MICHEL VAN VLIET

CONTROLE DE FREQUÊNCIA PARA ANDROID

LAVRAS - MG

2013

MICHEL VAN VLIET

CONTROLE DE FREQUÊNCIA PARA ANDROID

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso de Sistemas de Informação para obtenção do título de Bacharel em Sistemas de Informação**.**

Orientador

Bruno de Oliveira Schneider

LAVRAS-MG

2013

MICHEL VAN VLIET

CONTROLE DE FREQUÊNCIA PARA ANDROID

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso de Sistemas de Informação para obtenção do título de Bacharel em Sistemas de Informação.

APROVADA em 20 de Agosto de 2013

PhD. André Vital Saúde UFLA

PhD. José Monserrat Neto UFLA

Orientador

MSc. Bruno de Oliveira Schneider

LAVRAS-MG

2013

AGRADECIMENTO

À Universidade Federal de Lavras a ao Departamento de Ciência da Computação por todo conhecimento fornecido.

Ao professor Bruno de Oliveira Schneider pela oportunidade no desenvolvimento do projeto, pela atenção, paciência, tempo, confiança e dedicação cedidos ao longo do projeto.

Aos meus pais Wilhelmina A. M. van Noije van Vliet e Cornélio van Vliet por todos esses anos de paciência, confiança, amor e carinho.

Aos meus irmãos Marcela van Vliet e Arien van Vliet pelo apoio e auxilio em todos os momentos.

Aos meus amigos de república nos momentos de descontração e nos momentos difíceis pelo qual passei.

À minha namorada Luciana da Cunha Melo pelo apoio, persistência, incentivo e principalmente pela companhia.

RESUMO

Este projeto criou um aplicativo para realizar o controle de frequência dos alunos durante as aulas. A ideia básica do aplicativo é realizar a cobrança sem o uso de papel e de forma simples, diminuindo o tempo de todo processo e provendo um incentivo maior aos professores em realizar esta cobrança de forma eficaz. Com a sincronização dos dados pelo dispositivo móvel as falhas na verificação das listas de chamada podem ser reduzidas e ainda atualizações podem ser feitas pelo professor em tempo real. O aplicativo fornece as seguintes funcionalidades: importa os dados de arquivos CSV; insere e altera disciplinas; insere e altera alunos; registra os atrasos, abonos, presenças e ausências dos alunos; altera o status do aluno em diferentes datas; e ainda exporta os dados armazenados em arquivo CSV.

Palavras-chave: Controle de frequência. UML. Engenharia de *software*. JAVA. Android.

ABSTRACT

This work reports on the making of an attendance control application. Developed for Android cellphones, it is meant to eliminate paper control, reducing the time needed, providing an encouragement for teachers to undertake this activity in an effective way. Failures in the attendance control might be reduced because of data synchronization and manual edits that keep lists up to date. The application has the following features: import data from CSV files, insert and edit classes, insert and edit students, log presence and absence, log student tardiness, log excused absences, and export data to CSV files.

Keywords: Attendance control. UML. *Software* engineering. JAVA. Android.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface*
APK	*Android Package*
CSV	*Comma-Separated Values*
DAO	*Data Access Object*
DCC	Departamento de Ciência da Computação
DVM	*Dalvik Virtual Machine*
POJO	*Plain Old Java Object*
SIG	Sistema Integrado de Gestão
S.O.	Sistema Operacional
UFLA	Universidade Federal de Lavras
UI	*User Interface*
UML	*Unified Modeling Language*
VO	*Value Object*

SUMÁRIO

1. INTRODUÇÃO

Neste capítulo é apresentado a contextualização, motivação, objetivos e a estrutura geral do trabalho, o qual descreve todo o processo de criação e desenvolvimento do projeto.

1.1 Contextualização e Motivação

O interesse no aprendizado de uma nova tecnologia em sistemas operacionais, o Android, que foi um sistema inicialmente desenvolvido para aparelhos móveis, foi uma motivação para poder iniciar o projeto. O Android, assim como o iOS da empresa Apple tornaram os antigos celulares em mini computadores e mudaram a maneira como as pessoas interagem com dispositivos móveis, possibilitando a interação dos dispositivos com os computadores pessoais e permitindo a instalação de diversos aplicativos e jogos de forma eficiente e de grande utilidade. Pensando no aumento do número de alunos e a facilidade no uso de aparelhos móveis, surgiu a ideia de criar um aplicativo que facilitasse e agilizasse a verificação da frequência dos alunos durante as aulas.

É sabido a existência de aplicativos semelhantes ofertados na Play Store[1], sendo estes, desenvolvidos fora do Brasil e com objetivos mais específicos em outras áreas, como por exemplo, o aplicativo Teacher Aide Pro[2] que realiza o controle total das informações de um aluno, inclusive as notas. Outro aplicativo, com objetivo um pouco mais próximo deste projeto é o Attendance[3], o qual realiza

[1] Play Store: https://play.google.com/store

[2] Teacher Aide Pro: https://play.google.com/store/apps/details?id=com.glen.apps.TeacherAideProLite

[3] Attendance: https://play.google.com/store/apps/details?id=com.academics.attendance

o controle de frequência mas utiliza de planilhas armazenadas em nuvem, o que dificulta seu pois requer conexão com a internet para ser utilizado.

Além desses dois aplicativos que possuem mais pontos em comum existem outros: Attendance Roster; Attendance; Safety Attendance. Todos eles apresentam funcionalidades distintas e diferentes maneiras de lidar com as informações.

A ideia básica do aplicativo é realizar a verificação sem o uso de papel e de forma simples, diminuindo o tempo de todo processo e provendo um incentivo maior aos professores em realizar esta verificação de forma eficaz. É importante salientar que uma lista de frequência de alunos sofre muitas mudanças como por exemplo: mudança de alunos, turmas diferentes, horários diferentes para diferentes turmas, etc. Essas mudanças implicam em várias impressões para atualização das listas, algumas vezes de forma incorreta, tornando a verificação da lista de frequência um processo sujeito a falhas.

Com a sincronização dos dados pelo dispositivo móvel as falhas na verificação das listas de chamada podem ser reduzidas e, ainda, as atualizações podem ser feitas pelo professor em tempo real.

1.2 Objetivos

O objetivo no desenvolvimento do projeto é aplicar os conhecimentos específicos em engenharia de *software* e demais áreas de conhecimento de Sistemas de Informação e Ciência da Computação. Tem-se como meta, também, buscar o conhecimento em outras áreas, como a estruturação de dados, gerência de projetos, qualidade de *software*, e, ainda, o estudo da linguagem de programação JAVA sobre um novo Sistema Operacional: o Android.

Este trabalho produziu um aplicativo que facilita a realização do controle de frequência dos alunos com o uso de um dispositivo móvel (celular, *tablet*, etc.),

com sistema operacional Android. O sistema permite a redução de tempo em que esta cobrança é efetuada, podendo ser realizada de forma mais eficaz e minimizando possíveis erros.

1.3 Estrutura

Nos próximos capítulos serão abordadas quatro etapas da engenharia de *software*, alguns diagramas UML, além do conhecimento básico necessário sobre o Android e a linguagem de programação JAVA, assim como demais conceitos utilizados no desenvolvimento do projeto. No Capítulo 2 será apresentado o referencial teórico, descrevendo conceitos básicos para o entendimento do projeto. No Capítulo 3 será apresentado a metodologia utilizada para que o projeto pudesse ser concluído com êxito. Já no Capítulo 4 são mostrados os resultados do projeto, assim como as dificuldades encontradas durante sua fase de desenvolvimento e os benefícios gerados de acordo com as expectativas. No Capítulo 5 é descrito a conclusão das atividades realizadas. Por último, serão listadas as referências usadas para a realização do projeto e os apêndices como auxílio no entendimento e uso do aplicativo.

2. REFERÊNCIAL TEÓRICO

Neste capítulo são descritos os conceitos necessários para a realização deste projeto. Inicialmente é explicada a importância da engenharia de *software* e, posteriormente, os possíveis benefícios com o uso da UML - *Unified Modeling Language* (Linguagem de Modelagem Unificada), e ainda, uma breve descrição do Sistema Operacional “S.O.” Android e a linguagem JAVA.

2.1 *Stakeholders*

O conceito de *stakeholder* se faz necessário para a apresentação de alguns conceitos que serão vistos a seguir. De acordo com Freeman (1984), a definição de *stakeholder* é "qualquer grupo ou indivíduo que afeta ou é afetado pelo alcance dos objetivos de uma empresa". Seguindo este conceito, temos que a engenharia de *software*, assim como a criação e validação de um projeto são inteiramente relacionados com os *stakeholders*, já que os mesmos, interferem diretamente no desenvolvimento do *software* e seus resultados.

2.2 Engenharia de *Software*

"A engenharia de s*oftware* é uma disciplina de engenharia relacionada a todos os aspectos da produção de *software*, desde os estágios iniciais de especificação do sistema até sua manutenção, depois que este entrar em operação" (SOMMERVILLE, 2007).

De modo geral, a engenharia de *software* tem sua importância por vários fatores, todos eles relevantes à qualidade do *software*. Alguns desses fatores influenciam de forma direta e outros indiretamente na qualidade de *software*, como, por exemplo, na estrutura, organização e documentação.

A partir da Seção 2.2.1 são descritas as cinco atividades básicas de todo o processo de criação de um *software*. É válido lembrar que o que está descrito aqui não deve ser necessariamente seguido e muito menos tomado como regra obrigatória; são apenas passos comumente usados pela maioria dos desenvolvedores e coerentes com as boas práticas na produção de *software*.

2.2.1 Especificação de Requisitos

A especificação de requisitos, é considerada etapa fundamental para o início de um projeto. Segundo Sommerville (2007), uma das atividades da engenharia de requisitos é o contato com o cliente e os *stakeholders* de um sistema, para que se possa definir o escopo do projeto, seus serviços, restrições, observações, desempenho, etc.

Deve-se observar que a especificação de requisitos, difere-se da engenharia de requisitos, principalmente pelo fato da primeira tratar de uma etapa base relacionada a coleta de requisitos, já a engenharia de requisitos engloba todas as atividades de produção de um documento de requisitos, o qual possui as atividades de identificação; análise e negociação; especificação e documentação; e validação.

Existem diferentes maneiras de se realizar a coleta de requisitos e algumas delas são: entrevista, cenários, questionários, etc. Cada uma delas deve ser usada em diferentes situações, de acordo com o tipo de negócio.

Sommerville (2007) separa os requisitos em duas categorias distintas:

- **Requisitos Funcionais:** São requisitos que descrevem o que o sistema deve fornecer ou como certas funcionalidades do sistema devem ser realizadas.
- **Requisitos não funcionais:** São aqueles que influenciam no desenvolvimento do sistema, criando restrições ao mesmo.

