindo a identificação dos tempos gastos nas tarefas em cada etapa.

Outras discussões realizadas com estas reuniões foram às tarefas na etapa de aprovação, onde era o *Scrum Master* que as efetuava, concebendo os pontos que poderiam ser melhorados. Com estas reuniões a organização das implementações das tarefas progrediu, pelo fato do *Scrum Master* estar sempre contribuindo para aperfeiçoar o andamento do projeto e coordenando de forma significativa o desenvolvimento do mesmo.

7.1.6.4 Revisão da terceira *Sprint*

Com a conclusão do *Product Backlog* se tornou possível apresentar o protótipo para o *Product Owner*, onde foram revisados todos os passos percorridos até o momento de entrega final. Desta forma, foram exemplificadas todas as funcionalidades do protótipo, além de levantar todos os requisitos estabelecidos e validados para certificar que a aplicação foi concluída como esperado.

Com a demonstração do funcionamento do protótipo e a validação do mesmo por inteiro, o *Product Owner* verificou se o mesmo foi finalizado como esperado, permitindo concluir o projeto.

7.1.6.5 Retrospectiva da terceira *Sprint*

Esta retrospectiva concedeu ao desenvolvedor e *Scrum Master* analisar os pontos mais interessantes na construção não apenas *Sprint,* mas também do protótipo inteiro.

Com isto, foram expostos os pontos com maior dificuldade no decorrer do projeto, podendo ser citados estes como a implementação de algumas funcionalidades internas, que ocuparam uma significativa demanda de tempo.

Outros pontos expostos foram os positivos como a utilização das regras do *Scrum* e *Kanban* que disponibilizaram métodos para aperfeiçoar o desenvolvimento do projeto, a fim de agilizar e conceituar a finalização do mesmo.

7.2 RESULTADOS OBTIDOS

Como resultado foi criado uma aplicação utilizando as regras da metodologia ágil *Scrum* em conjunto com o *Kanban.* A aplicação do *Scrum* no desenvolvimento do projeto possibilitou listar primeiramente todos os requisitos e funcionalidades do protótipo e subdividir esta lista em listas de itens, gerando três *Sprints,* ordenadas por prioridade.

Com isto, foi possível executar a produção de cada lista de itens de forma organizada e com maior rapidez. Outro fator importante foi a participação da equipe em todo o projeto acrescentando de forma significativa, pelo fato de orientar o desenvolvedor quando necessário e auxiliar nas entregas das tarefas.

Com a aplicação das regras do *Kanban* foi possível desenvolver com maior qualidade o protótipo, proporcionando ao desenvolvedor uma forma de implementar sem interrupções e com um tempo suportado pelo mesmo, além de mostrar de forma visual as etapas de cada tarefa.

Deste modo, a aplicabilidade do *Scrum* e *kanban* resultou na produção do projeto com maior qualidade, organização e agilidade proporcionando o desenvolvimento de um protótipo de aplicativo que tem como objetivo permitir ao usuário acompanhar o dispêndio energético obtido em determinadas atividades físicas.

A aplicação apresenta inicialmente a tela principal, onde é possível acessar as funcionalidades do aplicativo, cadastrando os dados do usuário e calculando as devidas calorias que o mesmo pretende perder, posteriormente podem ser selecionadas as atividades que serão executadas e consultar os dados no perfil.

A tela para cadastro permite que o usuário insira seus dados e receba como retorno a classificação do IMC, referente às informações do mesmo. Estas informações são armazenadas no banco de dados para serem utilizada em outras telas ou funções, como na tela de cálculo das calorias. Esta busca receber outras informações do usuário e utilizar as informações cadastradas anteriormente com o objetivo de apresentar para o mesmo a sua taxa metabólica basal, o dispêndio energético total e o dispêndio energético que necessita atingir diariamente.

Após todos os dados cadastrados pode-se acessar a tela para selecionar as atividades. Esta tela possibilita a visualização das atividades, e suas respectivas características, como tempo ou distância e calorias gastas com a atividade. Deste modo, o usuário pode escolher uma atividade e registra-la como executada. Assim, o sistema salva esta atividade como feita e registra as calorias referentes a mesma.

Na tela de execução de atividades são exibidas diversas informações para que as atividades sejam praticadas de forma a obter o dispêndio energético apresentado visualmente. Além das informações expostas, como: atividades, tempo ou distância e quantidade de calorias que serão gastas, existe a frequência de treino em que o usuário deve executar a atividade. Esta frequência, quando exercida no nível correto, permite que a atividade consuma apenas a gordura corporal, sem retirar energia dos músculos.

Desta forma, o usuário pode registrar atividades diversas vezes, ou seja, marcar como feita uma ou mais atividades, onde as calorias são salvas no protótipo e podem ser visualizadas na tela de perfil.

As informações disponibilizadas nas telas de perfil permitem a visualização dos dados inseridos e o peso que já foi consumido pelo usuário. Deste modo, o protótipo procura verificar a quantidade de calorias gasta em cada dia, caso encontre um dia com o dispêndio energético menor do que o retornado anteriormente pelo software o mesmo é apagado e acrescentado um dia a mais para obter este gasto. As calorias e gastos retornados do protótipo para o usuário são

valores aproximados, por apresentar cálculos e a forma de praticar as atividades o protótipo considera que a atividade foi executada da forma com que foi exemplificada para o usuário.

A construção e implementação de todas as telas foi realizada de forma separada, criando uma demanda de tempo para cada produção, isto proporcionou um foco maior em cada etapa, permitindo priorizar as implementações mais importantes.

A utilização dos métodos híbridos *Scrum* e *Kanban* tornou possível agilizar e organizar melhor o desenvolvimento do projeto, pelo fato de integrar dois métodos distintos e utilizar suas regras conforme a necessidade do projeto. Os itens estabelecidos em cada *Sprint* do *Scrum* eram realizados aplicando regras do *Kanban,* assim, os prazos estabelecidos para entrega das *Sprints* eram balanceados conforme a capacidade do desenvolvedor, isto permitiu tempo para realizar testes e revisões do que foi desenvolvido, proporcionando um foco na qualidade.

A cada final de *Sprint* eram entregues os itens desenvolvidos ao *Product Owner,* proporcionando entregas frequentes ao mesmo. Este princípio fez com que o *Product Owner* criasse uma segurança no desenvolvimento do projeto.

Com isto, pode-se concluir que todos os objetivos do presente projeto foram cumpridos, de modo que estes exigiam a aplicabilidade dos métodos *Scrum* e *Kanban* em conjunto, para realizar o desenvolvimento de um protótipo de aplicativo utilizando o sistema operacional Android.

7.3 DIFICULDADES ENCONTRADAS

Algumas dificuldades encontradas surgiram no momento de divisão dos itens em *Sprints,* por buscar priorizar as tarefas mais importantes teve que ser analisado de forma detalhada para averiguar o melhor modo de dividir e qual a ordem mais interessante a ser aplicada.

CONCLUSÃO

De modo a atingir os objetivos propostos no presente trabalho, que consistiam na utilização dos métodos híbridos *Scrum* e *Kanban,* para a produção de um protótipo que monitore o dispêndio energético, foram realizados estudos sobre a plataforma Android, a metodologia ágil *Scrum*, o método *Kanban* e dispêndio energético. Para efetuar o desenvolvimento do aplicativo foram utilizadas tecnologias e ferramentas, como: IDE Eclipse, SDK, ADT *plugin* e o Google Android.

Deste modo, foi aplicado todo o estudo realizado, dos métodos citados, na construção do protótipo, identificando que estes contribuem de forma positiva na implementação de sistemas, auxiliando na criação de um aplicativo que oferece monitoramento do dispêndio energético de praticantes de atividades físicas.

Com tudo, pode ser concluído que a utilização dos métodos em conjunto para o desenvolvimento de um protótipo obteve sucesso, proporcionando uma produção com demanda de tempo suportada pelo desenvolvedor, sem interrupções, um visualização física da implementação das etapas, um desenvolvimento qualitativo, com maior agilidade e organização em todos os passos do projeto. Os métodos citados podem ser integrados de forma fácil, onde um contribui com o outro, permitindo amplificar as chances de sucesso no final.

