Afinador Eletrônico

Robson F. Loschi[1], Reinaldo S. Fortes[1]

[1]FACICS – Universidade Presidente Antonio Carlos (UNIPAC)
Barbacena – MG – Brazil

robby@barbacena.com.br, reifortes@yahoo.com.br

Abstract. *This article describes the musical, mathematical, physical and computational aspects concerning to the implementation of an electronic tuner. It describes, thus, the concepts and uses of fast Fourier transform, a mathematical tool essential for the accomplishment of this task, without it, the implementation of this software and many others, in innumerous knowledge areas would not be possible.*

Resumo. *Este artigo descreve os conceitos musicais, matemáticos, físicos e computacionais relativos à implementação de um afinador eletrônico. Descreve, assim, os conceitos e usos da transformada rápida de Fourier, ferramenta matemática essencial para esta tarefa, sem a qual a implementação deste software e de muitos outros, em inúmeras áreas do conhecimento seria impossível.*

1. Introdução

O afinador é um instrumento utilizado por músicos, geralmente por aqueles que se utilizam de instrumentos de cordas, para adequar o som de seus instrumentos a um padrão pré-estabelecido, permitindo, assim, que possam vir a tocar em conjunto.

Por ser de fundamental importância em qualquer estudo ou execução musical este pode, quase, ser considerado mais importante do que o próprio instrumento e tal fato pode facilmente ser explicado:

Basta imaginar dois violonistas (A e B) preparando-se para uma execução em conjunto. O violonista A utiliza uma freqüência de 426 Hz para afinar a sua corda Lá (A) e afina o restante das cordas com base nesta freqüência. O violonista B, por sua vez, utiliza a freqüência de 442 Hz para afinar a sua corda Lá (A) e afina o restante das cordas com base nesta freqüência. Após treinarem bastante sozinhos, com os violões devidamente afinados nas freqüências citadas, estes se juntam para sua execução em conjunto e o resultado é, obviamente, um completo desastre. Cada nota tocada por eles soa completamente diferente da tocada pelo companheiro.

Assim sendo, o afinador é fundamental mesmo para os possuidores de ouvidos absolutos.

Os dois tipos de afinador utilizados, hoje em dia, são:

- Diapasão;
- Afinador eletrônico.

O objetivo deste artigo é explicar todos os detalhes da implementação de um afinador eletrônico via software e está organizado da seguinte maneira: na seção 1 consta uma breve introdução sobre afinadores em geral, na seção 2 é abordado o processamento de sinais digitais, sua história e seus usos, na seção 3 é explicado como funciona a API Java Sound e como ela foi utilizada para o desenvolvimento deste trabalho, a seção 4 trata do formato de som *Pulse Code Modulation* (PCM), utilizado durante a implementação do software, a seção 5 aborda as transformadas de Fourier e, em especial, a transformada rápida de Fourier, essencial para o desenvolvimento do estudo do processamento de sinais digitais, a seção 6 relata os detalhes de implementação do software *My Tuner*, seu uso e os testes realizados com ele.

1.1 Diapasão

O diapasão (Figura 1), como fonte de freqüência padrão (som padrão), foi inventado por John Shore, trombeteiro real inglês, em 1711, mesma época em que o piano. [Netto 2001]

A freqüência do diapasão foi se alterando através dos tempos. Quando John Shore o inventou, era inicialmente de 419,9 Hz, a mais baixa da história. Atingiu a freqüência mais alta, de 457,2 Hz, em 1879, no diapasão proposto pelo fabricante de pianos Steinway & Sons, de Nova York. A freqüência atual, de 440 Hz, foi ratificada por Norma Internacional em 1987, na cidade de Toronto, Canadá. [Netto 2001]

Trata-se de um arco de metal que quando acertado por algo duro, geralmente por um martelo de borracha, vibra produzindo a freqüência correta para afinação da nota Lá pertencente à quarta oitava da maioria dos instrumentos. Depois de tocado, procura-se afinar a nota Lá no instrumento fazendo-a soar o mais próximo ao som do diapasão.

Figura 1. Diapasão

Hoje, existe um diapasão em forma de apito (Figura 2) com, geralmente, 4 ou 6 embocaduras, sendo que cada uma serve para afinar uma corda de um violão. Ao contrário do diapasão tradicional, o diapasão em forma de apito só pode ser usado para afinar instrumentos de corda com, no máximo, 6 cordas.