Como resultado da especificação de requisitos, obtêm-se o documento de requisitos, no qual são descritos todos os detalhes referentes ao "produto": suas características internas, sua relação com os usuários, restrições e demais conceitos importantes para o projeto. Neste projeto os requisitos coletados são descritos como parte integrante do trabalho.

2.2.2 Projeto

Durante a fase de projeto, ocorre a criação de toda a estrutura do *software*. É nessa fase que são criados os diagramas que ajudam a visualizar a interação do cliente com o sistema e as melhores maneiras de se obter o melhor resultado possível.

Para a realização da modelagem do *software*, definição do modelo de processo, cronograma de desenvolvimento e teste, utiliza-se a UML, a qual é descrita na Seção 2.3.

2.2.3 Codificação ou Desenvolvimento

A codificação ou fase de desenvolvimento é a fase, na qual, de fato, todo o *software* é produzido. Existem, atualmente, diversos sistemas operacionais e para cada um deles deve-se estudar a linguagem de programação que melhor se aplica ao projeto a ser desenvolvido.

Existem, também, diferentes maneiras de se estruturar a programação, e qual desses modos será utilizado deve ser estabelecido durante a fase de projeto. Atualmente, o desenvolvimento tende a um conceito mais recente, que é o desenvolvimento ágil.

"O *software* é parte de quase todas as operações de negócio e, assim, é essencial que um novo *software* seja desenvolvido rapidamente para aproveitar as

novas oportunidades e responder às pressões competitivas" (SOMMERVILLE, 2007).

Para se enfatizar a aplicação desse conceito, Pressman (2005) destaca 12 características do desenvolvimento ágil, e algumas delas são:

- A maior prioridade é satisfazer o cliente através da entrega antecipada e contínua do *software*;
- os clientes interessados e desenvolvedores devem trabalhar juntos o máximo possível durante o projeto;
- o método mais eficiente e eficaz de transmitir informação para uma equipe de desenvolvimento é a conversa entre as pessoas;
- o *software* funcionando é a principal medida de progresso;
- a atenção contínua à excelência técnica e um bom design aumenta a agilidade;
- simplicidade: a arte de otimizar a quantidade de trabalho ainda não feito é essencial.

2.2.4 Teste e Validação

Durante a fase de teste e validação, todo o projeto desenvolvido é testado e validado. Isso se faz necessário visando a validação dos requisitos estabelecidos com as funcionalidades esperadas, obtendo-se o resultado para o cliente e *stakeholders*.

Durante essa fase, novos requisitos podem ser observados e novas funcionalidades são criadas visando a satisfação das partes interessadas. Observa-se que todo o processo de desenvolvimento é um processo cíclico, no qual todo o *software* deve ser revisado quantas vezes forem necessárias, dentro do prazo previamente estabelecido.

2.3 UML

A UML - *Unified Modeling Language* (Linguagem de Modelagem Unificada) é hoje a base de estruturação para a fase de projeto de *software*. Como o próprio nome diz, a UML é uma metodologia, ou uma descrição das melhoras práticas a serem seguidas durante a modelagem de um *software*. Segundo Guedes (2009) a UML não se trata de uma linguagem de programação, e sim de uma notação, a qual tem como objetivo auxiliar durante todo o processo anterior ao desenvolvimento do *software* propriamente dito, influenciando no comportamento, estrutura e dinâmica dos processos.

A UML é controlada pela OMG - *Object Management Group* (Grupo de Gerenciamento de Objetos) desde 1997, com a colaboração de diversas pessoas envolvidas no ramo da engenharia de *software*. Ressalta-se que sua versão atual é a 2.0, lançada em 2005, possuindo atualizações mais recentes em definições específicas.

Na UML, são definidos diversos diagramas e é importante descrevê-los para melhor entendimento do projeto. Nesta Seção são apresentados os diagramas que dizem respeito à maioria dos projetos, sem especificar todos os existentes, já que alguns deles são definidos para projetos de grande porte.

2.3.1 Diagrama de Casos de Uso

Segundo Guedes (2009), este diagrama é de fato um diagrama que aborda uma modelagem mais informal. É comumente utilizado na fase de requisitos e pode servir de base para os demais diagramas. Fornece uma ideia geral de como o sistema irá se comportar com os usuários, identificando os mesmos e suas respectivas funções em relação ao sistema. Um exemplo é apresentado na Figura 1.

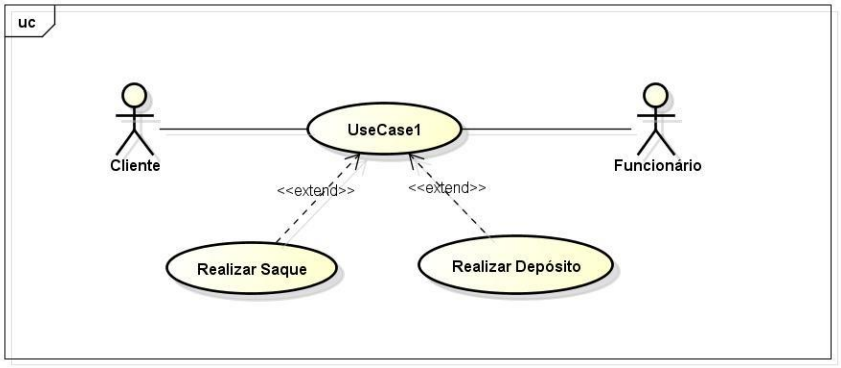

Figura 1 - Diagrama de Casos de Uso. Guedes (2009)

2.3.2 Diagrama de Classes

O diagrama de Classes é formado pela visualização das classes pertencentes ao sistema. As tabelas contém as funcionalidades de cada Classe, assim como o relacionamento entre elas. É o diagrama mais utilizado na maioria das situações e, segundo Guedes (2009), um dos mais importantes da UML.

Este diagrama é fundamental pelo fato de mostrar claramente onde cada funcionalidade do sistema está definida, facilitando a manutenção do sistema, assim como a adição de novas funcionalidades, como mostrado na Figura 2.

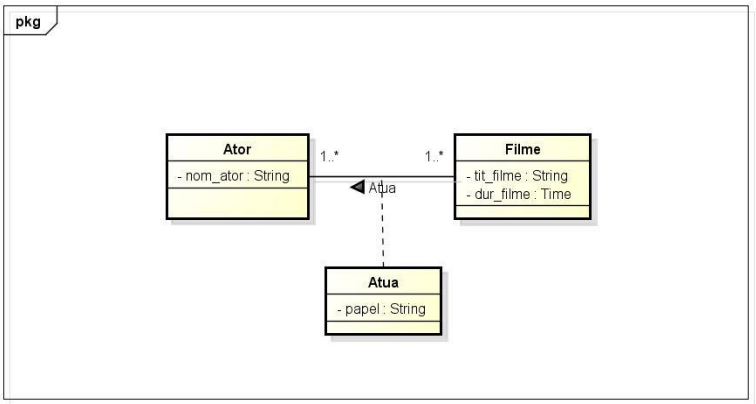

Figura 2 - Diagrama de Classes. Guedes (2009)

2.3.3 Diagrama de Objetos

Na Figura 3 é mostrado o diagrama, no qual são apresentados os valores que cada atributo da classe irá receber. O diagrama de Objetos é basicamente o mesmo modelo do diagrama de Classes, mas visualizado com valores de exemplo.

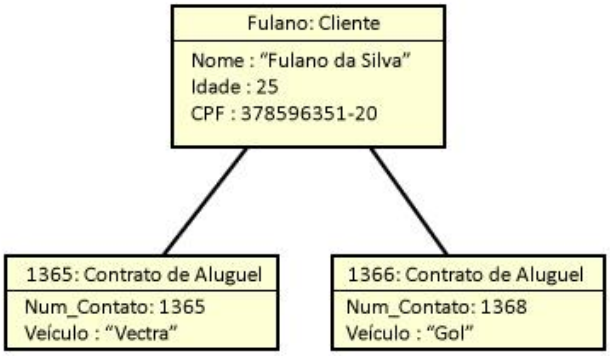

Figura 3 - Diagrama de Objetos

2.3.4 Diagrama de Pacotes

Segundo Guedes (2009), o diagrama de pacotes mostra os subsistemas englobados no sistema geral, de forma a determinar as partes que o compõem. Na Figura 4 é mostrado um exemplo. Este diagrama também pode ser utilizado para definir camadas de um *software* ou de um processo de desenvolvimento. Pode-se dizer que o diagrama de pacotes é a visualização das bibliotecas utilizadas ou criadas para o projeto.

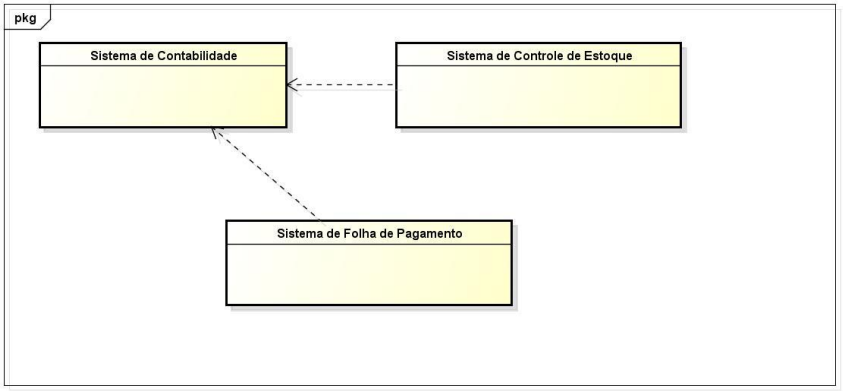

Figura 4 - Diagrama de Pacotes. Guedes (2009)

2.3.5 Diagrama de Sequência

O diagrama de sequência é utilizado para que se possa estabelecer a ordem temporal com que as classes são relacionadas de acordo com as interfaces criadas. Este diagrama é criado com o relacionamento dos diagramas de caso de uso e diagrama de classes, criando ações hipotéticas dos usuários e observando seu comportamento até sua conclusão, como pode ser observado na Figura 5.

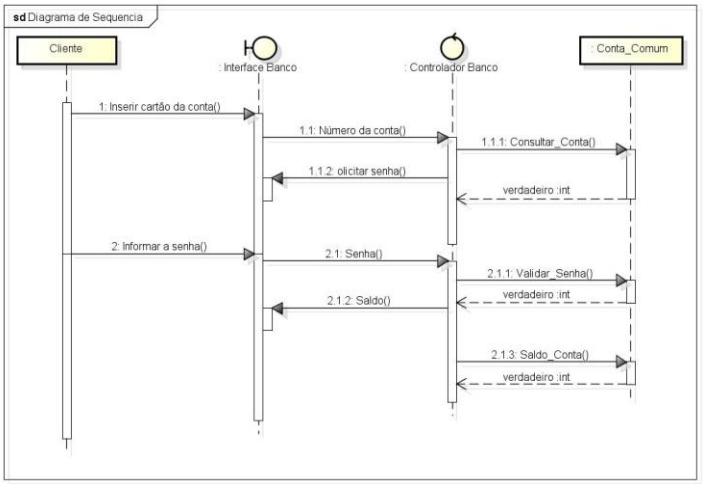

Figura 5 - Diagrama de Sequência. Guedes (2009)

2.3.6 Diagrama de Máquina de Estados

Segundo Guedes (2009), o diagrama na Figura 6, demonstra o comportamento de uma função, ou, de modo geral, uma classe durante as transições de uma atividade do sistema. O diagrama mostra, de forma sequencial, o próximo estado da atividade, fornecendo os passos seguintes após o resultado de cada transição.

