Capítulo 2

Desenvolvendo Soluções com Interface Baseada em Voz

Alexandre Maciel, Rodrigo Lins, Pablo Barros e Edson Carvalho

Abstract

Oral communication is undoubtedly the most natural form of human communication. As human-computer interaction becomes increasingly common, there is a natural demand for systems capable of interacting with users in a more practical and intuitive way. This tutorial aims to demonstrate the process of building applications using speech interfaces. This is achieved by using a development environment called FIVE (Framework for an Integrated Voice Environment) that allows an easy instantiation of speech engines abstracting complexities. Given this scenario, researchers in the areas of speech processing and human-computer interaction will have the possibility to develop applications using technologies through an environment that fully offers productive mechanisms for the popularization of applications with voice interfaces.

Resumo

A comunicação oral é, sem dúvida, a forma mais natural de comunicação humana. Em virtude da interação humano-computador se tornar cada vez mais comum, surge uma demanda natural por sistemas capazes de interagir com os usuários de maneira mais prática e intuitiva. Para isso, este tutorial tem como objetivo a demonstração do processo de construção de aplicações utilizando interfaces de voz. Para tal é utilizado um ambiente de desenvolvimento intitulado FIVE (Framework for an Integrated Voice Environment) que permite de forma fácil a instanciação de motores de fala abstraindo complexidades inerentes. Diante desse cenário, pesquisadores da área de processamento de voz e da área interação humano-computador terão a possibilidade de desenvolver aplicações utilizando tecnologias através de um ambiente que oferece de forma completa e produtiva mecanismos para a popularização das aplicações com interface de voz.

2.1 Introdução

O homem sempre buscou meios de comunicação que facilitassem a sua interação com a máquina e a interface com o usuário é parte fundamental dessa comunicação. Segundo Moran (1981), a interface com o usuário deve ser entendida como sendo a parte de um sistema computacional com a qual uma pessoa entra em contato – física, perceptiva ou conceitualmente. Ou seja, é por meio da interface que o usuário pode manipular um dispositivo, perceber ou interpretar comandos e respostas desencadeadas pela interação com um sistema.

O desenvolvimento de uma boa interface com o usuário é um fator crítico para o sucesso de qualquer sistema, visto que pode se tornar uma fonte de motivação para o usuário, ou então, se mal projetada, um ponto de forte rejeição ao sistema. Diante dessa preocupação, o desenho de uma interface amigável deve levar em consideração a maneira natural com a qual o homem interage com o próprio homem, tentando levar essa naturalidade aos sistemas computacionais. Isso é alcançado através de interfaces que simulem o sistema sensorial humano (visão, audição, olfato, paladar e tato).

A imaginação é o limite para definir as aplicações que este tipo de interface pode ter. Vários meios de interação natural vêm sendo desenvolvidos ao longo dos anos e utilizados em aplicações, seja isoladamente ou de modo integrado. São as chamadas aplicações com interface multimodal. Como exemplo de interfaces naturais pode-se citar telas sensíveis ao toque, luvas (*datagloves*), sistemas de câmeras que captam gestos, sistemas que controlam o movimento dos olhos, ou as interfaces de voz, que possibilitam a realização de diálogos por meio de reconhecimento e síntese voz.

Nos últimos anos, a área de interface de voz tem recebido grande atenção da academia, em virtude de dois fatores: primeiro, devido à melhoria na performance dos sistemas de processamento automático de fala, incluindo reconhecimento de fala, tradução de idiomas falados e síntese de voz; segundo, devido à convergência de dispositivos e à massiva produção de conteúdos multimídia, que passaram a requerer modos mais rápidos e eficientes de interação com os usuários (Salvador *et al.*, 2010).

Além de ser uma das formas mais naturais de comunicação entre os homens, a interface baseada em voz oferece inúmeras vantagens quando comparadas a outras formas de interface, por exemplo: a velocidade (a maioria das pessoas pode falar facilmente a taxas de 200 palavras por minuto, enquanto poucas conseguem digitar mais de 60 palavras por minuto); a mobilidade (em diversos ambientes não cabe a utilização de teclados e mouse ou os olhos do usuário devem permanecer fixos em um *display*); e a segurança (a voz é um mecanismo biométrico que pode ser usado para verificar o acesso a sistemas e a ambientes restritos) (Fechine, 2002).

Apesar destas vantagens, o processo de construção de uma Interface de Voz com o Usuário (no inglês: *Voice User Interface* - VUI) apresenta diferenças substanciais quando comparado a uma Interface Gráfica do Usuário (no inglês: *Graphical User Interface* - GUI). Como a voz não está "visível" ao usuário, como no caso das GUIs, e também devido seu caráter transiente, a interface de voz normalmente requer uma a carga cognitiva consideravelmente maior do usuário. Outro fator importante é que entradas por voz são muito mais rápidas que as entradas via teclado, no entanto, as saídas por voz são muito mais lentas do que leituras textuais feitas superficialmente.

Desse modo, cabe ao projetista da interface balancear a melhor forma de conjugar as opções de interface aproveitando as principais vantagens que cada uma delas oferece (Martins e Brasiliano, 2012).

Diante desse cenário, as pesquisas na área de interfaces de voz têm evoluído bastante no sentido de oferecer mecanismos que auxiliem o desenvolvimento de aplicações com interface de voz. Este tutorial, cujo público alvo são os projetistas de aplicações com interface de voz, irá demonstrar a utilização do ambiente FIVE (*Framework for an Integrated Voice Environment*), ferramenta construída com o objetivo de auxiliar o desenvolvimento integrado de aplicações com interfaces de voz com rápida curva de aprendizagem, independência de plataforma, extensibilidade a novas técnicas.

O restante deste documento está organizado da seguinte forma: a seção 2 aborda a fundamentação teórica acerca da área de interface de voz, bem como a apresentação do processo de construção e instanciação de motores de fala; a seção 3 apresenta o framework FIVE e um tutorial básico para sua utilização; a seção 4 apresenta um conjunto de exemplo de motores criados com o FIVE; a seção 5 apresenta aplicações de demonstração com interface baseada em voz; e por fim, a seção 6 apresenta as considerações finais sobre este tutorial.

2.2 Interface de Voz

Interface do usuário baseada em voz, em inglês *Voice User Interface* (VUI) consiste na interação de uma pessoa com um sistema através de voz, utilizando uma aplicação de linguagem falada (Shneiderman, 2000).

As VUI's tiveram sua origem em pesquisas de Inteligência Artificial, especialmente no desenvolvimento de "conversational interfaces", na década de 1950. Mas somente depois dos anos 90 é que a tecnologia contou com uma significativa melhoria (Cohen *et al.*, 2004).

A área de interface de voz com o usuário abrange uma grande variedade de subáreas que visam atender as diversas demandas da sociedade. Segundo Campbell (1997), as tecnologias de voz podem ser classificadas como pertencentes a uma das seguintes subáreas:

- **Codificação:** processo de compressão da informação em um sinal de voz assim como a sua transmissão de forma econômica por intermédio de um canal cuja largura de banda seja menor do que um sinal não comprimido.
- **Síntese de Voz:** processo de criação de uma réplica sintetizada de um sinal de fala a partir de uma mensagem textual, com o propósito de fazer saber a informação contida na mensagem.
- **Reconhecimento de Fala:** processo de extração de informação de uma mensagem contida em um sinal de fala que, na sua forma mais simples, permite controlar as ações de uma máquina em resposta aos comandos falados.
- **Reconhecimento de Locutor:** processo tanto de verificação como de identificação do locutor com o propósito de restringir acesso à informação (arquivos privados), a redes (computador ou telefônicas), ou permissões físicas.

- **Identificação de Linguagem:** processo de identificação do idioma que o locutor está falando a partir da voz do locutor.

Todas as tecnologias supracitadas podem desempenhar um papel importante no domínio das aplicações, contudo as áreas que têm recebido maior atenção por parte da comunidade acadêmica são: reconhecimento de fala, síntese de voz e reconhecimento de locutor (Maciel, 2012).

