

# Capítulo 4

## **Implementação de um sistema de transposição automática de seqüências musicais MIDI baseada na tessitura de um determinado cantor**

### **4.1 Observações Iniciais**

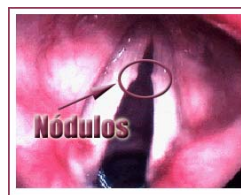
Esta dissertação apresenta uma solução de como resolver um relevante problema no domínio musical, podendo, o mesmo, acarretar danos à saúde e transtornos profissionais aos músicos em geral. Tais problemas surgem quando os músicos efetuam *performances* não condizentes com as esperadas tecnicamente na execução de uma determinada peça musical.

Diz-se isso quando um músico, principalmente os cantores, tenta executar uma peça musical e não consegue reproduzir, mesmo forçando, algumas freqüências (notas musicais) existentes na obra em questão. Neste caso, o menor transtorno que poderá ocorrer a este músico é a frustração de produzir uma interpretação desagradável para si mesmo e para o público que estiver presente.

Este tipo de execução forçada pode produzir um efeito colateral grave à saúde deste cantor, onde seu instrumento musical é o seu próprio sistema fonador. Tal problema ocorre devido ao fato do cantor ter que realizar um grande esforço para emitir

freqüências fora de sua tessitura vocal<sup>1</sup>, desafinando, ou, até mesmo, não conseguindo emitir determinadas notas musicais fora de seu range<sup>2</sup> vocal.

A repetição sistemática deste tipo de ação, ou seja, esforços excessivos em suas pregas vocais pode acarretar calosidades (nódulos) e outros problemas nas mesmas, conforme mostrado na figura 4.1 e no Capítulo 1 desta dissertação (com mais detalhes), os quais podem comprometer a carreira profissional deste músico.



**Figura 4.1** – Nódulos em pregas vocais

A ocorrência de nódulos vocais poderá levar o músico a ser submetido a longos tratamentos com fonoaudiólogos, e, até mesmo, a ter que fazer cirurgia(s) para remoção dos mesmos, podendo, em alguns casos, tornar tal profissional não mais apto a executar profissionalmente seus trabalhos com a qualidade necessária.

Não menos relevante do que a questão de possíveis danos à saúde do cantor é o dano profissional acarretado à carreira do mesmo. O fato de ele desafinar ao executar uma peça musical ou não conseguir emitir sons registrados na mesma durante uma apresentação, pode acarretar traumas que o incapacite de exercer novamente sua profissão com segurança, bem como pode levar o mesmo ao descrédito e discriminação pelo público em geral.

## **4.2 O problema da implementação do sistema**

Conforme visto no capítulo 2, o protocolo MIDI e os arquivos SMFs carregam potencialmente todas as informações musicais necessárias para que um sistema especialista possa realizar várias tarefas no domínio musical.

---

<sup>1</sup> Tessitura vocal – faixa de freqüência que um cantor consegue emitir com qualidade, sem distorção e sem esforço excessivo.

<sup>2</sup> Range – faixa de freqüência do espectro audível.

Diz-se potencialmente porque muitas das informações relevantes não estão explicitamente registradas nos arquivos SMF ou nas mensagens de tempo real do protocolo MIDI. Vários parâmetros musicais importantes, tais como: duração de uma nota musical (figura musical), divisão de compassos, tonalidade real e metrônomo devem ser identificados e quantificados através de um sistema especialista a ser desenvolvido pelo programador. Implementar tais sistemas demanda do programador um conhecimento profundo do protocolo MIDI, da estrutura dos SMFs, das especificidades dos fabricantes de equipamentos MIDI, de lógica e de manipulação de dados em sistema de numeração binário.

O conhecimento e as informações de como proceder a este tipo de implementação, principalmente no tocante a MIDI, são difíceis de serem obtidos, já que os mesmos não estão registrados em literatura nem disponibilizados pelos desenvolvedores de softwares ou mesmo pela sociedade afim (MMA-MIDI Manufacture Association).

Felizmente, tal tecnologia já é de conhecimento e domínio da equipe de pesquisadores da FEELT-UFU, a qual está registrada em vários trabalhos de mestrado, doutorado e iniciação científica desta instituição, bem como em dezenas de artigos, dicas publicadas e vários livros e revistas especializadas da área.

Um outro fator limitante, cuja solução está na escolha de um paradigma de programação adequado, se dá na programação de sistemas MIDI, onde se necessitam, na maioria das linguagens atuais, principalmente as procedimentais, de drivers e dlls específicos para controle e manipulação das placas e dos recursos sonoros do computador, os quais dependem de máquina e do sistema operacional utilizados.

A escolha pela linguagem funcional CLEAN, vem tornar o desenvolvimento e o código gerado legível e portátil, prescindindo-se de ter que atualizar drivers e dlls em cada plataforma e sistema operacional onde o aplicativo for rodar.

### 4.3 Requisitos mínimos necessários para implementação dos objetivos traçados

Para que se possa implementar um sistema que realize a transposição automática de uma seqüência MIDI para uma região de conforto de um determinado cantor com um range vocal específico, é necessário que o sistema, antes, seja capaz de identificar e realizar as seguintes tarefas:

1. Ser capaz de abrir um arquivo MIDI;
2. Seja capaz de reproduzir este arquivo para apreciação do cantor;
3. Separar os canais MIDI;
4. Identificar e separar em cada canal MIDI os eventos, as mensagens de ativação e desativação de notas das mensagens de sistema, para que se possa reconhecer as mensagens MIDI, através da transformação dos deltaTimes em unidades de tempo absoluto para cada nota musical existente na seqüência.(ver item 4.3.1);
5. O sistema deverá ser capaz de identificar ou classificar os eventos MIDI para que se possa detectar os status de Program Change (escolha de instrumento), e, desta forma, reconhecer todos os instrumentos existentes na seqüência (ver item 4.4.1.1);
6. Determinar o range de cada instrumento, ou seja, identificar a nota mais grave e a nota mais aguda de cada canal MIDI, para que se possa gerar uma lista ordenada de notas musicais por canal, onde a primeira nota será a mais grave e a última a mais aguda (ver item 4.4.1.1 e Capítulo 5, item 5.3);
7. Permitir que o cantor possa identificar, a qualquer momento, sua tessitura vocal e reconhecer com precisão quais as notas musicais limites de tal tessitura. Para tanto, o mesmo apenas deverá utilizar a ferramenta de transposição para ouvir os ranges de classificação vocal existentes (disponibilizado na ferramenta), identificando os limites que sua voz consiga reproduzir (ver item 4.4.1.1 – figuras 4.13, 4.14, 4.15 e Capítulo 5, item 5.1.2.1);
8. Definida a tessitura do cantor e escolhida a melodia de qual canal o mesmo deseja cantar, o sistema deverá ser capaz de transpor **automaticamente** todos os canais MIDI de forma a enquadrar a melodia escolhida (o canal MIDI) na tessitura (alcance, range vocal de conforto) do cantor. Identificado o range do

canal MIDI escolhido para cantar, bem como o range do cantor, o sistema poderá calcular quantos semitons deverá transpor a melodia para centralizar seu range no range do cantor, e, também, para manter todos os instrumentos da seqüência na mesma tonalidade da melodia. Deverá transpor todos os demais instrumentos (canais MIDI), menos a bateria, com o mesmo valor de semitons (ver item 4.4.1.1 – figuras 4.11, 4.12 e Capítulo 5, item 5.3 a 5.6);

9. Realizada a transposição, o sistema deverá ser capaz de gravar a nova seqüência de acordo com o nome de arquivo definido pelo cantor (ver item 4.4.1.1 e Capítulo 5, item 5.8).

Estes nove pontos básicos são os mínimos necessários para que se possa efetivar os objetivos propostos inicialmente nesta dissertação.

A seguir, serão analisadas as dificuldades e soluções inferidas para que se possa implementar um programa capaz de realizar as tarefas citadas.

### **4.3.1 Abrir um arquivo MIDI, separar os canais e identificar os eventos musicais**

A princípio pode parecer ser esta uma operação simples, já que foi afirmado que em um arquivo MIDI todas as informações necessárias para se registrar uma música estão no mesmo explicitamente registrados. Realmente esta afirmação é verdadeira, mas, devido à simplicidade da estrutura da máquina MIDI (ver capítulo 2), as informações musicais contidas na mesma devem passar por filtros de análise para reconstituir as formas musicais necessárias para um sistema de análise de range instrumental, bem como para uma futura transposição [35].

Um outro problema relevante é a operação inversa, ou seja, após a separação e identificação dos eventos e realizada a transposição, reconstituir o arquivo MIDI, sem perder nenhuma informação pré-existente no mesmo, tais como direitos autorais, letra da música, nome da música, formato, etc., é uma tarefa tão ou mais complexa do que a primeira.

A figura 4.2 mostra uma partitura musical contendo quatro *tracks* de música, cada *track* com um canal MIDI (Instrumento) específico. Logo a seguir é apresentado, em código hexadecimal, o arquivo MIDI equivalente em formato 1 (cada canal MIDI é colocado em um *track* específico).



*Figura 4.2 - Partitura da música a ser analisada*

No **primeiro Track da partitura**<sup>3</sup> tem-se um compasso musical contendo:

- O compasso inicia com a nota C3 tocando com duração de uma mínima (duas semínimas);
- Logo após a nota G3 é ativada com duração de uma semínima;
- Depois a nota E3 é ativada com duração de uma semínima;
- O metrônomo<sup>4</sup> da música = 100 bpm.

No **segundo Track da partitura** tem-se um compasso musical contendo:

- O compasso inicia com uma pausa de uma semínima;
- Logo após tem-se as notas D3 e F3 tocadas simultaneamente com duração de uma semínima;
- Depois a nota E3 é ativada com duração de uma semínima;
- Depois a nota G3 é ativada com duração de uma semínima.

<sup>3</sup> O fato de um *track* na partitura ter uma numeração, não significa que o *track* MIDI terá a mesma numeração equivalente. Em MIDI, os canais e *tracks* iniciam pelo índice 0.

<sup>4</sup> Metrônomo – determina o número de uma figura musical especificada por minuto. O mesmo dita o pulso da música. Em MIDI, a figura musical adotada em todos os casos é a semínima.

No terceiro *Track* da partitura tem-se um compasso musical contendo:

- O compasso inicia com a nota musical G3 sendo ativada com a duração de uma semínima;
- Logo após a nota F3 é ativada com duração de uma semínima;
- Depois a nota A3 é ativada com duração de uma semínima;
- Depois a nota C3 é ativada com duração de uma semínima.