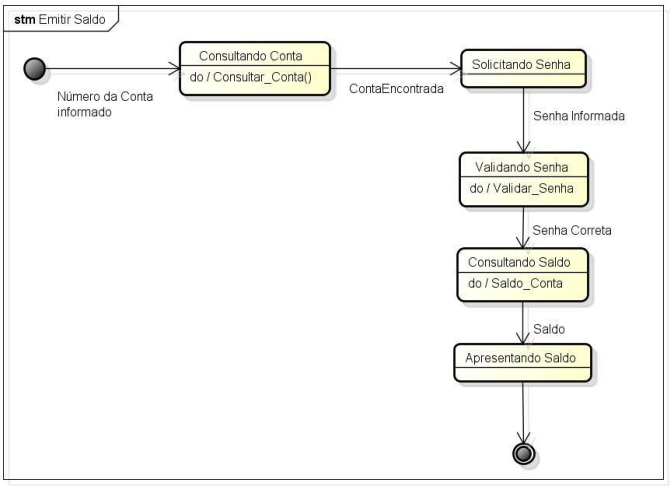

Figura 6 - Diagrama de Máquina de Estados. Guedes (2009)

2.3.7 Diagrama de Atividade

Neste diagrama obtém-se o diagrama de Máquina de Estados de forma mais detalhada. Através desse diagrama, é possível representar o grau de complexidade de uma funcionalidade específica, representando a variação dos estados de acordo com a entrada fornecida. Esse diagrama é importante, tendo em vista que o comportamento de uma função normalmente é variável de acordo com o tempo ou com diferentes valores de entrada, ou seja, seu comportamento atual depende do que aconteceu anteriormente. O diagrama é mostrado na Figura 7.

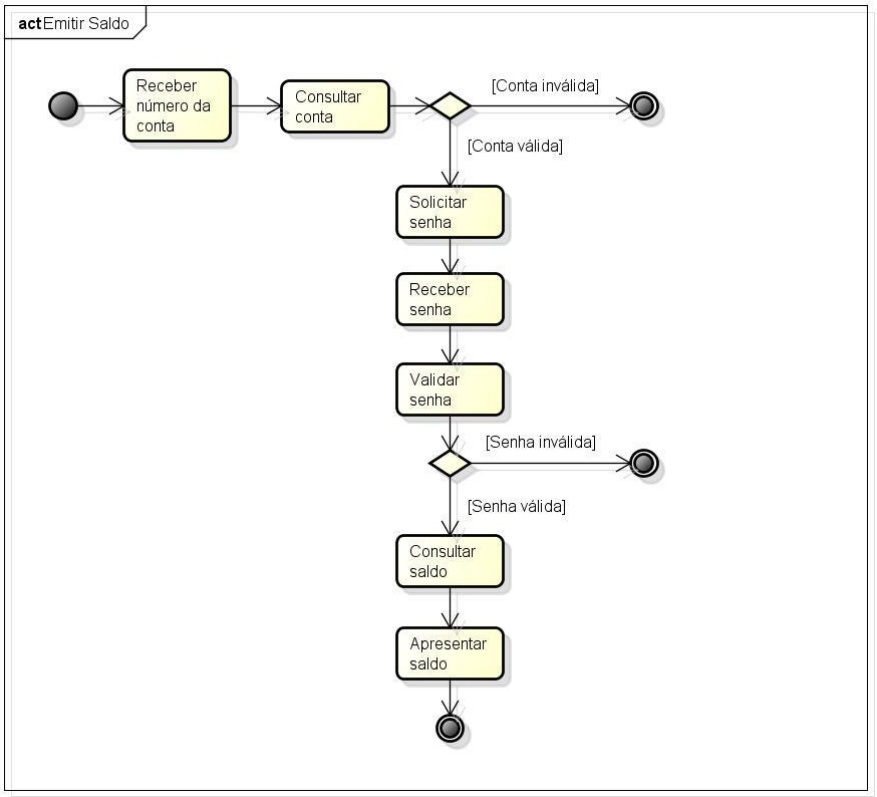

Figura 7 - Diagrama de Atividade. Guedes (2009)

2.4 Sistema Operacional Android

O Sistema Operacional Android surgiu a partir do ano de 2005 com a aquisição da Android Inc. pela empresa Google Inc. A partir daí foi desenvolvido um S.O. baseado no *kernel* Linux, um sistema de código aberto e livre que conta com o apoio de milhares de pessoas em todo mundo.

O Android é um S.O. especialmente criado para dispositivos móveis. Seu desempenho e segurança foram fundamentais para seu sucesso, além da sua criação em código aberto, possibilitando a interação de milhares de pessoas no desenvolvimento de novidades e aplicações.

O Android foi construído em algumas camadas, como se vê na Figura 8.

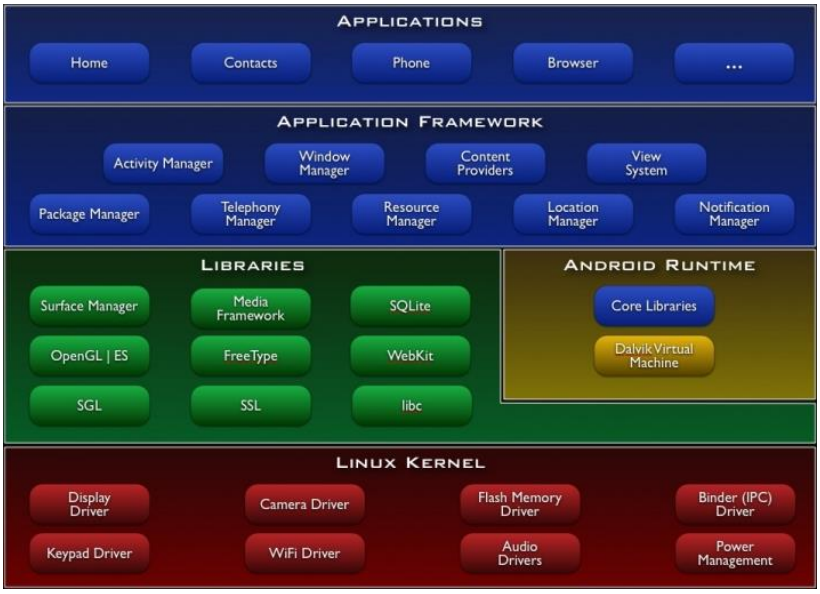

Figura 8 - Arquitetura Android (http://developer.android.com/guide/basics/what-is-android.html)

O *kernel* Linux é a base do sistema Android, sendo esta a primeira camada, ou seja, é a relação entre o *software* e o *hardware*, em que são definidos os programas de gerenciamento de memória, as configurações de segurança, o *software* de gerenciamento de energia e vários *drivers* de hardware.

A camada seguinte é conhecida como Camada de Tempo de Execução, na qual se encontram as bibliotecas básicas em linguagem JAVA, fundamentais para o desenvolvimento de aplicações. Esta camada funciona com o uso da *Dalvik Vitual Machine "*DVM*"* (Máquina Virtual Dalvik), uma máquina virtual que permite a execução de um S.O. sob qualquer *kernel*. Neste caso, a DVM realiza a execução de cada uma das aplicações do Android em um processo exclusivo, obtendo otimização da memória.

Segundo informações encontradas no site Wikipédia (2013), a DVM usa seu próprio byte code e o executa com a extensão ".dex". Resumidamente, o

código fonte Java é compilado pelo compilador Java. A partir disso, a ferramenta, que faz parte do SDK do Android, chamada de “dx” (Dexer) processa os arquivos “.class” em um formato de arquivo chamado DEX que contém o Dalvik bytecode. Os arquivos DEX agrupam todas as classes de uma aplicação em um único arquivo. Assim um novo aplicativo pode ser executado no S.O. Android.

No mesmo nível da camada de tempo de execução está a Camada de Bibliotecas, onde se encontram os códigos em C/C++ usadas por diferentes aplicações e componentes do sistema. São bibliotecas fundamentais para a organização do sistema operacional com diferentes tipos de dados.

Já a terceira camada é chamada de Camada de Ferramentas para Aplicação, fundamental para a interação do desenvolvedor com o sistema. Essa camada oferece ao desenvolvedor a oportunidade de otimizar, por exemplo, aspectos de hardware para a aplicação, podendo ter acesso às bibliotecas da Camada de Bibliotecas fazendo reuso de código, otimizando o sistema.

Na última camada, chamada de Camada de Aplicações, são armazenadas as aplicações que o usuário vê: são os programas de interface com o usuário, tais como contatos, navegador web, câmera, jogos, etc.

2.5 Linguagem JAVA

A linguagem JAVA é amplamente conhecida graças à sua portabilidade e consequentemente seu uso nos mais diversos dispositivos existentes. Seu conceito de desenvolvimento é a orientação a objetos, conceito muito difundido entre os programadores atualmente.

A criação da linguagem JAVA foi feita seguindo alguns objetivos para torná-la uma linguagem única na época. Alguns desses objetivos seguem abaixo:

- Linguagem orientada a objetos;
- Segurança;

- Portabilidade: Um mesmo código pode ser interpretado em qualquer Sistema Operacional com a *JAVA Virtual Machine* (Máquina Virtual JAVA);
- Plataforma Web: Seguir protocolos de rede, fornecendo serviços em lado servidor para Web.

3. METODOLOGIA

Seguindo as diferentes metodologias de pesquisa, as quais se enquadram na grande área Ciências Exatas e da Terra, é possível classificar esse projeto como uma pesquisa cientifica de natureza tecnológica ou aplicada, em que os requisitos levantados através de questionários são dados quantitativos, os quais mostram as melhores formas de estruturar o projeto de forma usual e funcional. Os dados observados através dos diagramas e testes são dados qualitativos, nos quais podem ser observados os pensamentos dos interessados e as melhores formas de se desenvolver o projeto. Quanto aos objetivos do projeto, o mesmo pode ser classificado como uma pesquisa descritiva e exploratória, no qual são identificados e analisados características e fatores relacionados ao projeto.

Como descrito anteriormente, o projeto foi desenvolvido seguindo as metodologias da Engenharia de *Software*. O modelo adotado foi o Desenvolvimento Iterativo pela facilidade de atualização e a possibilidade de alterações com o surgimento de novos requisitos.