De acordo com Zukerman e Litman (2001), é possível construir sistemas baseados em comandos de voz interativos e em tempo-real, em que as entradas do usuário sejam capturadas por um reconhecedor automático de voz e as saídas do sistema sejam enviadas através de um sintetizador de voz, ou através de mensagens previamente gravadas. Desse modo, para que uma aplicação com interface baseada em voz seja efetiva é preciso ela integre motores de fala diversos que "entendam" o que o usuário diz, desempenhem um processo de computação/transação, e respondam ao usuário de tal forma que dê prosseguimento à conversação e cumprimento dos objetivos do usuário.

Segundo Huang *et al.* (2001), a arquitetura típica para o desenvolvimento de uma aplicação com interface de voz possui três componentes principais: o primeiro representa o conjunto de motores que são responsáveis pelo reconhecimento de fala ou pela síntese de voz, o segundo consiste numa API (*Application Programming Interface*) que normalmente é usada para facilitar a comunicação entre os motores e a aplicação, e, o último consiste num conjunto de possíveis aplicações que podem ser desenvolvidas. A Figura 2.1 apresenta uma visão geral destes componentes dentro da arquitetura mencionada.

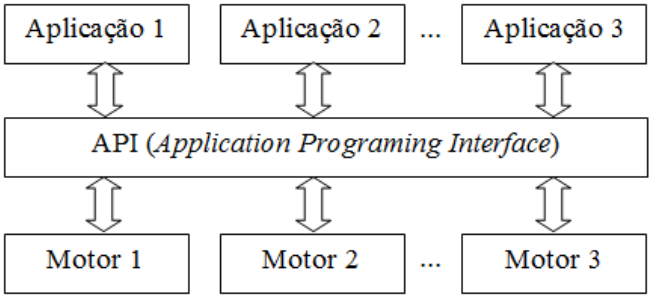

Figura 2.1. Arquitetura de aplicação com interface de voz (Huang *et al.*, 2001)

Esta arquitetura possibilita uma manutenção mais simplificada, já que é possível alterar um componente de uma camada sem influenciar a outra, possibilitando uma clara separação entre a tarefa de construção de motores e a instanciação deles na camada de aplicação. Desse modo, uma equipe integrada por especialistas em processamento de voz pode cuidar do desenvolvimento dos motores, seja de reconhecimento de fala, síntese de voz ou reconhecimento de locutor, enquanto outra equipe composta por projetistas de interfaces, artistas visuais, cientistas da computação e usuários pode ser direcionada para o desenvolvimento da interface (Martins, 2011).

2.2.1. Construção de Motores

A construção dos motores, seja para o reconhecimento fala, seja para a síntese de voz, consiste fundamentalmente em um processo de reconhecimento de padrões. Segundo Duda *et al.* (2000), este processo normalmente segue uma arquitetura comum a qualquer padrão (imagem estática, voz, dados financeiros, etc) e pode ser particionado em quatro módulos principais, como mostra a Figura 2.2.

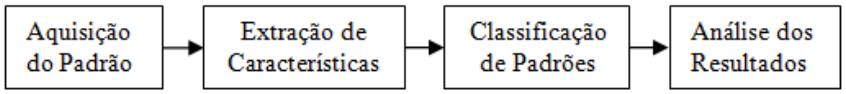

Figura 2.2. Componentes de um sistema de reconhecimento de padrões

Especificamente para os sistemas de reconhecimento de fala, o módulo de aquisição do padrão consiste na captura do sinal de voz e na conversão do mesmo em amostras digitais. O processo de captura das ondas de som normalmente é realizado por meio de um microfone e a conversão é realizada em conjunção com uma placa digitalizadora de som. Associada ao áudio adquirido, os sistemas da fala requerem a transcrição textual do conteúdo do áudio como um modo de ajustar os modelos de linguagem às amostras de áudio (McLoughlin, 2009).

O módulo de extração de características é o responsável por computar os dados do sinal de áudio e gerar as informações representativas necessárias ao módulo de classificação de padrões. Usando apenas as características importantes do sinal, a quantidade de dados utilizada para comparação é reduzida consideravelmente, assim, menos poder computacional é requerido e menos tempo de processamento é necessário. Segundo O'Shaughnessy (2008), os dois principais tipos de parâmetros de fala são os Coeficientes por Predição Linear (*Linear Prediction Coefficients* – LPC) e os Coeficientes Cepstrais em Escala Mel (*Mel-Frequency Cepstral Coefficients* – MFCC).

O módulo de classificação dos padrões consiste na utilização de abordagens algorítmicas para estabelecer representações consistentes dos padrões de voz a partir de um conjunto de amostras de treinamento rotuladas, e para realizar uma comparação fiável em conjunto de amostras de teste (Gaikwad *et al.*, 2010). Segundo O'Shaughnessy (2008), as abordagens mais utilizadas para a classificação de padrões são baseadas nos Modelos Ocultos de Markov, nas Redes Neurais Artificiais e nas Máquina de Vetor de Suporte. Especificamente para problemas de verificação de locutor, Holmes (2002) afirma que as principais abordagens associadas à classificação de padrões são baseadas na técnica de Quantização Vetorial ou nos Modelos de Misturas Gaussianas.

Por fim, o módulo de análise dos resultados consiste num conjunto de métricas utilizadas para oferecer uma melhor apresentação das saídas dos testes de classificação de padrões. Para os sistemas de reconhecimento de fala, a Taxa de Erro por Palavra (*Word Error Rate* – WER) associada a uma Matriz de Confusão é uma das métricas mais comumente usadas (Stehman, 1997). Já para os sistemas de síntese de voz há diversas abordagens para avaliar as vozes sintéticas. A escolha de qual abordagem escolher depende do propósito da avaliação e da aplicação específica a que se destina a voz sintética (Cryer and Home, 2010).

2.2.2. Instanciação de Motores

A etapa de instanciação dos motores de fala numa aplicação é normalmente suportada por uma camada de software, que oculta do desenvolvedor os detalhes de implementação de uma aplicação com interface baseada em voz. Este conjunto de software, habitualmente, designado API, permite o controle em tempo real dos motores de reconhecimento de fala e de síntese voz, bem como as interfaces de entrada/saída de áudio (Maciel *et al.*, 2008).

No contexto das aplicações com interface de voz, Huang (2001) afirma que o uso da API é importante porque garante que múltiplas aplicações possam trabalhar com uma grande quantidade de componentes (motores) disponibilizados por diferentes fornecedores de tecnologia de fala, em ambientes computacionais diversos.

Atualmente há uma grande variedade de iniciativas visando à disponibilização de API's para instanciação de motores de fala. A Microsoft Speech API e a Loquendo API são soluções proprietárias que permitem a interação com diversos tipos de motores independentes de fabricantes. Já a Java Speech API e o W3C VoiceXML (McGlashan *et al.*, 2010) são soluções criadas com o objetivo de oferecer um padrão que seja capaz de reconhecer e sintetizar fala independente de plataforma.

De modo geral, as funções disponíveis numa API são acessíveis somente por meio de programação. Ou seja, as requisições que os desenvolvedores desejam fazer à API são necessariamente realizadas por meio de chamadas inseridas no código fonte de suas aplicações. A sintaxe necessária para a realização dessas chamadas normalmente é definida na documentação disponibilizada junto com a API, sem a qual tal tarefa torna-se inviável.

A maioria da API's de fala disponibiliza suas funcionalidades segundo duas grandes áreas: reconhecimento e síntese. Isso se deve ao fato de que o projeto de uma aplicação com interface baseada em voz deve levar em consideração o modo natural como o homem se comunica com o próprio homem, ou seja, a interação por voz deve simular, de maneira natural, um diálogo com conversas caracterizadas por mudanças de iniciativas e feedbacks verbais e não verbais para poder indicar um verdadeiro entendimento (Maciel e Carvalho, 2007).