Figura 2. Diapasão de apito

1.2. Afinador eletrônico

O afinador eletrônico (Figura 3) é um dispositivo que não produz som algum, mas, ao contrário, recebe o som do instrumento a ser afinado através de um microfone ou uma entrada de linha. Após, informa através de um visor LCD (*Liquid Crystal Display*) ou um mostrador de cores, o quanto a nota tocada está próxima de uma das notas que este reconhece.

A maioria dos afinadores eletrônicos reconhece apenas as notas E, B, G, D, A e E, ou seja, a afinação padrão de um violão de 6 cordas. Com este tipo de afinador eletrônico não é possível afinar instrumentos com mais notas, como um piano, por exemplo.

Existem centenas de marcas e modelos diferentes disponíveis no mercado.

Figura 3. Afinador eletrônico

2. Processamento de sinais digitais

O processamento de sinais digitais (*Digital Signal Processing* - DSP) é a área da Ciência da Computação cujos objetos de estudo são os sinais sonoros (ondas mecânicas ou sinais analógicos) transformados em sinais digitais. Esse processamento é feito mediante complexos cálculos matemáticos.

A origem de tal processamento inicia-se, nos anos 60 com o uso de *mainframes* para a realização de pesados processamentos matemáticos como a Transformada Rápida de Fourier (Fast Fourier Transform - FFT), que permite que o espectro de freqüência de um sinal seja computado rapidamente. Tais técnicas foram muito pouco usadas na época

de sua criação porque os equipamentos compatíveis com estas funções eram encontrados apenas em universidades e centros de pesquisa [Anônimo 2001].

A criação dos microprocessadores, entre o fim dos anos 70 e o início dos anos 80 tornou possível uma aplicação muito maior das técnicas de DSP, entretanto, os microprocessadores de uso geral tais como a família Intel X86 não eram ideais para a quantidade de processamentos numéricos requeridos pelo DSP. Por esta causa, grandes companhias de eletrônicos como a *Texas Instruments*, *Analog Devices* e *Motorola* desenvolveram durante os anos 80 os processadores de sinais digitais, microprocessadores especiais cuja arquitetura fora especialmente criada para atender as necessidades requeridas para o processamento de sinais digitais [Anônimo 2001].

Através da DSP é possível:

- Captar sons de fontes como instrumentos ou microfones;
- alterar sons acrescentando efeitos ou eliminando ruídos, por exemplo;
- reproduzir sons utilizando um computador;
- criar sons digitalmente, sem a presença de músicos ou instrumentos.

A mais fundamental operação realizada através da DSP são as conversões A/D e D/A.

2.1. Conversões A/D e D/A

As ondas sonoras são produzidas por deformações provocadas pela diferença de pressão em um meio elástico qualquer (ar, metais, isolantes, etc), precisando deste meio para se propagar. Desta forma, percebe-se que o som é uma onda mecânica, não se propagando no vácuo. A maioria dos sons acaba sendo obtida através de objetos que estão vibrando, como é o caso do alto-falante. Quando o diafragma contido no alto-falante se movimenta para fora da caixa acústica ele cria uma região de alta pressão, pois comprime o ar que está nas proximidades (da mesma forma, ocorre uma rarefação quando o diafragma se move para dentro da caixa). Quando as variações de pressão chegam aos ouvidos, os tímpanos tentam imitar esta vibração e causam a sensação fisiológica do som [Bosquetti 1996].

Estas ondas, quando captadas por um microfone ou um captador magnético de um instrumento, passam a ser sinais analógicos e estes podem, facilmente, ser transmitidos através de cabos ou fios [Baldwin 2003].

Entretanto, para que estes sons sejam introduzidos e posteriormente trabalhados em um computador, deve ser realizada uma conversão Analógica/Digital (do inglês, Analogic / Digital – A/D). Tal conversão é, em alguns casos, realizada via *hardware* como nos casos do afinador eletrônico. Para os fins deste trabalho, esta conversão será realizada via software como será explicado na sessão 3.