A metodologia adotada no projeto seguiu os seguintes passos: Análise de Requisitos, Projeto, Codificação e Testes. Cada um deles é descrito nas subseções adiante.

3.1 Análise de Requisitos

Nesta etapa do projeto, foi realizado o levantamento de requisitos através de questionários e conversas com alguns professores. Através da análise dos requisitos, foi possível verificar aqueles que atendiam as expectativas do projeto. Esta etapa foi necessária para coletar dados para criar um sistema de acordo com as necessidades dos usuários. Com a documentação levantada, foi possível realizar o projeto e o planejamento de desenvolvimento com as atividades a serem

realizadas, além disso, com o Diagrama de Casos de Uso foi mostrado as interações dos usuários com as atividades necessárias no sistema (APÊNDICE A). Esses dados também podem ser usados como documento para justificar aos clientes a criação de atividades estabelecidas para a realização do projeto.

3.1.1 Requisitos

Os requisitos listados nas seções 3.1.1.1 e 3.1.1.2 foram estabelecidos através de questionários distribuídos aos professores do Departamento de Ciência da Computação (DCC) na Universidade Federal de Lavras (UFLA) e respondidos por oito deles. Estes requisitos forneceram as diretrizes básicas para o desenvolvimento do projeto, de forma que o mesmo ficasse adequado as expectativas dos interessados.

Para a especificação dos requisitos foram necessários que eles fossem separados em dois tipos distintos, relacionados a diferentes características essenciais do *software*. Primeiramente, são listados os requisitos funcionais, os quais são relativos às funcionalidades esperadas pelos usuários. Posteriormente são listados requisitos não funcionais, os quais determinam características essenciais para produção do *software*.

É importante ressaltar que o documento de requisitos sofreu algumas alterações pontuais durante a fase de desenvolvimento e testes. As alterações foram relacionadas principalmente com a criação de funcionalidades extras nas interfaces de disciplina e relatório, nas quais foram criadas atividades para reduzir o número de *clicks* na tela, não alterando os requisitos ou suas funcionalidades.

3.1.1.1 Requisitos Funcionais

Os requisitos coletados através dos questionários respondidos pelos possíveis usuários são listados a seguir.

Requisitos mínimos - Requisitos essenciais para o funcionamento básico do aplicativo.

RF01 - O aplicativo deverá ser capaz de gerenciar várias turmas. Cada disciplina deve conter os seguintes dados:

- Código, nome, apelido e número de aulas contidas em um dia de controle;
- turmas componentes;
- matrícula, turma e nome de cada aluno;
- data de realização de cada chamada.

RF02 - Uma turma poderá ser criada pela inserção manual de dados, ou pela importação de dados de um arquivo no formato CSV *(Comma Separated Values*) que contém a maioria dos dados (APÊNDICE B). O texto no arquivo deverá estar codificado em ISO-8859-1. Os dados restantes serão adicionados manualmente. Como uma disciplina pode ser composta de várias turmas, arquivos CSV adicionais poderão ser usados para incluir nomes de alunos pertencentes a diferentes turmas.

RF03 - Não será necessário definir antecipadamente as datas das aulas. O aplicativo deverá registrar a data automaticamente no momento da chamada. O aplicativo deverá permitir a correção da data, pois existe a possibilidade da data do dispositivo estar errada.

RF04 - A chamada deve poder ser realizada a cada aula. Cada aula de X minutos possui uma presença, sendo assim, o sistema deve permitir a escolha entre aulas únicas, duplas, triplas, etc.

RF05 - O aplicativo fará distinção entre presença, ausência, presença com atraso e ausência com abono. A alteração entre os vários estados será permitida a qualquer momento, não sendo necessário finalizar o registro de presença a cada aula.

RF06 - O aplicativo permitirá a consulta do total de faltas de um aluno, levando em consideração as faltas com abono e presenças com atraso.

RF07 - Salvar os dados do controle de frequência em arquivo no formato CSV.

Requisitos Extras - Requisitos levantados que não foram implementados e são recomendados para versões futuras.

RF08 - Personalizar o nome de chamada de cada aluno com o objetivo de poupar espaço em tela durante o uso. Para alunos com primeiro nome único, será usado o primeiro nome. Para alunos com primeiro nome repetido, será usado nome e sobrenome. A identificação de nomes repetidos será editável pelo usuário para incluir nomes diferentes que soam parecidos.

RF09 - Gerar porcentagens. Relação de alunos presentes x alunos ausentes. Relação de faltas x número de aulas.

RF10 - Manter uma lista de nomes diferentes que soam parecidos, para auxiliar a definição automática do nome de chamada de cada aluno.

RF11 - Permitir a importação de dados de alunos vindos de um arquivo no formato XLSX, conforme gerado pelo Sistema Integrado de Gestão (SIG) da UFLA.

RF12 - Gerar gráficos. Gráfico de relação entre alunos x faltas x número de aulas.

RF13 - Adicionar anotações para alunos selecionados. Em caso de apresentação de trabalhos um professor pode escrever comentários sobre alguns alunos. Em caso de atuação de destaque em sala de aula, o professor poderá atribuir pontos extra ao aluno.

3.1.1.2 Requisitos Não Funcionais

Abaixo são listados os requisitos não funcionais específicos para o desenvolvimento do aplicativo e posteriormente são listados alguns itens importantes para a qualidade do *software*.

RNF01 - Sistema Operacional Android.

RNF02 - Aplicativo com suporte ao S.O. Android versão 2.2 à 4.1+. Isso se faz necessário para que o aplicativo ofereça suporte à quase todas versões do Android atualmente.

Desempenho - Espera-se que o aplicativo funcione da maneira mais ágil possível e ofereça o desempenho mais próximo possível em todas as versões do Android ao qual o aplicativo oferece suporte.

Usabilidade - O aplicativo deve atender a todos requisitos mínimos descritos anteriormente, oferecendo a maior usabilidade possível aos usuários, atendendo suas expectativas.

Modularidade - O sistema deve permitir que novos módulos sejam adicionados permitindo novas funcionalidades ao sistema de acordo com novas necessidades e atendendo a outros requisitos estabelecidos por diferentes usuários de diferentes instituições.

3.2 Projeto

Durante a fase de projeto, foram criados os Diagramas de Classes, Diagrama de Objetos e Diagrama de Pacotes (APÊNDICE A). Com a criação destes diagramas foi possível identificar a relação das classes a serem criadas com as funções e seus possíveis estados de acordo com as entradas fornecidas. Durante esta etapa o foco principal foi a criação de uma interface que fosse intuitiva e que atendesse às expectativas. O projeto inicial de interface não era muito eficiente e

diversas mudanças foram realizadas para adequá-lo à uma maneira mais simples de interação com os usuários. Durante esta fase experimental, alguns requisitos também puderam ser melhor adequados par cada interface.

No desenvolvimento para dispositivos móveis a maior preocupação é com a interface, de forma a apresentar as informações da maneira mais clara possível, otimizando o número de toques na tela e ainda minimizando o processamento de dados. Neste projeto, isso não foi diferente e a maior dificuldade foi seguir o projeto, o qual sofreu diversas alterações na etapa de codificação.

Devido ao tamanho do projeto, alguns dos diagramas identificados anteriormente na Seção 2.3 não foram implementados. Os diagramas que não foram criados são: Diagrama de Sequência, Diagrama de Máquina de Estados e Diagrama de Atividades. Estes diagramas não foram necessários pois as atividades criadas para esse aplicativo não foram inter-relacionadas e as interfaces eram facilmente identificadas, já que cada atividade não possui mais de duas interfaces relacionadas, além disso as tarefas executadas não possuem alto grau de complexidade.

3.3 Codificação

Foi implementado o projeto com o conhecimento gerado nas fases anteriores. A linguagem de programação utilizada foi o JAVA, linguagem padrão para desenvolvimento no sistema operacional Android, e a plataforma de desenvolvimento utilizada foi o Eclipse.

O Android possui uma maneira robusta de desenvolvimento, no qual, seus aplicativos são comumente programados em JAVA, fazendo uso de arquivos XML para a criação das interfaces. A codificação foi baseada na *Application Programming Interface* (API) do Android e o código usufrui das funcionalidades

mais comuns dos aplicativos que são as listas, menus de contexto e menu de opções.

Após a definição dos objetos base da aplicação, que foram as classes DisciplinaVO, AlunoVO e ControleVO, assim como a definição da estrutura do banco de dados (Figura 16), foi possível dar início à criação das classes com relação às interfaces, criando as funcionalidades observadas no diagrama de casos de uso da Figura 15.

O processo de desenvolvimento seguiu uma estrutura muito utilizada pelos desenvolvedores de linguagens orientada a objetos. A estrutura é baseada em *Models*, *Views* e *Controllers* conhecido como MVC. Os objetos que foram criados no pacote POJO (*Plain Old Java Object*), podem ser chamados de *model*, os quais possuem apenas os atributos básicos, os quais são utilizados para criar a estutura de armazenamento do banco de dados, por esse motivo os objetos também podem ser chamados de *Value Object* (VO), ou seja objeto de valores. Já os objetos armazenados no pacote DAO (*Data Access Object*) fazem a manipulação dos objetos POJO e os dados armazenados no banco de dados, os métodos dessas classes realizam a inserção, alteração, busca e remoção dos dados armazenados utilizando a implementação de acesso ao banco de dados fornecido pela classe SQLiteOpenHelper do SDK Android. As classes conhecidas como *View*, são aquelas que fornecem a interface com o usuário e foram criadas no pacote UI (*User Interface*).

A estrutura do banco de dados foi criada para o uso de banco de dados relacional, o SQLite, fornecido por padrão pelo S.O. Android. Foram criadas três tabelas com colunas relacionadas aos atributos dos objetos base do pacote POJO. A tabela “alunos” possui uma chave estrangeira na coluna “disc_id” que faz relação com a tabela “disciplinas”, dessa forma, o sistema pode identificar um aluno pertencente a uma ou mais disciplinas. A tabela “controle_freq” possui duas chaves estrangerias, a chave na coluna “disc_id”, que também tem relação com a

tabela “disciplinas” e a chave na coluna “aluno_id”, a qual tem relação com a tabela “alunos”. Dessa forma é possível realizar os processos de inserção, alteração e exclusão de dados de forma mais eficiente, além da melhor organização da estrutura de desenvolvimento.

Como o aplicativo faz uso de arquivos CSV, foi utilizado um pacote de leitura e escrita em arquivos CSV anteriormente implementado, sob a licença Apache versão 2.0. O código, chamado de “Open CSV”[4] foi utilizado exclusivamente para a interação com os arquivos, tendo em vista que o código atendia aos requisitos necessários para essa aplicação.