A Figura 2.3 mostra um exemplo de um diálogo entre um usuário e uma aplicação com interface de voz.

```
[sistema]: Bom dia. Bem vindo ao serviço de apoio ao passageiro. O que
deseja?
[usuário]: Qual o horário do próximo ônibus do Central Parque para o
Brooklin?
[sistema]: O próximo ônibus com origem Central Parque e destino
Brooklin sai às catorze horas e trinta minutos. Ajudo em algo mais?
[usuário]: Qual o preço do bilhete?
[sistema]: O bilhete custa dois dólares e quinze cents. Algo mais?
[usuário]: Não, obrigado.
[sistema]: Nós é que agradecemos. Tenha um bom dia.
```

Figura 2.3. Exemplo de diálogo com iniciativa mista

2.3 Framework FIVE

O FIVE (*Framework for an Integrated Voice Environment*) é um framework construído com o objetivo de auxiliar o desenvolvimento integrado de aplicações com interfaces de voz. Com ele, é possível gerar motores de reconhecimento de fala, de verificação de locutor e de síntese de voz e, por meio de uma API (*Application Programming Interface*) própria, instanciar estes motores em aplicações desenvolvidas em múltiplas plataformas (Maciel e Carvalho, 2010).

Os requisitos utilizados na construção do FIVE levam em consideração uma pesquisa realizada junto à comunidade acadêmica da área de processamento de voz em (Maciel, 2012). Segundo esta pesquisa, os especialistas consultados mencionaram que os requisitos essenciais para uma ferramenta produtiva que auxilie o desenvolvimento de aplicações com interface baseadas em voz são: rápida curva de aprendizagem, portabilidade entre ambientes computacionais, e capacidade de extensão para as técnicas de extração de características e de classificação de padrões.

Diante desses requisitos, a arquitetura geral do FIVE foi desenhada de modo independente, baseada em três módulos: o CORE, que consiste num conjunto de classes abstratas parcial ou completamente implementadas definidas de modo sequencial e extensível para auxiliar o processo de criação dos motores de fala; o módulo API, que consiste numa implementação simples, robusta, transparente e portável que otimiza a instanciação dos motores na camada de aplicação; e o módulo GUI (*Graphical User Interface*) que consiste numa interface gráfica que auxilia o desenvolvedor através de uma interface sequencial e de fácil compreensão. A Figura 2.4 mostra os módulos da arquitetura geral do FIVE.

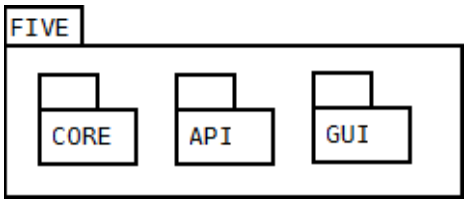

Figura 2.4. Arquitetura geral do FIVE

O FIVE é um framework orientado a objetos, desenvolvido em linguagem de programação Java, utilizando o modelo de divisão de responsabilidades MVC (*Model-View-Controller*) (Gamma *et al.*, 1995) e o framework de persistência Hibernate para proporcionar independência de sistema de banco de dados.

Desenvolvido como parte da tese de doutorado intitulada *Investigação de um Ambiente para o Desenvolvimento Integrado de Interface de Voz* (Maciel, 2012), o FIVE é um software livre, registrado no Instituto Nacional de Propriedade Intelectual do Brasil (INPI), com os direitos autorais pertencentes à Universidade Federal de Pernambuco. A ideia dos mantenedores do FIVE é criar uma grande rede de colaboração para que a comunidade acadêmica possa ajudar no aperfeiçoamento e na implementação de melhorias para a ferramenta. Mais informações sobre os termos de uso e normas de colaboração estão disponíveis no site oficial: www.cin.ufpe.br/~five.

A partir da consolidação da ferramenta, foi construído um tutorial como modo de auxiliar os desenvolvedores desde sua inicialização até a geração do motor de fala. As seções a seguir apresentam o passo a passo deste tutorial.

2.3.1. Inicialização

Para iniciar o FIVE, basta fazer o registro no site oficial do framework, descompactá-lo e executar o arquivo *FIVE_GUI.bat*, caso esteja utilizando o ambiente Windows, ou *FIVE_GUI.sh*, caso esteja utilizando o ambiente Linux. Cabe observar se a pasta *lib* se encontra com os arquivos FIVE_CORE.jar e FIVE_API.jar e todas as outras bibliotecas que acompanham a distribuição do FIVE.

2.3.2. Projeto

Para criar um projeto no framework FIVE, o usuário deve clicar no menu *Arquivo* e escolher a opção *Novo*. A tela *Novo Projeto* será apresentada e, nela, o usuário deve escolher o tipo de projeto que deseja criar: Reconhecimento de Fala, Reconhecimento de Locutor ou Síntese de Voz. Em seguida, o usuário deve clicar no botão *Avançar* e informar o nome e o local que deseja dar ao seu projeto. Por fim, é preciso especificar o banco de dados que deseja utilizar para armazenamento das informações dos cadastros realizados e dos resultados obtidos. A Figura 2.5 mostra a sequência de abas da tela *Novo Projeto*.

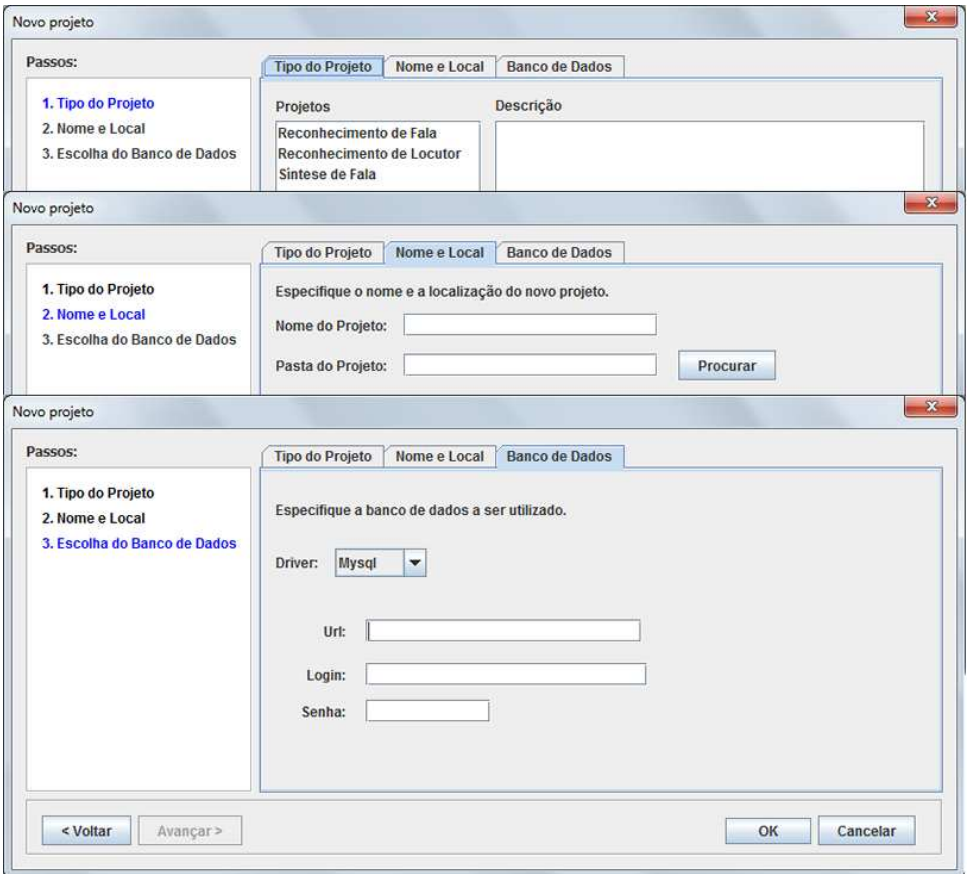

Figura 2.5. Tela Novo Projeto

Assim que for criado o projeto, o painel principal do FIVE apresentará um conjunto de abas com as seções necessárias à construção de um motor de fala e uma estrutura de arquivos e pastas será criada no caminho informado.