No quarto *Track* da partitura tem-se um compasso musical contendo

- O compasso inicia com a nota musical A2 sendo ativada com a duração de uma semínima;
- Logo após a nota D3 é ativada com duração de uma semínima;
- Logo após tem-se as notas E3 e G3 tocadas simultaneamente com duração de uma semínima;
- Logo após tem-se as notas C3, E3 e G3 tocadas simultaneamente com duração de uma semínima.

A seguir, apresenta-se o arquivo MIDI equivalente à partitura da Figura 4.2.

```
4D 54 68 64 00 00 06 00 01 00 05 00 A8 4D 54 72 6B 00 00 00 59 00 FF 03 08 75 6E 74 69 74 6C 65 64
00 FF 02 20 43 6F 70 79 72 69 67 68 74 20 A9 20 32 30 30 36 20 62 79 20 6C 75 63 69 61 6E 6F 20 6C 69
6D 61 00 FF 01 0C 6C 75 63 69 61 6E 6F 20 6C 69 6D 61 00 FF 58 04 04 02 18 08 00 FF 59 02 00 00 00
FF 51 03 09 27 C0 00 FF 2F 00 4D 54 72 6B 00 00 00 28 00 FF 03 07 54 72 61 63 6B 20 31 00 C0 00 00 90
3C 64 82 50 3C 00 00 43 64 81 28 43 00 00 40 64 81 28 40 00 00 FF 2F 00 4D 54 72 6B 00 00 00 2F 00 FF
03 07 54 72 61 63 6B 20 32 00 C1 18 81 28 91 41 64 00 3E 64 81 28 3E 00 00 41 00 00 40 64 81 28 40 00 00
43 64 81 28 43 00 00 FF 2F 00 4D 54 72 6B 00 00 00 2F 00 FF 03 07 54 72 61 63 6B 20 33 00 C3 0C 00 93
43 64 81 28 43 00 00 41 64 81 28 41 00 00 45 64 81 28 45 00 00 3C 64 81 28 3C 00 00 FF 2F 00 4D 54 72
6B 00 00 00 41 00 FF 03 07 54 72 61 63 6B 20 34 00 C6 34 00 96 39 64 81 28 39 00 00 3E 64 81 28 3E 00
00 40 64 00 43 64 81 28 43 00 00 40 00 00 43 64 00 40 64 00 3C 64 81 28 3C 00 00 40 00 00 43 00 00 FF
2F 00
```

Um sistema que analise o código MIDI registrado neste arquivo deverá obter os mesmos resultados da partitura grafada e detalhada anteriormente. Para tanto, a seguir será mostrado com detalhes como fazer a leitura e análise correta de um arquivo MIDI formato 1.

### 4.3.1.1 Identificando o cabeçalho principal do arquivo MIDI

O Cabeçalho principal de um arquivo MIDI possui um número fixo de bytes. No *track* principal, diferente dos demais *tracks* de música, utiliza-se os 8 bits<sup>5</sup> de cada byte.

O número de Bytes do cabeçalho do *track* principal<sup>6</sup> é 14 Bytes.

Assim, o cabeçalho deste arquivo é mostrado a seguir:

**4D 54 68 64 00 00 00 06 00 01 00 05 00 A8**

Onde:

<b>4D 54 68 64</b>	Caracteres M (4D) T (54) h (68) d (64)
<b>00 00 00 06</b>	Número de Bytes que faltam para completar o <i>track</i>
<b>00 01</b>	Formato do arquivo = formato 1
<b>00 05</b>	Número de <i>tracks</i> = 5
<b>00 A8</b>	Valor da ppq, do tempo relativo de uma semínima = $A8_H = 168_{10}$

**Tabela 4.1** – Cabeçalho de um arquivo MIDI

O sistema projetado deve extrair deste cabeçalho as informações sobre o tipo de formato. Cada formato possui uma sintaxe diferente, ou seja, no formato zero tem-se um *track* musical (MTrk) para cada canal MIDI da seqüência, já no formato 1 se tem apenas um *track* musical para todos os canais.

Assim, a forma de armazenar as informações nestes dois tipos de formatos difere estruturalmente uma da outra.

Apenas o *track* principal MThd é sintaticamente igual em todos os formatos, de onde um programa de análise poderá extrair o valor da contagem de pulsos por semínima (ppq), formato e número de *tracks* musicais MTrk.

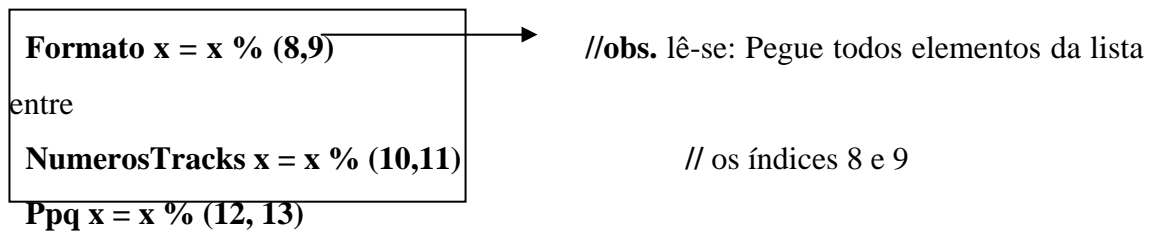
Através do valor da ppq lida pode-se inferir o valor dos tempos de cada nota musical grafada no arquivo MIDI, partindo de Bytes codificados no formato de deltaTimes.

As funções básicas, em CLEAN, que fazem a leitura e identificação da ppq, formato e número de *tracks* é:

<sup>5</sup> Nos *tracks* de música utilizam-se apenas os 7 bits menos significativos de cada Byte para dados, o oitavo é um sinalizador que informa se o Byte é um byte de status (1) ou um byte de dados (0).

<sup>6</sup> O *track* principal, o primeiro do arquivo MIDI, é denominado de MThd – MasterTrack head.





Onde:

- $x$  é uma lista contendo o arquivo binário MIDI lido e convertido para uma lista.
- $\%$  é uma função de CLEAN que pega elementos de uma lista entre os índices entre parênteses (inclusive)

De posse do conhecimento do número de *tracks* que o arquivo MIDI possui, o sistema, o programa em CLEAN, pode então separá-los. Para tanto, existem duas maneiras possíveis de se separar tais *tracks*:

1. No cabeçalho de cada *track* musical (denominado por MTrk), assim como no MThd, existe a informação de quantos bytes o mesmo possui. Desta forma, é só ler esta quantidade e extrair do arquivo *track* por *track*.
2. A outra opção é baseada no fato de que cada *track* MTrk possui uma mensagem de fim de arquivo contendo três bytes (FF 2F 00). Assim, pode-se efetuar a leitura do arquivo até que esta mensagem seja encontrada, e, desta forma, separar todos os *tracks* do arquivo.

A primeira opção é mais simples de ser implementada, principalmente em paradigma funcional.

Assim, o primeiro procedimento é eliminar da lista o *track* MThd já analisado, liberando o resto do arquivo para análise.

A função principal em CLEAN que faz isto é:

<b>eliminaMThd</b> $x = \text{drop } 14 \ x$
--

Onde:

- $x$  é uma lista contendo o arquivo binário MIDI lido e convertido para uma lista.

- **drop** é uma função do CLEAN que elimina um número especificado de elementos de uma lista qualquer.

A informação de quantos bytes um *track* possui está registrada nos bytes de índice 4 a 7 (quatro bytes) da lista, ou seja, quatro bytes logo após o nome MTrk (em ASCII<sup>7</sup>) que inicia todos os *tracks* musicais. Assim, um MTrk possui o número de Bytes definido por 4 bytes do cabeçalho mais 8 bytes (4 do MTrk e mais 4 da contagem).

A função principal em CLEAN, e suas auxiliares diretas, que devolve um *track* **MTrk** da lista de *tracks* é:

```
PegaTrack x = take (contagem +8) x
```

```
where
```

```
contagem = transforma2BemCont qBytes
```

```
qBytes = x % (4,7)
```

```
transforma2BemCont n:[y] = y
```

```
transforma2BemCont n:[z,y] = z*256 + y
```

```
transforma2BemCont n:[w,z,y] = w* 256*256 + z*256 + y
```

```
transforma2BemCont n:[t,w,z,y] = t* 256*256 * 256 + w* 256*256 + z*256 + y
```

Onde:

- **qByte = x % (4,7)** é responsável por pegar os 4 bytes contendo o valor da quantidade de bytes restantes do MTrk corrente
- **contagem** é responsável por aplicar uma função (**transforma2BemCont qByte**) que converte n Bytes (de um a quatro bytes) em um valor inteiro.
- **contagem + 8** = número de bytes do *track* MTrk corrente
- **take contagem x** pega da lista x (o arquivo MIDI sem o *track* MThd), um número de bytes igual ao valor de **contagem** (o byte corrente)

A cada *track* lido, elimina-se o mesmo da lista e passa-se a analisar e separar um novo *track* do arquivo MIDI restante, utilizando a mesma função descrita anteriormente. O

<sup>7</sup> ASCII = American Standard Code for Information Interchange. Codificação de símbolos, caracteres, dígitos, e outras informações utilizando 127 símbolos ou 256 símbolos (ASCII entendido). Ver Anexo 1.

processo se repete até a lista do arquivo MIDI ficar vazia. Quando isto ocorre, a separação dos *tracks* estará pronta.

Este algoritmo só é utilizado para separar *tracks* de formato 1, onde cada *track* (trilha) já vem separado no arquivo SMF.

No caso de formato 0, o algoritmo é bem mais complexo, já que no mesmo só existe um *track* musical contendo todos os canais misturados.

O formato 0 é montado com os eventos MIDI sendo inseridos no arquivo SMF na seqüência exata com que as notas e eventos musicais vão ocorrendo.

Como o raciocínio da implementação de leitura dos dois formatos é semelhante, o mesmo não será apresentado neste documento para não torná-lo mais extenso ainda.

A função que elimina um *track* após ser lido é semelhante à função de leitura do mesmo, só que, ao invés de se pegar um número *x* de bytes contendo o *track* MTrk, deve-se eliminá-lo da lista.