Nesta etapa, diversas alterações foram realizadas, tanto no código quanto na interface, com o objetivo de adequar as funcionalidades à realidade dos dispositivos móveis.

3.4 Testes

Foram realizados testes para melhorar a usabilidade, portabilidade, funcionalidade e demais questões que não foram previstas.

As etapas acima forneceram parte do conhecimento necessário para que o projeto pudesse ser iniciado de forma coerente com os requisitos encontrados, sem a construção de funcionalidades desnecessárias e principalmente minimizando a existência de erros. Durante esta etapa, o aplicativo foi testado e várias correções foram realizadas, principalmente no tratamento de exceções encontradas com incompatibilidades de versões do Android.

Vários problemas foram encontrados na interação do aplicativo com alguns dos gerenciadores de arquivos testados para realizar a busca do arquivo

[4] http://opencsv.sourceforge.net

CSV no dispositivo, o que trouxe mudanças pontuais na codificação relacionadas à esses problemas.

Durante esse processo também foram feitas alterações no layout para fazer uso de novas funcionalidades criadas nas versões mais recentes do Android, como por exemplo a adição dos botões mais utilizados do menu de opções na barra superior do aplicativo, otimizando a quantidade de toques na tela do dispositivo.

4. RESULTADOS

O projeto foi baseado em dados coletados através de oito questionários distribuídos aos professores do departamento de Ciência da Computação (DCC) da UFLA, com o intuito de desenvolver o projeto com as funcionalidades mais requisitadas pelos professores nas respostas dos relatórios. Neste capítulo é apresentado o aplicativo através de suas etapas de desenvolvimento e suas funcionalidades.

4.1 Menu e contexto no S.O. Android

No desenvolvimento para dispositivos móveis existe uma enorme preocupação com relação ao espaço e principalmente nas informações que devem ou não ser apresentadas ao usuário em cada uma das interfaces de um aplicativo. No desenvolvimento deste projeto, isso não foi diferente, e a maior dificuldade se estabeleceu durante o entendimento de como os usuários, em geral, lidam com a interface dos dispositivos móveis. Durante todo o desenvolvimento do projeto foi necessário um levantamento das funcionalidades mais importantes em cada contexto, de forma a apresentar ao usuário apenas as funções e botões que eram mais importantes em cada situação.

O uso de menus é fundamental para minimizar a quantidade de funcionalidades mostradas de forma direta ao usuário. Nesta aplicação, são utilizados dois dos principais menus fornecidos pela API do Android e seu uso nesta aplicação é descrito nas seções 4.1.1 e 4.1.2.

4.1.1 Menu de Contexto

Como citado anteriormente, cada interface criada pelo aplicativo possui um contexto. Segundo a API do Android, o contexto da aplicação é o repositório central para as funcionalidades de todas as aplicações de nível superior no Android. Este contexto é usado quando se deseja acessar configurações e recursos compartilhados entre as várias interfaces da aplicação. Nesta aplicação, o menu de contexto fornece uma lista de opções para alterar um item em uma lista específica através de um *click* longo. Em cada contexto o número de opções varia de acordo com as funcionalidades definidas em cada interface.

Um exemplo deste tipo de situação é quando se tem uma lista de produtos e o usuário pode desejar alterá-lo. Como, neste caso, o contexto da interface é a lista de produtos, ao usuário pode dar um *click* longo sobre o produto e assim realizar alguma tarefa, como alterar o nome do produto, excluí-lo, ou fazer qualquer outra tarefa que seja conveniente neste contexto.

4.1.2 Menu de Opções

Ao contrário do menu de contexto da Seção 4.1.1, este menu está relacionado com as opções que abrangem diferentes funcionalidades de uma interface, podendo esta fazer relação com outros objetos da aplicação. O menu de opções criado nesta aplicação fornece funcionalidades que são muito importantes, mas a princípio pouco usadas em relação aos demais botões apresentados de forma direta ao usuário. Isso ocorre, por exemplo, na interface “Gerenciador Disciplina”, em que o menu de opções fornece dois itens, sendo eles Importar e Exportar. Este menu é um exemplo da importância do contexto nas aplicações Android, em que uma funcionalidade, dentro de um contexto pode interagir com métodos mais

abrangentes relacionados com um ou mais objetos da aplicação. As funcionalidades de cada um dos itens citados acima é descrita na Seção 4.3.

4.2 Main Disciplina (Interface Inicial)

Como visto na Seção 3.1.1, as disciplinas devem ser adicionadas com cinco campos distintos, os quais informam ao usuário em qual disciplina será realizada a chamada. Esta interface fornece um detalhe importante, que é a lista de turmas pertencentes à determinada disciplina, tendo em vista que uma mesma disciplina pode ser ministrada para diferentes turmas em diferentes dias e horários.

Nesta atividade do aplicativo é possível realizar cinco funcionalidades relacionadas com as disciplinas. Cada uma dessas funcionalidades é descrita nas seções 4.2.1 a 4.2.5.

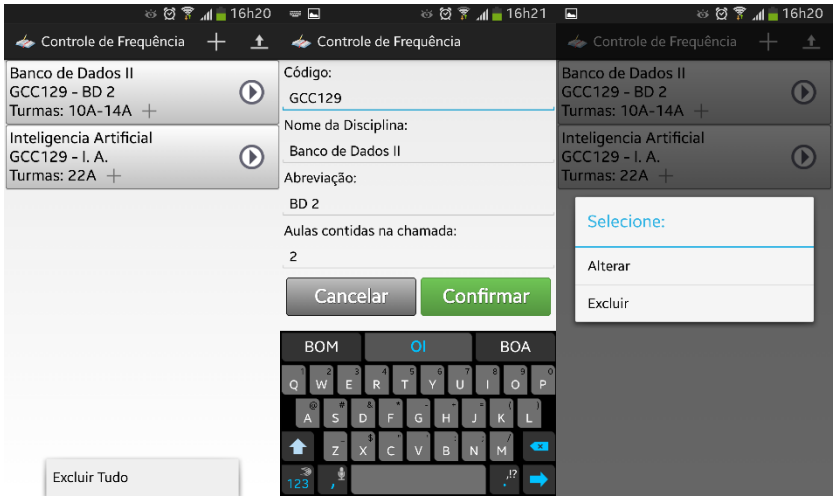

Figura 9 - Interfaces da atividade Disciplina

4.2.1 Importar Disciplina

Uma disciplina pode ser importada de um arquivo CSV, através de diferentes gerenciadores de arquivos. O arquivo pode estar no armazenamento interno do dispositivo ou em "nuvem", através do uso de aplicativos como o DropBox e SkyDrive. O aplicativo cria uma pasta chamada "ControleDeFrequencia" no armazenamento interno do celular, se houver. Após a leitura dos dados no arquivo de origem, o usuário será direcionado para a interface "Nova Disciplina", na qual poderá verificar as informações e fazer alterações descritas na Seção 4.2.2.

4.2.2 Nova Disciplina

Além da importação dos dados é possível que uma disciplina seja adicionada manualmente pelo usuário através do teclado do dispositivo. Todos os campos são obrigatórios e devem ser preenchidos corretamente de acordo com o exemplo dado em cada campo. Cada um dos campos são autoexplicativos mas o último deles, denominado "Aulas contidas na chamada" merece uma atenção especial. Este campo é fundamental para um controle de frequência eficiente, é através dele que o controle é realizado de diferentes maneiras de acordo com o número de créditos de uma dada disciplina em uma data específica. Através do levantamento dos requisitos, foi observado que nem todos os professores realizam uma verificação de presença por aula, no caso das disciplinas da UFLA, uma aula possui 50 minutos, o que é equivalente à um crédito. Portanto, um professor pode realizar um controle de frequência para cada aula, ou então um controle para cada duas aulas consecutivas em uma mesma data e assim por diante. Foi identificado que a maioria dos respondentes dos questionários fazem apenas um controle para cada duas aulas consecutivas, sugerindo que o aplicativo iniciasse este campo com

o padrão 2. Por esse mesmo motivo, a interface de inclusão de disciplina também é mostrada quando se importam os dados de um arquivo, pois o usuário deve verificar, se na disciplina importada, ele irá realizar um ou mais controles de frequência dos alunos, de acordo com o número de créditos da disciplina em uma determinada data.

4.2.3 Alterar Disciplina

Por motivo de segurança e simplicidade ao usuário, também foi criada uma funcionalidade para alterar os dados de uma disciplina. Para isso basta que o usuário mantenha o dedo pressionado (*click* longo) em uma das disciplinas listadas e selecione a opção "Alterar", a partir daí, a mesma interface usada para adicionar uma disciplina é mostrada. O principal uso desta funcionalidade é permitir ao usuário a alteração do número de aulas por controle de frequência e a criação de uma abreviação para o nome da disciplina, que facilita a leitura em dispositivos menores. É importante ressaltar que o número de aulas por controle pode ser alterado a qualquer momento, permitindo ao usuário realizar quantidades de controle de frequência diferentes em qualquer data que tenha interesse.

4.2.4 Excluir Disciplina

Uma disciplina também pode ser excluída. Em alguns casos podem ocorrer do usuário realizar a adição ou importação de uma disciplina já existente no dispositivo, ou então, uma disciplina é adicionada por engano. Quando uma disciplina é excluída, todos os dados hierárquicos pertencentes a ela, tais como, lista de alunos e todo o controle de frequência pertencente a ela são excluídos.

4.2.5 Excluir Tudo

Além da exclusão de uma única disciplina, também se faz necessário a exclusão de todas elas. Ao término de um semestre, e início do próximo, as listas de alunos e suas respectivas turmas poderão ser alteradas, assim como, as disciplinas ministradas por um professor podem sofrer alterações. Esta ferramenta, foi criada para facilitar a exclusão e garantir que todos os dados armazenados são apagados com apenas um passo.

4.3 Gerenciador de Disciplina

Como pode ser observado na Figura 10, a interface Gerenciador de Disciplina direciona o usuário à iniciar um novo controle de frequência em uma data específica ou então visualizar a lista de alunos. O gerenciador fornece a opção de alterar a data do controle de frequência a ser realizado e ainda permite importar os alunos de uma determinada turma através de um arquivo CSV e também exportar todos os dados armazenados da disciplina, sendo exportado um relatório em formato CSV (codificação UTF-8) com o nome da disciplina, as turmas que a compõem e a lista de alunos com o respectivo controle de frequência de cada aluno separado por data. Um exemplo da estrutura do arquivo para importação é apresentada no APÊNDICE B.