2.3.3. Locutor

Para cadastrar um locutor, o usuário deve clicar no botão *Incluir*, na aba *Locutores*, e informar os dados solicitados da tela *Locutores*. Informações como Região, Idade e Gênero são úteis caso o usuário deseje especializar o seu motor de fala, fazendo para isso o uso de opções de filtros que possibilitam essa especialização. A Figura 2.6 mostra a Tela *Locutores*.

2.3.4. Locução

Para cadastrar uma locução o usuário deve clicar no botão *Incluir* na aba *Locuções* e escrever a locução desejada na tela *Locuções*. O conteúdo escrito neste espaço pode variar de acordo com o tipo de projeto que o usuário esteja criando. Para motores de reconhecimento de fala, podem ser inseridas palavras isoladas ou palavras contínuas. Para motores de reconhecimento de locutor, podem ser inseridas palavras isoladas, que funcionam como uma senha para o reconhecimento dependente do texto ou textos foneticamente balanceados, pelos quais os modelos fonéticos são extraídos para o reconhecimento independente do texto. Para motores de síntese de voz, devem ser inseridos textos foneticamente balanceados buscando extrair os melhores modelos fonéticos para a geração de fala sintética. A Figura 2.7 mostra a tela *Locuções*.

Figura 2.6. Tela de Locutores

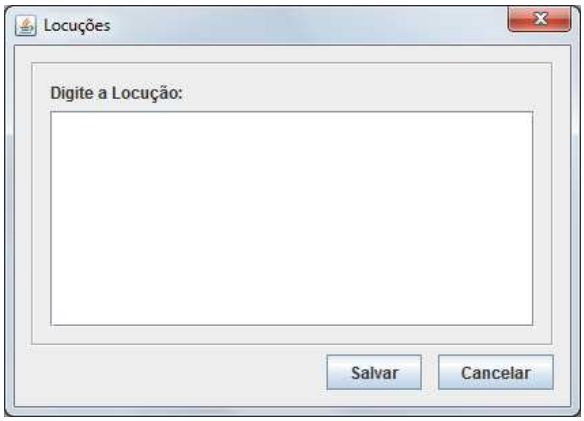

Figura 2.7. Tela de Locuções

A princípio, o processo de cadastro de locuções parece uma ação bastante simplista, mas, na verdade ele realiza um passo fundamental para o desenvolvimento do restante do sistema: o Processamento da Linguagem Natural (PLN). Este processo é dividido em duas fases, a primeira consiste no pré-processamento do texto, transformando abreviações, siglas, e números em representações textuais. A segunda consiste na realização da conversão grafema-fonema, da divisão silábica e da determinação da sílaba tônica das palavras contidas na locução.

2.3.5. Amostras de Áudio

Para cadastrar uma amostra de áudio o usuário deve clicar no botão *Incluir* na aba *Base de Dados* e preencher os campos informados na tela *Amostra*. Para que uma amostra de áudio seja inserida com sucesso, o usuário deve informar todos os campos solicitados na tela: o locutor que irá realizar a gravação, a locução que será falada, o ambiente em que a gravação está sendo realizada, se é locução é genuína ou impostora (informação necessária para motores de verificação de locutor), as características da amostra, e, por fim, o nome do arquivo de áudio que será salvo na pasta *samples* do projeto. A Figura 2.8 mostra a tela *Amostra*.

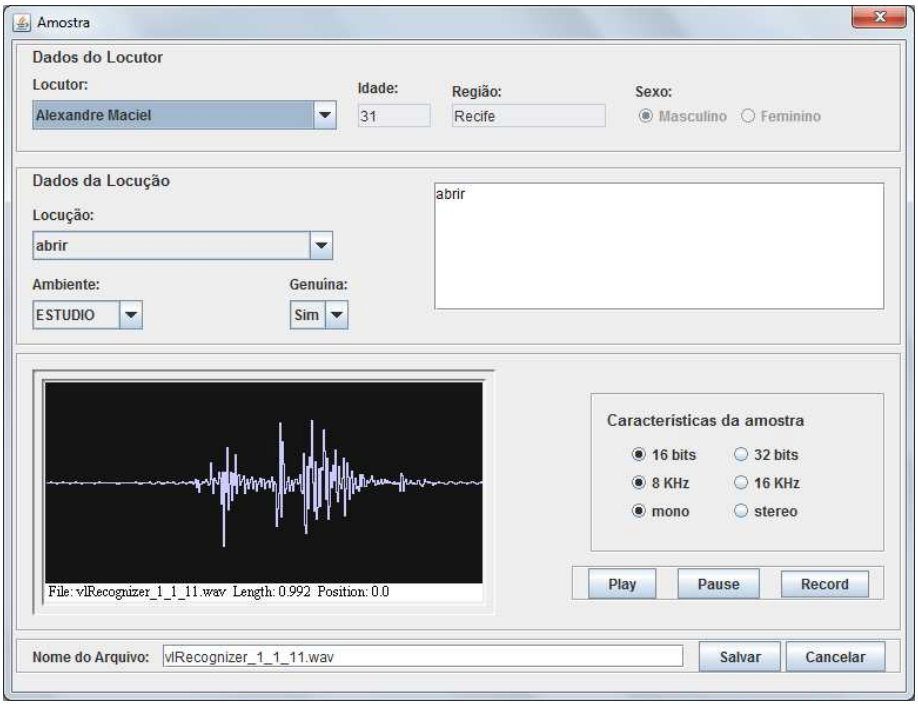

Figura 2.8. Tela de Amostras de Áudio

Para realizar a gravação do áudio, o usuário deve clicar no botão Record, que, assim que é clicado, muda o nome para Stop. Caso deseje ouvir o áudio gravado, o usuário deve clicar no botão Play.

2.3.6. Extração de Características

Para realizar a extração de características das amostras de áudio existentes no projeto o usuário deve clicar no botão *Incluir* na aba *Extração de Características* e preencher os campos informados na tela *Parâmetros da Extração de Características*. Nesta tela, o usuário encontra alguns parâmetros necessários à extração de características. Caso deseje filtrar apenas algumas amostras de áudio a serem extraídas, pode utilizar as opções de Filtro disponibilizadas no topo da tela. Alguns parâmetros solicitados para a extração são fixos, como: Superposição, Duração do Frame, Pré-ênfase e Janelamento. A depender da técnica escolhida, outros parâmetros exclusivos dela devem ser informados. A Figura 2.9 mostra a tela *Parâmetros da Extração de Características.*

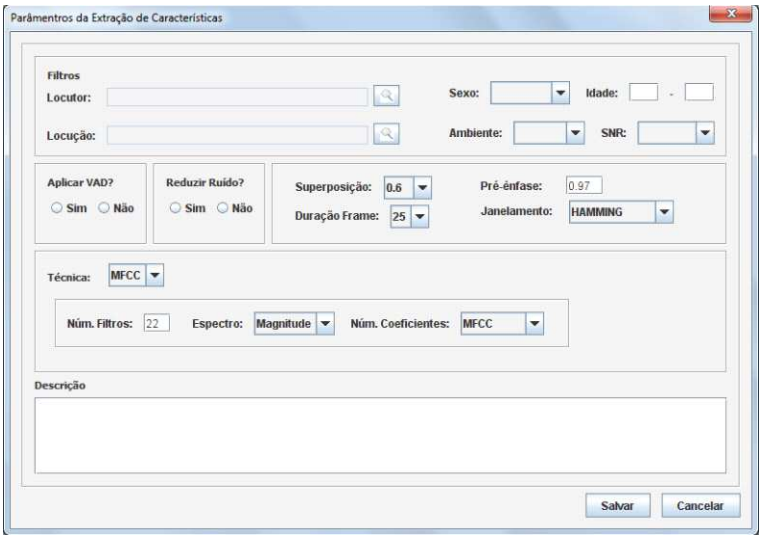

Figura 2.9. Tela Parâmetros de Extração de Características

2.3.7. Classificação de Padrões

Para realizar a classificação dos padrões das características extraídas no projeto o usuário deve clicar no botão *Incluir* na aba *Classificação de Padrões* e preencher os campos informados na tela *Parâmetros da Classificação de Padrões*. Nesta tela, o usuário encontra alguns parâmetros necessários à classificação dos padrões. Caso deseje filtrar apenas algumas amostras de áudio a serem classificadas, pode utilizar as opções de Filtro disponibilizadas no topo da tela. Alguns parâmetros solicitados para a classificação são fixos como: Percentual de Treino, de Teste e *Threshold* (limiar de aceitação de reconhecimento). A depender da técnica escolhida outros parâmetros exclusivos dela devem ser informados. A Figura 2.10 mostra a tela *Parâmetros da Classificação de Padrões.*