Por fim, de modo a concluir o presente estudo são sugeridos alguns trabalhos futuros:

a) utilizar outro método em conjunto com estes;

b) comparar com a utilização de outra metodologia ágil;

c) evoluir as funcionalidades do aplicativo e implementar para iOS;

d) aplicar testes reais.

REFERÊNCIAS

ABLESON, W. Frank et al. **Android em ação.** 3.ed. Rio de Janeiro: Elsevier, 2012. 656p.

ALLEN, Grant; OWENS, Mike. **The definitive guide to SQLite.** 2 ed. New York: Apress, 2010. 238 p.

ANDERSON, David. **Kanban:** Successful Evolutionary Change for Your Technology Business. Washington, Blue Hole Press. 2010

APPLE. **Ipad Air.** Disponível em: <http://www.apple.com/br/ipad-air/>. Acesso em: 10 Set. 2014.

ARAUJO, A. S et al. Fatores motivacionais que levam as pessoas a procurarem por academias para a prática de exercícios físicos. **Revista Digital**, Buenos Aires, ano 12, n 115, 2007. Disponível em: <http://www.efdeportes.com/efd115/fatoresmotivacionais-que-levam-as-pessoas-a-procurarem-por-academias.htm>. Acesso em 04 nov. 2014

BARBIERI, Aline Fabiane; MELLO, Rosângela Aparecida. As causas da obesidade: uma análise sob a perspectiva Materialista histórica. **Revista da Faculdade de Educação Física da Unicamp,** Campinas, v. 10, n. 1, p.133-153, jan./abr. 2012. Disponível em: <http://fefnet178.fef.unicamp.br/ojs/index.php/fef/article/viewFile/653/396>. Acesso em: 16 Set. 2014.

BESSA, Antonio Sousa. **Programação Gambas:** Banco de dados. OpenASB, 2014. 313 p.

BONOME, Karoline da Silva et al. Disseminação do uso de aplicativos móveis na atenção à saúde. In: CONGRESSO BRASILEIRO EM INFORMÁTICA EM SAÚDE, 13., 2012, São Paulo. **Anais...** 2012. p. 1-6. Disponível em: <http://www.sbis.org.br/cbis2012/arquivos/807.pdf>. Acesso em: 15 Nov. 2014.

BRASIL, (Gov.). **Pesquisa revela aumento na prática de atividades físicas.** Disponível em: <http://www.brasil.gov.br/saude/2014/05/pesquisa-revela-aumentona-pratica-de-atividades-fisicas>. Acesso em: 19 out. 2014.

BRAY, Tim. **What Android Is**. Disponível em: <http://www.tbray.org/ongoing/When/201x/2010/11/14/What-Android-Is>. Acesso em: 12 out. 2014.

CAMPOS, Marcos Vinhal (Org.). **Atividade física passo a passo:** saúde sem medo e sua preguiça. Brasília: Thesaurus, 2002. 248 p.

CASTRO, Ana Lucia de. **Culto ao corpo e sociedade:** mídia, estilos de vida e cultura de consumo. 2. ed. São Paulo: Annablume, 2007. 150 p.

CINAR, Onur. **Android Apps with Eclipse.** New York: Apress, 2012. 372 p.

CLAYTON, David; VANDERKAM, Laura. **Como ser saudável sem ser chato:** uma dieta para pessoas como você, saúde sem abrir mão dos prazeres. Rio de Janeiro: Elsevier, 2006. 200 p. Tradução de Ana Beatriz Rodrigues.

COMUNICAÇÕES, Ministério Das (Org.). **História da Telefonia: Linha do Tempo**. Disponível em: <http://www.mc.gov.br/component/content/article/44-historia-dascomunicacoes/22463-historia-da-telefonia>. Acesso em: 08 Set. 2014.

COSTA, Daniel Gouveia. **Java em rede:** Recursos avançados de programação. Rio de Janeiro: Brasport, 2008. 344 p.

DARCEY, Lauren; CONDER, Shane. **Learning Android application programming for the kindle fire:** A hands-On guide building your first Android application. New Jersey: Addison-Wesley Professional, 2012. 352 p.

DARIDO, Suraya Cristina; SOUZA JUNIOR, Osmar Moreira de. **Para ensinar educação física:** Possibilidades de intervenção na escola. Campinas: Papirus, 2007. 352 p.

DARIVA, Roberto. **Gerenciamento de dispositivos móveis e serviços de telecom**. Rio de Janeiro: Elsevier, 2011. 160 p.

DAVID, A. Aker. **Relevância de marca:** como deixar seus concorrentes para trás. Porto Alegre: Artmed, 2011 342 p.

DUBEY, Abhishek; MISRA, Anmol. **Android security:** attacks and defenses. California: Hardback, 2013. 280 p.

ETTINGER, D.; A engrenagem do scrum. 2012 disponível em: http://danielettinger.com/2011/04/06/a-engrenagem-do-scrum/; acessado em: julho/2012.

FRANCO, Eduardo Ferreira. **Um modelo de gerenciamento de projetos baseadonas metodologias ágeis de software e nos princípios da produção enxuta.** 2007. Dissertação (Mestrado em Sistemas Digitais) – Universidade de São Paulo, São Paulo.

FIGUEIREDO, Carlos Maurício Seródio; NAKAMURA, Eduardo. **Computação Móvel:** Novas Oportunidades e Novos Desafios. **T&C Amazônia**, Manaus, n. 2, p.16-28, 01 jun. 2003. Semestral. Disponível em: <https://portal.fucapi.br/tec/imagens/revistas/ed02_completo.pdf> Acesso em: 02 set. 2014.

FONSECA, Isabella; CAMPOS, Alberto. **Por que Scrum?** Engenharia de Software Magazine, Rio de Janeiro, Edição 4, p. 30-35, 2008.

GHISI, Thiago. Kanban no desenvolvimento de projetos de software.**Engenharia de Software Magazine**, ed 45 p. 11-16, 2012.

GOOGLE. **Android Developers**. Disponível em: <http://developer.android.com/>. Acesso em: 15 Out. 2014.

GRUMAN, Galen. **The iPad's victory in defining the tablet: What it means**: Apple's view of the tablet is now the accepted model, but one that most commodity competitors still haven't figured out .InfoWorld. Disponível em: <http://www.infoworld.com/d/mobile-technology/the-ipads-victory-indefining- the-tablet-what-it-means-431>. Acesso em: 20 Set. 2014.

GUALANO, Bruno; TINUCC, Taís. Sedentarismo, exercício físico e doenças crônicas. **Revista Brasileira de Educação Física e Esporte,** São Paulo, v. 25, n. 1, p.37-43, dez. 2011. Disponível em: <http://www.scielo.br/pdf/rbefe/v25nspe/05.pdf>. Acesso em: 16 Set. 2014.

GUEDES, Dartagnan Pinto. **Manul prático para avaliação em educação física.** Barueri: Manole, 2006. 484 p.

HOLZNER, Steven. **Inside XML.** Massachusetts: New Riders, 2001. 1152 p.

IBGE. **PNDA:** De 2005 para 2011, número de internautas cresce 143,8% e o de pessoas com celular, 107,2%. Disponível em: <http://saladeimprensa.ibge.gov.br/noticias?view=noticia&id=1&idnoticia=2382&busca=1&t=pnad-2005-2011-numero-internautas-cresce-143-8-pessoas-celular-107-2>. Acesso em: 08 Set. 2014.

ILYAS, Mohammad; AHSON, Syed A. **Smartphones:** research report, Chicago: International Engineering Consortium**,** 2006. 249 p.

ITU, International Telecommunication Union; **New ITU ICT Development Index compares 154 countries.** Disponível em <http://www.itu.int/newsroom/press_releases/2009/07.html> Acesso em: 09 Set 2014.

JANDOZA, William; PRADO, Antonio Francisco do. **Scrum em Aplicações Móveis.** São Carlos, v. 2, n. 3, p. 211 -21 9, set-dez 201 3.

JOBSTRAIBIZER, Flávia. **Criação de Aplicativos para Celulares com Google Android.** São Paulo: Universo dos livros, 2009. 128 p.

