# 2 DESENVOLVIMENTO

Neste capítulo será apresentado o método de desenvolvimento escolhido para a gerência desse trabalho, as ferramentas utilizadas, a versão final da arquitetura e os graus de dificuldade encontrados.

## 2.1 Método de Desenvolvimento

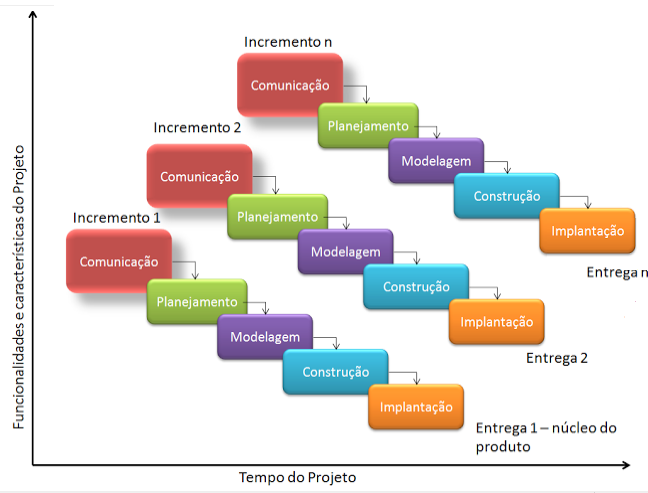
A fim de auxiliar no desenvolvimento do sistema descrito na monografia, a metodologia de desenvolvimento de software escolhida foi o modelo incremental. Com isso, fez-se com que o sistema fosse dividido em quatro incrementos, sendo que, ao final de cada iteração é produzido um conjunto de itens utilizáveis.

## 2.1.1 Modelo Incremental

O modelo incremental (Pressman, 2006) tem como vantagem dividir a construção do sistema em vários incrementos, no qual, em cada um dos incrementos é realizado todo o ciclo de desenvolvimento de software, desde a definição dos requisitos até os testes. Isso tudo, faz com que ao final de cada incremento, o sistema seja testado e analisado minuciosamente, fazendo com que ele esteja o mais estruturado possível para o próximo incremento e ajudando a diminuir o risco de problemas graves serem encontrados no final.

Para que a metodologia ofereça melhores resultados, é necessário que cada estágio seja bem planejado e gerenciado, buscando identificar o máximo de erros e riscos possíveis para produzir um sistema confiável e dentro do cronograma.

A Figura 1 apresenta uma visão melhor sobre o modelo incremental, suas funcionalidades e características.



1. Modelo Incremental

Em cada incremento, esse método de desenvolvimento apresenta as seguintes funcionalidades:

* Fase de comunicação, onde reuniões frequentes são realizadas com o cliente a fim de extrair o máximo de requisitos possíveis e estabelecer quais objetivos seriam traçados.
* Fase de planejamento, onde são definidos os prazos e caminhos a serem traçados para alcançar os objetivos.
* Fase de modelagem, que é a fase onde são criados diagramas contendo a arquitetura do sistema junto com todos os procedimentos e detalhes utilizados na implementação dos requisitos.
* Fase de Construção ou desenvolvimento, que é a fase responsável pelo desenvolvimento físico do software, é feita a codificação para uma linguagem legível pela máquina, onde ela possar interpretar todas as funcionalidades desenvolvidas. Testes e análises também são realizados a fim de garantir que os objetivos foram atingidos e identificar erros e riscos; na fase de implantação, é a fase onde uma versão executável do software é gerada e transferida para ambiente de produção.
* Fase de implantação, é a fase onde o software é transferido do estado de desenvolvimento para a área de produção.

Pelo fato deste sistema não ter sido implantado em nenhuma área de produção, a fase de implantação não foi utilizada e ao invés dela, foi utilizado a fase de integração, que consiste em integrar o incremento que acabou de ser produzido a todos os outros incrementos que já foram desenvolvidos anteriormente.