Após o sinal ser criado ou transformado dentro do computador, antes que possa ser transmitido para outros dispositivos (amplificadores, caixas acústicas e etc.) deve ser transformado em sinal analógico através de uma conversão Digital/Analógica (do inglês, Digital / Analogic – D/A). Assim como a conversão A/D, esta também é realizada, em alguns casos, via *hardware*, mas será realizada via *software* para este trabalho.

3. Java Sound API

Segundo a empresa responsável pelo desenvolvimento, a Sun Microsystems, a API Java Sound é uma API de baixo nível para controlar a entrada e saída de áudio e aplicar efeitos a estas. Ela provê controle explícito das capacidades normalmente requeridas para entradas e saídas de áudio, num *framework* extensível e flexível [Meloan 1999].

Esta API não provê GUIs para o tratamento de sons, mas funções para que estas sejam facilmente implementadas.

Com Java Sound é possível tratar dois tipos de sons: Midi e áudio Sampleado. Como este trabalho não utiliza sons midi em nenhum momento estes serão, desde já, descartados.

Áudio sampleado (Figura 4) não é mais do que uma série de dígitos que representam a amplitude ou a intensidade das ondas sonoras e sua manipulação depende de dois pacotes java: javax.sound.sampled e javax.sound.sampled.spi, para este trabalho, apenas o primeiro será utilizado.

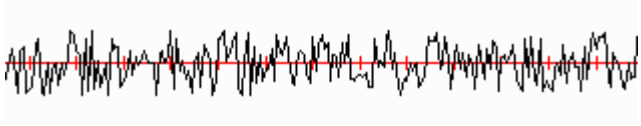

Figura 4. Representação de uma onda de áudio sampleado

Como exemplo de áudio sampleado, pode-se citar a informação contida em um CD de áudio no qual está impressa a representação eletrônica do som criado por um artista.

Para a captura de áudio sampleado, necessita-se, geralmente, de dois passos:

- Uso de um microfone ou captador magnético para captar a pressão das ondas sonoras e transforma-las em sinais elétricos.
- Realizar uma conversão A/D para transformar estes sinais em sinais digitais.

Java Sound, trata os sons de entrada como se entrassem através de um Mixer de áudio (Mesa de som) (Figura 5). Esses mixers, normalmente aceitam diversas fontes de som como entrada, cada uma chamada de linha (*Line*). A API foi construída utilizando os conceitos de *Mixer* e *Line*. [Baldwin 2003]

Figura 5. Mixer

4. Pulse Code Modulation (PCM)

Existem diversos formatos digitais para representar sons, tais como: *Wave* (.wav), *Real Audio* (.ra), *Moving Pictures Experts Group Layer 3* (.mp3). Entretanto, trabalhando-se

com Java Sound, a melhor opção de formato é o *pulse code modulation* (PCM), pois a API já o possui definido em uma de suas classes.

Para entender o funcionamento do formato PCM, basta observar uma onda que represente um som qualquer (Figura 6) [Anônimo 2000]

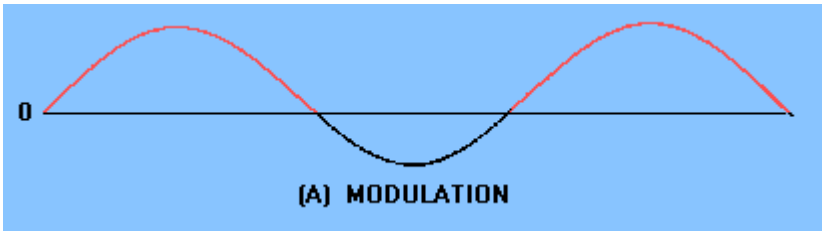

Figura 6. Onda representando um som qualquer

Quando deseja-se representar o som digitalmente num computador, é necessário discretizar esta onda, ou seja, representá-la em função de sua amplitude (eixo y) durante um intervalo de tempo (eixo x) em valores discretos (Figura 7). A forma usada pelo formato PCM consiste em guardar num vetor as amplitudes em determinados intervalos de tempo [Anônimo 2000].