<p><b>EliminaTrack <i>x</i> = drop (contagem +8) <i>x</i></b> <b>where</b> <b>contagem = transforma2BemCont <i>qBytes</i></b> <b><i>qBytes</i> = <i>x</i> % (4,7)</b></p>
---

Onde:

- **drop** é uma função do CLEAN que elimina um número especificado de elementos de uma lista qualquer.

O formato 1 possui uma particularidade de se ter um *track* MTrk a mais do que o número de *tracks* musicais, o qual é responsável por especificar alguns parâmetros musicais necessários à execução da música ou para grafia em partitura, tais como:

- valor do metrônomo
- tonalidade
- nome da música
- direitos autorais
- armadura de clave
- outros

Este *track* é sempre o primeiro após o *track* principal MThd. A leitura deste *track* é feita da mesma forma com que se procedeu na leitura dos demais *tracks* MTrk.

#### 4.3.1.2 Separando os *tracks* do Arquivo MIDI de exemplo

##### Track principal MThd

4D 54 68 64 00 00 00 06 00 01 00 05 00 A8

Número de bytes  
faltantes do *track*

##### Primeiro Track MTrk (MetaEventos)

4D 54 72 6B 00 00 00 59 00 FF 03 08 75 6E 74 69 74 6C 65 64 00 FF 02  
20 43 6F 70 79 72 69 67 68 74 20 A9 20 32 30 30 36 20 62 79 20 6C 75 63 69  
61 6E 6F 20 6C 69 6D 61 00 FF 01 0C 6C 75 63 69 61 6E 6F 20 6C 69 6D 61  
00 FF 58 04 04 02 18 08 00 FF 59 02 00 00 00 FF 51 03 09 27 C0 00 FF 2F 00

Número de bytes faltantes  
do *track* =  $59_H = 89_{10}$

Fim de *track*

##### Segundo Track MTrk

4D 54 72 6B 00 00 00 28 00 FF 03 07 54 72 61 63 6B 20 31 00 C0 00 00 90  
3C 64 82 50 3C 00 00 43 64 81 28 43 00 00 40 64 81 28 40 00 00 FF 2F 00

Número de bytes  
faltantes do *track* =  $28_H$

Fim de *track*

### Terceiro Track

4D 54 72 6B 00 00 00 2F 00 FF 03 07 54 72 61 63 6B 20 32 00 C1 18 81  
 28 91 41 64 00 3E 64 81 28 3E 00 00 41 00 00 40 64 81 28 40 00 00 43 64 81  
 28 43 00 00 FF 2F 00

Número de bytes faltantes do track = 2F<sub>H</sub>

Fim de track

### Quarto Track

4D 54 72 6B 00 00 00 2F 00 FF 03 07 54 72 61 63 6B 20 33 00 C3 0C 00  
 93 43 64 81 28 43 00 00 41 64 81 28 41 00 00 45 64 81 28 45 00 00 3C 64 81  
 28 3C 00 00 FF 2F 00

Número de bytes faltantes do track = 2F<sub>H</sub> = 47<sub>10</sub>

Fim de track

### Quinto Track

4D 54 72 6B 00 00 00 41 00 FF 03 07 54 72 61 63 6B 20 34 00 C6 34 00 96  
 39 64 81 28 39 00 00 3E 64 81 28 3E 00 00 40 64 00 43 64 81 28 43 00 00 40  
 00 00 43 64 00 40 64 00 3C 64 81 28 3C 00 00 40 00 00 43 00 00 FF 2F 00

Número de bytes faltantes do track = 41<sub>H</sub> = 65<sub>10</sub>

Fim de track

De posse dos *tracks* separados, cabe ao sistema agora separar e identificar os eventos existentes em cada um.

A análise dos *tracks* musicais é igual em todos os *tracks* MTrk, mesmo no primeiro que traz apenas mensagem contendo metaEventos, os quais são eventos que contêm informações estruturais da música.

### 4.3.1.3 Analisando o *track* de metaEventos ( o *track* 1)

Os metaEventos são musicalmente bem descritos por MACHADO [19] em sua dissertação de mestrado, e formalizados computacionalmente por LOPES [22] também em sua dissertação de mestrado. Desta forma, para não repetir conceitos já suficientemente registrados nesta mesma instituição, será passado direto a implementação de como proceder tais análises.

A máquina MIDI, conforme mostrado no capítulo 2, é uma máquina que recebe um valor de contagem (denominada de *deltaTime*), e, logo a seguir, uma mensagem contendo uma instrução a ser executada por ela.

Esta contagem é precedida, portanto, de um *deltaTime* que pode ter de 1 a 4 Bytes.

O problema, portanto, na separação das mensagens MIDI, é determinar quando começa e quando finaliza um *deltaTime*, e, logo após, de se determinar qual é a mensagem MIDI contida no arquivo e quantos bytes de dados cada uma possui (este valor pode variar desde 0 Bytes até 2139062271 Bytes).

Assim, elimina-se de cada *track* musical 8 bytes correspondendo aos 4 bytes contendo a palavra MTrk e mais 4 bytes contendo a contagem de quantos bytes o *track* ainda possui após o mesmo.

A função que faz isto é:

**Elimina8Bytes x = drop 8 x**

Análise e identificação dos MetaEventos do primeiro *Track*:

4D 54 72 6B

✂

00 00 00 59

<b>00</b>	<b>DeltaTime = 0</b> -> o evento seguinte deve ser executado instantaneamente, sem nenhuma espera.
<b>FF 03</b>	<b>MetaEvento de Título</b>
<b>08</b>	O meta evento possui 8 bytes
75 6E 74 69 74 6C 65 64	<b>untitled -</b> <b>u=75,n=6E,t=74,i=69,t=64,l=6C,e=65,d=64</b>
<b>00</b>	<b>DeltaTime = 0</b>
<b>FF 02</b>	<b>MetaEvento de Direito Autoral</b>
<b>20</b>	O meta evento possui 32 bytes (20 <sub>H</sub> )
43 6F 70 79 72 69 67 6 8 74 20 A9 20 32 30 30 36 20 62 79 20 6C 75 63 69 61 6E 6F 20 6C 69 6D 61	<b>Copyright .....</b>
<b>00</b>	<b>DeltaTime = 0</b>
<b>FF 01</b>	<b>MetaEvento de Texto</b>
<b>0C</b>	O meta evento possui 12 Bytes (0C <sub>H</sub> )
6C 75 63 69 61 6E 6F 20 6C 69 6D 61	<b>luciano lima. l=6C,u=75, ...</b>
<b>00</b>	<b>DeltaTime = 0</b>
<b>FF 58</b>	<b>Fórmula de Compasso</b>
<b>04</b>	O meta evento possui 4 bytes
04 02 18 08	<b>Fórmula 4 por 4 (2<sup>2</sup>)</b>
<b>00</b>	<b>DeltaTime = 0</b>
<b>FF 59</b>	<b>Meta Evento de Armadura de Clave</b>
<b>02</b>	O meta evento possui 2 bytes
<b>00 00</b>	<b>Tonalidade de Dó maior</b>
<b>00</b>	<b>Delta Time = 0</b>
<b>FF 51</b>	<b>Meta Evento de Tempo</b>
<b>03</b>	O meta evento possui 3 bytes

09 27 C0	<b>Metrônomo = 100 bpm<sup>8</sup></b> = Este meta evento informa o tempo de uma semínima em microssegundos.  Assim, se tem um tempo de 0927C0 <sub>H</sub> = 600.000 microssegundos = 0,6 segundos -> <b>metrônomo</b> = 60s/0,6s = 100 bpm.
00	<b>Delta Time = 0</b>
FF 2F 00	<b>Meta Evento de fim de track</b>  FF 2F com 00 bytes de dados

*Tabela 4.2 – Análise e identificação dos MetaEventos do primeiro Track*

Do resultado desta análise pode-se criar uma lista, mostrada a seguir, contendo o *track*, onde cada elemento da lista é uma lista contendo um evento.

[  
[00,FF,0,08,75,6E,74,69,74,6C,65,64],  
[00,FF,02,20,43,6F,70,79,72,69,67,68,74,20,A9,20,32,30,30,36,20,62,79,20,6C,75,63,69,61,6E,6F,20,6C,69,6D,61],  
[00,FF,01,0C,6C,75,63,69,61,6E,6F,20,6C,69,6D,61],  
[00,FF,58,04,04,02,18,08],  
[00,FF,59,02,00,00],  
[00,FF,51,03,09,27,C0],[  
00,FF,2F,00]  
]

Uma função genérica em CLEAN que busca um determinado evento (uma mensagem de MetaEvento, por exemplo) em uma lista MTrk, como a descrita aqui, é mostrada a seguir:

**PegaMetaEventoGen** lt me = [ x \\ x<- lt | (lt!!2) == me]

Onde:

<sup>8</sup> bmp – batidas por minuto



- **!!** é uma função do Clean que pega um elemento de uma lista, conforme o índice fornecido logo após esta função.
- **It!!2** retorna o terceiro elemento da lista, ou seja, o elemento de índice 2.
- **[ x \ x<- It | (It!!2) == me]** retorna todos as listas de mensagens pertencentes a **It** que obedeçam à regra de que o terceiro elemento da lista de mensagem seja igual ao código do metaEvento solicitado (**me**).

Observe que a implementação em CLEAN desta função de busca é extremamente aderente à formalização lógica do problema proposto.

## Regra geral dos metaEventos

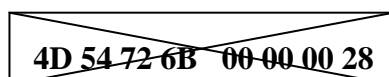
Existem vários outros tipos de metaEventos, todos possuindo a mesma sintaxe, ou seja, inicia por um Byte = FF (255), seguido de um byte que especifica qual é o metaEvento em questão, seguido do número de bytes de dados (segue a mesma regra do deltaTime de 1 a 4 Bytes), seguido dos bytes de dados do metaEvento.

### Analisando os *tracks* de música

O princípio é o mesmo da análise do *track* de metaEventos, o qual é, também, um MTrk.

Assim, após eliminar os 8 (oito) primeiros Bytes do *track*, passa-se à análise do *track* musical, iniciando por identificar o primeiro deltaTime, seguido do primeiro evento e assim sucessivamente até que se atinja o fim de *track* com mensagem de metaEvento FF 2F 00.

A seguir é feita uma análise do *track* 2, canal 0, para exemplificar como o processo é feito.