**2.1.2 Gerência do Projeto**

A gerência desse projeto utilizando-se o modelo incremental foi iniciada após terem sido recolhidos todos os requisitos iniciais do sistema. Dados esses requisitos, foi definido que o sistema seria dividido em quatro incrementos, todos com a mesma duração.

A divisão dos incrementos foi feita da seguinte forma:

* O primeiro incremento consistia em desenvolver o módulo relacionado aos cadastros dos usuários, que possuia as funcionalidades de login, redefinição de senha e configurações de cadastro.
* No segundo incremento consistia em implementar os módulos de integração com as redes sociais Deezer e Facebook. A integração levava em conta implementar também as funcionalidades de manipulações e compartilhamento de músicas, além da funcionalidade que oferecia a possibilidade de usuários se cadastrarem no sistema através de sua conta no Facebook.
* O terceiro incremento foi definido que fossem desenvolvidos dois módulos. O módulo relacionado a manipulações de playlists dos usuários e o módulo que controlava as amizades e requisições de amizade entre os usuários.
* O quarto e último módulo consistia na implementação do algoritmo de recomendação de músicas do sistema.

No início de cada incremento, um reunião era feito junto ao cliente com o intuito de extrair o máximo de requisitos possíveis relacionados ao incremento que fosse se iniciar. Dados esses requisitos, era feito um planejamento para definir quais funcionalidades deveriam ser implementas primeiro, qual o tempo de duração de cada uma delas e quais seriam mais suscetíveis a erros, exigindo maior atenção. Com o planejamento desenvolvido, todas as tabelas e dados eram modelados e gerados, dando ínicio ao desenvolvimento das funcionalidades. Após cada incremento ter sido desenvolvido ele era integrado aos outros.

Durante o desenvolvimento do sistema, houveram alguns atrasos que acarretaram em mudanças no cronograma do projeto. Os atrasos, na maioria maior parte dos casos, foram relacionado ao fato do autor não possuir um bom domínio das linguagens de programação utilizadas no projeto, outros atrasos que surgiram foram acarretados pela quantidade de disciplinas que o autor estava cursando na faculdade.

Os problemas de atrasos citados acima ocorrem no primeiro incremento do projeto. Para isso, foi necessário aumentar o número de horas trabalhadas neste incremento, diminuindo a quantidade de horas dos outros três incrementos. Com essa mudança, foi possível concluir todas as atividades no prazo pré-determinado.

.

**2.2 Tecnologias e Ferramentas**

Neste item serão descritas as ferramentas e tecnologias utilizadas para o desenvolvimento do sistema.

**2.2.1 Eclipse**

O Eclipse (Eclipse, 2014) é um ambiente de desenvolvimento portátil e de código aberto, utilizado para desenvolvimento da linguagem Java. Essa plataforma oferece diversas facilidades para o programador como: refatoração do código, identificação de erros e depuração de projetos. Toda a codificação do sistema foi realizada nesse ambiente.

## 2.2.2 PostgreSQL

O banco de dados relacional escolhido para o armazenamento dos dados gerados pelo sistema foi o PostgresSQL (Postgres, 2014). Ele foi escolhido por ser um sistema de código aberto com muita qualidade, eficiência e por oferecer um software chamado pgAdmin, que auxilia na gerência do banco de dados PostgreSQL.

## 2.2.3 Java

Java (ORACLE, 2014) é uma linguagem de programação orientada a objeto portátil, gratuita e bastante difundida. Ela foi escolhida para a codificação do sistema pelo fato de ser bem documentada e padronizada, possuir *frameworks* que auxiliam na implementação do sistema, possuir comunidades ativas em que usuários discutem seu funcionamento, ser orientada a objetos e possuir uma interface JDBC de comunicação com o banco de dados.

## 2.2.4 XHTML

*Extensible HyperText Markup Languag*, ou XHTML (W3C, 2014), é uma linguagem de construção de páginas na internet criada a partir do HTML (*HyperText Markup Language*) juntamente com o XML (Extensible Markup Language).