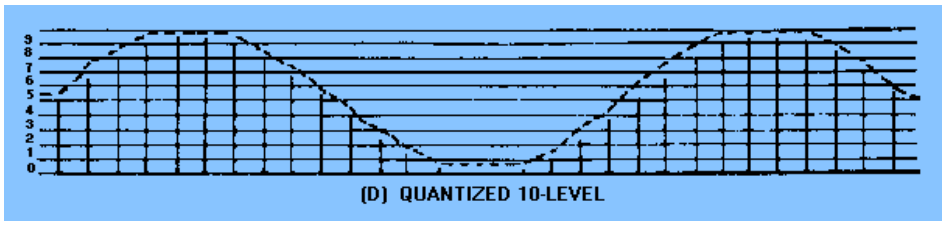

Figura 7. Onda discretizada

Quanto aos possíveis valores da amplitude pode-se, por exemplo, usar 8 bits para cada posição do vetor. O valor da amplitude pode ser então de -127 a 127. Uma característica do formato PCM é que ele realiza um mapeamento linear da intensidade do som para este intervalo [Nagato 2002].

A discretização na dimensão horizontal corresponde a uma taxa de *frames* por segundo(fps). Um frame é uma amostra do som em um determinado instante. Neste caso, foi utilizada uma taxa de 16000 *frames* por segundo. Isto equivale a 16000 posições no vetor de bytes para representar um segundo de som (taxa de *samples* por segundo). Observe-se que um frame pode conter vários canais, cada um correspondendo a um *sample* em um determinado instante. Ou seja, se a gravação for feita em modo estéreo serão utilizados 2 canais. Neste exemplo foi utilizado um canal apenas (mono) para simplificar a representação e o algoritmo de busca de freqüência. Se fossem usados 6 canais um frame teria 6 bytes e um segundo de som teria 16000 * 6 = 96000 bytes. Se, além disso, fossem usados 2 bytes para representar a amplitude haveriam 192000 bytes por segundo (12 bytes por frame) [Nagato 2002].

5. Transformadas de Fourier

Jean Baptiste Joseph Fourier foi um importante matemático e físico francês que viveu entre os séculos XVIII e XIX. Suas principais contribuições matemáticas foram as séries trigonométricas batizadas em sua homenagem como séries de Fourier e suas

transformadas também batizadas em sua homenagem como transformadas de Fourier. [Iezzi 2000]

As transformadas de Fourier são transformadas integrais que expressam funções em termos de funções de base sinusoidal (trigonométrica). Existem diversas variações desta transformada, dependendo do tipo de função a transformar.

Quando algum sinal é capturado por meio de um microfone ou captador magnético, junto com este sinal desejado, obtêm-se, geralmente, alguns sinais indesejados também, chamados genericamente de ruídos.

Esses ruídos podem ser perceptíveis a ouvidos humanos quando se encontram em uma faixa de freqüência entre 20 Hz e 20 kHz, ou imperceptíveis quando fora desta faixa. Porém, para o processamento de sinais digitais qualquer ruído, perceptível ou não atrapalha consideravelmente o processo. No processo de captura de sinais, geralmente os sinais imperceptíveis aparecem muito mais vezes do que os perceptíveis.

É amplamente documentado que a tentativa procurar freqüências dos sinais analógicos capturados e transformados em digitais é tarefa praticamente impossível quando se tenta utilizar os métodos físicos tradicionais como tentar medir a distância entre duas cristas consecutivas (comprimento da onda) e dividindo a taxa de samples por segundo por este valor ou contar o número de cristas e dividi-las pelo tempo em que estas aparecem, por exemplo, justamente por causa dos ruídos [Nagato 2002].

Para resolver este problema, a solução empregada é o uso da transformada rápida de Fourier (*fast Fourier transform* - FFT) [Nagato 2002].

Jean Baptiste Joseph Fourier foi um importante matemático e físico francês que viveu entre os séculos XVIII e XIX. Suas principais contribuições matemáticas foram as séries trigonométricas batizadas em sua homenagem como séries de Fourier e suas transformadas também batizadas em sua homenagem como transformadas de Fourier. [Iezzi 2000]

As transformadas de Fourier são transformadas integrais que expressam funções em termos de funções de base sinusoidal (trigonométrica). Existem diversas variações desta transformada, dependendo do tipo de função a transformar.