00	Delta Time = 0, isto significa que o evento seguinte deve ser iniciado imediatamente																
FF 03	Meta Evento de Título																
07	Meta Evento com 7 Bytes																
54 72 61 63 6B 20 31	Track 1, T=54, r=72, .....																
00	Delta Time = 0																
C0 00	Mensagem de dois bytes de mudança de instrumento (C0 -> C = mudança de instrumento, 0 = canal 0) = piano acústico <sup>9</sup> (00)																
00	Delta time = 0																
90 3C 64	Mensagem de três bytes de ativação de nota no canal 0 (9 = ativa nota, 0 = canal 0). A nota a ser ativada é a 3C <sub>H</sub> (60 <sub>10</sub> ) <sup>10</sup> que é a nota C3 (o dó central do piano). O volume da mesma deverá ser 64 <sub>H</sub> (100 <sub>10</sub> )																
82 50	<p>Meta Evento com dois Bytes, já que o primeiro byte possui valor superior a 7F<sub>H</sub> (127<sub>10</sub>). Para calcular o valor real do tempo em ppqs, deve-se transformar o Delta Time em ppqs, ou seja, transformar a informação de tempo com 7 bits por Byte em uma com 8 bits por Byte. Assim, tem-se</p> <table><tr><td>8</td><td>2</td><td>5</td><td>0</td></tr><tr><td>1000</td><td>0010</td><td>0101</td><td>0000</td></tr></table> <p>Para tanto, elimina-se os bits mais significativos e remontam-se os Bytes com os oito bits, ficando</p> <table><tr><td>0</td><td>1</td><td>5</td><td>0</td></tr><tr><td>0000</td><td>0001</td><td>0101</td><td>0000</td></tr></table> <p>O valor, portanto, do tempo, é de 0150<sub>H</sub> = 336<sub>10</sub></p> <p>Como uma semínima vale 168, isto significa que o</p>	8	2	5	0	1000	0010	0101	0000	0	1	5	0	0000	0001	0101	0000
8	2	5	0														
1000	0010	0101	0000														
0	1	5	0														
0000	0001	0101	0000														

<sup>9</sup> Ver tabela de código de instrumentos no Anexo 2.

<sup>10</sup> Ver tabela de códigos de notas musicais no Anexo 3.

	tempo deste delta time equivale ao tempo de duas semínimas (uma mínima), ou seja, deve-se esperar o tempo de duas semínimas para executar o próximo evento MIDI.
3C 00	<p>Mensagem sem o Byte de status que define a ação do evento. Um byte de status deve iniciar com o bit mais significativo setado em 1, ou seja, com um byte maior ou igual ao valor 80<sub>H</sub>(128<sub>10</sub>). No caso, o evento começou com um valor igual a 3C, inferior a 80. Quando isto ocorre, adota-se o último status declarado como sendo o status do evento. A isto denomina-se Running Status<sup>11</sup>. No caso, o último status foi 90, ou seja, ativar uma nota no canal zero.</p> <p>Esta mensagem, sem o running status, ficaria assim: 90 3C 00, ou seja, ativar a nota C3 (3C<sub>h</sub>) com o volume igual a 0, ou seja, desativar a nota C3. Ativar nota com volume zero corresponde a se desativar esta nota. O código utilizado para desativar uma nota no canal zero é o código 80, no caso, a mensagem sem running status ficaria: 80 3C 00.</p>
00	Delta Time = 0
43 64	<p>Novamente a mensagem começa com um byte menor que 80<sub>H</sub>. Isto significa que novamente estar-se-á utilizando o running status, ou seja, o último status utilizado, o qual continua sendo o 90<sub>H</sub>. Assim, esta mensagem completa seria: 90 43 64, ou seja, ativar a nota G3 (43<sub>H</sub>) imediatamente com o volume igual a 64 (100<sub>10</sub>).</p>
81 28	Delta Time com dois bytes, já que o primeiro Byte é

<sup>11</sup> O running status foi criado para economizar espaço de memória, o que era um fator importante nos anos 80.

	<p>maior que 80<sub>H</sub> e o segundo menor. Assim, tem-se</p> <table><tr><td>8</td><td>1</td><td>2</td><td>8</td></tr><tr><td>1000</td><td>0001</td><td>0010</td><td>1000</td></tr></table> <p>Para tanto, elimina-se os bits mais significativos e remontam-se os Bytes com os oito bits, ficando</p> <table><tr><td>0</td><td>0</td><td>A</td><td>8</td></tr><tr><td>0000</td><td>0000</td><td>1010</td><td>1000</td></tr></table> <p>O valor, portanto, do tempo, é de A8<sub>H</sub> = 168<sub>10</sub>, cujo valor é o valor de uma ppq, ou seja, do tempo de uma semínima.</p>	8	1	2	8	1000	0001	0010	1000	0	0	A	8	0000	0000	1010	1000
8	1	2	8														
1000	0001	0010	1000														
0	0	A	8														
0000	0000	1010	1000														
43 00	Novamente uma mensagem com running status, no caso o 90 <sub>H</sub> , gerando uma mensagem igual a 90 43 00, que significa que se deve desativar a nota musical 43 <sub>H</sub> (G3).																
00	Delta time = 0																
40 64	Mensagem com running status, novamente o 90 <sub>H</sub> , indicando que se deve ativar a nota 40 <sub>H</sub> (E3) com o volume 64 <sub>H</sub> (100 <sub>10</sub> ).																
81 28	<p>Delta Time com dois bytes, já que o primeiro Byte é maior que 80<sub>H</sub> e o segundo menor. Assim, tem-se</p> <table><tr><td>8</td><td>1</td><td>2</td><td>8</td></tr><tr><td>1000</td><td>0001</td><td>0010</td><td>1000</td></tr></table> <p>Para tanto, elimina-se os bits mais significativos e remontam-se os Bytes com os oito bits, ficando</p> <table><tr><td>0</td><td>0</td><td>A</td><td>8</td></tr><tr><td>0000</td><td>0000</td><td>1010</td><td>1000</td></tr></table> <p>O valor, portanto, do tempo, é de A8<sub>H</sub> = 168<sub>10</sub>, cujo valor é o valor de uma ppq, ou seja, do tempo de uma semínima.</p>	8	1	2	8	1000	0001	0010	1000	0	0	A	8	0000	0000	1010	1000
8	1	2	8														
1000	0001	0010	1000														
0	0	A	8														
0000	0000	1010	1000														
40 00	Mensagem com running status = 90 <sub>H</sub> , indicando que se deve ativar a nota E3 (40 <sub>H</sub> ) com volume zero, ou																

	seja, desativar tal nota.
<b>00</b>	<b>Delta Time = 0</b>
<b>FF 2F 00</b>	<b>Meta evento de Fim de <i>Track</i></b>

**Tabela 4.3** – *Análise e identificação dos MetaEventos do segundo Track*

A análise mostra que este *track* informa que se deve iniciar ativando uma nota musical C3 com um tempo de uma mínima (duas semínimas), logo a seguir ativar uma nota musical G3 com o tempo de uma semínima, e, para finalizar, ativar a nota musical E3 durante o tempo de uma semínima, confirmando o que está grafado na partitura exemplo e na análise da mesma feita anteriormente.

A partir da análise das mensagens deste *track*, o sistema pode montar uma lista de lista de eventos MIDI, com os running status eliminados contendo o mesmo, conforme mostrado a seguir:

```
[
  [00,FF,03,07,54,72,61,63,6B,20,31],
  [00,C0,00],
  [00,90,3C,64],
  [8250, 90,3C,00],
  [00, 90,43,64],
  [8128, 90,43,00],
  [00, 90,40,64],
  [8128, 90,40,00],
  [00,FF,2F,00,
]
```

Montadas as listas de eventos, no formato apresentado, onde o DeltaTime é o primeiro elemento e o status é o segundo, fica simples de se implementar funções de análise e reconhecimento das mensagens MIDI (deltaTime + evento). Para exemplificar, a seguir é mostrada a implementação de uma função genérica que retorna uma lista contendo todas as mensagens especificadas no argumento da mesma. A função é **PegarEvento**. O primeiro argumento da mesma é a lista **lt** de *tracks* e **ev** é o evento desejado.

**PegarEvento**  $lt\ ev = [x \setminus x \leftarrow lt \mid (lt!!2) == ev]$

Onde:

- **!!** é uma função do Clean que pega um elemento de uma lista, conforme o índice fornecido logo após esta função.
- **lt!!2** retorna o terceiro elemento da lista, ou seja, o elemento de índice 2.
- $[x \setminus x \leftarrow lt \mid (lt!!2) == ev]$  retorna todas as listas de mensagens pertencentes a **lt** que obedecem à regra de que o terceiro elemento da lista de mensagem seja igual ao código do Evento solicitado (**ev**).

Pode-se perceber que realmente as informações estão todas declaradas nos arquivos SMF, mesmo que implicitamente.

Percebe-se, também, um relativo grau de complexidade para “ensinar” o computador a reconhecer, extrair e montar as listas de eventos musicais contidas nos mesmos.

A maior dificuldade, como ficou claro, está em se obter os conhecimentos necessários para que se possa reconhecer todos os tipos de eventos que um arquivo MIDI pode utilizar, para que, posteriormente, funções simples como as que foram mostradas aqui como exemplo, possam realizar as tarefas pretendidas.

#### **4.4 IHM (Interface Homem Máquina) – a interface com o usuário**

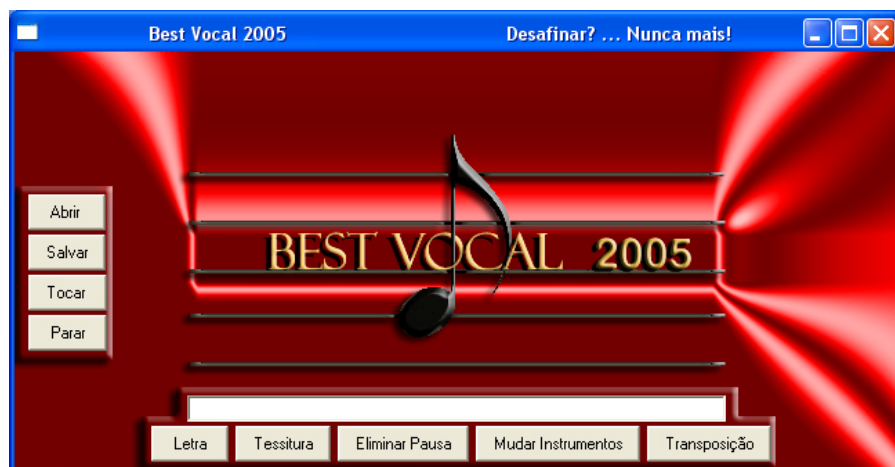
A maioria dos usuários alvo do sistema proposto nesta dissertação são usuários leigos em computação. Os mesmos possuem relativa dificuldade na manipulação de aplicativos complexos, principalmente quando os sistemas possuem um número significativo de menus, submenus, links e outros recursos interativos que as interfaces visuais possuem atualmente.