JORDAN, Lucas; GREYLING, Pieter. **Practical Android Projects.** New York: Apress, 2011. 424 p.

LADD, Eric et al. **Using XHTML, XML and Java 2.** Washington: Que, 2001. 1413 p.

LECHETA, Ricardo. R. **Googloe Android:** aprenda a criar aplicações para dispositivos móveis com o Android SDK. 3. ed. São Paulo: Novatec, 2013. 824 p.

LEE, Wei-meng. **Beginning Android™ Application Development**. Indianápolis: Wiley, 2011.

MACARINI, Luan Porfírio. **Estudo do comportamento da bateria de um dispositivo móvel com sistema operacional android, com base no modelo analítico linear.** 2012. 62 f. Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) - Universidade do Extremo Sul Catarinense, Criciúma. Disponível em: <http://www.kironunesc.net.br/tcc/arquivos/trabalhos/343.pdf>. Acesso em: 16 Set. 2014.

MAREGA, Marcio; MALUF, José Antônio. **Manual de atividades físicas para prevenção de doenças.** Rio de Janeiro: Elsevier, 2012. 304 p.

MARIOTTI, Flavio S. Kanban: o ágil adaptativo.**Engenharia de Software Magazine**, ed 45 p. 6-10, 2012.

MARTINS, Elaine; MORAES, Regina Lúcia de Oliveira. **Método ágil aplicado ao desenvolvimento de software confiável baseado em componentes.** Dissertação (Mestrado em Ciência da Computação) – Universidade estadual de campinas, Campinas. 2013.

MEIER, Reto. **Professional Android™ 2 Application Development.** Indianapolis: Wrox, 2010. 576 p.

MORIMOTO, Carlos E. **Uma análise crítica do iPhone**. Disponível em: <http://www.hardware.com.br/analises/iphone/>. Acesso em: 20 Set. 2014.

NEUWALD, Leonardo Alves**. Diabetes control:** uma aplicação mobile aplicada ao gerenciamento de informações médicas referentes ao controle do diabetes. 2012. Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) – Universidade do Extremo Sul Catarinense, Criciúma.

OLIVEIRA, Vivian Rodrigues de; PAULINO, Rita de Cássia Romeiro. Construção e estrutura da notícia nas interfaces dos tablets. **E-Com,** v.6, n.1, 2013. Disponível em: <http://revistas.unibh.br/index.php/ecom/article/view/1015/580> Acesso em: 21 Set. 2014.

PEREIRA, Lúcio Camilo Oliva; SILVA, Michel Lourenço da. **Android para desenvolvedores.** Rio de Janeiro: Brasport, 2009. 240 p.

PEREIRA, Paulo; TORREÃO, Paula; MARCA, Ana Sofia. **Entendendo Scrum para Gerenciar Projetos de Forma Ágil.** 2007.

PIERIN, Angela Maria Geraldo. **Hipertensão arterial:** uma proposta para o cuidar. Barueri: Manole, 2004. 372 p.

PRENTICE, Willian E. **Fisioterapia na prática esportiva:** Uma abordagem baseada em competências. 14. ed. Artmed, 2012. 878 p. Tradução de Denise Regina de Sales e Maiza Ritomy Ide.

PRESSMAN, Roger S.; **Engenharia de software:** uma abordagem profissional. 7. ed. Porto Alegre: AMGH, 2011. 775 p. Tradução Ariovaldo Griesi e Mario Moro Fecchio.

SALABERRI, Diogo Bonoto. **Desenvolvimento de Aplicações Embarcadas com Android.** 2011. 52 f. Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) - Universidade Federal de Pelotas, Pelotas. Disponível em: <http://inf.ufpel.edu.br/nopcc/lib/exe/fetch.php?media=monografias:2010:2010-monodiogo_salaberri.pdf>. Acesso em: 15 Nov. 2014.

SANTOS, Maria Cesariana Gandara Barbosa.**Impacto das atividades fisicas nos indicadores de saúde de sujeitos adultos: programa mexa-se.** 2009. Tese (Doutorado em Educação Física) - Universidade Estadual de Campinas, Campinas.

SCHWARZ, Ronan et al. **The Android developer's cookbook:** Building application with the Android SDK. 2. ed. 2013. 464 p.

SENAC, DN. **Beleza:** desafios e conquistas da ciência e da tecnologia. Rio de Janeiro: Senac Nacional, 2005. 128 p.

SERSON, Roberto Rubinstein. **Programação orientada a objetos com java.** Rio de Janeiro: Brasport, 2007. 492 p.

SILVEIRA, Paulo et al. **Introdução a arquitetura e design de software:** Uma visão sobre a plataforma Java. Rio de Janeiro: Elsevier, 2012. 284 p.

SQLITE. **About SQLite.** Disponível em: < http://www.sqlite.org/about.html> Acesso em: 22 Out. 2014.

SOARES, João Cesar Castro. **Dieta dissociada:** emagrecer com saúde comendo de tudo. São Paulo: MG Editores, 2008 144 p.

VETTORAZZI, Felipe J; SCHUTZ, F. **Taxi calculator:** Aplicação android para controle de rotas de táxi**. Revista Eletrônica Científica Inovação e Tecnologia,** Paraná, v. 2, n. 2, p.48-59, 6 mar. 2012. Disponível em: <http://revista.md.utfpr.edu.br/sis/index.php/IT/article/viewFile/208/pdf>. Acesso em: 15 Nov. 2014.

WILMORE, Jack , Costill David L. **Fisiología del esfuerzo y del deporte.** 6.ed. Buenos Aires: Paidotribo, 2007. 744 p.

APÊNDICE D – ARTIGO

PROTÓTIPO DE APLICATIVO COM A UTILIZAÇÃO DA METODOLOGIA ÁGIL SCRUM E DO CONCEITO KANBAN PARA ACOMPANHAMENTO DO DISPÊNDIO ENERGÉTICO EM ATIVIDADES FÍSICAS

Thalles J. Vieira[1], Luciano Antunes[2], Joni M. de Farias[3]

[1]Acadêmico do Curso de Ciência da Computação – Departamento de Ciência da Computação – Universidade do Extremo Sul Catarinense (UNESC) – Criciúma, SC – Brazil
[2]Professor do Curso de Ciência da Computação – Departamento de Ciência da Computação – Universidade do Extremo Sul Catarinense (UNESC) – Criciúma, SC – Brazil
[3]Professor do Curso de Educação Física – Departamento de Educação Física – Universidade do Extremo Sul Catarinense (UNESC) – Criciúma, SC - Brazil

Abstract. *This study sought to apply Agile Scrum together with the Kanban concept in developing a prototype application, the Android platform, which seeks to provide a monitoring of energy expenditure obtained in physical activities. Thus, the above methods were merged using the strengths of each to perform application development. Therefore, it was possible to obtain higher quality and agility in creating a prototype application that provides a monitoramente of energy expenditure obtained in certain physical activities.*

Resumo. O presente trabalho buscou aplicar a metodologia ágil Scrum em conjunto com o conceito Kanban no desenvolvimento de um protótipo de aplicativo, na plataforma Android, que busque disponibilizar um acompanhamento do dispêndio energético obtido em atividades físicas. Deste modo, foram mesclados os métodos citados, utilizando os pontos positivos de cada, para realizar o desenvolvimento do aplicativo. Com isto, foi possível obter maior qualidade e agilidade na criação de um protótipo de aplicativo que proporcione um monitoramente do dispêndio energético obtido em determinadas atividades físicas.

1.INTRODUÇÃO

O crescimento do mercado de dispositivos móveis vem se mostrando surpreendente nos últimos anos, graças à rápida evolução da tecnologia neste setor. Deste modo existe a necessidade de explorar ao máximo os recursos da tecnologia móvel (VETTORAZZI; SCHUTZ, 2013).

Junto a este significativo aumento no mercado consumidor eclode a evolução dos sistemas operacionais, em especial o Android, com um vasto número de recursos já integrados o qual tem se mostrado muito eficiente, como código aberto, *middleware* e ferramentas de desenvolvimento (SALABERRIA, 2011).

