Um Sistema para Melhorar a Usabilidade de um Gerenciador de Correio Eletrônico Baseado em Reconhecimento de Fala

Josué Dantas, Rafael Oliveira, Hugo Santos, Nelson Neto e Aldebaro Klautau

[1]Laboratório de Processamento de Sinais – LaPS
Universidade Federal do Pará – UFPA
Rua Augusto Correa, 1 – 660750-110 – Belém, PA, Brasil
http://www.laps.ufpa.br/falabrasil

{josue,rafaelso,nelsonneto,aldebaro}@ufpa.br, hugo.santos@itec.ufpa.br

Resumo. *O tempo investido em lidar com correio eletrônico tem crescido para a maioria dos usuários de computadores. O processamento da fala, tanto o reconhecimento quanto a síntese, pode ser usado para o aumento da produtividade do usuário ou mesmo melhoria da acessibilidade para pessoas com necessidades especiais. Este trabalho descreve o desenvolvimento de uma extensão ao software Mozilla Thunderbird a qual permite não apenas controlar o software através de comandos de voz, mas também converter o texto em fala. O projeto segue a filosofia de software livre e se beneficia dos recentes avanços nas pesquisas em processamento da fala específicas ao português brasileiro. Uma das contribuições do trabalho é melhorar a taxa de reconhecimento do sistema pela incorporação de suporte as gramáticas de comando e controle. Versões anteriores da interface de programação utilizada estavam limitadas a utilizar modelos de linguagem baseados em N-grama, os quais são mais adequados para aplicações como a de ditado.*

Abstract. *The time invested in dealing with email has significantly increased for most computer users. Speech technologies, such as recognition and synthesis, can he used to improve the productivity of user or even accessibility for those who have special needs. This work describes the development of a plugin for the Mozilla Thunderbid that allows not only controlling the software via voice commands, but also to convert text into speech. The project relies on the free software philosophy and benefits from recent advances in speech processing research targeting Brazilian Portuguese. One of the contributions of the work is to significantly improve the system recognition rate by supporting command-and-control grammars. Previous versions of the adopted programming interface were limited to N-gram language models, which are more adequate to applications such as dictation.*

1. Introdução

Um importante passo para a melhoria da interação dos usuários com aplicativos de computador é a possibilidade de comunicação através de linguagem natural falada. Esse desafio é considerado por muitos como um dos mais importantes da computação moderna. Para isso dois importantes processos são requeridos: primeiramente o processo pelo qual o computador interpreta o que o usuário fala, *automatic speech recognition* (ASR) [Taylor 2009], e o segundo, igualmente importante, é a produção de linguagem natural pela

Proceedings of the 8th Brazilian Symposium in Information and Human Language Technology, pages 107–114,
Cuiabá, MT, Brazil, October 24–26, 2011.

máquina através de um sistema *text-to-speech* (TTS) [Huang et al. 2001]. Esse tipo de interação, que há algum tempo era vista como inovação futurística [Canny 2006], atualmente tem amadurecido pelo estabelecimento de soluções para problemas específicos. Por exemplo, o ASR em Português Brasileiro (PB) começa a ser utilizado em projetos de software livre no desenvolvimento de aplicativos que se beneficiam do suporte a *ditado*, como o SpeechOO [Colen e Batista 2010].

Nesta fase, que pode ser considerada embrionária em relação à comunicação usando linguagem natural, um aplicativo que pode se beneficiar de uma interface aural é o que gerencia correio eletrônico. O envio e recebimento de mensagens por meio da Internet é uma atividade que só tem crescido. Estima-se que em abril de 2010 havia cerca de 2,9 bilhões de contas de e-mail [Email Marketing Reports 2011]. Entre os gerenciadores de e-mail enquadrados na categoria de software livre, um dos destaques é o Mozilla Thunderbird, com cerca de 15 milhões de usuários [Thunderbird 2011].