Realizada uma pesquisa com os usuários alvos, principalmente músicos da noite, verificou-se que os mesmos possuem pouco tempo e interesse no uso do computador, a não ser para jogos e atividades recreativas sociais (msn, orkut e outros afins).

Assim, desta pesquisa conclui-se que criar uma interface com menus e sub-menus fogem do paradigma de soluções e aplicações que tais profissionais utilizam no dia a dia.

Uma interface simples, aderente de ser utilizada, para tais grupos, seria uma interface apenas com poucos botões, cada um com uma função distinta e específica. Tais interfaces não podem possuir elementos que permitam ações diferentes que necessitem do julgamento do usuário, bem como a mesma não deverá possuir campos de texto onde o usuário tenha que escrever ações que o sistema deverá fazer, evitando que erros na grafia venham a travar o sistema, ou, na melhor das hipóteses, não efetuar o que se quer.

Com base nestas informações e pesquisas, prototipou-se várias interfaces, sendo que a que mais agradou e foi de encontro aos objetivos traçados é a que é apresentada na figura 4.3, onde o sistema implementado foi denominado de BEST VOCAL 2005.

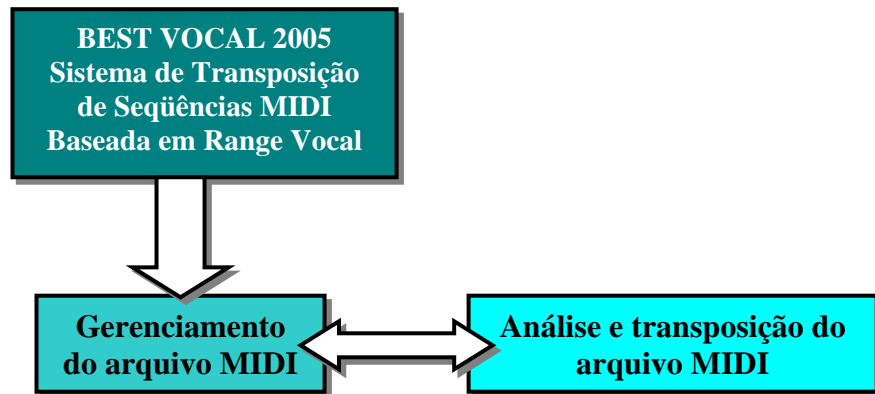


*Figura 4.3 - Interface do Best Vocal 2005*

### **Estrutura geral da interface**

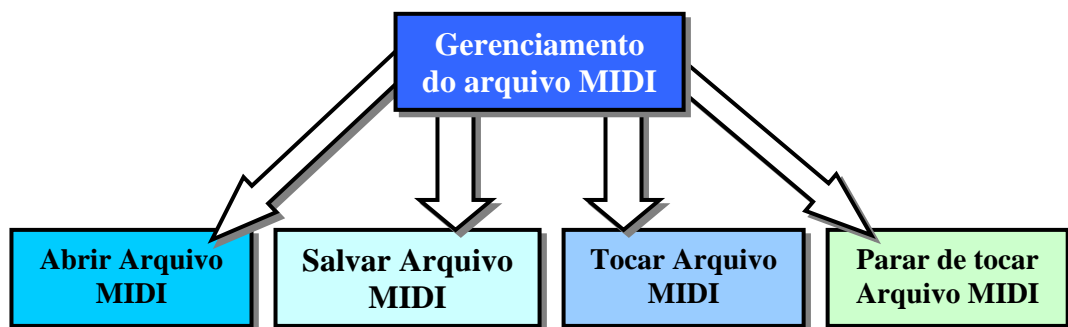
Conforme pode-se ver na figura 4.3, a mesma possui dois conjuntos distintos de botões, cada um com uma funcionabilidade distinta, a saber:

- Gerenciamento de arquivos MIDI formato 0 ou formato 1
- Análise, modificação e transposição dos arquivos MIDI



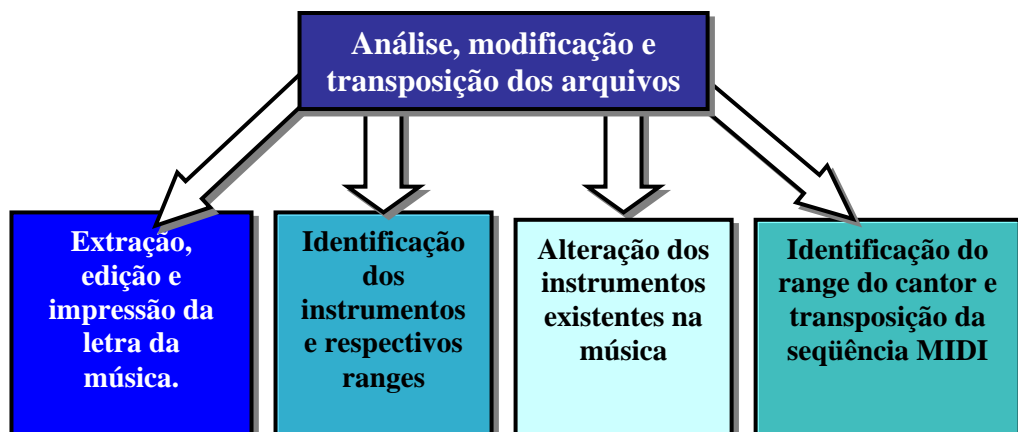
*Diagrama 4.1* – Sistema de transposição de seqüências MIDI baseada em range vocal

A ferramenta de gerenciamento possui quatro potencialidades: Abrir, Salvar, Tocar e Interromper a execução de um arquivo MIDI, seja em formato 0 ou formato 1.



*Diagrama 4.2* – Gerenciamento do arquivo MIDI

A ferramenta de análise e modificação dos arquivos MIDI, possui também quatro potencialidades básicas: letra da música, tessitura dos instrumentos, substituição de instrumentos da música, Identificação de range vocal do cantor/transposição da música.





***Diagrama 4.3 – Análise, modificação e transposição dos arquivos MIDI***

Para implementação da interface de entrada, optou-se por construir uma interface principal **MDI**<sup>12</sup> (**M**ultiple **D**ocument **I**nterace) que permitisse que as janelas de todas as ferramentas ficassem abertas e ativas ao mesmo tempo, já que, como será visto ainda neste capítulo, é imprescindível que as ferramentas e respectivas janelas fiquem abertas ao mesmo tempo para que se possa efetivar uma transposição adequada e de uma forma mais simples e intuitiva.

Na implementação das janelas das ferramentas, optou-se pela criação de interfaces **NDI** (**N**o **D**ocument **I**nterface), as quais são simples de serem criadas e facilitam e flexibilizam a construção e retrabalho no layout (as mesmas se auto-ajustam com o layout dos botões e campos de texto).

A seguir são apresentadas, detalhadamente, as ferramentas desenvolvidas.

## **4.4.1 Descrição da utilização da interface e do sistema de transposição**

### **4.4.1.1 Potencialidades e características**

O sistema desenvolvido permite ao usuário realizar as seguintes tarefas:

#### **1-Abrir um arquivo MIDI, seja em formato 0 ou formato 1**

Para se utilizar o Best Vocal 2005, é necessário, em primeiro lugar, abrir um arquivo MIDI qualquer.

Apesar de ser uma tarefa complexa, o sistema desenvolvido permite que o usuário abra arquivos MIDI de qualquer formato, evitando que o mesmo tenha que se preocupar em

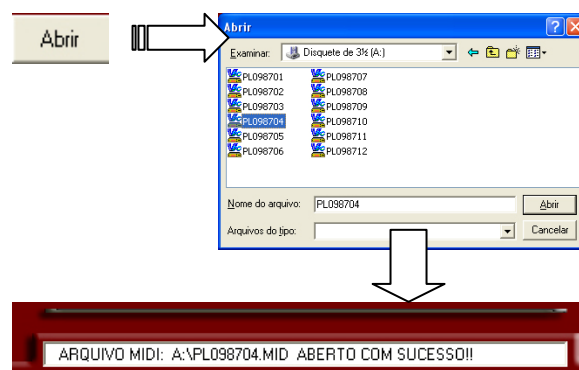
---

<sup>12</sup> A implementação deste tipo de interface, em CLEAN, pode ser visto com detalhes no Anexo 4 e na dissertação deo trabalho de mestrado de MARTINS em seu trabalho de mestrado na FEELT-UFU [36].

qual formato o arquivo foi gerado, simplificando a tarefa dele, e, conforme projeto, evitando que tenha que tomar decisões conflitantes e muitas vezes complexas para ele (já que poucas pessoas compreendem as diferenças e aplicações das implementações dos vários formatos de arquivo MIDI).

Enquanto um arquivo não for aberto, nenhuma das demais ferramentas funcionará.

Para abrir um arquivo MIDI, portanto, basta clicar no botão Abrir. Ao fazer isto, o gerenciador de arquivos é aberto para que o usuário escolha um arquivo. Se o arquivo escolhido for um arquivo MIDI válido, o sistema emite a mensagem mostrada na figura 4.4, caso contrário, o mesmo emite uma mensagem de erro.



*Figura 4.4 - Abrindo um arquivo MIDI qualquer*

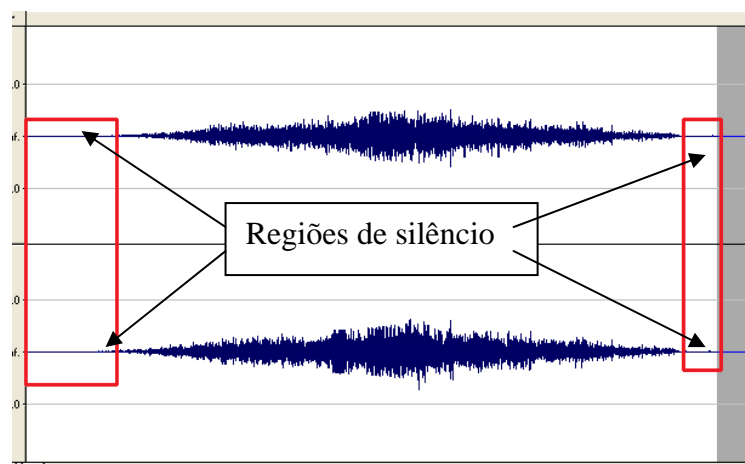
## 2-Tocar ou interromper a execução do arquivo aberto ou do modificado

Para executar um arquivo MIDI, o sistema detecta a placa de som ativa do computador e a utiliza, sem que seja necessário ao usuário configurar a mesma no sistema. Esta é uma característica que atende a estrutura do projeto que é a de simplificar a utilização do sistema. Assim, para tocar um arquivo MIDI, basta, depois de abri-lo, é claro, clicar no botão **Tocar** e para interromper a execução, basta clicar no botão **Parar** (ações bastante intuitivas).