Essa linguagem foi utilizada para a construção de todas páginas web do sistema.

## 2.2.5 CSS

*Cascading Styles Sheets*, ou CSS (W3C, 2014), é uma linguagem utilizada para a construção de layouts de páginas *web*.

Ela foi utilizada na personalização do layout das páginas *web* do sistema, adicionando estilos agradáveis em cada uma delas.

## 2.2.6 ERMaster

ERMaster (ERMaster, 2014) é um *plugin* do Eclipse, que foi utilizado para criar a modelagem de banco de dados do projeto.

## 2.2.7 JSF

*Java Server Faces*, ou JSF (ORACLE, 2014),é um *framework* MVC utilizado para criar aplicações Java para *web*. Ele foi escolhido pelo fato de possuir componentes visuais pré-prontos e possuir Java APIs para a representação desses componentes no lado servidor, possibilitando a manipulação de eventos.

## 2.2.8 JavaScript

JavaScript (MOZILLA DEVELOPER NETWORK, 2014), é uma linguagem de programação interpretada, utilizada para que scripts executados no lado cliente interagissem com o usuário sem a necessidade de passar pelo lado servidor. Ele foi utilizado para auxiliar na integração com o Facebook e com o Deezer e na confecção de páginas *web* do sistema.

## 2.2.9 JSON

JavaScript Object Notation, ou JSON (JSON GROUP, 2014), é uma formatação leve de troca de dados que usa convenções similiares às diversas linguagens de programação sendo de fácil entendimento para seres humanos e máquinas. Ele foi utilizado na consulta de dados relaciondados à músicas no *webserver* do The Echo Nest.

## 2.2.10 APIs Utilizadas

Para este sistema foram utilizadas as seguintes APIs: The Echo Nest, Facebook, Primefaces e Deezer.

A primeira, The Echo Nest, é uma empresa de plataforma de inteligência musical que oferece serviços de música para desenvolvedores e empresas. Sua API foi utilizada para fazer requisições de dados relacionados á músicas e bandas no *webserver* da empresa.

A API do Facebook foi utilizada para obter dados musicais, como páginas de bandas curtidas do usuário que possui cadastro integrado com o Facebook e oferecer ao usuário, a oportunidade de compartilhar músicas no Facebook.

O Primefaces é uma biblioteca para JSF que possui componentes prontos, permitindo criar interfaces para aplicações *web* de forma bastante eficiente e simples. Neste projeto, alguns componentes do Primefaces foram utilizados com o intuito de melhorar o funcionamento das interfaces e deixar o *layout* do sistema mais agradável aos olhos dos usuários.

A API do Deezer foi utilizada para realizar o *streaming* das músicas no sistema.

## 2.2.11 Controle de Versões

Os software de controle de versão utilizado neste projeto foram: o Git e GitHub. O Git é um sistema de controle de versão e gerenciamento de código fonte e o GitHub nada mais é do que um serviço de *web hosting* compartilhado para projetos que usam controle de de versionamento Git. Foram feitas as configurações necessárias para que o controle das versões do projeto de TCC pudessem ocorrer, estas configurações estam no Apêndice A.

## 2.2.12 *Backup*

## 2.3 Arquitetura do Sistema

## 2.4 Grau de Dificuldade – Aspectos de Aprimoramento

Nas subseções a seguir são descritos os aspectos de aprimoramento envolvidos no projeto.

## 2.4.1 Algoritmos de Recomendação

O projeto conta com uma inteligência artificial responsável por recomendar músicas que tendem a agradar ao gosto do usuário. Para isso, foi necessário muito estudo sobre algoritmos de recomendação existentes para escolher o que melhor se encaixasse na realidade do projeto e qual seria a melhor forma de implementá-lo.

Incialmente, foi pensado em se utilizar o algoritmo Slope-One, usado em Filtragem Colaborativa, que se baseia em calcular predições a partir da comparação de avaliações de usuários a certos itens. Porém após muitos estudos e discussões com a orientadora a fim de encontrar um algoritmo que pudesse melhorar a qualidade das recomendações feitas aos usuários, o algoritmo K-Means foi escolhido.