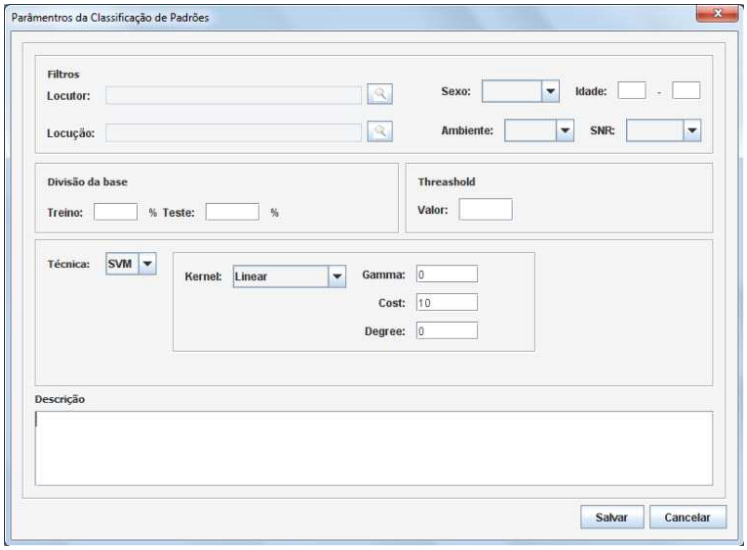

Figura 2.10. Tela de cadastro de classificação de padrões

2.3.8. Geração do Motor

Um motor de fala consiste no agrupamento de alguns artefatos produzidos ao longo do processo construção do motor. Esses artefatos variam de acordo com o tipo de projeto escolhido pelo usuário. Para gerar o motor de fala o usuário deve clicar no botão *Incluir* na aba *Geração do Motor* e preencher os campos informados na tela *Geração do Motor.* Nesta tela, o usuário encontra os parâmetros, nome do motor, arquivo de configuração e de modelos acústicos e, a depender do tipo de projeto escolhido, outros parâmetros exclusivos dele devem ser informados. A Figura 2.11 mostra a tela *Geração do Motor.*

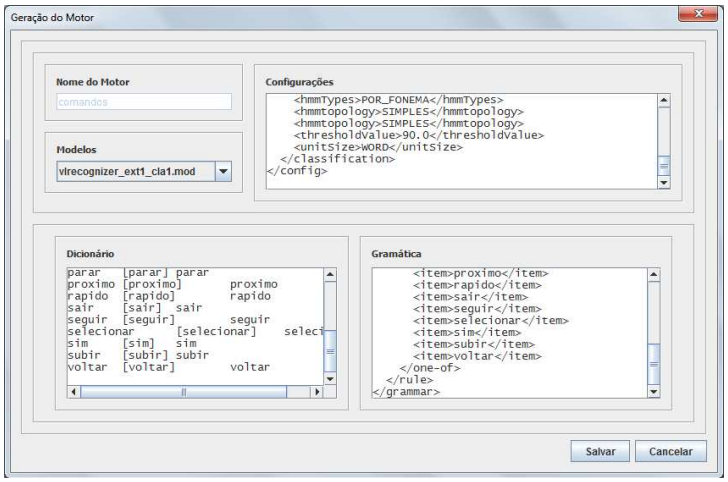

Figura 2.11. Tela de cadastro de motor

Ao final do processo de geração do motor, o usuário deve observar os arquivos gravados na pasta *engine* do projeto. É esta pasta que junto ao FIVE_API leva a interface de voz ao nível de aplicação. A Figura 2.12 mostra o conteúdo da pasta *engine*.

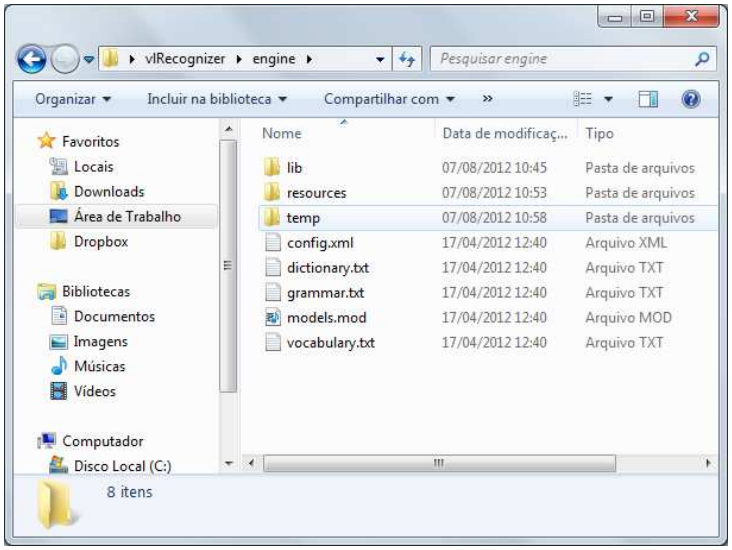

Figura 2.12. Conteúdo da pasta engine

2.4 Exemplos de Motores

Com o propósito de otimizar a demonstração de instanciação de motores em aplicações de exemplo, decidimos utilizar dois motores previamente desenvolvidos. Estes motores fazem parte do trabalho de Maciel (2012). As seções a seguir apresentam os parâmetros e os resultados utilizados na criação de um motor de reconhecimento de fala isolada e de um motor de síntese de voz.

2.4.1. Motor de Reconhecimento de Fala

Para validar os resultados dos motores de fala construídos usando o FIVE, decidiu-se pela criação de um motor de reconhecimento de palavras isoladas. 40 locutores voluntários (30 masculinos e dez femininos) com faixa etária média de 23,5 anos, da região nordeste do Brasil foram selecionados e 20 palavras isoladas, que representam comandos de controle, foram escolhidas para fazer parte do motor de fala.

O processo de aquisição das amostras de áudio ocorreu com a gravação de cinco amostras por locutor para cada uma das palavras, com a frequência de amostragem de 8 kHz, 16 bits por amostra, mono, em ambiente com ruído controlado. Um total de 4000 amostras foi gravado e, após uma averiguação amostral foram eliminadas algumas gravações de má qualidade, restando 3.933 arquivos.

Em seguida, realizou-se a extração de características utilizando o algoritmo MFCC padrão, o algoritmo MFCC seguindo o formato do HTK e, por fim, o algoritmo MFCC baseado no padrão ETSI. Depois de vários testes, numa escala intervalar de parâmetros empíricos, achou-se a configuração paramétrica ideal: pré-ênfase de 0,97, superposição de 60%, duração do frame de 25 milissegundos, e um banco de 22 filtros.