Com empresas e indivíduos fazendo uso do e-mail para propagar serviços, idéias, etc., diariamente os usuários recebam uma quantidade relevante de mensagens. Estudos realizados por [Barley et al. 2010] concluíram que a grande quantidade de e-mails recebidos atualmente, sobretudo acerca de negócios / trabalho, trazem estresse para os usuários pois demandam parte significativa de seu tempo.

O Outlook da Microsoft possibilita a sua manipulação através de comandos de voz que permitem acessar os menu, editar emails, entre outras funcionalidades. Porém , entre as linguagens suportadas (no momento, chinês simplificado, chinês tradicional, inglês americano e japonês) não se encontra o PB. Nota-se que algumas das funcionalidades são permitidas somente para o inglês [Outlook 2011].

Este trabalho descreve o desenvolvimento de um sistema computacional (software), na forma de extensão (*plugin*), que torne mais eficaz a utilização de correio eletrônico (email) a partir do uso de uma interface aural. O desenvolvimento de tal aplicação se baseia na existência de ferramentas que possibilitam reconhecimento e síntese de voz para a língua alvo. Como requisito, a arquitetura do software deve ser flexível, permitindo o uso de diversas linguagens, mas o foco do trabalho será o PB, que é a língua usada para validação do sistema. Para se ter suporte ao PB, são necessários modelos acústicos e de liguagem específicos. Versões desses recursos foram recentemente disponibilizadas pelo grupo FalaBrasil [Neto et al. 2010].

O trabalho encontra-se organizado da seguinte forma. Na Seção 2 estão descritas as ferramentas utilizadas para a execução do projeto. Já a Seção 3 descreve os recursos desenvolvidos. A Seção 4 apresenta o resultado de simulações usando o reconhecimento em modo de gramática livre de contexto e usando o modelo de linguagem N-grama. Por fim, a Seção 5 apresenta a conclusão e sugere pesquisas futuras.

2. Tecnologias Utilizadas

A presente seção descreve as linguagens e a forma como os aplicativos Mozilla são desenvolvidos, e também as ferramentas que permitem o desenvolvimento de aplicativos com interface aural para o PB.

2.1. Recursos usados pela Fundação Mozilla

Para o desenvolvimento de aplicativos Mozilla, dentre outras, faz-se necessário a manipulação de duas linguagens: XML *Interface User Language* (XUL) e o javascript. O XUL é uma linguagem de marcação desenvolvida pela Mozilla para desenvolvimento do layout de suas aplicações, sendo que uma *engine*, o Gecko, é utilizada para interpretar as *tags* e gerar uma interface ao usuário. Para prover funcionalidade aos elementos desenvolvidos a partir do código XUL utiliza-se o javascript, por permitir a construção de aplicativos mais robustos, além de possibilitar a chamada de métodos nativos de outras linguagens, tal como Java neste caso, conforme ilustrado na Figura 1. Dentre as formas de realizar essa integração destaca-se o Liveconnect por causa da praticidade: o mesmo não necessita da utilização de aplicativos para geração de interfaces e outros arquivos para a comunicação entre os códigos.

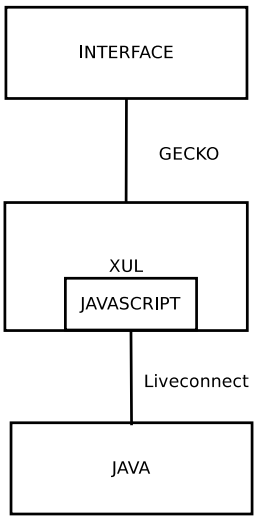

Figura 1. Tecnologias utilizadas para o desenvolvimento do Thunderbird

O Thunderbird, bem como outros aplicativos da Mozilla, é desenvolvido de forma a facilitar a construção de extensões, pois este se constitui de sobreposições das páginas utilizadas na aplicação. Assim, pode-se criar uma sobreposição para determinada página facilmente, basta somente especificar qual página deseja-se sobrescrever no arquivo chrome.manifest, de forma que a aplicação poderá retornar ao estado anterior, sendo necessário para isto remover os módulos adicionados.