O algoritmo K-Means (MacQueen, 1967) é um algoritmo de aprendizado não supervisionado que visa agrupar dados em um número finito de grupos definido pelo usuário. Neste projeto, ele foi utilizado para realizar o *clustering* de atributos presentes nas músicas avaliadas por usuários a fim de encontrar músicas a serem recomendadas a eles.

## 2.4.2 Layout Responsivo

Como atualmente o meio que os usuários acessam aplicativos web são diversos, foi definido que este projeto possuísse layouts que fossem adaptavéis a desktops e smartphones. Com isso foi necessário muito empenho por parte do autor para que todas as interfaces do sistema fossem feitas sob medida para obter o melhor resultado possível quando forem visualizadas nos dispositivos citados acima.

## 2.4.3 Multiusuário

Este projeto é um sistema multiusuário, pelo fato dele suportar o acesso de múltiplos usuários simultaneamente. Para isso, foi necessário maior estudo sobre o funcionamento deste tipo de sistema e como ele diferencia usuários diferentes permitindo que eles se cadastrem, façam login ou realizem qualquer outro tipo de ação no sistema, ao mesmo tempo sem ocorrer manipulação de informações erradas.

## 2.4.4 Utilização de APIs externas

Para a integração deste sistema, foram utilizadas as seguintes API’s dos seguintes sistemas externos: Facebook e Deezer. Para ambos foi necessária a leitura de forúns e tutoriais pela internet, pois ambas são integradas com a linguagem de programação JavaScript, que não era de muito conhecimento do usuário até então.

## 2.4.5 Programação *Web*

Um dos maiores desafios desse trabalho foi a programação *web* pois o autor não possuia conhecimento sobre o gerenciamento de requisições assíncronas, como era o ciclo de vida das páginas da internet e além disso, não tinha possuia conhecimento sobre nenhuma das linguagens de programação escolhidas: JSF, JavaScript, CSS e XHTML. Isso tudo fez com que o autor lesse livros específicos sobre as linguagens citadas e procurasse alguns tutoriais na internet para alcançar conhecimento necessário e superar essa dificuldade.

## 2.4.6 Envio de mensagem de email ?

# 4 Recmusic – sistema de *Steaming* e recomendação de músicas

Neste capítulo será apresentada a arquitetura do sistema com todos os módulos que ela contém e uma explicação detalhada de todas as funcionalidades contidas nesses módulos.

## 4.1 Arquitetura do RecMusic

O diagrama de arquitura da Figura ? oferece uma visualização simplificada dos principais componentes que fazem parte do sistema. Ele tem como objetivo mostrar como são distribuídas todas as funcionalidades presentes nos módulos.

Como pode ser visto no diagrama, o sistema foi dividio em seis módulos: Cadastro, Amigos, Facebook, Músicas, Playlists e Recomendação. O módulo de Cadastro faz parte do primeiro incremento do sistema, ele foi representado pela cor verde no diagrama de arquitetura. Já os módulos Música e Facebook, que fazem parte do segundo incremento, são representados pela cor vermelha. O terceiro incremento, em que foi implementado os módulos Amigos e Playlists, pode ser visualizado no diagrama de arquitetura pela cor laranja. Por fim, o módulo Recomendação, representado pela cor azul, compõe o quarto e último incremento do sistema.

A arquitetura final do sistema não sofreu modificações relevantes se comparada a que arquitetura elaborada inicialmente. A única modificação que merece destaque, foi a adição da funcionalidade que permite o *upload* de fotos por parte dos usuários para complementar os seus perfis.

Nas subseções a seguir, são descritos todos os módulos do sistema acompanhandos de todas as suas funcionalidades.

## 4.1 Módulo Cadastros

O módulo de Cadastro consiste na manipulação de cadastros de todos os usuários do sistema. O RecMusic não possui nenhum tipo especial de usuário, portanto todos tem permissões iguais para todas as interfaces do sistema. Porém, o cadastramento no site pode ocorrer de duas maneiras: pelo Facebook ou na forma de cadastro tradicional em que o usuário digita seus dados e escolhe uma senha com tamanho maior que seis caracteres.