Além destas ferramentas existem outros métodos que podem ser utilizados para aperfeiçoar o desenvolvimento destas aplicações, são as metodologias ágeis. Dentre diversos tipos procura-se uma que proporcione maior identificação com o desenvolvimento da aplicação, no caso deste o *Scrum.* Possibilitando um melhor desempenho e gerenciamento do projeto, permitindo a criação de um software em menos tempo e com maiores chances de sucesso (ETTINGER, 2012).

Com a utilização do *Scrum* permitiu-se aplicar outro conceito para obter um gerenciamento visual de cada passo dado na criação da aplicação, o método *Kanban.* Com este é possível verificar o projeto em todas as etapas e poder alocar uma nova tarefa no desenvolvimento da aplicação (GHISI, 2012).

Atualmente existem diversos aplicativos que auxiliam de alguma forma na vida das pessoas, sendo que muitos destes são voltados diretamente para a área da saúde e lazer, atendendo as necessidades de cada individuo (BONOME et al, 2012).

Hoje em dia o número de pessoas que pratica algum tipo de atividade física está aumentando (BRASIL, 2014). Este número vem crescendo principalmente devido à busca pela perda de peso e ganho de massa muscular, consequentemente procurando estar próximo do seu peso ideal e assim enfatizar um corpo esteticamente bonito e saudável (ARAUJO et al, 2007; Castro, 2007).

No entanto, as pessoas praticam estes exercícios físicos sem saber qual o gasto calórico exercido. Com isto surgi à necessidade de uma tecnologia de fácil portabilidade, simples manuseio, rápido acesso e que possibilite informar aos usuários o dispêndio energético resultado em cada atividade desempenhada. Isto auxiliará no que diz respeito a motivar as pessoas a continuarem a prática de atividades físicas.

Para atender a todos os requisitos do trabalho teve-se a utilização de um conjunto de dois métodos, citados a cima, desta forma podem ser minimizados os riscos no desenvolvimento, agilizando os processos com menos foco em documentação e mais em implementação, além de permitir mudanças no decorrer do projeto e possibilitar a entrega mais frequente das atividades desenvolvidas, assim pode-se atingir um elo de confiança maior com o usuário.

2 Dispositivos móveis

Dispositivos móveis podem ser definidos como aparelhos que tenham um tamanho reduzido, para serem transportados com maior facilidade, possuam uma capacidade de processamento e ainda permitam a troca de informações via rede. No entanto, outras características também são importantes, como apresentar uma bateria de longa duração, para que não seja necessário estar conectado a rede elétrica, interferindo na mobilidade do mesmo, e ainda ter acesso a tecnologia de rede sem fio (FIGUEIREDO; NAKAMURA, 2003).

A usabilidade destas tecnologias está aumentando devido à disponibilidade de tantos recursos. De acordo com pesquisas realizadas, com pessoas de 10 anos ou mais, em 2011, existiam 115,4 milhões de pessoas (69,1%) que tinham telefone celular para uso pessoal (IBGE, 2013).

3 Sistema operacional android

O Android é um SO móvel baseado em uma versão modificada do Linux. Foi originalmente desenvolvido por uma equipe com mesmo nome, Android, Inc. (LEE, 2011).

O Android foi desenvolvido inicialmente para *smartphones*, no entanto hoje é utilizado em diversos dispositivos. Seu SO foi baseado no *Kernel* 2.6 do Linux, responsável por realizar todo o controle da memória. Desta forma o *Kernel* consegue gerenciar processos e aplicativos, que podem ser executados simultaneamente, *threads* dos processos e a segurança dos arquivos e pastas (LECHETA, 2013).

A arquitetura do Android é formada basicamente por cinco camadas, conforme mostra a figura 7 (GOOGLE, 2014).

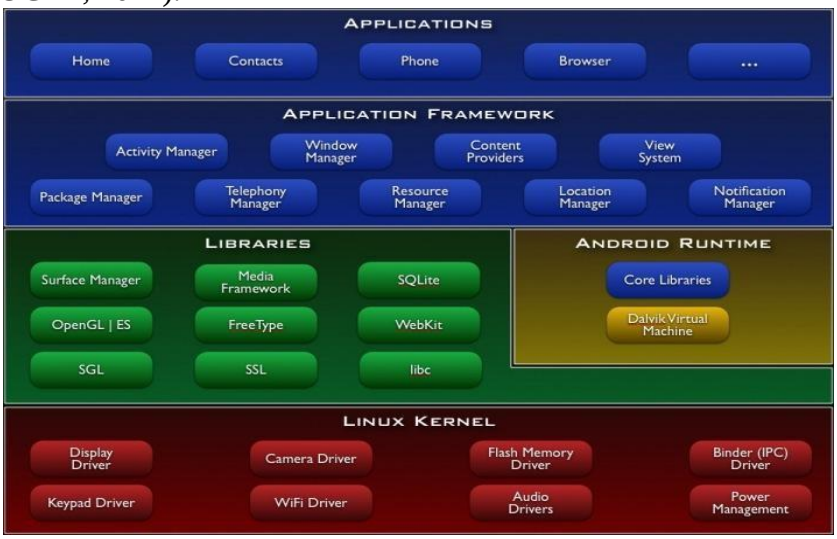


Figura 1 - Arquitetura do Android.

Na camada de aplicações se encontram todos os aplicativos essências do Android. Na camada *Frameworks* são encontradas todas as APIs e recursos que são utilizados pelos aplicativos. O *Runtime* é uma instância da Máquina Virtual *Dalvik*, que é criada para cada aplicação executada no Android. A última camada é o Linux Kernel, responsável por serviços centrais do sistema, como segurança, gestão de memória e processos (PEREIRA; SILVA, 2009).

4 Engenharia de software

Segundo Pressman (2011) a engenharia de software consiste em diversos princípios, métodos, conceitos e ferramentas que são recorridos no planejamento e desenvolvimento de um software. A utilização desta prática permite que a criação de programas computacionais seja mais organizada, proporcionando seguir um roteiro que pode detalhar cada etapa e assim obter uma chance maior de sucesso.

4.1 Metodologias e conceitos ágeis

Metodologias ágeis e conceitos podem ser definidos como métodos que melhorem as aplicações, capazes de aperfeiçoar e agilizar os processos no desenvolvimento destes softwares. Estes métodos buscam implementações em curtos períodos, minimizando os riscos e tornando o desenvolvimento eficiente e rápido (JANDOZA; PRADO, 2013).

4.1.1 Método ágil *Scrum*

Scrum é considerado um *framework* utilizado para gerenciar o desenvolvimento ágil de softwares, no qual se podem aplicar diversas técnicas ou processos, podendo ser usado para identificar o que é necessário ser feito para desenvolver software de qualidade em pouco tempo. Desta forma pode-se melhorar a eficácia das práticas de gerenciamento e desenvolvimento destes softwares (PEREIRA; TORREÃO; MARCAL, 2007). Além destas características o Scrum possui ótimas práticas de gerenciamento, estas, por sua vez, aceitam ajustes rápidos, acompanhamento e visibilidade constantes (FONSECA; CAMPOS, 2008).

Projetos que utilizam o método Scrum são compostos por equipes pequenas que desenvolvem suas atividades a partir de requisitos ou do *Product Backlog*, explicado mais a

baixo. O desenvolvimento do projeto é dividido em ciclos chamados de *Sprint*, este organiza o desenvolvimento de software em iterações de 2 a 4 semanas (ETTINGER, 2012).

Existem alguns conceitos do *Scrum* que possibilitam um melhor entendimento sobre o funcionamento do mesmo, além de papéis importantes dentro do projeto.