Para saber quais páginas devem ser sobrepostas é importante a utilização de um plugin, que neste caso foi adicionado ao Thunderbird, chamado Chrome List, o qual permite a exploração das pastas que contém as páginas que podem ser manipuladas. É também importante a utilização do DOM inspector, outro plugin que inspeciona a dinâmica do aplicativo de modo a facilitar o entendimento dos eventos invocados a partir da interação com usuário, auxiliando na identificação dos eventos alvo para a construção do plugin.

2.2. Reconhecimento de Voz

Em relação a algumas outras línguas como a inglesa, não existem muitas ferramentas que possibilitem a adição do módulo de ASR e TTS aos softwares para Português Brasileiro

(PB). Segundo [Oliveira et al. 2011], mesmo soluções comerciais como o Dragon e IBM Via Voice não incluem o PB. A Microsoft recentemente disponibilizou uma versão de seu kit de desenvolvimento baseado na *Speech Application Programming Interface* (SAPI), o qual permite a construção de aplicativos para o PB e outras 25 linguagens, com suporte para síntese e reconhecimento [SAPI 2011].

Em [Silva et al. 2010] tem-se a descrição do Coruja, o qual consiste em uma API que utiliza o Julius, descrito em [Lee et al. 2001]. O Coruja é um engine ASR completo integrado à plataforma .NET, possibilitando o desenvolvimento de aplicativos com ASR em PB para o Windows. O objetivo dos autores do Coruja é prover ferramentas que permitam o desenvolvimento de aplicativos aurais, sem que isso dependa de plataforma. Por isso o Coruja foi expandido como a adição do JLaPSAPI, conforme a Figura 2. A JLaPSAPI permite a utilização da *Java Speech Application Programming Interface* (JSAPI) [JSAPI 2011], possibilitando usar-se a importante característica de portabilidade que Java possui.

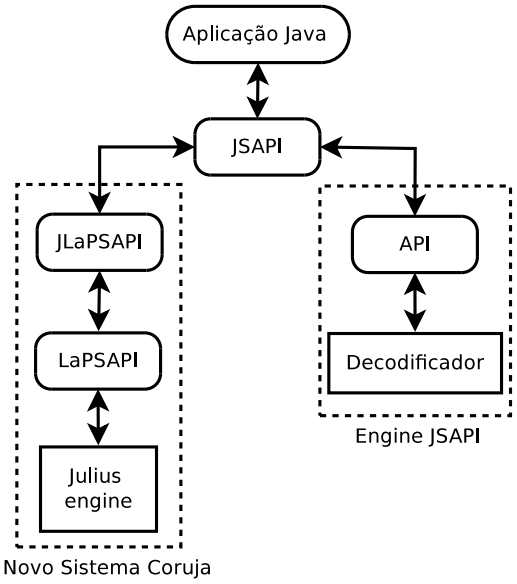

Figura 2. Nova arquitetura do sistema Coruja com suporte a JSAPI

3. Recursos Desenvolvidos

Nas seções seguintes serão descritos os recursos que já foram desenvolvidos na execução desse projeto.

3.1. LaPSMailBenchMark

Com o intuito de obter uma boa avaliação de desempenho e possibilitar a comparação de resultados com outros grupos de pesquisas, foi construído o corpus de audio LaPSMail-BenchMark. Buscou-se criar um corpus de referência com características mais próximas da operação de um sistema ASR em ambientes ruidosos.

Para a construção do corpus LaPSMailBenchMark foram utilizadas as sentenças que proporcionam o controle do aplicativo, bem como nomes próprios que representam contatos de um usuário de email. Atualmente, o corpus possui 25 locutores (homens

e mulheres) com 89 frases cada, o que corresponde a 85 minutos de áudio. A taxa de amostragem utilizada foi de 16.000 Hz e cada amostra foi representada com 16 bits. Como mencionado, o ambiente não foi controlado, existindo a presença de ruído de escritório nas gravações.