Somente será possível alterar as configurações de cadastro do usuário se ele estiver logado no sistema. Caso ele tenha esquecido a sua senha de autenticação, ele poderá clicar na opção “Esqueceu sua senha?”. Com isso ele será redirecionado a uma página que requisita a inserção de seu endereço de email cadastrado no sistema. Validado o endereço de email inserido pelo usuário, o sistema enviará uma mensagem de email, que contém um *link* que redirecionará o usuário a uma página de redefinição de senha.

### 4.2 Módulo Amigos

Este módulo, consiste na manipulação de requisições de amizade e visualizações de perfil entre usuários do sistema.

Como toda rede social, esta permite que usuários possuam amigos, na qual podem visualizar informações pessoais e musicais de cada um desses amigos. Para dois usuários se tornarem amigos, é preciso que um deles procure pelo outro no sistema; através de seu nome, login ou email; e ao entrar em seu perfil, o envie uma solicitação de amizade. Porém a amizade será confirmada apenas quando o outro usuário responder confirmando a amizade.

Caso um usuário não queria mais ser amigo de outro usuário, ele tem a opção de desfazer a amizade, basta ele entrar no perfil desse usuário determinado e clicar no botão correspondente à “Desfazer Amizade” que a amizade será desfeita sem a necessidade de nenhuma confirmação por parte do outro usuário.

Cada usuário do sistema possui uma página de perfil própria que contém suas informações pessoais: login, nome, sobrenome, sexo e email; informações sobre suas playlists, músicas avaliadas e a sua lista de amigos. Essas páginas de perfil, só poderão ser completamente visualizadas por outros usuários caso eles sejam amigos, caso contrário só serão visualizadas as informações pessoais e a lista de amigos.

O sistema possui uma interface própria para o usuário visualizar sua lista de amigos e pesquisar por outros usuários do sistema, onde ele pode visualizar seus perfis e enviar requisições de amizade. Esta interface também oferece ao usuário, a lista de requisições de amizade feitas por outros usuários a ele. O usuário tem a opção de aceitar ou recusar essas requisições.

### 4.3 Módulo Músicas

Como o próprio nome diz, este módulo consiste na manipulação de músicas no sistema.

A pesquisa de música pode ser realizada em um espaço na parte superior da interface do sistema onde, cada usuário pode optar por procurar músicas apenas pelo nome, nome da banda ou ainda, pelo nome da música e nome da banda; porém nesse último caso, o caracter “-” teria de ser adicionado entre o nome da música e nome da banda como, por exemplo, “*Black – Pearl Jam*”. Para sinalizar como será a pesquisa, dois *checkbox’s* são disponibilizados, um indicando que a pesquisa será apenas por banda e o outro por música; caso os dois estiverem ticados na hora da pesquisa, o sistema procurará por músicas e bandas.

Para a pesquisa de músicas, o sistema faz uma requisição na API do The Echo Nest, refenciada na Subseção 2.2.10, que devolve no formato JSON, informações relacionadas às músicas procurada como: *id* da música no Deezer, nome da música, nome da banda, álbum da música, informações sobre à BPM da música, url da imagem da capa do álbum da música.

O *streaming* de músicas na interface do usuário acontece através de um player disponibilizado pela API do Deezer. Na requisição deste player, é passado como parâmetro o id das músicas que serão reproduzidas. Para que o *streaming* das músicas seja completo, é necessário que o usuário esteja logado no Deezer, caso contrário, o usuário só poderá ouvir apenas 30 segundos de cada música. A interface ainda oferece uma lista de músicas ,relacionadas a música em questão, para que o usuário possar conferir.