### 3-Eliminar pausas existentes no início do arquivo

Este recurso foi bastante solicitado pelos usuários que testaram o sistema, e, apesar de não ser relevante para o objetivo desta dissertação, ou seja, a transposição automática da seqüência MIDI, é uma ferramenta bastante útil e atrativa para o público alvo. Na realidade, muitos usuários adquiriram o programa especialmente para utilizar esta ferramenta em suas seqüências, já que a mesma não existe nos softwares comerciais existentes no mercado.

Eliminar pausas é uma tarefa tão ou mais complexa do que transpor uma seqüência musical. Eliminar pausas em arquivos do tipo Wave é uma tarefa simples, ou seja, é só retirar a região inicial e final do arquivo onde a amplitude do mesmo estiver abaixo de um determinado valor.



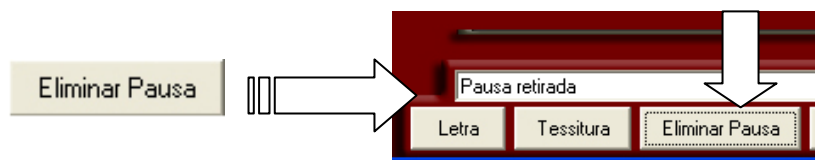
**Figura 4.5** - Arquivo wave com silêncio no início e fim do arquivo

Já em um arquivo MIDI, não existe uma região explícita de silêncio. Um silêncio em um arquivo MIDI é um Delta Time diferente de zero em algum evento existente antes da primeira nota musical ser ativada em alguns dos 16 canais possíveis em uma seqüência MIDI. Uma pausa no fim de um arquivo MIDI também é um Delta Time diferente de zero em algum evento após a última nota musical ser desativada. Desta forma, um programa que se destine a fazer isto tem que analisar todos os eventos de um arquivo MIDI e zerar todos os delta time existentes antes desta primeira nota em um dos canais do arquivo, bem como, também, zerar o Delta Time de todos os eventos que existirem em todos os canais MIDI após a última nota de um destes canais ser desativada.

O paradigma e linguagem de programação escolhida facilitam bastante esta tarefa, já que, para tanto, pode-se utilizar novamente a notação Zermelo-Frankel para definir o domínio (a lista de *tracks* contendo as listas de eventos) e as restrições do mesmo (deltaTimes diferentes de zero antes da primeira nota de qualquer lista de *track* ser ativada), gerando o conjunto solução (uma nova lista de *tracks* com os deltaTimes zerados de todas as listas de eventos antes da primeira nota de qualquer canal ser ativada).

O **Best Vocal**, portanto, examina todos os canais MIDI existentes no arquivo, identifica a pausa existente no início<sup>13</sup> da execução musical de cada um deles e elimina a menor pausa encontrada em todos os canais. Assim, o arquivo salvo, com pausas iniciais removidas, ao ser executado é iniciado imediatamente.

Para utilizar esta ferramenta, basta clicar no botão **Eliminar Pausa**. Quando o sistema termina de realizar a ação, uma mensagem de sucesso é emitida, conforme mostrado na figura 4.6.



**Figura 4.6 - Eliminando Pausas**

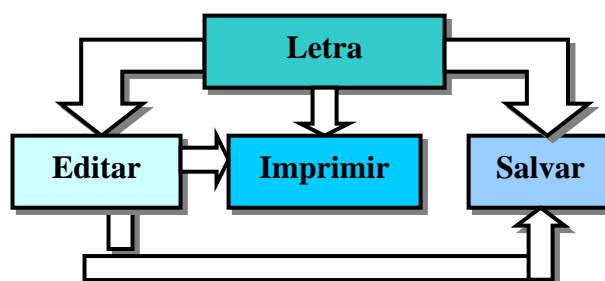
## 4-Visualizar, salvar e imprimir a letra da música contida no arquivo

Esta ferramenta também não é necessária para que se cumpram os objetivos do projeto. A mesma foi implementada, também, pela solicitação dos usuários do programa sob teste.

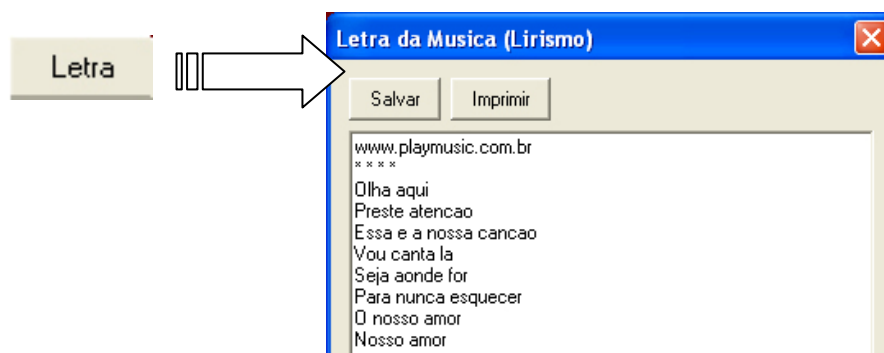
<sup>13</sup> Não se implementou a eliminação de pausas finais, já que as mesmas raramente são encontradas em seqüências profissionais e na maioria da amadoras.

Já que o objetivo do sistema é transpor uma seqüência musical MIDI para uma tonalidade que o cantor consiga cantar sem esforço, bem como, também, já que muitos arquivos MIDI possuem a letra da música (lirismo<sup>14</sup>) incorporada no mesmo, permitir que o usuário possa visualizar e imprimir a letra da música é uma potencialidade aderente ao projeto e que o torna um produto também mais atrativo, principalmente para os cantores que não possuem uma boa memória.

Para ativar esta ferramenta, basta clicar no botão **Letra**. Após alguns segundos, a letra aparece em uma interface NDI. Detalhes do uso desta ferramenta pode ser visto no diagrama a seguir e na figura 4.7.



*Diagrama 4.4 – Letra*





*Figura 4.7 – Janela da ferramenta Letra da Música*

Esta ferramenta (**Letra da Musica**) permite ao usuário editar a letra da música. Como o texto é editável também permite que se acrescente acordes logo acima das palavras e outras características personalizadas que se desejar.

<sup>14</sup> Lirismo – Embora na língua portuguesa esta palavra, traduzida literalmente do inglês Lyrics, dê dupla conotação, vem sendo aceita erroneamente por músicos, principalmente por amadores, como sinônimo de letra de uma música.



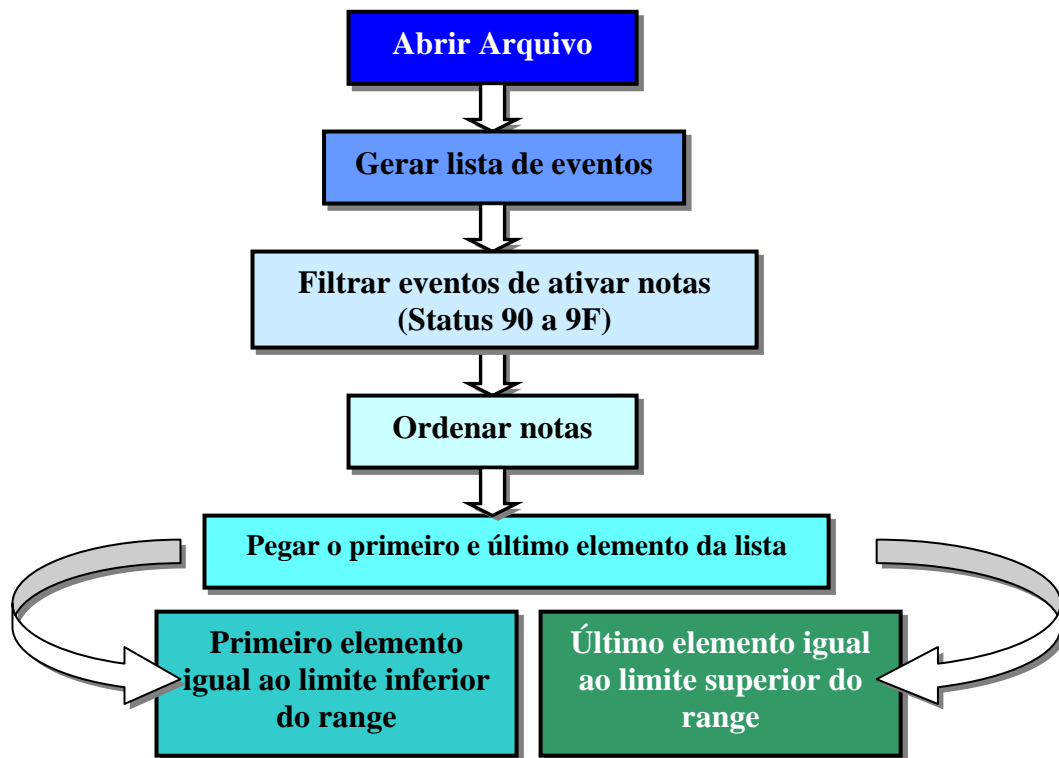
*Figura 4.8 – Acréscimo de acordes acima das palavras*

O profissional pode salvar a letra em arquivo texto ou imprimí-la direto da janela. Para isto, basta que ele clique no botão  ou no botão .

## **5-Visualizar, salvar e imprimir o range (Tessitura) de cada instrumento (canal MIDI) do arquivo**

Ao se analisar um arquivo MIDI para determinar as notas limites utilizadas em cada canal, a definição de range e tessitura se confundem, já que em MIDI são utilizados instrumentos virtuais, sintetizados, e, neste caso, os mesmos sempre terão uma qualidade tão boa quanto se queira ou tão boa quanto for a qualidade do sintetizador.