Papéis e fases do *Scrum:*

a) *scrum team*: é composto por *Product Owner*, equipe de desenvolvimento e o *Scrum Master*;

product owner: tem como responsabilidade gerenciar o *Backlog* do projeto, ou seja, definir os requisitos iniciais e gerais,

b) *scrum master*: responsável por coordenar a equipe de desenvolvimento e garantir que a rotina imposta pelo *Product owner* está sendo seguida(PEREIRA; TORREÃO; MARCAL, 2007);

c) *product backlog*: é uma lista de tudo o que foi especificado para ser desenvolvidas no projeto inteiro, prioridades de itens, requisitos e tudo que envolve o projeto como um todo;

d) *sprint backlog*: são itens do *Product Backlog* que serão desenvolvidos em uma sprint. Assim podem ser definidas as funcionalidades que serão implementadas de forma separada;

e) *sprint*: é uma repetição onde novas funcionalidades são desenvolvidas e acrescentadas ao projeto. A duração do *Sprint* é de até 30 dias, cada vez que um *Sprint* é finalizado outro começa automaticamente com novas funções para serem implementadas. Ao decorrer do período do *Sprint* não são feitas alterações que afetem o propósito final do mesmo, desta forma os riscos de complexidade e entrega final diminuem. O *Sprint* pode ser antecipado ou cancelado, porém apenas o *Product owner* tem este poder;

g) *daily meeting*: reunião diária feita com duração de no máximo 15 minutos, com o objetivo de avaliar o progresso do projeto;

h) revisão do *sprint*: esta é feita no final dos *Sprints*, assim pode-se analisar se o projeto final saiu como proposto no início, os pontos fundamentais para a conclusão do mesmo e os pontos negativos ocorridos, tendo como responsável pela avaliação o *Product Owner* (ETTINGER, 2012).

4.1.2 Kanban

O *Kanban* é um conceito ou abordagem que tem como objetivo transformar o trabalho em desenvolvimento visível para toda equipe. Deste modo é possível sinalizar as atividades em andamento, o progresso do projeto, os prazos de entregas, as dificuldades encontradas e erros cometidos (MARIOTTI, 2012).

O *Kanban* é composto por um painel de cartões, onde estes representam a capacidade limite acordada em cada fase de um projeto que são colocadas em circulação(MARIOTTI, 2012).

O modelo *Kanban* pode ser resumido em três etapas:

d) visualizar os processos;

e) limitar o trabalho em processo do inglês WIP (*work in progress*);

f) gerenciar do *lead-time*, ou seja, verificar o tempo que a atividade leva para passar por todas as fases até a sua entrega (MARIOTTI, 2012).

O sinalizador visual *Kanban* funciona como um quadro de sinalização de processos, deixando visível o andamento do projeto. Este quadro de sinalização é composto pelas seguintes etapas: análise, desenvolvimento, aceitação e implementação, podendo estas ser adaptadas conforme as necessidades do projeto. Estas etapas são inseridas como colunas no quadro de visualização, assim cada coluna representa uma fase do desenvolvimento da tarefa, cada vez que a atividade é concluída em uma fase ela passa para a próxima, até estar pronta para a entrega (MARIOTTI, 2012).

Na figura 2 pode ser observado um exemplo de quadro visual *Kanban*, onde o número em cada coluna representa o máximo de cartões e as cores em cada cartão o responsável pela tarefa.

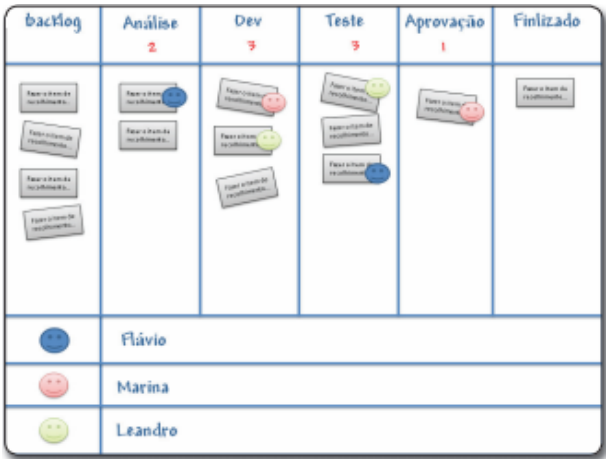


Figura 25 - Quadro visual Kanban.

Para que a utilização do *Kanban* em um projeto ou organização tenha sucesso e resultados rápidos é preciso seguir algumas etapas:

g) focar na qualidade;

h) limitar a quantidade de trabalho em progresso;

i) entregar frequentemente;

j) balancear demanda com a capacidade máxima;

k) priorizar tarefas;

l) atacar fontes de variedade (GHISI, 2012).

Estas devem ser obedecidas na ordem em que estão.

5 Atividades físicas

Movimentos corporais produzidos pelos músculos, que proporcionam um aumento no gasto de energia acima do normal são considerados atividades físicas. A prática regular de atividades é um dos principais componentes na prevenção do crescimento mundial de doenças crônicas. Atualmente existem informações suficientes, sobre como a realização destes exercícios são benéficos, para que todo individuo exerça alguma atividade e melhore sua condição, tanto física como de saúde (MAREGA; MALUF, 2012).

5.1 Quilocalorias

O corpo humano está constantemente queimando quilocalorias para manter ativas as atividades e funções realizadas diariamente, desta forma necessita sempre repor a energia gasta para manter sua sobrevivência, buscando a mesma em proteínas, gorduras, carboidratos e álcool disponibilizados na digestão de alimentos (CAMPOS, 2002).

5.2 Dispêndio energético

O dispêndio energético nada mais é que o gasto das quilocalorias ingeridas. O corpo necessita de uma quantia de quilocalorias para seu funcionamento diário, todavia este consumo é diferente entre homens e mulheres (CAMPOS, 2002).

O gasto calórico para alguém que pratica atividade física, ou que possui uma massa muscular mais elevada, é gradativamente maior, dependendo da frequência que exerce alguma atividade, do peso e dos tipos de atividades realizadas, com tudo, também é preciso uma absorção maior de calorias para manter estas funções ativas (CLAYTON; VANDERKAM, 2006).

As atividades físicas são fundamentais para se alcançar um dispêndio energético maior que o normal, assim como a alimentação, que também está diretamente ligada a este fato (SOARES, 2008).

7 Calorias control – protótipo de aplicativo para acompanhamento do dispêndio energético com utilização das técnicas scrum e kanban

O presente estudo mostrou como resultado a eficiência da utilização da metodologia ágil *Scrum* em conjunto com o *Kanban*, aplicados na construção de um protótipo que possibilita um controle de dispêndio energético, dos usuários, em atividades físicas, tornando o desenvolvimento do projeto ágil e qualitativo.

Os métodos pesquisados e utilizados contribuíram de forma significativa em todo o processo da criação do software, proporcionando um acompanhamento das etapas do projeto, tornando-as visíveis e permitindo detectar erros logo no início, contribuindo assim com uma rapidez na evolução do mesmo.

O aplicativo criado recebe os dados inseridos pelos usuários e possibilita um monitoramento do dispêndio energético obtido em atividades praticadas.

7.1 Metodologia

Para que fosse possível desenvolver o projeto de forma mais organizada, ágil e qualitativa, foram estudados os métodos *Kanban* e *Scrum*, mesclando-os e aplicando as regras de cada um conforme as necessidades para a criação do projeto. Por fim, foram apresentadas neste trabalho, de forma mais breve, assuntos sobre o dispêndio energético, descrevendo os pontos mais importantes e utilizados na aplicação.

O projeto teve início com a aplicabilidade das regras do *Scrum*, utilizando o *Kanban* em conjunto quando preciso, possibilitando melhor organização nas etapas. Com isto, foi feita uma reunião inicial para definir todas as funcionalidade e pontos a serem produzidos, após esta foram feitas *Sprints* para produzir todas as fases do projeto de forma separada e organizada.

7.1.1 Aplicabilidade do *Scrum* e *Kanban*

Para o desenvolvimento bem sucedido do protótipo foram estabelecidas regras a serem seguidas, estas convém dos métodos utilizados, visando proporcionar organização, qualidade no aplicativo, rapidez na construção e visibilidade das etapas, permitindo que o desenvolvimento seja eficiente e proporcionando um limite de tarefas suportável, em um determinado limite de tempo, para serem produzidos.

7.1.2 Definição dos papéis no *Scrum*

Inicialmente, com o objetivo de organização foram estabelecidos os papéis dentro do *Scrum* para a equipe, definindo o coorientador do presente trabalho como o *Product Owner*, o orientador como o *Scrum Master*, e por fim, o acadêmico na parte do desenvolvimento.