De acordo com cada música pesquisada, o sistema permite que o usuário avalie cada uma delas. A avaliação corresponde a uma nota de 1 a 5 e é feita na mesma interface em que a música é reproduzida. Essas avaliações serão utilizadas no módulo de recomendações de músicas, que será descrito na subseção 4.?.

O sistema disponibiliza duas interfaces que disponibilizam respectivamente, uma lista com as 25? músicas melhores avalidas por todos usuários dos sistema e uma lista com todas as músicas avaliadas pelo usuário em questão.

### 4.4 Módulo Facebook

O módulo Facebook é responsável por realizar toda a comunicação do sistema com a rede social Facebook. A intregação com o Facebook, como explicada na seção 2..2.10, é feita com o uso de uma API disponibilizada pelo próprio Facebook, com ela, o usuário tem a opção de se cadastrar no sistema através de seu *login* do Facebook. Caso isso aconteça, o sistema coletará automaticamente algumas informações sobre o gosto musical desse usuário. Essas informações serão utilizadas na recomendação de músicas.

Através da integração com o Facebook, o sistema ainda oferece a possibilidade dos usuários compartilharem músicas, do RecMusic, no Facebook, o que, poderá de qualquer forma, divulgar o RecMusic para outras pessoas.

### 4.5 Módulo *Playlist*

O módulo *Playlist* é responsável por controlar as manipulações das playlists criadas pelos usuários do sistema.

Cada usuário poderá criar quantas playlists ele quiser e para a criação de cada uma delas há apenas um campo a ser preenchido que diz respeito ao nome da playlist. Cada playlist poderá conter um número ilimitado de músicas e a adição delas é feita na mesma tela em que as músicas são reproduzidas.

Cada playlist criada pelo usuário possui uma interface específica. Nesta interface, o sistema lista e oferece o *streaming* de todas as músicas da playlist. Caso o usuário não goste de alguma música, esta interface oferece a possibilidade do usuário removê-la da playlist.

### 4.6 Módulo Recomendação

O sistema de recomendação criado neste projeto combina duas abordagens diferentes: a filtragem baseada em conteúdo e a filtragem colaborativa. A mistura de duas abordagens diferentes foi utilizada para que uma tente fortificar e superar as desvantagens da outra, fazendo com que a recomendação obtenha resultados mais precisos. Pelo fato de ser utilizadas duas abordagens, faz com que esse sistema seja caracterizado com um sistema de recomendação híbrido.

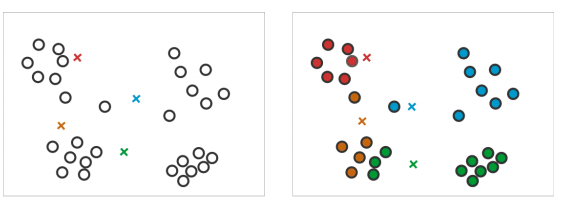
A filtragem baseada em conteúdo consiste em recomendar itens semelhantes a que o usuário preferiu no passado, levando-se em conta atributos desses itens. Nesse projeto, todas as recomendações de músicas são realizadas em cima dos gêneros de músicas que o usuário avaliou positivamente no passado. Para que seja considerado uma avaliação positiva, as músicas devem receber notas maiores ou iguais a 3.

A filtragem colaborativa, basea-se em recomendar itens que pessoas com gostos semelhantes preferiram no passado. A grande qualidade dessa técnica, está na troca de experiências entres os usuários, assumindo que pessoas de mesmo gosto possuem os mesmos interesses. No caso desse sistema, para encontrar usuários diferentes mas, que possuem gostos comuns, foi utilizado o valor médio da BPM (Batidas Por Minuto)? das músicas avaliadas positivamente pelos usuários. A BPM é um atributo da música, onde é calculado a quantidade de batidas que a música possui a cada minuto, esse atributo ajuda a diferenciar músicas mais lentas das músicas mais rápidas. Através desse atributo, o sistema classifica usuários que têm preferência por músicas mais lentas como tendo gostos semelhantes a outros usuários que também preferem músicas lentas e usuários que gostem de músicas agitadas com gostos semelhantes a outros usuários que preferem músicas agitadas.