Por fim, realizaram-se os experimentos de classificação dos padrões utilizando as técnicas HMM e SVM. Para os experimentos com HMM, as características utilizadas foram as MFCC_HTK e HTK_ETSI e os modelos fonéticos adotados foram: palavra inteira, fonemas e trifonemas. Para os experimentos utilizando SVM as características utilizadas foram as MFCC e as funções Kernel utilizadas foram: Linear, Polinomial, Sigmoide. A Tabela1.1 mostra as melhores taxas de acertos obtidos nesses experimentos.

Tabela 1.1. Taxas de acerto para reconhecimento de fala.

Técnica	Unidade Fonética	Taxa de acerto
HMM-HTK	Palavra	88,12%
	Fonema	95,72%
	Trifonema	87,89%
HMM-ETSI	Palavra	96,21%
	Fonema	97,28%
	Trifonema	89,10%
SVM	Linear	98,30%
	Polinomial	98,30%
	Sigmoide	97,67%

2.4.2. Motor de Síntese Voz

A validação dos resultados dos motores de síntese foi realizada em duas etapas: a primeira consistiu no treinamento para obtenção dos modelos acústicos de duas vozes sintéticas (uma masculina e uma feminina), e a segunda consistiu numa pesquisa de opinião para avaliar a qualidade dessas vozes.

A etapa de treinamento iniciou com a seleção de dois locutores profissionais, com tratamento fonoaudiológico adequado para evitar sotaques e regionalismos, e com a seleção de 800 locuções foneticamente balanceadas, que foram gravadas pelos locutores em um ambiente de estúdio profissional. O formato das amostras de áudio foi: frequência de amostragem de 44.100 kHz, 16 bits por amostra e modo estéreo.

O processo de treinamento utilizado no FIVE para a obtenção dos modelos acústicos das vozes sintéticas seguiu o modelo baseado em HMM proposto por Maia (2006). Para isso, algumas adaptações no processo de extração de características foram realizadas, a fim de se obter os parâmetros de aperiodicidade e a frequência fundamental. Também foram realizados ajustes no processamento de linguagem natural a fim de se obter informações contextuais das locuções utilizadas no treinamento.

Finalizada a etapa de treinamento, iniciou-se a etapa de avaliação, com a sintetização de 30 frases foneticamente balanceadas com quantidade de palavras variadas, interrogativas e afirmativas. Estas frases foram sintetizadas com as vozes masculina e feminina baseadas nos modelos de difones do Mbrola (Mbrola, 2012); nos modelos acústicos obtidos com o FIVE; e no sintetizador comercial da Loquendo.

O processo de avaliação foi realizado por 30 voluntários e iniciou com um cadastro para a avaliação do perfil dos entrevistados, necessário para avaliar o nível de percepção deles para realizar a transcrição das amostras de áudio, o segundo passo da avaliação. Nesta etapa foi solicitado aos entrevistados que ouvissem uma sequência de áudios, transcrevessem o conteúdo e avaliassem a qualidade segundo três critérios de qualidade: volume, dicção e pontuação.

A Figura 2.13 apresenta os resultados para as seis vozes analisadas (MM:Mbrola masculino, MF:Mbrola feminino, FF:FIVE feminino, FM:FIVE masculino, LF:Loquendo feminino e LM:Loquendo masculino). Para avaliar a qualidade da transcrição das frases faladas calculou-se o percentual de frases transcritas por inteiro e palavras individuais transcritas corretamente.

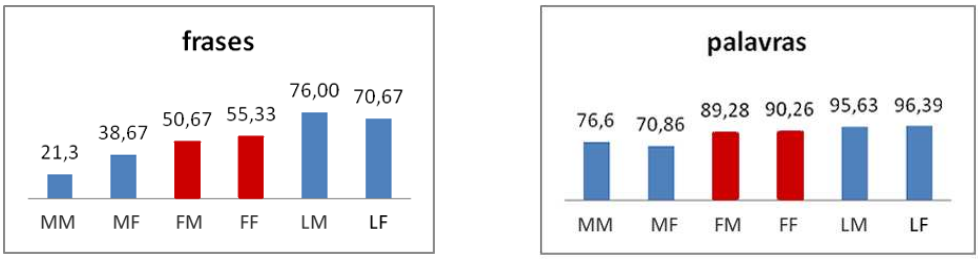

Figura 2.13. Taxa de acerto para transcrição de fala sintética

2.5 Aplicações de Demonstração

Com o objetivo de realizar a validação do processo de instanciação de motores criados a partir do framework FIVE foram desenvolvidas duas aplicações de demonstração que buscaram testar e avaliar o módulo API quanto as suas funcionalidades de reconhecimento de fala e de síntese de voz.

Estas aplicações foram desenvolvidas utilizando o ambiente de desenvolvimento integrado *NetBeans* e, por uma questão de otimização de tempo e esforço, tiveram seu escopo reduzido. A aplicação de reconhecimento de fala é apresentada em duas telas para demonstrar os dois modos de instanciação da API de reconhecimento: a partir de uma amostra de áudio pré-gravada, e a partir da captura de áudio por meio de um microfone. Já a aplicação de síntese de fala consiste numa tela simples onde o usuário digita a frase que deseja sintetizar.

Para criar um projeto de demonstração no *NetBeans* o usuário deve clicar na opção de Menu *Arquivo* e em seguida escolher a opção *Novo Projeto*. Feito isto será mostrada a tela *Novo Projeto*. Nela o usuário deve escolher a opção "Aplicativos Java" e em seguida informar na tela *Novo Aplicativo Java* o nome e a localização do projeto. A Figura 2.14 mostra as telas *Novo Projeto* e *Novo Aplicativo Java*.

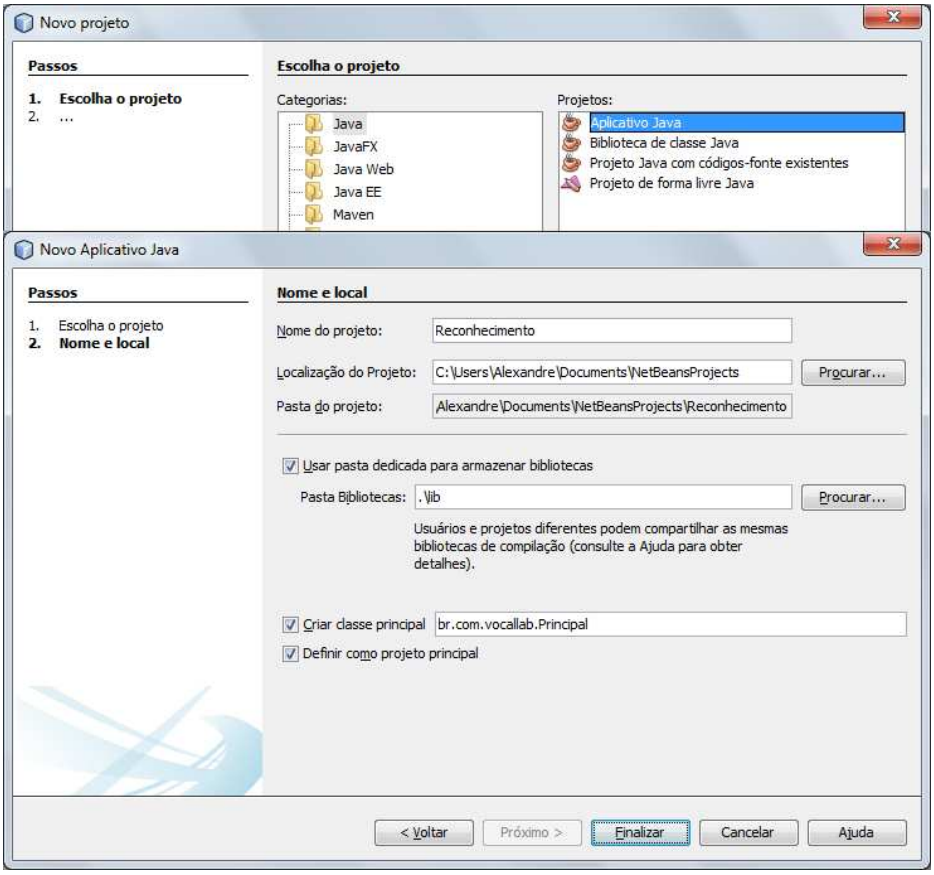

Figura 2.14. Passo a passo da tela Novo Projeto

Criado o projeto, o segundo passo para a construção de uma aplicação de demonstração, que seja capaz de reconhecer ou sintetizar fala, consiste na importação das bibliotecas do FIVE. Para inserir as bibliotecas do FIVE, o usuário deve clicar com o botão direito do mouse sob a pasta *Bibliotecas* na árvore de conteúdo do projeto e escolher a opção *Adicionar JAR/pasta*. Todas as bibliotecas disponibilizadas pelo FIVE devem ser selecionadas. A Figura 2.15 mostra o passo a passo para a inclusão das bibliotecas do FIVE.