7.1.3 Reunião de definição do *Product Backlog*

Foram verificadas as informações necessárias para o funcionamento do aplicativo, estabelecendo o fluxo de informações entre o usuário e o protótipo, como pode ser verificado na figura 3. Com isto, foram definidas as informações que seriam importantes retornar para o usuário.

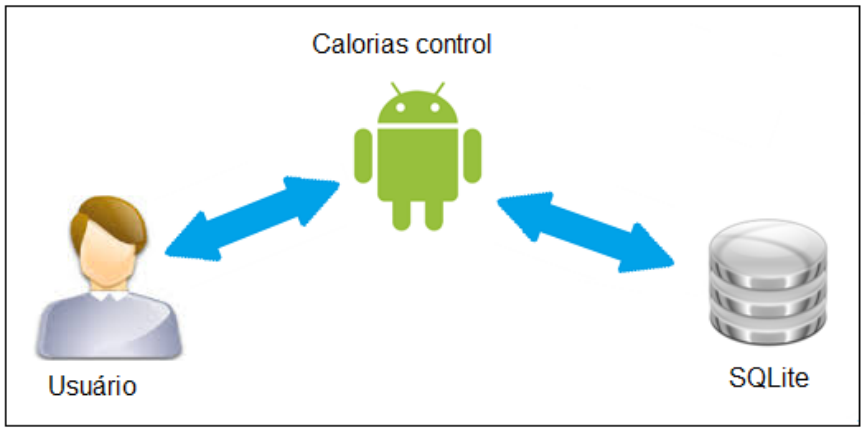


Figura 26 - Modelagem do fluxo de informações do aplicativo.

Como pode ser visto na figura 3 existe um envio e recebimento de informações entre o usuário, aplicação e o banco de dados, onde o usuário entra com os dados na aplicação e a mesma apresenta os retornos citados anteriormente, entre outros. Estas informações são salvas e retroalimentadas no banco de dados SQLite e podem ser consultadas pelo usuário.

Posteriormente, com uma gerência maior do *Scrum Master*, mas com o acompanhamento do desenvolvedor e do *Product Owner* foram analisadas as etapas, para a produção do protótipo, de uma forma mais prática. Estas consistiam em desenvolver um diagrama de casos de uso, um modelo dos protótipos de *wireframes* e uma modelagem do banco de dados.

7.1.4 Desenvolvimento da primeira *Sprint*

A primeira *Sprint* identificou como objetivo a divisão de itens prioritários e de grande necessidade para o inicio do desenvolvimento do projeto, permitindo que fossem desenvolvidas etapas interessantes para a obtenção de sucesso no final.

7.1.4.1 Reunião de planejamento da primeira *Sprint*

Nesta fase foi reunida toda a equipe novamente para identificar itens importantes do *Product Backlog* e formar a primeira *Sprint Backlog*. Deste modo, conforme a equipe propôs foram selecionados para serem desenvolvidos inicialmente na primeira *Sprint* o caso de uso da aplicação, a modelagem das telas e a modelagem do banco de dados.

7.1.4.2 Desenvolvimento da primeira *Sprint*: modelagem

Neste trabalho foram utilizadas ferramentas gratuitas, aplicadas na modelagem de caso de uso, na modelagem do banco de dados e na criação de *wireframes*, com o intuito de apresentar algumas telas.

7.1.4.2.1 Modelagem do caso de uso

O desenvolvimento do caso de uso se deu pela manipulação da linguagem *Unified Modeling Language* (UML), esta modelagem apresenta de forma visual os artefatos da construção do aplicativo, auxiliando o conhecimento da aplicação.

A figura 4 demonstra o diagrama de caso de uso da aplicação, onde o ator representa o usuário em comunicação com a aplicação.

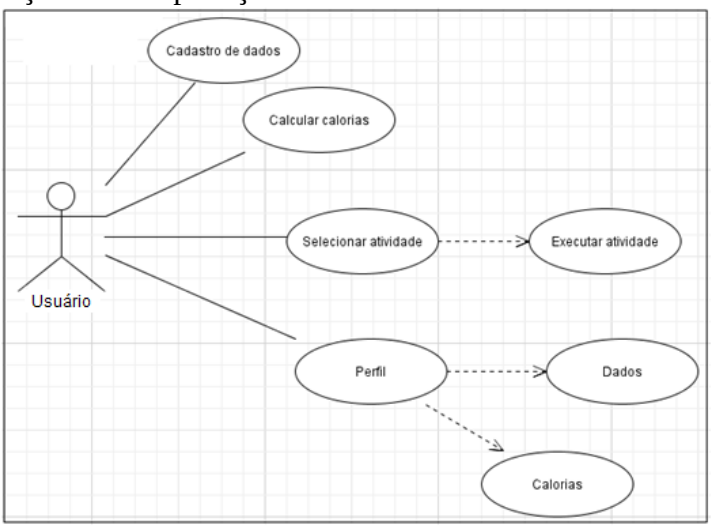


Figura 27 - Diagrama de Caso de Uso da aplicação.

Conforme pode ser visualizado na figura 4 o usuário tem comunicação direta com a aplicação, podendo navegar entre as telas inserindo e consultando dados. Os casos de uso inseridos com ligação direta ao usuário são considerados da tela inicial do protótipo, proporcionando extensões para outras telas.

Com a criação do caso de uso da aplicação efetuado a etapa seguinte se refere à realização da modelagem das telas.

7.1.4.2.2 Modelagem das telas (wireframe)

Com o objetivo de criar as telas inicialmente, sem que houvesse grandes ajustes futuramente, se deu o uso da modelagem das mesmas. Desta forma foram construídos *wireframes* utilizando uma ferramenta gratuita, sem que precisasse desenvolver diretamente no XML da aplicação, assim foi possível evitar atrasos e seguir os padrões de telas modelados.

Na figura 5 pode ser visualizada a tela inicial do aplicativo, esta apresenta quatro botões, os quais possuem ações de redirecionamento para outras telas.

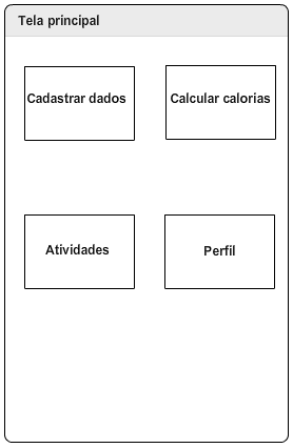


Figura 28 - Wireframe da tela principal.

A tela inicial da aplicação foi modelada com o intuito de disponibilizar ao usuário uma seção de escolhas. Estas são representadas pelos botões que permitem a navegação entre as opções da aplicação.

Posteriormente à modelagem de alguns *wireframes* que constituem a aplicação foi possível modelar o banco de dados, facilitando assim a criação do mesmo no software.

7.1.4.2.3 Modelagem do banco de dados

De modo a construir a base de dados da aplicação, com a possibilidade de visualizar as tabelas e analisar a forma coerente de inserção dos dados foi modelado o banco de dados. Desta forma foi possível ter uma visão do comportamento dos dados.

A modelagem do banco de dados é apresentada na figura 7 no seu modelo físico.

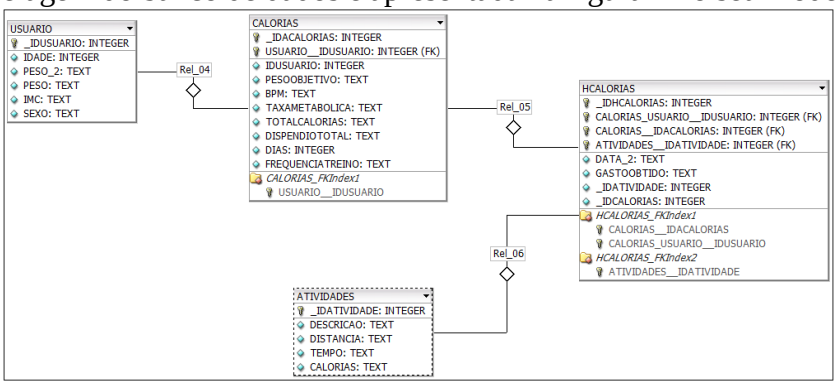


Figura 7 - Modelagem do banco de dados.

