ParentsAssistance

Projeto Arquitetural

There is guidance within this template that appears in a style named InfoBlue. This style has a hidden font attribute that allows you to toggle whether it is visible or hidden in this template. Use the Microsoft® Word® menu **Tools > Options > View > Hidden Text** check box to toggle this setting. There is also an option for printing: **Tools > Options > Print**.

# Objetivo

Este documento tem o objetivo de descrever filosofia, decisões, restrições, justificativas, premissas e quaisquer outros aspectos que dão forma ao projeto e sua implementação.

# Metas Arquiteturais e Filosofia

O fator predominante ativo que fez necessário a criação desse projeto foi a necessidade de um maior controle e acompanhamento das atividades dos filhos pelos pais, visto que conforme a sociedade foi mudando, mudou também a rotina de acompanhamento da educação familiar, com pais e mães tornando-se bastante ocupados até mesmo para seus filhos; por este motivo se fez necessário o ParentsAssistance, este aplicativo permitirá aos pais acompanharem de perto diariamente a rotina de estudos dos filhos, através de dispositivos móveis com Sistema Operacional Android com base de dados local; O Android teve  
iniciativa da Google de prover uma plataforma aberta para Web móvel.

Para a proposta da arquitetura, foram considerados fatores como a finalidade do sistema, tipo de usuários e ambiente de execução. Sendo assim, a arquitetura a ser adotado preciso atender às seguintes características:

* Modularidade: o sistema deve ser desenvolvido em camadas, havendo uma interface de comunicação bem definida entre as mesmas.
* Reusabilidade: a arquitetura do sistema deve ser tal que permita a utilização de classes e componentes em outros projetos, favorecendo o tempo de produção e a qualidade do produto gerado.

# Premissas e Dependências

# Requisitos Críticos da Arquitetura

**Segurança**

Se a segurança for um requisito crítico, a arquitetura deve ser Se a segurança for um requisito crítico, a arquitetura deve ser projetada de modo que as operações relacionadas à segurança projetada de modo que as operações relacionadas à segurança estejam todas localizadas em um único subsistema (ou em um estejam todas localizadas em um único subsistema (ou em um pequeno). Isso reduz os custos e os problemas de validação e pequeno). Isso reduz os custos e os problemas de validação e segurança e torna possível fornecer esse serviço a sistemas de segurança e torna possível fornecer esse serviço a sistemas de proteção relacionados.

**Disponibilidade** 

Se a disponibilidade for um requisito crítico, a arquitetura deve ser projetada para incluir componentes redundantes e, assim ser projetada para incluir componentes redundantes e, assim que possível, substituir e atualizar componentes sem parar o que possível, substituir e atualizar componentes sem parar o sistema.

**Facilidade de manutenção **

Aqui a arquitetura de sistema deve ser projetada usando Aqui a arquitetura de sistema deve ser projetada usando componentes de baixa componentes de baixa granularidade e autocontidos que possam ser prontamente mudados. Os criadores de dados possam ser prontamente mudados. Os criadores de dados devem ser separados dos clientes, e estruturas de dados devem ser separados dos clientes, e estruturas de dados compartilhadas devem ser evitadas.

**Os conflitos entre os requisitos na arquitetura** 

Podem ser resolvidos com o uso de arquiteturas diferentes podem ser resolvidos com o uso de arquiteturas diferentes para as diferentes partes do sistema.