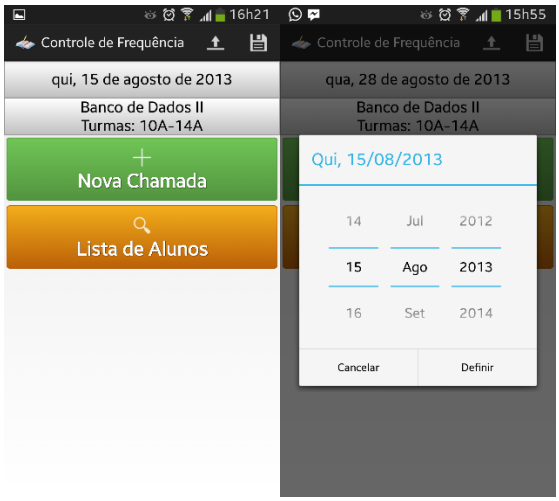

Figura 10 - Interface do Gerenciador de Disciplina

4.4 Controle de Frequência

A atividade de controle de frequência apresenta alguns detalhes do aluno, como seu nome, matrícula e a turma a qual ele pertence. Esta interface realiza o controle de presença dos alunos pertencentes à disciplina atual, de acordo com a data selecionada no gerenciador de disciplina. Quando o usuário clica sobre o botão Presente ou Ausente o sistema passa para o próximo aluno da lista. Por questões de segurança, quando o usuário clica no botão voltar do dispositivo o controle e finalizado com o status “Indefinido” visando manter a quantidade de aulas dadas igual para todos alunos da disciplina. A lista de alunos é apresentada por ordem crescente da turma e posteriormente por ordem alfabética de alunos.

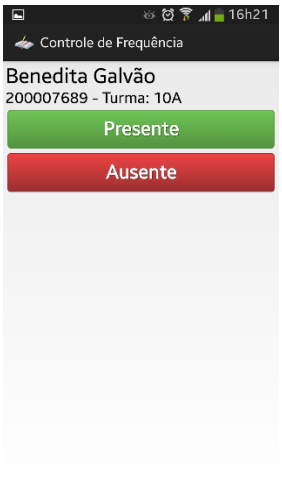

Figura 11 - Interface do Controle de Frequência

4.5 Lista de Alunos

A lista de alunos é formada por todos alunos pertencentes à uma disciplina, através dessa lista o usuário pode visualizar um relatório do controle de frequência realizado para um determinado aluno. Além dessa funcionalidade, também é possível adicionar, alterar e excluir um aluno ou excluir todos os alunos listados e consequentemente todos seus dados relacionados, inclusive os controles de frequência realizados para cada aluno.

Ao clicar sobre um aluno será apresentada uma nova interface, na qual é apresentado um relatório sobre os status armazenados do aluno de acordo com os controles realizados anteriormente. As interfaces podem ser vistas na Figura 12.

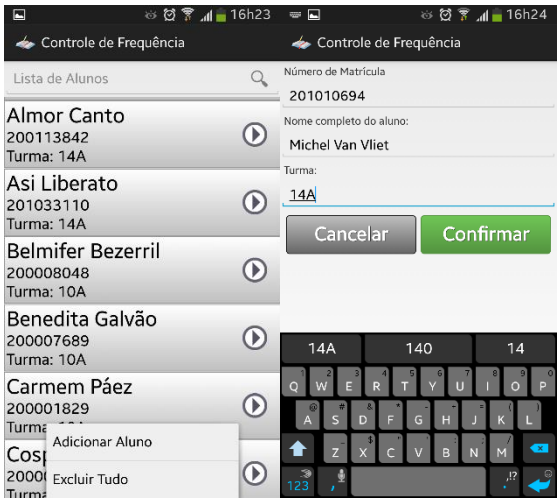

Figura 12 - Interfaces da Lista de Alunos

4.5.1 Relatório do Controle de Frequência

O relatório do controle de frequência mostra ao usuário a relação da soma de todos os status armazenados para um aluno até a data atual. É mostrado de forma fácil, o total de presenças, atrasos, abonos e ausências. A interface pode ser vista na Figura 13. Se, na data atual, o usuário tiver realizado um controle, aparecerá mais botões de acordo com o número de controles realizados para que se possa fazer a alteração do status para o aluno em uma aula. Também é apresentado dois botões, Próximo e Anterior, os quais percorrem a lista de alunos mais facilmente, mostrando o relatório para cada um deles com apenas um toque. Ao clicar no botão Ver Histórico o usuário também poderá ter acesso ao histórico detalhado do aluno atual.

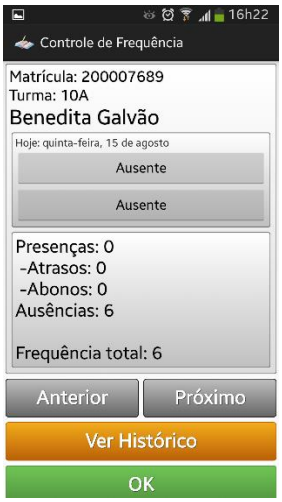

Figura 13 - Interface de Relatório por aluno

4.5.2 Histórico do Controle de Frequência

Visando facilitar a possibilidade de alterações no histórico do controle de frequência, foi criada a interface de histórico, na qual é possível que o usuário faça alterações em um controle de frequência realizado em uma determinada data. Isso se fez necessário para alterar o status, no qual um aluno pode passar de ausente para presente, ou de ausente para abono, por exemplo. Quando um aluno possui o status alterado para Abono, o mesmo passa a ter um total de faltas menor, ou seja, o Abono é considerado como presença, de forma que o usuário não precisa recalcular os totais no relatório. Os atrasos, assim como, os abonos são totalizados junto com as presenças. É possível ainda excluir um controle de frequência, ao manter pressionado sobre o status desejado em uma determinada data. Ao excluir um controle de frequência é necessário lembrar que o total de controles realizados será alterado e deve ser verificado posteriormente no relatório e no arquivo CSV exportado. A interface pode ser vista na Figura 14.

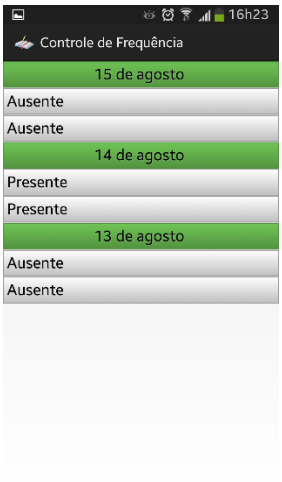

Figura 14 - Interface de Histórico detalhado

4.6 Direitos Autorais

O aplicativo será distribuído seguindo a licença GPLv2[5], permitindo liberdade de uso e de modificação futura, ao mesmo tempo que garante que as melhorias desenvolvidas deverão ser disponibilizadas de volta à UFLA e que o autor poderá comercializar uma versão do programa.

[5] http://www.gnu.org/licenses/gpl-2.0-standalone.html

5. CONCLUSÃO

O desenvolvimento de aplicações móveis tem sido amplamente explorada e o uso de diversos tipos de aplicativos tem facilitado muito algumas atividades diárias das pessoas. O aplicativo desenvolvido realiza atividades que facilitam o controle de frequência de pessoas em diversas ocasiões, onde seu uso se faz necessário.

A aplicação oferece recursos que eram realizados de forma manual, e com o uso de papel, além de realizar cálculos e relatórios de forma rápida, oferecendo maior dinâmica na visualização dos dados coletados. A possibilidade de acompanhar com mais detalhes a frequência dos alunos pode auxiliar os professores de escolas e universidades no acompanhamento dos mesmos em determinadas situações em que o controle de frequência é eficaz.

Todo o projeto foi construído seguindo as melhores práticas adquiridas durante os estudos anteriores e foram fundamentais para o desenvolvimento do projeto de forma coordenada. A estruturação do banco de dados (APÊNDICE A) e da programação em JAVA auxiliaram no melhor desenvolvimento e na segurança dos dados armazenados, fornecendo a base necessária para a geração dos relatórios.

Os relatórios são criados de forma dinâmica e com informações extras que não eram verificadas de forma manual. O acompanhamento das informações em tempo real resolve situações comuns, como a correção de controles realizados anteriormente e também a informação da situação atual aos alunos.

O objetivo de criar um aplicativo para dispositivos móveis com o S.O. Android foi alcançado e seu uso pode ser realizado em diferentes versões, aumentando ainda mais suas possibilidades de uso. Através das novas plataformas Android, o aplicativo pode ser apresentado de diferentes formas, com as mesmas funcionalidades, facilitando ainda mais a interface com o usuário.

Além de desenvolver o aplicativo, também foi possível utilizar, na prática, diversos conceitos vistos durantes as disciplinas cursadas e as dificuldades que podem ser encontradas em todas as fases da engenharia de software, principalmente durante as atividades de especificação de requisitos, projeto e desenvolvimento, nas quais a maioria dos problemas aconteceram e foram corrigidos.

Projetos futuros podem ser realizados com o objetivo de integrar o aplicativo a sistemas online e em tempo real de armazenamento[6]. O aplicativo também pode ser adequado para ser utilizado com diferentes formatos de arquivo com diferentes tipos de dados.

[6] Exemplo: SIG – Sistema integrado de Gestão; utilizado na UFLA

REFERENCIAL BIBLIOGRÁFICO

FREEMAN, R. E. **Strategic management:** a stakeholder approach. Boston: Pitman, 1984.

GOOGLE INC. **Android Developers.** Disponível em: <http://developer.android.com >. Acesso em: 03 Maio 2012.

GUEDES, Gilleanes T. A. **UML 2**: Uma abordagem prática. São Paulo: Novatec Editora, 2009.

ORACLE. **Saiba mais sobre a tecnologia Java.** Disponível em: <http://www.java.com/pt_BR/download>. Acesso em: 03 Maio 2012.

PRESSMAN, Roger S. ***Software* Engeneering**: a practitioner's approach. 6ª edição. Nova York: McGraw-Hill, 2005.

SOMMERVILLE, Ian. **Engenharia de *Software***. 8ª edição. São Paulo: Pearson Addison-Wesley, 2007.

WIKIPEDIA. **Dalvik.** Disponível em: <http://en.wikipedia.org/wiki/Dalvik_(software)>. Acesso em: 26 Agosto 2013.

APÊNDICES

APÊNDICE A – DIAGRAMAS

A Figura 15 mostra o Diagrama de Casos de uso utilizado para definir as atividades básicas realizadas pelos usuários do aplicativo.