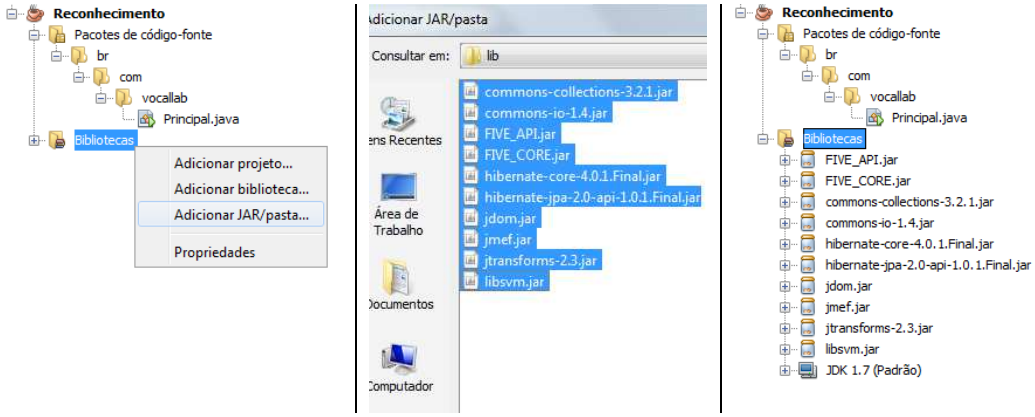

Figura 2.15. Passo a passo para inclusão das bibliotecas do FIVE

Finalizada a importação das bibliotecas, o usuário está livre para codificar sua aplicação. As seções a seguir apresentam trechos de código utilizados para a instanciação da API utilizando as funcionalidades de reconhecimento de fala e de síntese de voz, além de apresentar as telas resultantes das aplicações de demonstração.

2.5.1. Aplicação de Reconhecimento de Fala

A aplicação desenvolvida para validação dos recursos de reconhecimento de fala da API do FIVE levou em consideração o motor de reconhecimento de palavras isoladas construído conforme apresentado na seção 1.4.1. A Figura 2.16 mostra o trecho de código do modo de instanciação baseado em amostras de áudio pré-gravado.

```
try{
   String enginePath = "C:\Users\Alexandre\engine";
   Recognizer rec = new Recognizer(enginePath);

   Result res = rec.recognize(selectedFile);

   taText.append(res.getAnswer ();

catch(RecognizerException ex){
   System.out.println(ex.getMessasge());
}
```

Figura 2.16. Trecho de código para reconhecimento de amostra de áudio

No código apresentado, vê-se a especificação do atributo *enginePath* informando o local que o motor de reconhecimento de fala encontra-se e em seguida a instanciação do objeto *Recognizer*, que recebe o atributo *enginePath*. O processo de reconhecimento

a partir de uma amostra de áudio ocorre por meio da chamada do método *recognize*, passando como parâmetro o arquivo de áudio selecionado, e, por fim, o resultado obtido é apresentado numa área de texto através da chamada do método *getAnswer*. É importante salientar que, durante o reconhecimento, é possível que algum erro cause uma exceção do tipo *RecognizerException*, que pode ser capturada e tratada.

A Figura 2.17 mostra a tela construída para exibição dos resultados deste modo de instanciação. Nesta tela, o usuário busca o arquivo de áudio a ser reconhecido por meio do botão *Buscar*, e em seguida clica no botão *Reconhecer*, que apresenta o resultado no campo *Texto*.

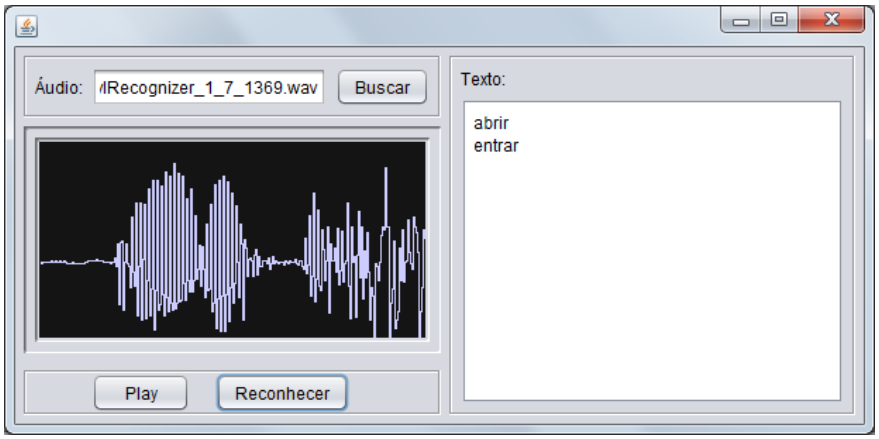

Figura 2.17. Tela de reconhecimento de fala a partir de amostra de áudio

A segunda aplicação para validação dos recursos de reconhecimento de fala da API do FIVE também levou em consideração o motor de reconhecimento de palavras isoladas construído conforme apresentado na seção 1.4.1. A Figura 2.18 mostra o trecho código responsável pelo reconhecimento de fala a partir da captura do som através de um microfone.

```
public class Principal implements RecognizerListener {

   public static void main (String args[]) {
      try{
         String enginePath = "C:\Users\Alexandre\engine";
         Recognizer rec = new Recognizer(enginePath);

         rec.addRecognizerListener(this);
         rec.startRecognition();

      catch(RecognizerException ex){
         System.out.println(ex.getMessasge());
      }
   }

   public void onRecognition(Result res){
      System.out.println(res.getAnswer());
   }
}
```

Figura 2.18. Reconhecimento de fala a partir de microfone

No código apresentado, a classe *Principal* implementa o recurso da classe *RecognizerListener* da API para a criação de eventos. Desse modo, o método *addRecognizerListener* direciona todos os eventos ocorridos na API para a classe *Principal*. Em seguida é apresentada chamada do método *StartRecognition*, responsável pelo abertura do microfone e início de captura de eventos. Como a classe *RecognizerListener* é uma classe abstrata, a classe *Principal* precisa implementar o evento *onRecognition* para que sempre que houver um reconhecimento, o resultado dele (*Result*) seja impresso para o usuário.

A Figura 2.19 mostra a tela construída para a exibição dos resultados deste modo de instanciação. Nesta tela, o usuário clica no botão *Start* e ao falar alguma palavra do dicionário, o sistema a reconhece e apresenta na seção *Palavras Reconhecidas*. Para obtenção de melhores resultados, ao iniciar o microfone o usuário deve esperar alguns segundos para calibração do nível de ruído ambiente.