5.1. Transformada contínua de Fourier

Geralmente, a denominação "Transformada de Fourier" refere-se à Transformada de Fourier para funções contínuas (Figura 8), que representa qualquer função integrável *f*(*t*) como a soma de exponenciais complexas com freqüência angular ω e amplitude complexa *F*(ω).

$$f(t) = \mathcal{F}^{-1}(F(\omega)) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(\omega)e^{i\omega t}\,d\omega.$$

Figura 8. Transformada contínua de Fourier

5.2. Transformada discreta de Fourier

Para uso em computadores, seja para aplicações científicas ou em processamento digital de sinais, é preciso ter valores x_k discretos. Para isso existe a versão da transformada para funções discretas (Figura 9).

$$x_k = \frac{1}{n}\sum_{j=0}^{n-1} f_j e^{\frac{2\pi i}{n}jk} \qquad k = 0, \ldots, n-1$$

Figura 9. Transformada discreta de Fourier

5.3. Transformada rápida de Fourier

A (FFT) consiste numa implementação rápida da transformada discreta de Fourier, tirando partido da estrutura repetitiva do cálculo. A transformada discreta de Fourier pode-se escrever como o produto de uma matriz por um vetor. A matriz é chamada matriz de transformação e o vetor contém as amostras do sinal a transformar. O sucesso da FFT deve-se, por um lado, ao vasto número de aplicações e por outro, à estrutura particular da transformada discreta de Fourier que faz com que a matriz de transformação possua uma grande redundância nos seus elementos facilitando a sua implementação recursiva. [Jesus 2005]

Para o processamento digital de sinais utiliza-se, normalmente, a transformada rápida ao invés da discreta porque calcula o espectro de um sinal mais rapidamente. O algoritmo da transformada discreta possui uma complexidade O(n^2), enquanto o da transformada rápida possui a complexidade O(n log n).

6. My Tuner

My Tuner (Figura 10) é um software, desenvolvido após os estudos dos tópicos já citados neste artigo, que reproduz as funções de um afinador eletrônico. Pode reconhecer qualquer nota natural e qualquer acidente, ou seja, todas as notas da escala cromática (C, C#, D, D#, E, F, F#, G, G#, A, A#, B) em qualquer oitava de qualquer instrumento musical. Comparado a um afinador eletrônico normal, apresenta a vantagem de poder afinar qualquer instrumento da forma que se desejar, não ficando limitado apenas a 6 notas.