Para realizar esta tarefa, o programa deve implementar uma função que retorne todas as notas de cada canal e faça uma ordenação das mesmas. Feito isto, ter-se-á uma lista contendo mensagens iniciadas pelo status de 90 a 9F (ativar nota no canal 0 ao canal 15). Para determinar o range de cada canal, basta pegar, na lista gerada por canal, a primeira e última nota musical.



*Diagrama 4.5 – Detecção do range dos canais MIDI*

```

range lt = [primeiro, ultimo]
where
  primeiro = hd (sort [x!!2 \\ x <- lt | isMember x [ "90", "91", "92",
    "93", "94", "95", "96", "97", "98", "99", "9A",
    "9B", "9C", "9D", "9E", "9F"]])
  ultimo = last (sort [x!!2 \\ x <- lt | isMember x [ "90", "91", "92",
    "93", "94", "95", "96", "97", "98", "99", "9A",
    "9B", "9C", "9D", "9E", "9F"]])
  
```

**Obs: x!!2 – Código da nota ativada**

**hd = função que pega o primeiro elemento de uma lista**

**last = função que pega o último elemento de uma lista**

**sort = função que ordena uma lista**

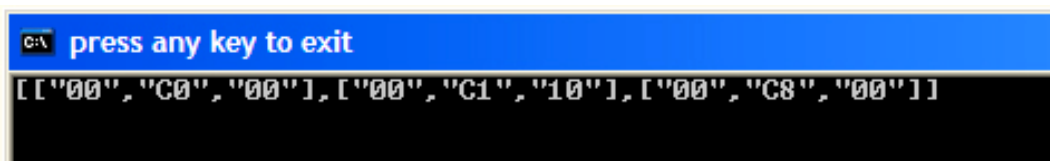
Para determinar o instrumento do canal, basta, novamente, implementar uma função utilizando a notação Zermelo-Frankel que devolva somente as mensagens de escolha ou mudança de instrumento (status de C0 a CF). Feito isto, é só ver na tabela de instrumentos MIDI (Anexo 2) qual instrumento corresponde ao código da mensagem. A função, a seguir, ilustra como gerar, em CLEAN (Anexo 4), uma lista contendo

Para tanto, utiliza-se a função **isMember** do CLEAN para checar se um dado elemento pertence ou não a uma determinada lista.

```
arqMIDI = [
["00", "C0", "00"], ["00", "90", "3C", "64"], ["8250", " 90", "3C", "00"],
["00", " 90", "43", "64"], ["8128", "90", "43", "00"], ["00", "FF", "2F", "00"],
["00", "C1", "10"], ["00", "91", "20", "64"], ["8250", " 91", "20", "00"],
["00", " 91", "23", "64"], ["8128", " 91", "23", "00"], ["00", "FF", "2F", "00"],
["00", "C8", "00"], ["00", "98", "4C", "64"], ["8250", " 98", "4C", "00"],
["00", " 98", "44", "64"], ["8128", " 98", "44", "00"], ["00", "FF", "2F", "00"]
]
```

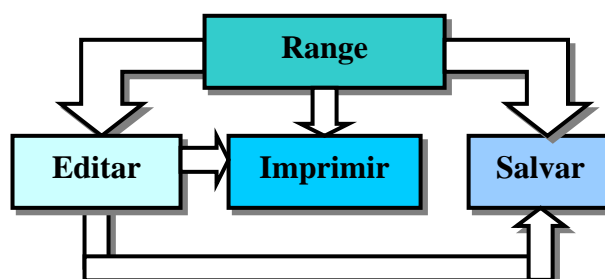
[illegible]

Executando esta função (*pegaEventoInst lt*) no CLEAN, com **lt** = **arqMIDI** (**Start** = **pegaEventoInst arqMIDI**), tem-se:

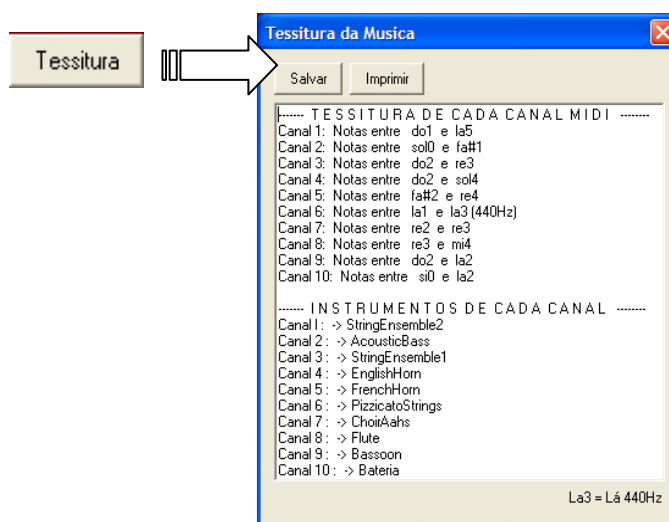


Para ativar a ferramenta de análise de tessitura, basta clicar no botão **Tessitura**. Ao fazer isto, alguns segundos após a janela **NDI** é aberta contendo o range (tessitura) de cada canal MIDI, bem como o nome de cada instrumento dos respectivos canais. Detalhes do uso desta ferramenta pode ser visto no diagrama a seguir e na figura 4.9.





*Diagrama 4.6 – Range*



*Figura 4.9 – Range dos instrumentos*

O conhecimento do range e dos instrumentos de cada canal são de fundamental importância para que o cantor possa identificar qual melodia e de qual canal, deseja cantar.

O usuário, novamente, pode optar por salvar os ranges em arquivo texto ou imprimir direto da janela.

## 6-Visualizar e mudar os instrumentos de cada canal MIDI do arquivo (cada arquivo pode possuir até 16 canais MIDI)

Em alguns programas como o Finale ou mesmo o Sibelius, mudar um instrumento de um determinado canal ou staff, em arquivos já criados, não é uma tarefa tão simples para a maioria dos usuários. No **Best Vocal** esta tarefa é bastante simples, bastando ao músico clicar em um botão, escolher, em um pop-up, o instrumento que quer mudar e

escolher o instrumento que vai substituí-lo. Apesar de não ser uma ferramenta fundamental para o que o programa se propõe: **Transpor músicas para uma região confortável e adequada a um cantor ou determinado instrumento solo**, a ferramenta é bem vinda por, novamente, ser simples de ser utilizada.

Para utilizar esta ferramenta, basta clicar no botão **Mudar Instrumentos**. Ao fazer isto, uma janela com dois pop-ups é aberta para que o usuário escolha o instrumento que quer modificar e o novo instrumento que ficará em seu lugar, conforme seqüência apresentada na figura 4.10.

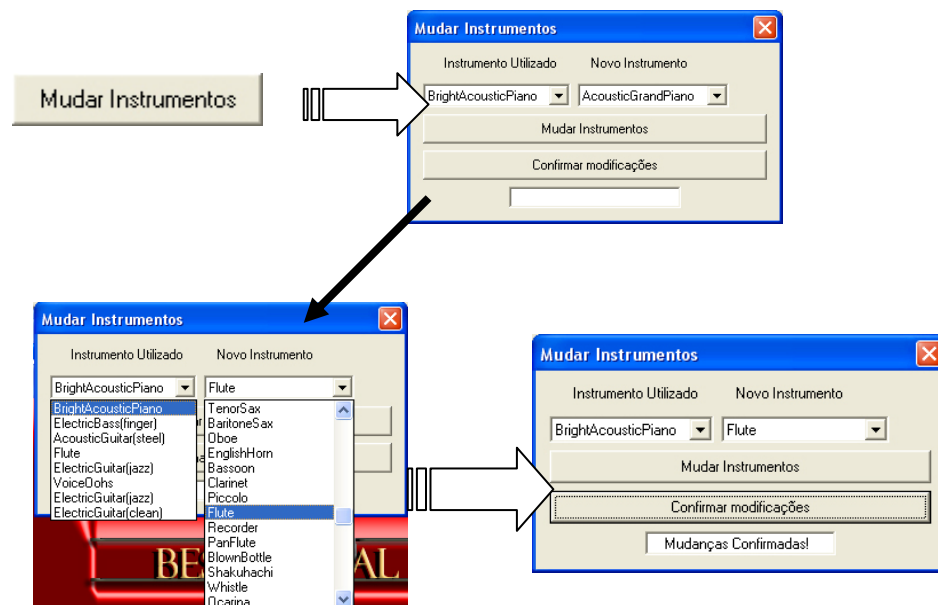


Figura 4.10 - Mudança de instrumentos

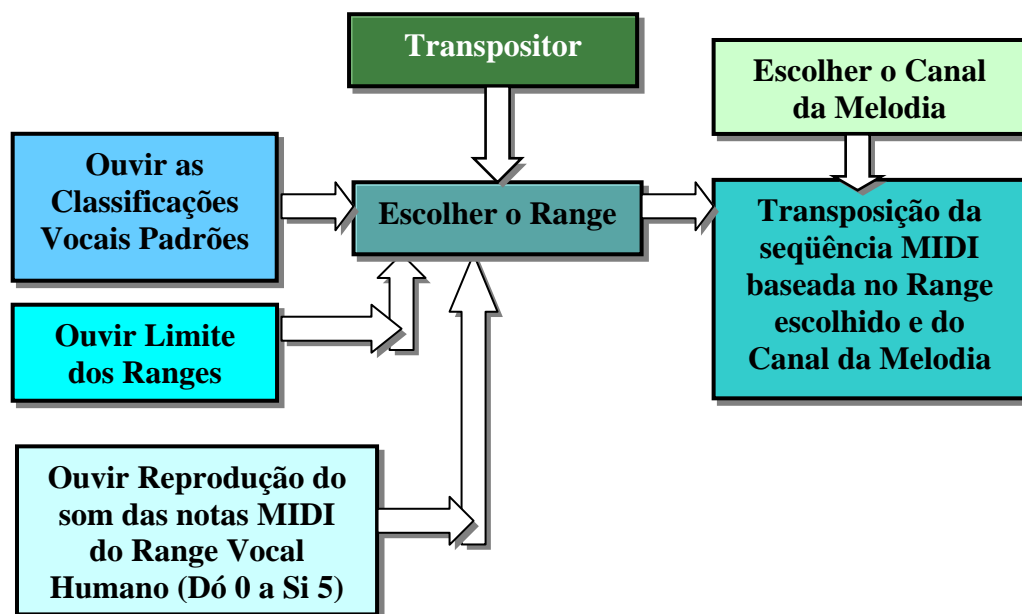
## 7-Transpor a música conforme tessitura vocal do músico ou instrumento que deseja utilizar com a música de playback, de acordo com a melodia de um determinado canal MIDI existente no arquivo

Geralmente um músico faz um teste de classificação vocal com profissionais da área, tais como: fonoaudiólogos e especialistas em canto. Este tipo de teste visa descobrir qual é o melhor range de conforto vocal (tessitura) para um determinado cantor cantar, sem desafinar, falhar ou forçar suas pregas vocais.

