# MMUnB

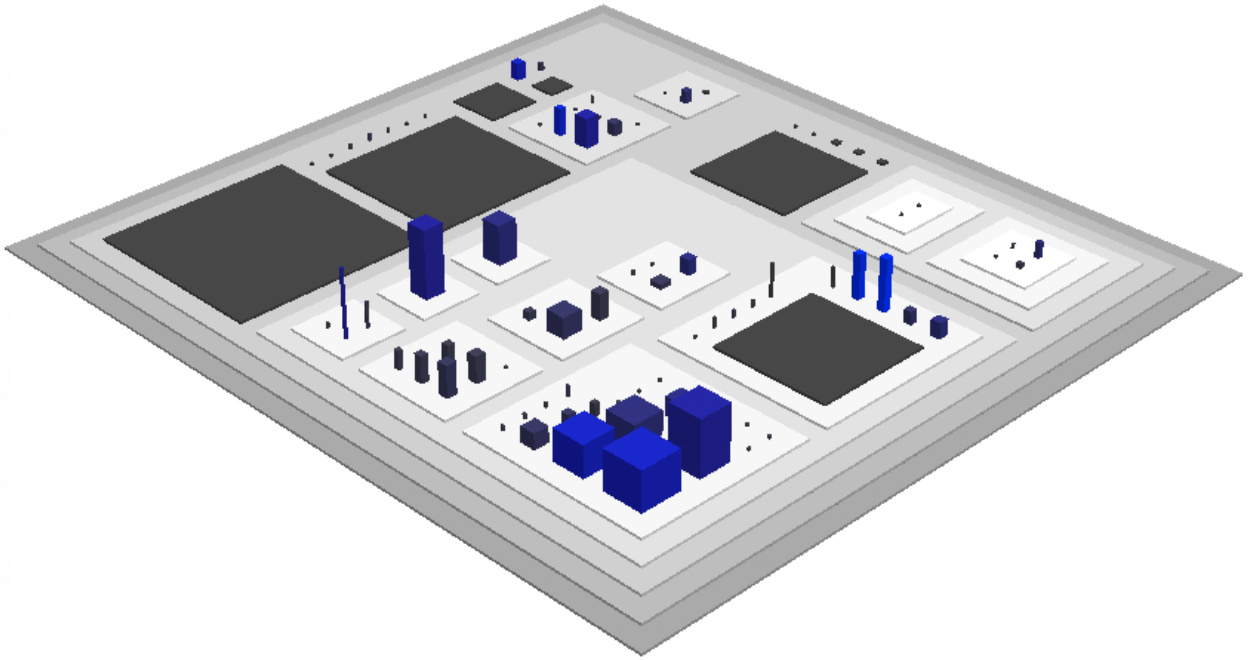
# MMUNB

Esse projeto tem como objetivo servir como base para o ensino do desenvolvimento na plataforma Android.

Consiste em uma nova versao do Mobile Media, player multimidia para aplicacões móveis e bem disseminado como estudo de caso para avaliar implementacões de Linhas de Produtos de Software.

As versões anteriores eram para a plataforma Java Micro Edition.

# MMUNB - CIDADE



# MMUNB - CIDADE

1) Os prédios que tem uma largura muito grande, porém sem altura, significa que que essa classe (prédio) tem muitos atributos e poucos métodos, como por exemplo:

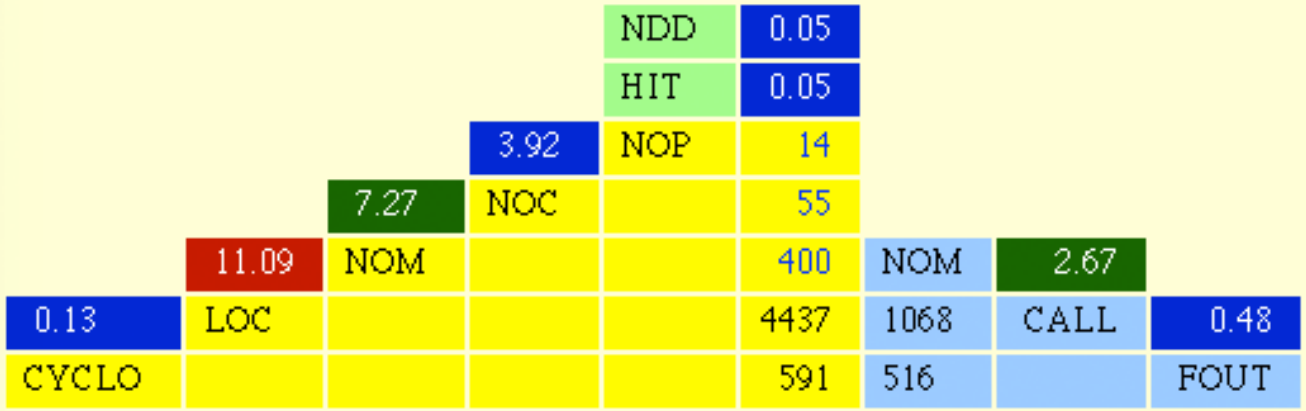
classes drawable ( GUI ) , ID, Strings, Constantes
( Muitas frases já escritas, e também ).

2) Já os prédios com altura (azul) tem métodos e classes, e azul significa um tamanho razoável.

3) Temos muitos quarteirões que são pacotes.