Figura 10. My Tuner

6.1. Implementação

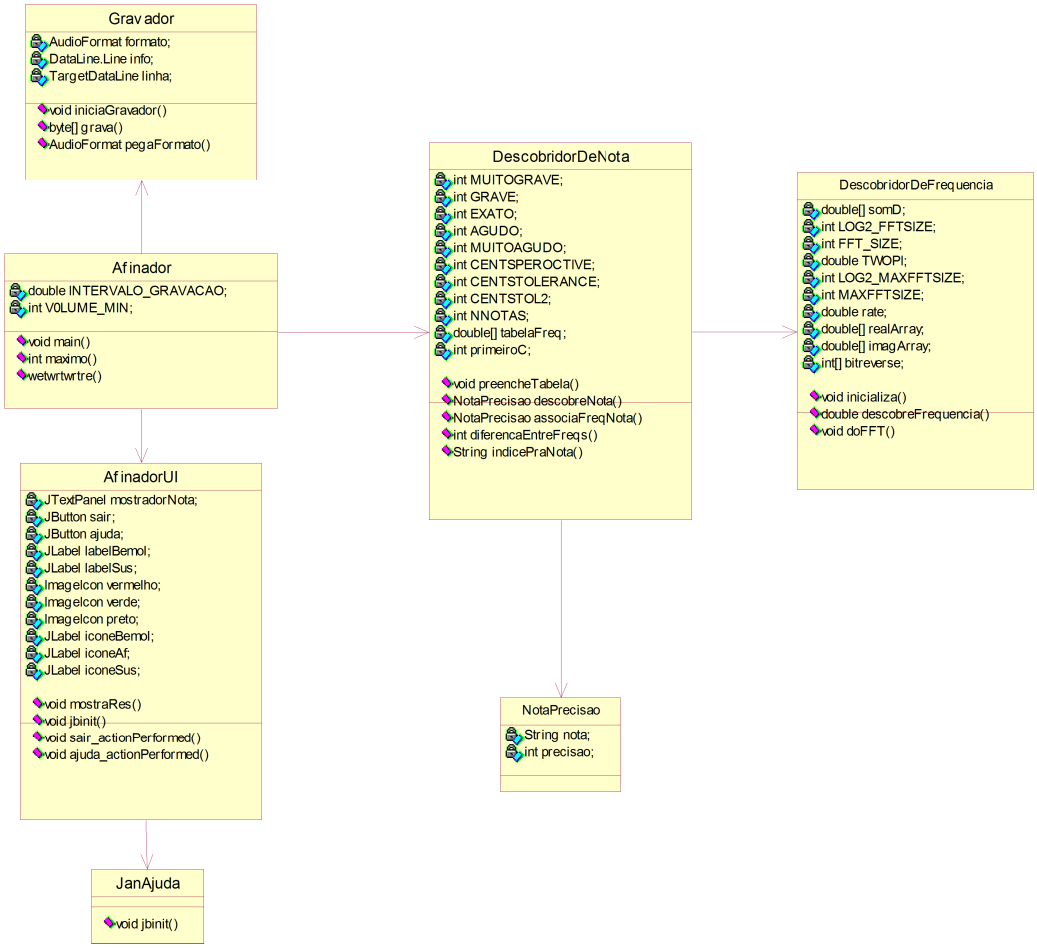

Figura 11. Diagrama de classes

O software *My Tuner* foi implementado na linguagem Java para poder explorar seus recursos de portabilidade. Sendo assim, é provável que funcione em qualquer sistema operacional.

A classe principal do software (Afinador.java) possui um *loop* infinito e dentro deste há uma chamada ao método grava da classe gravador (Gravador.java). Este método, implementado utilizando funções da API Java Sound, recebe como parâmetro o tempo que deve gravar o som proveniente do microfone do computador. O tempo *default* é de 0,5 segundos e o som recebido é armazenado em um vetor de bytes.

A seguir, ainda dentro do *loop* infinito, é realizada uma chamada ao método descobreNota da classe DescobridorDeNota (DescobridorDeNota.java). Este método recebe como parâmetro o vetor de bytes com o som gravado e faz uma chamada ao método descobreFrequencia da classe DescobridorDeFrequencia (DescobridorDeFrequencia.java). O método descobreFrequencia, realiza o complexo cálculo da FFT e retorna a freqüência encontrada (Figura 11).

Esta freqüência é associada a uma nota através do método associaFreqNota da classe DescobridorDeNota. Caso a freqüência encontrada não seja exatamente igual à de uma nota, é calculada a distância em *cents* da nota mais próxima e é atribuída uma precisão à nota de acordo com uma tolerância pré-estabelecida.

Existem outros métodos dentro das classes citadas, mas estes servem apenas como auxílio para que estas funções sejam implementadas.

O programa possui ainda as seguintes classes:

- AfinadorUI, cuja função é implementar uma interface para que o usuário possa utilizar o programa.
- JanAjuda, que implementa uma janela de ajuda ao usuário.
- NotaPrecisao, é uma classe auxiliar cuja função é a de juntar uma nota e uma precisão para posterior exibição ao usuário.