Para melhorar o processamento na hora da recomendação, o sistema armazena automaticamente informações sobre os gostos dos usuários a medida que eles vão avaliando novas músicas. Para cada música avaliada positivamente pelo usuário, ele tem o seu cadastro atualizado como tendo gostado de todos os gêneros daquela música e para cada um desses gêneros, é calculado a média do valor médio da BPM de todas as músicas, desse mesmo gênero, avalidas positivamente pelo usuário. Ou seja, cada usuário possui uma tabela com todos os gêneros de músicas que ele gostou e cada um desses gêneros possui a média das BPM’s dessas músicas.

Durante a recomendação, são sorteados 3 gêneros que o usuário tenha gostado; caso não tenha gostado de pelo menos 3 gêneros, o sistema recomenda músicas em cima dos gêneros que existirem e para complementar a recomendação, é recomendado uma lista com as músicas com melhores avaliações no sistema. Após selecionados os gêneros, o sistema consulta no banco de dados todos os outros usuários do sistema que também gostaram daquele gênero, guardando o valor correspondente a média das BPM’s das músicas avaliadas positivamente por esses usuários aquele gênero. Em cima dessa média das BPM’s do usuário que está recebendo a recomendação e a de todos os usuários descritos anteriormente, é executado o algoritmo *K-Means*.

O algoritmo *K-Means* (K-MEANS CLUSTERING, 2010) é um algoritmo de agrupamento de dados, tendo como base encontrar a melhor divisão para agrupar um conjunto de dados em um número definido de grupos. A Figura ? mostra um exemplo com um conjunto de dados antes e depois de serem agrupados em blocos diferentes.



1. Agrupamento de dados.

Nesse sistema, foi definido que a quantidade de grupos, cujos dados serão agrupados, será 3 e que os dados de entrada serão a lista com a média das BPM’s das músicas de um deterterminado gênero do usuário que está recebendo a recomendação e a média das BPM’s das músicas de todos os outros usuários que gostaram daquele gênero.

Os passos para execução do *K-Means* são:

1. Dada a lista com os dados de entrada, são atribuídos a cada uma das três centróides, que são os valores iniciais de cada *cluster*, os três primeiros valores da lista. O valor relativo ao usuário que está recebendo a recomendação sempre fica no final da lista.
2. Guarda-se os valores das centróides dos três blocos.
3. Percorre todos os elementos da lista e atribui-os ao bloco onde a diferença para a centroíde do bloco seja a menor.
4. Recalcula-se o valor das centóides de cada grupo, sendo esta, a média do valores dos objetos atuais do grupo.
5. Através dos valores guardados das centróides iniciais de cada bloco, verifica-se se eles são iguais aos valores atuais, se isso acontecer o algoritmo converge e termina sua execução. Caso os valores sejam diferentes, o algoritmo volta para o passo II. Para evitar que o algoritmo entre em loop, foi colocado um máximo de 150 iterações, caso ele não termine ao final dessas 150 iterações, a execução do algoritmo é finalizada automaticamente.

Ao final da execução do algoritmo, são selecionados todos os usuários que estiverem no mesmo bloco do usuário que estiver recebendo as recomendações. Através desses usuários, são escolhidas músicas que eles tenham gostado e que sejam do mesmo gênero que está sendo feita a recomendação. Cada recomendação seleciona seis músicas por gênero para serem recomendadas porém, uma vez que essas músicas tenham sido recomendadas, a menos que não exista outras músicas a serem recomendadas, elas so serão recomendadas novamente depois de passados 20 minutos.

Para cada recomendação de músicas por gênero, o usuário tem a opção de avaliá-las atribuindo uma nota de 1 a 5, como acontece na avaliação das outras músicas. Ao selecionar uma nota, ela será automaticamente atribuída para todas as músicas recomendadas do gênero em questão. Essa foi uma maneira encontrada para incentivar o usuário a avaliar cada vez mais músicas, para aumentar a precisão das recomendações.