. Abaixo há uma pequena lista de requisitos críticos da arquitetura:

* Requisitos de interação com outras aplicações;
* Manipulação de sensores;
* Requisitos web que resultam em aplicações hibridas (mobile - web);
* Diferentes famílias de hardware;
* Consumo de energia;

<http://developer.android.com/index.html>

<http://www.romulonunes.pro.br/es/material/Aula08ES-ProjetoDeArquitetura.pdf>

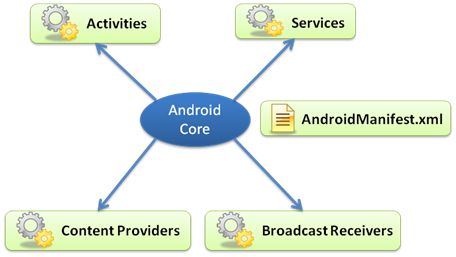
# Decisões, Restrições e Justificativas

* **Não utilizar arquitetura Web:** O sistema deverá ser construído de forma a ser mantida a comunicação e o gerenciamento de dados apenas na plataforma Android, ou seja, não se faz necessário um portal ou página web para tal, ou seja, a arquitetura é Android e apesar de utilizar conexão com a web para obter os dados do servidor, o desenvolvimento do aplicativo não deverá ser construído em cima da arquitetura web.
* **Facilidade de uso:** O sistema deverá prezar pela usabilidade, possuindo botões com descrição objetiva da funcionalidade a ser acessada, não sendo necessário o auxílio de links externos para o entendimento das funcionalidades.

# Mecanismos Arquiteturais

**Camada de aplicações:** onde se localizam todos os aplicativos que são executados sobre o sistema operacional, tais como, cliente de SMS e MMS, cliente de e-mail, navegador, mapas, calculadora, dentre outros.   
**Camada de Bibliotecas:** é a camada que possui as bibliotecas C/C++ que são utilizadas pelo sistema, e também bibliotecas de multimídia, visualização de camadas 2D e 3D, funções para navegadores web, funções de aceleradores de hardware, renderização 3D, funções para gráficos, fontes bitmap e vetorizadas e funções de acesso a banco de dados SQLite.   
**Camada de Runtime:** nessa camada se instancia a máquina virtual Dalvik, criada para cada aplicação executada no Android. Essa maquina virtual é a melhor referente a desempenho, maior integração com a nova geração de hardware e projetada como executar vários processos paralelamente.  
**Camada de Kernel Linux:**, o núcleo do sistema operacional Android é derivado do kernel 2.6 do Linux, herdando diversas características dessa plataforma. Parte importante que se utiliza do Linux na concepção do Google Android é o de controle de processos, gerenciarem memória, threads, protocolos de rede, modelo de drives e a segurança dos arquivos.

# Principais Abstrações



# Camadas do Framework da Arquitetura

**Activities** são as representantes das telas da aplicação. Associada a uma *activity* normalmente existe uma *view*, que define como será feita a exibição visual para o usuário. As *activities* são responsáveis por gerenciar os eventos de tela e também coordenam o fluxo da aplicação.

Os **Services** são códigos que executam em segundo plano. Normalmente são utilizados para tarefas que demandam um grande tempo de execução.

Os **Content Providers** (provedores de conteúdos) são a maneira utilizada pela plataforma para compartilhar dados entre as aplicações que executam no dispositivo. Um exemplo bem claro disto é a aplicação de gerenciamento de contatos do Android, que é nativa. Aplicações desenvolvidas por terceiros podem utilizar um content provider a fim de ler os contatos armazenados no dispositivo de forma simples.

Os Broadcasts **Receivers** são componentes que ficam "escutando" a ocorrência de determinados eventos, que podem ser nativos ou disparados por aplicações. Uma aplicação pode, por exemplo, utilizar um *broadcast receiver* para ser avisada quando o dispositivo estiver recebendo uma ligação e, com base nessa informação, realizar algum tipo de processamento.

Junto os estes componentes, existe o arquivo de manifesto **AndroidManifest.xml**. Ele é obrigatório e único para cada aplicação. É nele que são feitas as configurações gerais da aplicação e dos componentes que fazem parte dela. E, juntando tudo isto, existe a figura do **Android Core**, que na verdade não é um componente específico, mas sim a plataforma Android propriamente dita. É ele quem proporciona a interação entre os componentes e as aplicações e torna possível a execução do código.

# Visões Arquiteturais