# MMUNB – PIRÂMIDE DE MÉTRICAS

System Detail: /Users/michaeldfti/Desktop/MMAndroid/Fontes




Interpretation of the Overview Pyramid for module /Users/michaeldfti/Desktop/MMAndroid/Fontes

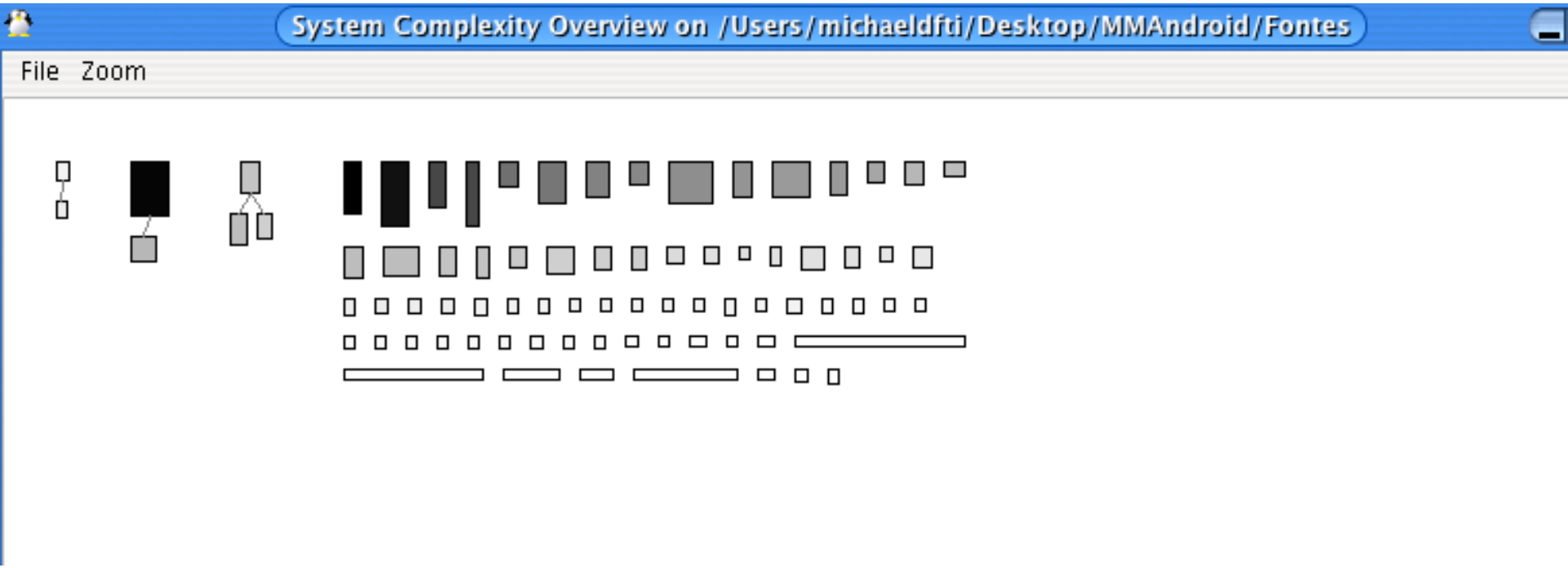
# MMUNB – PIRÂMIDE DE MÉTRICAS

**LOC:** Linhas de código do software

**NOM:** Número de métodos

**NOC:** Número de classes

# MMUNB – VISÃO POLIMÉTRICA



# MMUNB – DESCRIÇÃO IMPLEMENTAÇÃO MICHAEL

Em reunião ficou definido que eu iria implementar algums features para o projeto já em andamento MMUnB.

Após o usuário entrar no aplicativo, deveria-se verificar a existência de algum áudio no banco de dados, se o banco estiver vazio, uma mensagem pedindo autorização para sincronização de pastas com o banco aparecerá, caso o usuário aceite, o sistema faz uma verificação de todas as músicas do celular, e inclui no banco uma a uma.

# MMUNB – DESCRIÇÃO IMPLEMENTAÇÃO MICHAEL

O aplicativo se inicia pela classe **MMUnBActivity** dentro dessa classe, chamo o método **DefaultAudioListDAO countAudio.countListAudioBanco()** no qual realiza a contagem de músicas que tem cadastradas no banco, caso a contagem seja 0, o método **mensagemSincronizarVazio** da própria classe **MMUnBActivity** é chamado.

Caso o usuário aceitar a sincronização, o método **sincronizarTudo** é chamado.

# MMUNB – DESCRIÇÃO IMPLEMENTAÇÃO MICHAEL

O método **sincronizarTudo,** recolhe todas as músicas de **ListAllFiles getAllMusic().** Assim música por música de todas as pastas do dispositivos são incluídas no banco.

Com auxílio da classe **DefaultAudioExtractor** informações de album e autor são extraídas do arquivo mp3.

# MMUNB – DESCRIÇÃO IMPLEMENTAÇÃO MICHAEL

Após colher o nome do autor, eu verifico se esse autor já existe no banco com o método **ManageAuthor** da classe **DefaultAuthorDAO**, onde a busca é feita pelo ID.

O ID é gerado pelo método **nameUnicAuthor** da classe **DefaultAudioExtractor** assim: Retira todos os caracteres especiais do nome do autor, inclusive espaços, e trasnforma para maiúsculo, após isso gera um hashcode dessa transformação.

# MMUNB – DESCRIÇÃO IMPLEMENTAÇÃO MICHAEL

Caso esse ID do autor não esteja no banco, cadastraremos um autor com o método **adicionaAuthor.**

Assim, matemos uma lista única de autores.

# MMUNB – OBSERVAÇÕES

1) A dinâmica de cadastramento de autores não estava dentro do escopo do projeto inicial do projeto planejado em reunião.

2) Também foram criadas ideias para desenvolvimento da GUI do aplicativo.

3) Alteramos o ícone do aplicativo.

4) O aplicativo está no github em MMAndroid, no branch BetterUI