O que ocorre é que este range identificado pelo profissional nem sempre é o mesmo em qualquer hora do dia, principalmente se o músico já tiver cantado em excesso, se tiver se embriagado, conversado alto ou estiver, por exemplo, resfriado.

Assim, a ferramenta de transposição do **Best Vocal** tem como objetivo garantir que o músico sempre consiga cantar seu repertório sem forçar, falhar ou se preocupar em desafinar. Nos casos em que nenhuma tonalidade atenda tais requisitos, ter este conhecimento prévio é uma garantia de sucesso para o cantor, o qual poderá modificar seu repertório a tempo, sem comprometer seu nome devido a más interpretações.

Para ativar esta ferramenta, basta clicar no botão **Transposição**. Ao fazer isto, uma janela NDI é aberta para que o profissional possa transpor suas músicas na tonalidade que melhor se encaixe em seu range. O esquema de como a transposição é realizada é mostrado logo a seguir, sendo que a explicação mais detalhada é apresentada após o mesmo.



*Diagrama 4.7 – Transposição*

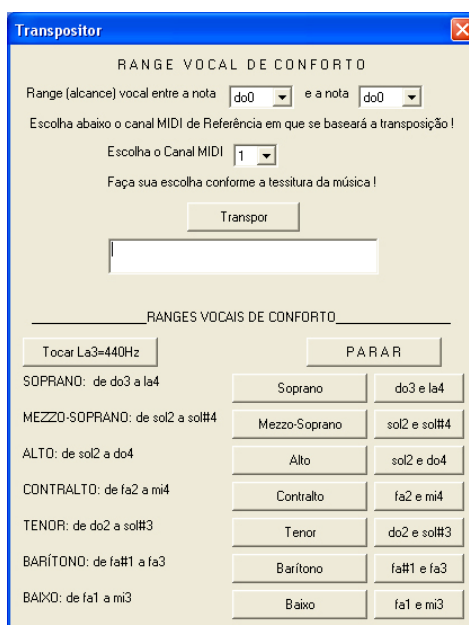
A janela **Transpositor** realiza duas ações distintas, mas destinadas a um mesmo objetivo, ou seja: a Transposição automática de todos os canais de uma seqüência MIDI baseada em um range vocal determinado pelo usuário e de uma melodia existente em um dos canais MIDI. Pode-se dividir a função desta janela, portanto, em duas partes:

- **Avaliação pessoal do range vocal:** Na metade inferior da janela **Transpositor**, mostrada na figura 4.11, tem-se alguns botões destinados para que o cantor

possa identificar um range vocal que julga ser o de melhor conforto e melhor qualidade timbral para ele. Para fazer isto, o sistema disponibiliza algumas classificações vocais (*vocal type*) consagradas na literatura, as quais podem ser ouvidas pelo usuário ao clicar nos respectivos botões.

- **Transposição da seqüência MIDI conforme range vocal declarado pelo usuário:**

Na metade superior da janela **Transpositor**, figura 4.11, a seguir, o profissional, após identificar um ou mais ranges de classificação vocal que julga alcançar com qualidade, pode fazer um ajuste fino, mais preciso, deste range. Este ajuste é feito através de dois pop-ups contendo notas musicais de **Dó 0** a **Si 5**. Nestes pop-ups, quando o usuário escolhe uma nota musical qualquer, o sistema emite o som da mesma, permitindo ao cantor tentar reproduzi-la com qualidade. Uma vez definido o range vocal definitivo (notas limites), pode-se efetuar a transposição da seqüência MIDI, de forma a centralizar o range da melodia que o cantor deseja cantar com o range de conforto escolhido pelo usuário. Para tanto, a janela disponibiliza um outro menu pop-up contendo a numeração dos 16 canais MIDI existentes, de forma que o usuário escolhe o canal onde está a melodia em questão, identificada na janela de **Tessitura da Música** (acionada pelo botão **Tessitura**). Ao clicar no botão **Transpor** desta janela, o sistema transpõe todos os canais MIDI (menos a bateria, canal 10) para o range de conforto do cantor.



**Figura 4.11** - Janela de transposição

A seguir, mostra-se com mais detalhes como utilizar esta ferramenta em conjunto com a ferramenta de tessitura já abordada anteriormente.

Na ferramenta de tessitura o cantor verifica o canal e instrumento da melodia que deseja cantar.

Ao mesmo tempo, o cantor seleciona no pop-up de range vocal a nota mais grave e a nota mais aguda que julga cantar sem esforço e com qualidade.

Feito isto, no pop-up **Canal MIDI** o profissional escolhe o canal da melodia identificada.

Pronto, agora é só clicar no botão **Transpor** e aguardar.

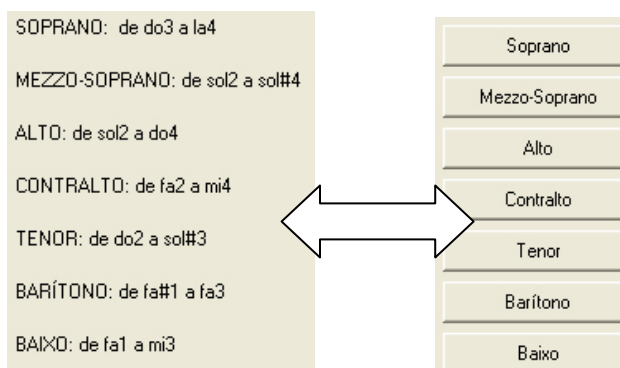
Se a transposição for possível, ou seja, se o range da melodia for menor que o do cantor, o sistema realizará a transposição da seqüência e emitirá uma mensagem de sucesso, caso contrário, emitirá uma mensagem de erro.



**Figura 4.12** - Seqüência de ações para a transposição

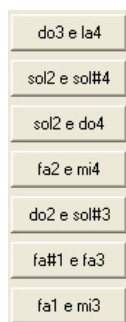
Caso o cantor não conheça sua tessitura, o mesmo pode utilizar as dicas de classificação para determinar os limites de seu range, de seu alcance vocal de conforto. Para tanto existem três possibilidades de dica:

- Ouvir os ranges vocais mais usuais (em uma escala sonora)



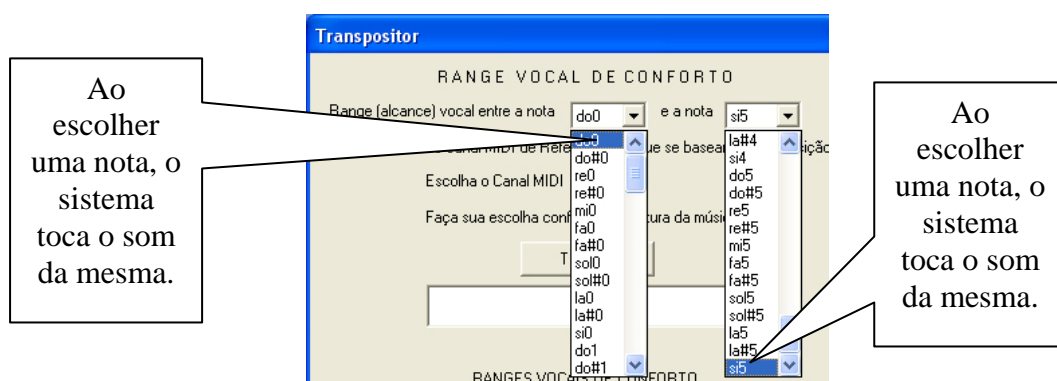
**Figura 4.13** - Classificações padrões

- Ouvir apenas o som das notas musicais limites de cada classificação vocal



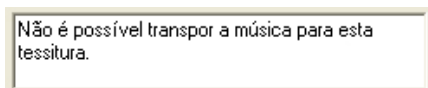
**Figura 4.14** – Botões de limites de range das classificações

- Ouvir o som individual de cada nota musical entre a nota **Dó0** (32,7 Hz) e a nota **Sí5** (1975,5Hz), observando que para o Best Vocal, a nota Lá diapasão é o Lá 3 (440Hz) (adotado por várias escolas brasileiras)



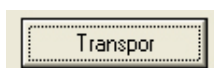
**Figura 4.15** – Pop-ups com limites do range do cantor

Se o range do cantor não for adequado para a melodia escolhida, o **Best Vocal** emite uma mensagem avisando.



**Figura 4.16** – Mensagem de inviabilidade de transposição

Realizadas as configurações, basta ao usuário clicar no botão de transposição da mesma janela para concluir a tarefa.



**Figura 4.17** – Botão de Transposição

## 8-Salvar o arquivo mantendo as configurações do arquivo original acrescido das modificações realizadas

Esta ferramenta é realmente interessante. A princípio simples, mas, vista em profundidade, complexa de ser implementada. Realizadas as modificações desejadas pelas ferramentas anteriormente citadas, o **Best Vocal** salva o arquivo mantendo todas as características do arquivo inicial, tais como: título, copyright, lirismo, metrônomo e todos os eventos e mensagens MIDI exatamente iguais às existentes no arquivo original. Para utilizá-la, basta clicar no botão **Salvar**. Após realizado o salvamento, se o profissional clicar no botão **Tocar**, o mesmo executará a música com as alterações procedidas.



*Figura 4.18 - Salvando um arquivo modificado*

### Observações finais

#### Pontos relevantes

Fazendo uma análise macroscópica, os pontos fortes e atrativos deste sistema estão principalmente:

- na ferramenta de eliminação de pausas iniciais dos arquivos MIDI que tanto incomodam os músicos, principalmente os que utilizam seqüências MIDI como playback;
- na ferramenta de transposição baseada na tessitura vocal do músico, permitindo ao mesmo sempre cantar na região de conforto vocal, evitando desafinar;
- por salvar o arquivo modificado no mesmo formato de abertura, mantendo as demais características, eventos, meta-eventos e mensagens existentes no arquivo original.