Figura 2. 19. Tela da aplicação de reconhecimento de fala

2.5.2. Aplicação de Síntese de Voz

A aplicação desenvolvida para apresentação dos recursos de síntese da API do FIVE levou em consideração o motor o motor de síntese de voz construído conforme apresentado na seção 1.4.2. A Figura 2.20 mostra o trecho código responsável pela síntese de voz.

```
try{
   String enginePath = "C:\Users\Alexandre\engine";
   Synthesizer syn = new Synthesizer (enginePath);

   String phrase = taPhrase.getText();
   String speaker = cbSpeaker.getSelectedItem();

   syn.Synthesize(phrase, speaker);

catch(RecognizerException ex){
   System.out.println(ex.getMessasge());
}
```

Figura 2.20. Trecho de código para síntese de voz

Assim como no reconhecimento, na síntese, para instanciar o objeto *Synthesizer*, é preciso informar o caminho (*enginePath*) para acessar o motor. Com o objeto *Synthesizer* criado, o desenvolvedor pode realizar a chamada ao método *synthesize* fornecendo os parâmetros: *phrase* (frase a ser sintetizada) e *speaker* (voz do locutor de que irá narrar a frase). Durante o processo de síntese, algum erro pode causar uma exceção do tipo *SynthesizerException*, que pode ser capturada e tradada.

A Figura 2.21 mostra a tela para exibição dos resultados deste modo de instanciação. Nela, são informados os parâmetros: locutor, escolhido através do campo *Voz*, e a frase a ser sintetizada. Com os parâmetros preenchidos, o usuário clica no botão *Falar* e o sistema toca o áudio sintetizado.

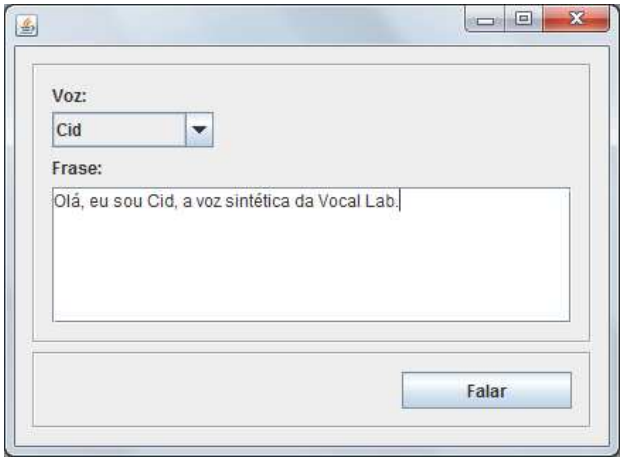

Figura 2.21. Tela da aplicação de síntese de voz

2.6 Considerações Finais

A comunicação oral é, sem dúvida alguma, a forma mais natural de comunicação humana. Em virtude da comunicação homem-máquina se tornar cada vez mais comum, surge uma demanda natural por sistemas capazes de interagir com os usuários de maneira mais prática e intuitiva.

Apesar da evolução da área de interface de voz nas últimas décadas, as aplicações com este tipo de recurso ainda são escassas no mercado e isso se deve a dois fatores: a tecnologia de processamento de voz requer avançados conhecimentos específicos na geração de motores e as ferramentas existentes não se encontram aptas a integrar as principais subáreas da interface de voz.

Este tutorial apresentou o ambiente produtivo FIVE para o desenvolvimento de aplicações com interface de voz que integra motores de fala diversos, com rápida curva de aprendizagem, de modo extensível a várias técnicas e independente de plataforma. Com essa ferramenta espera-se que este tutorial tenha proporcionado aos profissionais da área de Tecnologia da Informação, seja especialistas na área de processamento de voz, seja projetistas de interface, um rápido entendimento deste poderoso recurso, e que em estudos aprofundados posteriormente possam proporcionar a disponibilização de novas aplicações com interface baseada em voz no mercado.

Referências

Campbell, J.P. (1997) "Speaker Recognition: A Tutorial", In: Proceedings of the IEEE, v.85, p.1437-1462.

Cohen, M. H., Giangola, J. P., Balogh, J., Voice User Interface Design, Addison Wesley, 2004.

Cryer, H. and Home, S. (2010) "Review of Methods for Evaluating Synthetic Speech", In: RNIB Centre for Accessible Information (CAI), Technical report # 8.

Duda, R.O., Hart, P.E., Stork, D.G., Pattern Classification, Wiley-Interscience. 2000.

Fechine J. M. (2002) "Reconhecimento Automático de Identidade Vocal utilizando Modelagem Híbrida: Paramétrica e Estatística", Tese (Doutorado em Engenharia Elétrica). Universidade Federal da Paraíba.

Gaikwad, S.K., Gawali, B.W., Yannawar, P. (2010) "A Review on Speech Recognition Technique", In: International Journal of Computer Applications, v.10, No 3.

Gamma, E., Helm, R., Johnson, R. Vissides, J., Design Patterns: Elements of a Reusable Object-Oriented Software, Addison Wesley, 1995.

Holmes, J., Speech Synthesis and Recognition, CRC Press, 2002.

Huang, X., Acero, A., Hon, H.W., Spoken Language Processing – A Guide to Theory, Algorithm, and System Development, Prentice Hall, 2001.

Maciel, A. M. A. (2007) "Investigação de um Ambiente de Processamento de Voz utilizando VoiceXML". Dissertação (Mestrado em Ciência da Computação). Universidade Federal de Pernambuco.

Maciel, A. M. A., Veiga, A., Neves, C., Lopes, J., Lopes, C., Perdigão, F., Sá, L. A. (2008) "Robust Speech Command Recognizer for Embedded Applications". In: Proceedings of International Conference on Signal Processing and Multimedia Application. Porto.

Maciel, A. M. A., Carvalho, E. C. B. (2010) "FIVE – Framework for an Integrated Voice Environment". In: Proceedings of International Conference on Systems, Signal and Image Processing, Rio de Janeiro.

Maciel, A. M. A. (2012) "Investigação de um Ambiente de Desenvolvimento Integrado de Interface de Voz". Tese (Doutorado em Ciência da Computação). Universidade Federal de Pernambuco.

Martins, V. F. (2011) "Avaliação de Usabilidade para Sistemas de Transcrição Automática de Laudos em Radiologia". Tese (Doutorado em Engenharia) Universidade de São Paulo.

Marins, V. F. e Brasiliano, A. (2012) "Interface do Usuário Baseada em Voz como Ferramenta para Promover o Ensino/Aprendizagem de Língua Estrangeira". In: Revista Eletrônica do Alto Vale do Itajaí, vol. 1.

Maia, R. S. (2006) "Speech Synthesis and Phonetic Vocoding for Brazilian Portuguese based on Parameter Generation from Hidden Markov Models". Tese (Doutorado em Engenharia). Nagoya Institute of Technology.

MBROLA, "The Mbrola Project". (2012) Disponível em <http://tcts.fpms.ac.be/synthesis>. Setembro.

Mcglashan, S. *et al*. (2010) "Voice Extensible Markup Language (VoiceXML) 3.0". Disponível em <http://www.w3.org/TR/voicexml30>. Julho.

Mcloughlin, I., Applied Speech and Audio Processing. Cambridge, Cambridge University Press, 2009.

Moran, T. (1981) "The Command Language Grammars: a Representation for the User Interface of Interactive Computer Systems", In: International Journal of Man-Machine Studies. v.15, p.3-50.

O'Shaughnessy, D. (2008) "Automatic Speech Recognition: History, Methods and Challenges". In: Pattern Recognition, v.41, p.2965-2979.

Salvador, V.F.M., Oliveira NETO, J.S., Paiva, M.G. (2010) "Evaluating Voice User Interfaces in Ubiquitous Applications". In: Designing Solutions-Based Ubiquitous and Pervasive Computing Editado por Milton Mendes, Pedro Fernandes. New Issues and Trends. Natal, RN: IGI Global.

Shneiderman, B. (2000) "The limits of speech recognition", In: Communications of the ACM, v. 43, n. 9, p. 24-27.

Stehman, P. (1997) "Selecting and Interpreting Measures of Thematic Classification Accuracy", In: Remote Sensing of Environment, v.62, p.77–89.

Zukerman, I., Litman, D. (2001) "Natural language processing and user modeling: synergies and limitations". In: User Modeling and User-Adapted Interaction n. 11, p.129-158, Khuwer Academic Publishers.