Conforme mostra a figura é possível verificar todas as ligações entre as tabelas e os respectivos campos de cada.

Esta modelagem permitiu o entendimento e melhor organização dos campos. E assim facilitar na criação do banco, além de permitir visualizar a ligação entre as tabelas.

7.1.5 Desenvolvimento da segunda *Sprint*

A segunda *Sprint* deu início após o termino da primeira, buscando reunir toda a equipe novamente para uma nova seleção de itens do *Product Backlog.*

7.1.5.1 Reunião de planejamento da segunda *Sprint*

Primeiramente, como antes, foi estabelecida uma nova reunião, porém com novos itens e necessidades a serem desenvolvidos, acrescentados os itens propostos pelo *Product Owner* na reunião de revisão da primeira *Sprint.*

Foi determinada para a segunda *Sprint* a criação do banco de dados na aplicação, a criação das telas de cadastro de dados e calculo das calorias, no XML da aplicação, e as funcionalidades das mesmas.

Como propósito de qualificar e agilizar ainda mais os processos de desenvolvimento aplicou-se as regras do *Kanban*, estas foram utilizadas no modo de desenvolvimento e na determinação do tempo de realização da *Sprint*, onde se procurou estimar quinze dias para a finalização dos itens selecionados, outras regras aplicadas foram na priorização de tarefas e na escrita de testes antes da implementação.

7.1.5.2 Implementação da segunda *Sprint*

Primeiramente, foi realizada a criação das tabelas do banco de dados, concebendo a criação de quatro classes em um pacote, onde cada uma destas apresenta o seu SQL de criação.

Após a criação do banco estabelecida foi iniciada a criação das telas definidas para esta *Sprint* e suas funcionalidades, utilizando um quadro visual *Kanban* para o desenvolvedor e *Scrum Master* acompanharem em conjunto os passos de cada tarefa a ser desenvolvida.

O quadro visual *Kanban* foi modelado conforme os requisitos do projeto e adicionadas tarefas determinadas na *Sprint Backlog*. Assim foi possível controlar as atividades que estavam sendo realizadas e adicionar novas sem ocasionar interrupções. O quadro visual *Kanban* inicial pode ser verificado na figura 8.

Sprint Backlog	Análise	Desenvolvimento	Testes	Aprovação	Finalização
Layout da tela de Cadastro de dados Layout da tela de Cálcular calorias Funcionalidades da tela principal Funcionalidades da tela de Cadastro de dados Funcionalidade da tela deCalcular cálorias	Layout da tela principal				
Desenvolvedor	X	X	X		X
Scrum Master				X	

Figura 8 - Quadro visual Kanban inicial da segunda Sprint.

Conforme mostra o quadro *Kanban* existem diversas etapas por onde cada tarefa passa, sendo que a aprovação é concedida pelo *Scrum Master*, pelo fato de produzir uma validação do que foi produzido.

Inicialmente, a primeira tarefa foi a de construir o *Layout* da tela principal, posteriormente às demais definidas no quadro *Kanban*. As elaborações das telas foram baseadas nas modelagens dos *wireframes*, assim, estas passaram pela etapa de análise e foram introduzidas na etapa de desenvolvimento de forma separadas, sendo produzida unitariamente, assim uma nova tarefa começa apenas quando a outra for finalizada, respeitando a regra do *Kanban* de limitar a quantidade de trabalho em progresso.

As próximas tarefas a serem realizadas foram às implementações das funcionalidades de cada tela.

Para realizar a implementação das funcionalidades foram criadas novas classes para cada tela, assim foi possível manipular os campos elaborados no XML acessando a classe R do Android. Com o intuito de organizar de forma correta foram criadas classes de controle referentes a cada classe de tela, estes controles são responsáveis por organizar os dados e enviar requisições de inserções ou consultas para a classe Banco. Esta tem finalidades como as de abrir e fechar as conexões com o banco de dados, realizar inserções e consultas, as quais são retornadas as classes de controles.

De modo a qualificar as implementações foram realizados os testes, apresentando as telas criadas no XML e as funcionalidades desenvolvidas. Com os testes escritos logo no início e a adição de outros no decorrer da implementação foi possível rever todos os possíveis equívocos e revisar o código fonte. Quando eram encontrados erros nas funcionalidades, estas voltavam para a etapa de desenvolvimento até estarem aptas para a aprovação e finalização.

Permitindo obter um foco maior melhor no projeto outra regra do *Kanban* foi utilizada, no que diz respeito a atacar fontes de variedades, isto possibilitou deixar o desenvolvedor focado no projeto, retirando alguns itens que poderiam afetar a concentração, tornando melhor e mais ágil o desenvolvimento.

7.1.6 Desenvolvimento da terceira *Sprint*

Assim como as anteriores, a terceira *Sprint* procurou reunir toda a equipe novamente para analisar os itens prioritários a serem desenvolvidos. Porém, como a demanda restante era pequena possibilitou a finalização de todo o *Product Backlog* na terceira *Sprint*.

Esta *Sprint*, por ser parecida com a segunda, obteve a aplicação das mesmas regras, unindo *Scrum* e *Kanban* para realizar a implementação das telas faltantes.

7.1.6.1 Reunião de planejamento da terceira *Sprint*

Esta *Sprint* proporcionou a criação das telas de perfil e atividades, buscando implementar posteriormente as funcionalidades referente as mesmas e funções para unir o protótipo por completo.

A realização desta *Sprint* se deu com a utilização de regras do *Kanban* novamente e principalmente com a utilização do quadro visual, este que proporcionou uma organização importante e o conhecimento de etapas a serem percorridas, dentre outras regras aplicadas também na segunda *Sprint* que possibilitaram melhor entendimento do progresso do projeto.

Por fim, foi essencial fazer um teste completo do aplicativo, buscando avaliar se todas as funcionalidades aplicadas estavam coerentes.

7.1.6.2 Implementação da terceira *Sprint*

A terceira *Sprint* iniciou com a produção do quadro visual *Kanban*, procurando deixar explicitas as tarefas desenvolvidas, assim o *Scrum Master* obtia um controle do crescimento do projeto, bem como seus atrasos.

Cada tarefa passou por todas as etapas do quadro *Kanban* de forma individual, assim o tempo de desenvolvimento de cada tarefa foi específico para ela, contribuindo desta forma com a qualidade do software.

Após o desenvolvimento de cada funcionalidade se deu a realização do teste de cada implementação de forma separada, primeiramente revisando o código e as funções que a própria tela executa, com o objetivo de enviar para aprovação sem erros.

Com a finalização de todas as tarefas da *Sprint* foi possível realizar um teste do protótipo por inteiro a fim de qualificar o mesmo. Deste modo, os testes contribuíram para revisar os possíveis equívocos que poderiam ocorrer.

7.1.7 Reuniões diárias das *Sprint* (*Daily Meeting*)

Estas reuniões foram executadas com encontros entre o desenvolvedor e *Scrum Master,* onde aconteciam de forma pessoal, *online*, via troca de emails, entre outros meios de comunicação, sendo executadas não diariamente, mas três vezes por semana. Com isto foi possível identificar erros que poderiam ocasionar o atraso do projeto e possibilitar a coordenação sobre o desenvolvedor, determinando passos para aperfeiçoamento das etapas.

7.1.8 Revisão das *Sprint*

Estas revisões, de modo geral, foram realizadas de modo a reunir toda a equipe em uma reunião e apresentar ao *Product Owner* as etapas produzidas. Todavia, possibilitou explicar ao mesmo, detalhes da aplicação, dentre estes como seria o funcionamento e as informações retratadas em cada tela.

7.1.9 Retrospectiva das *Sprint*

A retrospectiva das *Sprinst* concedeu reunir o desenvolvedor e *Scrum Master* com o intuito de diagnosticar os pontos negativos e positivos das *Sprints.* Nestas reuniões foram discutidos os passos alcançados e analisado os pontos a serem melhorados para as *Sprints* posteriores.

7.2 Resultados obtidos

Como a utilização do *Scrum* e *Kanban* foi possível executar a produção de cada lista de itens de forma organizada e com maior rapidez. Outro fator importante foi a participação da equipe em todo o projeto acrescentando de forma significativa, pelo fato de orientar o desenvolvedor quando necessário e auxiliar nas entregas das tarefas.