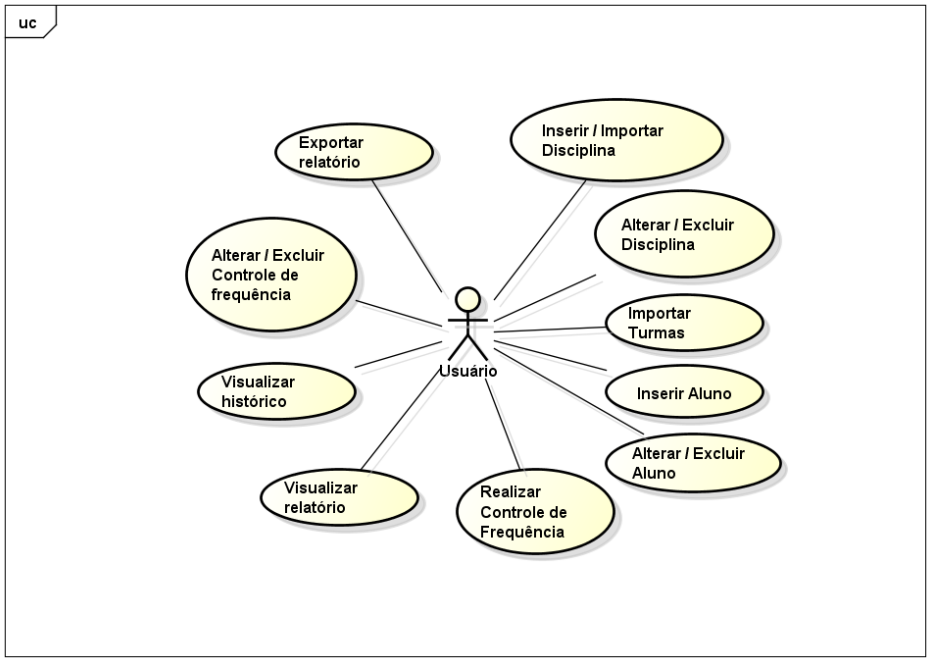

Figura 15 - Diagrama de Casos de Uso

A Figura 16 mostra a modelagem do banco de dados utilizado pelo aplicativo de Controle de Frequência.

Os losangos em azul são campos que não podem ser nulos.

Os losangos em vermelho são chaves estrangeiras.

Os símbolos de uma chave amarela são as chaves primárias de cada tabela.

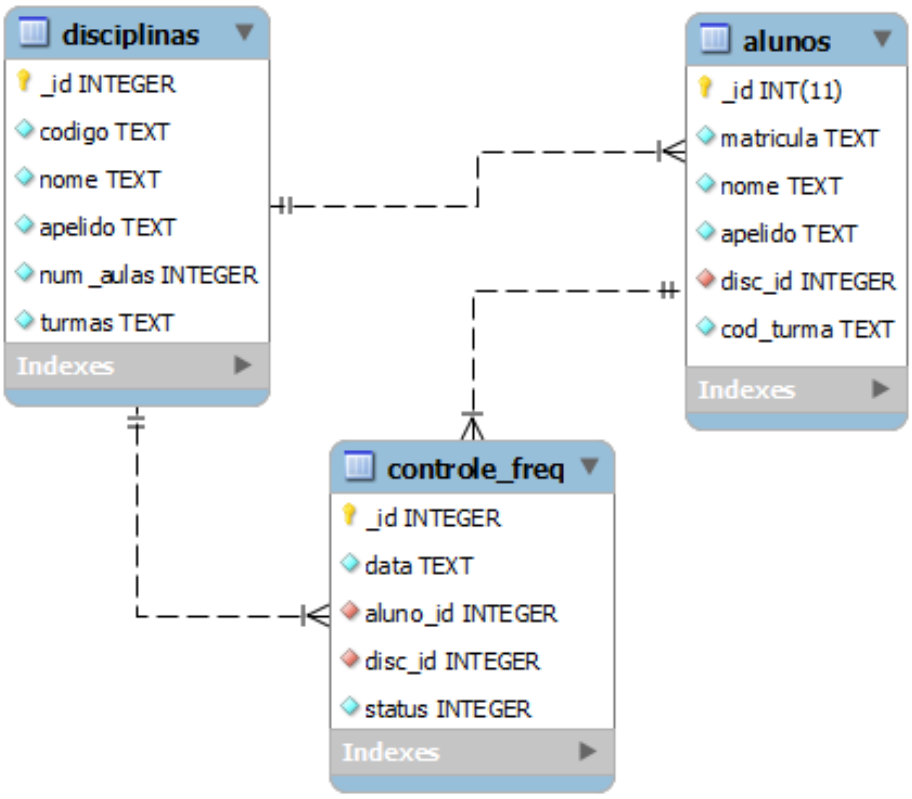

Figura 16 - Modelagem do banco de dados

O Diagrama de Classes da Figura 17 é um diagrama resumido e mostra a relação de todas as classes pertencentes ao aplicativo. As classes com o nome final "Activity" são as interfaces.

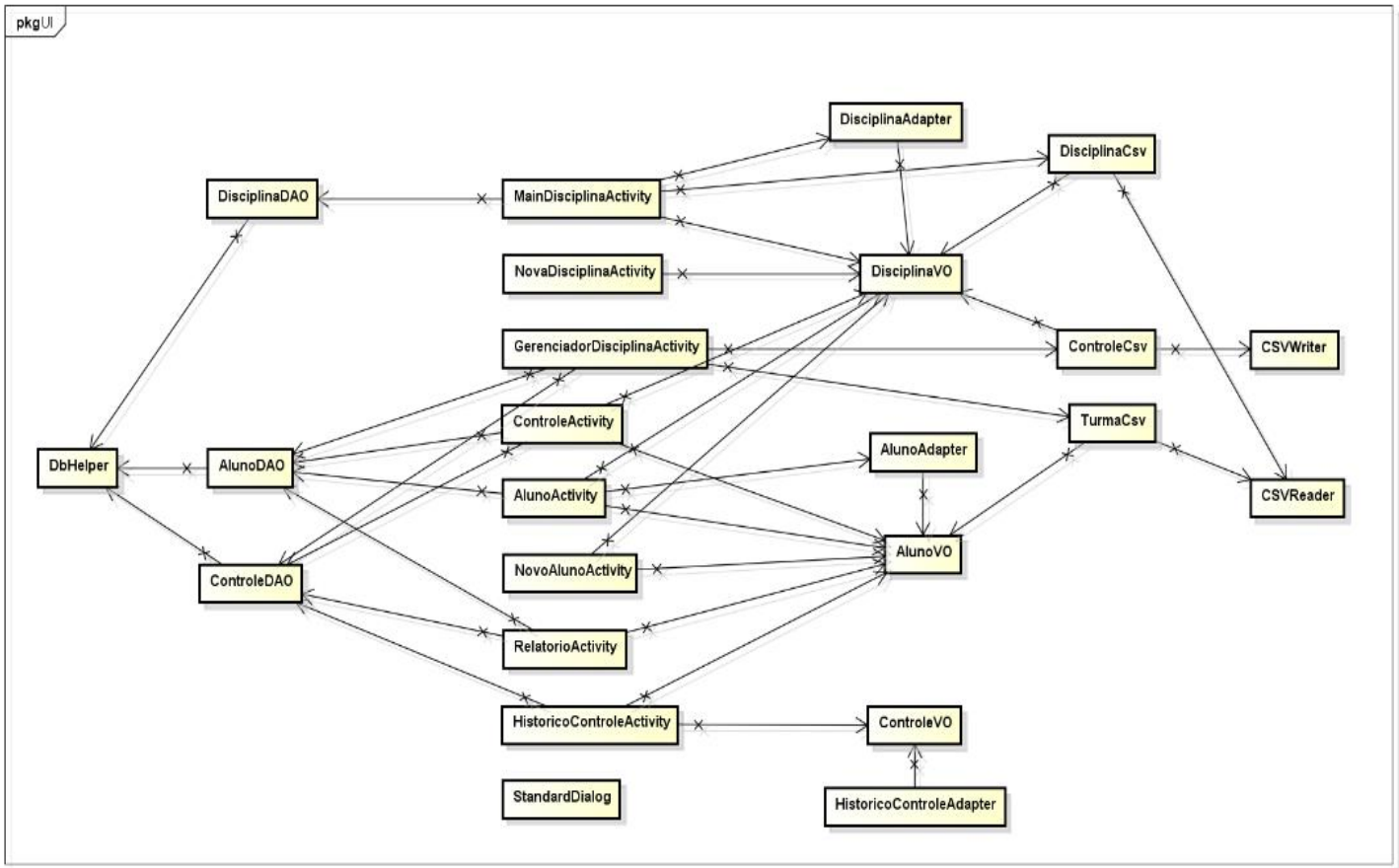

Figura 17 - Diagrama de Classes Resumido (Todos pacotes)

A Figura 18 mostra os atributos pertencentes às classes básicas do aplicativo, e seus possíveis valores.

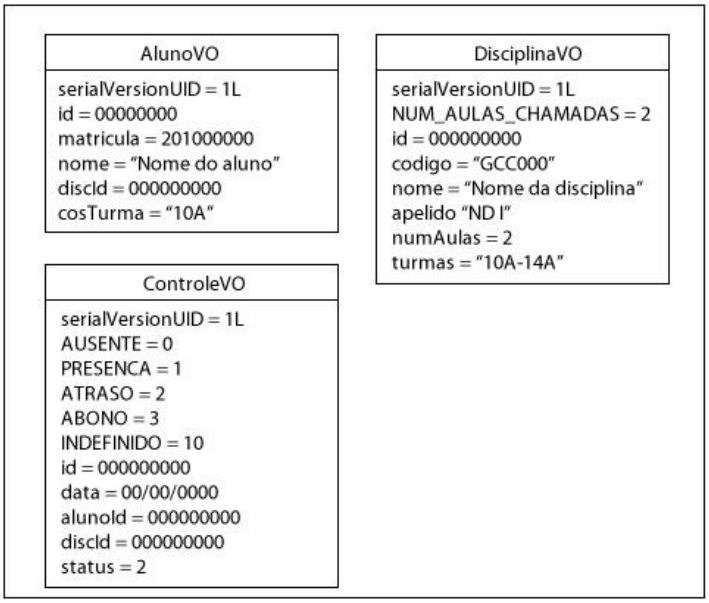

Figura 18 - Diagrama de Objetos

A Figura 19 mostra a interação entre os pacotes criados durante o desenvolvimento do aplicativo.

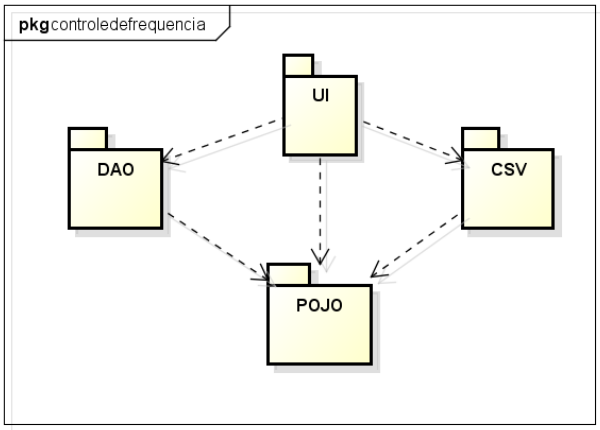

Figura 19 - Diagrama de Pacotes

APÊNDICE B - ESTRUTURA DO ARQUIVO CSV

Os arquivos devem estar salvos no formato CSV (*Comma Separeted Values*) seguindo a estrutura da Figura 20 com codificação ISO-8859-1. Esta estrutura deve ser seguida na importação de dados da disciplina e das turmas para uma disciplina.