6.2. Uso (Figura 12)

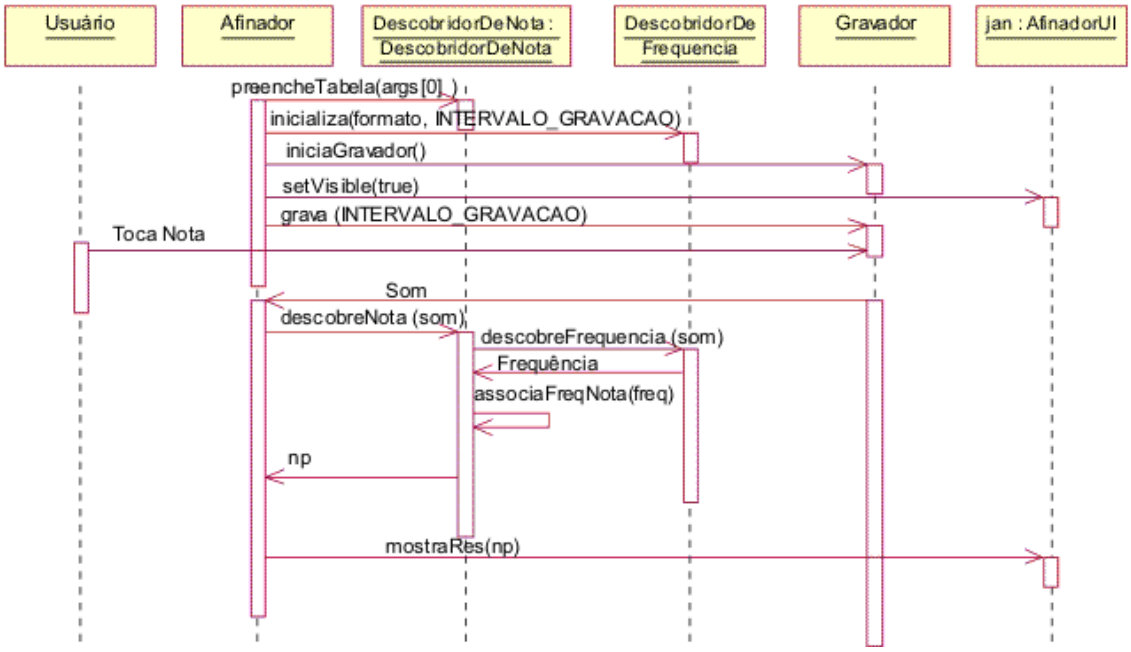

Figura 12. Diagrama de seqüência

Quando o software é inicializado deve ser passado como parâmetro um arquivo texto que contém, em sua primeira linha, o índice do primeiro C (dó) encontrado neste arquivo, seguido do número de freqüências que o arquivo contém. A partir da terceira linha, o arquivo possui em cada linha a freqüência de uma nota. Estas freqüências devem estar ordenadas de forma crescente. Junto com o software, é fornecido o arquivo texto que possui as freqüências para a afinação padrão (A4=440 Hz), para utilizar o software com outras afinações, basta trocar este arquivo.

O instrumento a ser afinado, deve estar ligado à entrada de microfone do computador e o volume deste deve estar suficientemente alto.

A seguir, basta tocar a nota desejada diversas vezes e a nota tocada vai aparecer no visor do programa. Caso esta esteja perfeitamente afinada, aparecerá uma luz verde acima do visor. Caso esteja um pouco mais grave ou aguda (Figura 13), luzes vermelhas

aparecerão acima dos símbolos de sustenido (#) e bemol (*b*) e a nota deve ser ajustada até estar completamente afinada (Figura 14).

Figura 13. My Tuner – Nota E (Mi) um pouco grave

Figura 14. My Tuner – Nota E (Mi) afinada

7. Conclusão

O processamento de sinais digitais é uma área da ciência da computação absolutamente inexplorada neste país, mas de grande importância na implementação de ferramentas para músicos e também na implementação de ferramentas de uso geral, como o *Eliminador de ruídos.*

Entretanto, seu uso depende diretamente de complexos cálculos matemáticos, aí incluídas as transformadas de Fourier.

Referências

Netto, L. (2001) “Inteferência sonora no diapasão”, http://www.feiradeciencias.com.br/sala10/10_36.asp, Novembro

Anônimo (2001) “Digital Signal Processing”, http://www.dsptutor.freeuk.com/intro.htm, Novembro

Bosquetti, D. (1996) “Física da Música”, http://www.cdcc.sc.usp.br/ondulatoria/musica.html, Novembro

Meloan, M. (1999) “Sound Technology”, http://java.sun.com/developer/technicalArticles/Media/JavaSoundAPI/, Novembro

Baldwin, R. (2003) “Java Sound, Getting Started”, http://www.developer.com/java/other/article.php/1572251, Novembro

Anônimo (2000) “Pulse-Code Modulation”, http://www.tpub.com/neets/book12/49l.htm, Novembro

Nagato, M (2002) “Trabalho de formatura”, http://www.linux.ime.usp.br/~cef/mac49902/monografias/mfnagato/, Novembro

Iezzi, G. (2000) “Fundamentos de matemática elementar”, Atual Editora

Jesus, S (2005) “Introdução à FFT”, http://w3.ualg.pt/~sjesus/aulas/pds/node19.html, Novembro