Com a aplicação das regras do *Kanban* foi possível desenvolver com maior qualidade o protótipo, proporcionando ao desenvolvedor uma forma de implementar sem interrupções e com um tempo suportado pelo mesmo, além de mostrar de forma visual as etapas de cada tarefa.

Deste modo, a aplicabilidade do *Scrum* e *kanban* resultou na produção do projeto com maior qualidade, organização e agilidade proporcionando o desenvolvimento de um protótipo de aplicativo que tem como objetivo permitir ao usuário acompanhar o dispêndio energético obtido em determinadas atividades físicas.

A aplicação apresenta inicialmente a tela principal, onde é possível acessar as funcionalidades do aplicativo, cadastrando os dados do usuário e calculando as devidas calorias que o mesmo pretende perder, posteriormente podem ser selecionadas as atividades que serão executadas e consultar os dados no perfil.

Nas telas de cadastro de dados e cálculo das calorias são salvos os dados do usuário para posteriormente gerar atividades para serem executadas.

Após todos os dados cadastrados pode-se acessar a tela para selecionar as atividades. Esta tela possibilita a visualização das atividades, e características, como tempo ou distância e calorias gastas com a atividade. Deste modo, o usuário pode escolher uma atividade e registra-la como executada.

Desta forma, o usuário pode registrar atividades diversas vezes, ou seja, marcar como feita uma ou mais atividades, onde as calorias são salvas no protótipo e podem ser visualizadas na tela de perfil.

As informações disponibilizadas nas telas de perfil permitem a visualização dos dados inseridos e o peso que já foi consumido pelo usuário. Deste modo, o protótipo procura verificar a quantidade de calorias gasta em cada dia, caso encontre um dia com o dispêndio energético menor do que o retornado anteriormente pelo software o mesmo é apagado e acrescentado um dia a mais para obter este gasto.

A construção e implementação das telas de forma separada proporcionou um foco maior em cada etapa, permitindo priorizar as implementações mais importantes.

Os itens estabelecidos em cada *Sprint* do *Scrum* eram realizados aplicando regras do *Kanban*, assim, os prazos estabelecidos para entrega das *Sprints* eram balanceados conforme a capacidade do desenvolvedor, isto permitiu tempo para realizar testes e revisões do que foi desenvolvido, proporcionando um foco na qualidade.

A cada final de *Sprint* eram entregues os itens desenvolvidos ao *Product Owner*, proporcionando entregas frequentes ao mesmo. Este princípio fez com que o *Product Owner* criasse uma segurança no desenvolvimento do projeto.

Com isto, pode-se concluir que todos os objetivos do presente projeto foram cumpridos, de modo que estes exigiam a aplicabilidade dos métodos *Scrum* e *Kanban* em conjunto, para realizar o desenvolvimento de um protótipo de aplicativo utilizando o sistema operacional Android.

Conclusão

Foi aplicado todo o estudo realizado, dos métodos citados, na construção do protótipo, identificando que estes contribuem de forma positiva na implementação de sistemas, auxiliando na criação de um aplicativo que oferece monitoramento do dispêndio energético de praticantes de atividades físicas.

Com tudo, pode ser concluído que a utilização dos métodos em conjunto para o desenvolvimento de um protótipo obteve sucesso, proporcionando uma produção com demanda de tempo suportada pelo desenvolvedor, sem interrupções, um visualização física da implementação das etapas, um desenvolvimento qualitativo, com maior agilidade e organização em todos os passos do projeto. Os métodos citados podem ser integrados de forma fácil, onde um contribui com o outro, permitindo amplificar as chances de sucesso no final.

REFERENCIAS

VETTORAZZI, Felipe J; SCHUTZ, F. **Taxi calculator:** Aplicação android para controle de rotas de táxi**. Revista Eletrônica Científica Inovação e Tecnologia,** Paraná, v. 2, n. 2, p.48-59, 6 mar. 2012. Disponível em: <http://revista.md.utfpr.edu.br/sis/index.php/IT/article/viewFile/208/pdf>. Acesso em: 15 Nov. 2014.

FONSECA, Isabella; CAMPOS, Alberto. **Por que Scrum?** Engenharia de Software Magazine, Rio de Janeiro, Edição 4, p. 30-35, 2008.

JANDOZA, William; PRADO, Antonio Francisco do. **Scrum em Aplicações Móveis.** São Carlos, v. 2, n. 3, p. 211 -21 9, set-dez 201 3.

ETTINGER, D.; A engrenagem do scrum. 2012 disponível em: http://danielettinger.com/2011/04/06/a-engrenagem-do-scrum/; acessado em: julho/2012.

PEREIRA, Paulo; TORREÃO, Paula; MARCA, Ana Sofia. **Entendendo Scrum para Gerenciar Projetos de Forma Ágil.** 2007.

MARIOTTI, Flavio S. Kanban: o ágil adaptativo.**Engenharia de Software Magazine**, ed 45 p. 6-10, 2012.

CLAYTON, David; VANDERKAM, Laura. **Como ser saudável sem ser chato:** uma dieta para pessoas como você, saúde sem abrir mão dos prazeres. Rio de Janeiro: Elsevier, 2006. 200 p. Tradução de Ana Beatriz Rodrigues.

CAMPOS, Marcos Vinhal (Org.). **Atividade física passo a passo:** saúde sem medo e sua preguiça. Brasília: Thesaurus, 2002. 248 p.

MAREGA, Marcio; MALUF, José Antônio. **Manual de atividades físicas para prevenção de doenças.** Rio de Janeiro: Elsevier, 2012. 304 p.
PEREIRA, Lúcio Camilo Oliva; SILVA, Michel Lourenço da. **Android para desenvolvedores.** Rio de Janeiro: Brasport, 2009. 240 p.

GOOGLE. **Android Developers**. Disponível em: <http://developer.android.com/>. Acesso em: 15 Out. 2014.

FIGUEIREDO, Carlos Maurício Seródio; NAKAMURA, Eduardo. **Computação Móvel:** Novas Oportunidades e Novos Desafios. **T&C Amazônia**, Manaus, n. 2, p.16-28, 01 jun. 2003. Semestral. Disponível em: <https://portal.fucapi.br/tec/imagens/revistas/ed02_completo.pdf> Acesso em: 02 set. 2014.

IBGE. **PNDA:** De 2005 para 2011, número de internautas cresce 143,8% e o de pessoas com celular, 107,2%. Disponível em: <http://saladeimprensa.ibge.gov.br/noticias?view=noticia&id=1&idnoticia=2382&busca=1&t=pnad-2005-2011-numero-internautas-cresce-143-8-pessoas-celular-107-2>. Acesso em: 08 Set. 2014.

LEE, Wei-meng. **Beginning Android™ Application Development**. Indianápolis: Wiley, 2011.

GHISI, Thiago. Kanban no desenvolvimento de projetos de software.**Engenharia de Software Magazine**, ed 45 p. 11-16, 2012.

BONOME, Karoline da Silva et al. Disseminação do uso de aplicativos móveis na atenção à saúde. In: CONGRESSO BRASILEIRO EM INFORMÁTICA EM SAÚDE, 13., 2012, São Paulo. **Anais...** 2012. p. 1-6. Disponível em: <http://www.sbis.org.br/cbis2012/arquivos/807.pdf>. Acesso em: 15 Nov. 2014.

BRASIL, (Gov.). **Pesquisa revela aumento na prática de atividades físicas.** Disponível em: <http://www.brasil.gov.br/saude/2014/05/pesquisa-revela-aumentona-pratica-de-atividades-fisicas>. Acesso em: 19 out. 2014.

SALABERRI, Diogo Bonoto. **Desenvolvimento de Aplicações Embarcadas com Android.** 2011. 52 f. Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) - Universidade Federal de Pelotas, Pelotas. Disponível em: <http://inf.ufpel.edu.br/nopcc/lib/exe/fetch.php?media=monografias:2010:2010-monodiogo_salaberri.pdf>. Acesso em: 15 Nov. 2014.