- A primeira linha faz referência ao código da disciplina, o qual, deve, obrigatoriamente, ter seis caracteres.
- A segunda linha deve conter o nome da disciplina e a turma, separados, obrigatoriamente, por um "-".
- A terceira linha é o cabeçalho usado para informar o significado das linhas abaixo. Esta linha pode ser vazia, mas ela deve existir.
- Todas as linhas seguintes são usadas para listar os alunos pertencentes à respectiva turma. Primeiro escreve-se o nome do aluno e posteriormente o número da matrícula, obrigatoriamente com nove números inteiros. O nome e a matricula devem ser separados por uma vírgula "," ou ponto e vírgula ";".

	A	B	C
1	Código da Disciplina GCC129,		
2	Inteligência Artificial - 22A,		
3	Nome,Matrícula		
4	Acacio Costa,200009528		
5	Aldo Linhares,200007635		
6	Aldone Quintal,200000013		
7	Belmifer Bezerril,200008048		
8	Benedita Galva,200007689		
9	Carmem Paz,200001829		
10	Cosperranho Alvim,200008779		
11	Fernando Espargosa,200006243		
12	Florinda Higuera,200001631		
13	Gerson Lagoa,200009742		
14	Godofredo Fartaria,200005335		
15	Maximiliano Varella,200000926		
16	Miru Temes,200002238		
17	Salvador Chousa,200007030		
18	Vlademiro Jocame ,200000368		
19			

Figura 20 - Estrutura arquivo CSV (ISO-8859-1)

APÊNDICE C
GUIA RÁPIDO DE USO

1. Como criar o arquivo de dados CSV

Considerando que o usuário deseja importar os dados através de um arquivo CSV, o mesmo deve realizar alguns passos para que o arquivo seja criado de forma correta.

Primeiramente o usuário deve salvar o arquivo XLSX gerado pelo SIG da UFLA. Após abrir o arquivo, o mesmo deve ser salvo no formato CSV. Para que o arquivo seja salvo no novo formato, o usuário pode fazer uso de diferentes programas que fazem a leitura de planilhas, como por exemplo, o Microsoft Office Excel, Libre Office, etc. Cada programa interage de forma diferente com o usuário mas as regras para a criação do arquivo CSV são as mesmas.

Para que o arquivo de dados seja gerado corretamente e todos os caracteres possam ser interpretados de forma correta pelo aplicativo o usuário deve salvar o arquivo com a codificação ISO-8859-1. Além da codificação ISO-8859-1 o novo arquivo deve ser salvo utilizando como parâmetro separador, a vírgula “,” ou o ponto e vírgula “;”. Veja mais detalhes no APÊNDICE B. Isso deve ser configurado no momento em que se salva o arquivo, onde, dependendo do programa utilizado, essas informações podem ser alteradas.

Um breve passo a passo é escrito a seguir:

- Fazer o download do arquivo XLSX através do SIG/UFLA, ou criar um novo arquivo manualmente, seguindo o APÊNDICE B;
- caso esteja usando o arquivo gerado pelo SIG, o usuário deve abri-lo em um programa leitor de planilhas de dados, por exemplo, Microsoft Excel, Libre Office, etc.;

- salvar o arquivo em formato CSV, configurando-o com o separador “,” ou “;” entre as colunas e com codificação ISO-8859-1.

2. Como instalar um aplicativo no seu dispositivo

Novos aplicativos são comumente instalados através do site Play Store[7] da Google, ou pela própria aplicação no dispositivo, onde o usuário realiza a busca por um aplicativo e pode comprá-lo e instalá-lo ou apenas instalá-lo, naqueles que são gratuitos. O usuário deve possuir uma conta Google para que possa realizar a instalação. Após efetuar o login, ele pode selecionar o aplicativo desejado, para que o mesmo possa ser baixado para o dispositivo escolhido. Após o download o aplicativo é automaticamente instalado e pode ser usado pela primeira vez.

Além dessa opção, o S.O. Android ainda permite a instalação de aplicações através do armazenamento do próprio dispositivo, onde o usuário deve possuir o arquivo da aplicação com extensão “apk”, formato padrão das aplicações Android. Para que a instalação seja feita de forma manual é necessário que uma opção de segurança seja alterada, permitindo a instalação de aplicações desconhecidas. Para alterar essa configuração, o usuário deve ativar a opção de “Fontes desconhecidas”, nas configurações do dispositivo, normalmente encontrada na opção “Segurança” do S.O. Android, esta opção também pode estar localizada em “Gerenciador de aplicações”.

Após realizar o procedimento acima o usuário deve utilizar um gerenciador de arquivos para encontrar o arquivo da aplicação, e assim, instalá-lo de forma manual, para isso, basta clicar sobre o arquivo, e o instalador do Android lhe informará as opções para aquela aplicação.

Um breve passo a passo através da Play Store é descrito abaixo.

[7] https://play.google.com

- Baixar a aplicação pela Play Store;
- O aplicativo é instalado automaticamente;
- abrir o aplicativo e começar a usar.

Um breve passo a passo através do instalador em arquivo:

- Baixar, ou transferir o arquivo “apk” para o armazenamento interno ou externo do dispositivo;
- verificar se a opção “Fontes desconhecidas” está ativa nas “Configurações” do dispositivo, em “Segurança” ou “Gerenciador de aplicações”;
- utilizar um gerenciador de arquivo para chegar ao arquivo no dispositivo;
- abrir o arquivo do aplicativo com a extensão “apk”;
- o instalador do Android dá início à instalação;
- após a instalação, abra o aplicativo instalado e comece a usar.

3. Manual básico do aplicativo Controle de Frequência

O manual apresentado aqui, fornece o conhecimento básico para iniciar o uso do aplicativo, algumas funcionalidades não são exemplificadas e o usuário deve explorar todas as opções em cada interface de acordo com seu interesse.

Ao abrir a aplicação pela primeira vez, é mostrado um aviso, informando ao usuário de que não existe nenhum dado cadastrado. Ao abrir o aplicativo pela primeira vez é criada uma pasta nominada “ControleDeFrequencia”, a qual é recomendada para armazenar os arquivos CSV.

Para usufruir de todos recursos do aplicativo o usuário deve seguir as informações abaixo.

3.1 Instalar um gerenciador de arquivos

Um gerenciador de arquivos é necessário para que o aplicativo possa importar os dados de um arquivo CSV. Para realizar a instalação do gerenciador de arquivos faça o seguinte:

- Vá até o site da Play Store, ou acesse o aplicativo pelo dispositivo;
- procure por “Gerenciador de arquivos”;
- escolha um de sua preferência;
- instale o aplicativo.

3.2 Importar uma disciplina de um arquivo CSV

- Transfira o arquivo CSV para a pasta “ControleDeFrequencia” no armazenamento do dispositivo;
- abrir o aplicativo;
- clicar no botão de menu do dispositivo;
- clicar no botão do menu “Importar Disciplina”;
- um gerenciador de arquivos será aberto, vá até o arquivo CSV transferido para o dispositivo na pasta “ControleDeFrequencia”;
- selecione o arquivo CSV;
- na nova interface confira os dados da disciplina;
- salve a nova disciplina.

Após adicionar uma disciplina, o usuário pode alterá-la o excluí-la, para isso, basta que o usuário de um toque longo sobre a disciplina desejada.

3.3 Importar turma

É possível adicionar uma nova turma à uma disciplina de duas maneiras, como se segue:

Através do botão “+” na lista de disciplinas:

- Toque no botão “+” na disciplina desejada;

- o gerenciador de arquivos instalado será aberto;
- vá até o arquivo CSV transferido na pasta "ControleDeFrequencia";
- selecione o arquivo desejado;
- os dados são importados e uma mensagem é mostrada com o resultado da importação.

Através da interface "Gerenciador de Disciplinas":

- Na lista de disciplinas, toque na disciplina desejada;
- na nova interface, toque no botão "Lista de alunos";
- toque no botão de menu do dispositivo;
- toque no botão do menu "Importar Turma";
- o gerenciador de arquivos instalado será aberto;
- vá até o arquivo CSV transferido na pasta "ControleDeFrequencia";
- selecione o arquivo desejado;
- os dados são importados e uma mensagem é mostrada com o resultado da importação..

Um aluno de uma turma pode ser adicionado através do menu na "Lista de Alunos", além disso, o aluno também pode ser alterado ou excluído, para isso basta dar um toque longo no aluno desejado e selecionar a opção alterar, ou então excluir o aluno.

3.4 Efetuar o Controle de Frequência

Para realizar o controle de frequência siga os passos a seguir:

- Selecione a disciplina desejada na lista de disciplinas;
- No gerenciador de disciplinas toque no botão "Nova Chamada"

- Uma interface com os dados do aluno é apresentada, toque em “Presente” ou “Ausente” de acordo com a presença ou ausência do aluno.
- Ao clicar em um dos botões acima, o próximo aluno da lista é apresentado até o fim da lista, onde um alerta é gerado.

3.5 Alterar o Controle de Frequência

É possível alterar o controle de frequência de duas maneiras distintas. Para alterar o controle de frequência na data atual:

- Vá até a “Lista de Alunos”;
- toque sobre o aluno que deseja alterar o controle de frequência;
- se um controle tiver sido realizado na data atual, é possível alterá-lo diretamente no relatório apresentado, toque no controle e selecione o novo status;
- se nenhum controle tiver sido realizado na data atual, toque em “Ver Histórico”;
- na lista apresentada toque sobre o controle de frequência realizado na data desejada e altere seu status.

3.6 Exportar o controle de frequência de uma disciplina

Para exportar o controle de frequência gerado ao longo do tempo para um disciplina, siga os passos:

- Na lista de disciplinas, toque na disciplina que deseja exportar;
- na interface “Gerenciador de Disciplina”, toque no menu do dispositivo;
- nas opções do menu, toque em “Exportar Dados”;

- Os dados serão exportados na pasta “Arquivos Exportados” em “ControleDeFrequencia” no armazenamento interno do dispositivo.

O arquivo CSV exportado possui codificação UTF-8. Para abrir esse arquivo o usuário deve possuir um programa gerenciador de planilhas como por exemplo o Libre Office ou Microsoft Office Excel. Abaixo é apresentado um breve passo a passo para o Office Excel (a partir da versão 2007):

- Abrir o Microsoft Office Excel com uma planilha vazia;
- Clicar sobre a opção DADOS;
- No campo de importar dados clicar em “de Texto”;
- Selecionar o arquivo CSV exportado;
- Configurar a codificação para UTF-8 no campo “Origem do Arquivo”. Clique em avançar;
- Em delimitadores selecione o campo “vírgula”
- Em Qualificador de Texto deixe selecionado as aspas duplas (”);
- Clique em concluir.