3.2. A extensão para suporte a reconhecimento de voz

No atual estágio do projeto deu-se prioridade à ASR, em detrimento de TTS. A extensão proposta usa as sobreposições para acionar eventos do Thunderbird através da voz, sendo priorizado o PB. O desenvolvimento desta extensão usou as ferramentas para o reconhecimento de voz para o PB citadas, dentre as quais destaca-se a JLaPSAPI, que é uma interface entre a JSAPI e o Julius. A utilização destas ferramentas permitiu a construção de um arquivo .jar que recebe um sinal de voz pelo microfone e retorna texto usando ASR.

As *engines* de reconhecimento de voz trabalham tipicamente em dois modos, o ditado e o de gramática. No primeiro, um modelo de linguagem probabilístico baseado em N-gramas vai buscando a melhor hipótese levando em conta todas as palavras existentes no dicionário. Já no modo de gramática, há uma restrição na quantidade de palavras que podem ser retornadas, tendo em vista o fornecimento de um arquivo que restringe a quantidade de palavras a serem comparadas. Seguiu-se o tutorial disponível em [JLaPSAPI 2011] para o desenvolvimento do reconhecedor.

Um passo após a criação do reconhecedor foi a integração do código em java com a extensão, usando o Liveconnect. Para o uso do Liveconnect uma boa referência é encontrada em [Liveconnect 2011], sendo que um exemplo é disponibilizado para download. Este apresenta o código que elucida a chamada do código em Java. Na Figura 3 pode-se visualizar a integração existente entre os módulos desenvolvidos e as outras tecnologias.

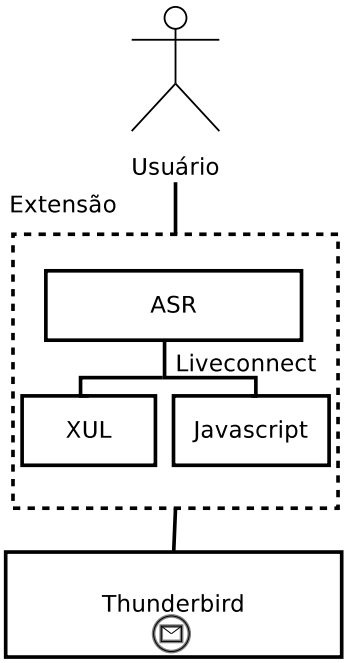

Figura 3. Esquema da interação entre o usuário e o Thunderbird usando a extensão proposta.

Através do Chrome list e do DOM Inspector foi possível identificar quais páginas deve-se sobrescrever, sendo estas a messenger.xul, msgAccountCentral.xul e a messengercompose.xul. A messenger é basicamente a página principal do Thunderbird. A msgAccountCentral é a página que é apresentada quando o usuário abre o aplicativo, desde que este já tenha armazenado as informações de sua conta de e-mail. Já a messengercompose é a tela que possibilita a criação de novas mensagens. Cada página desta deve ser informada no arquivo chrome.manifest juntamente com a página que irá sobrescrevê-la, conforme a linha abaixo: overlay chrome://messenger/content/messenger.xul chrome://speech/content/messengerOverlay.xul

3.3. Gramática

Como citado, a *engine* de reconhecimento usada é o Julius. Para ele construiu-se uma gramática conforme especificado em [Jgram 2009], sendo esta composta por dois arquivos o .grammar, que define basicamente as regras que compõem a gramática e o .voca, que por sua vez apresenta quais palavras compõem cada regra.

Para exemplificar como os arquivos são construídos abaixo lista-se uma parte do .grammar, que especifica a gramática.

S : NS_B SENT NS_E

SENT : ACTION

O S é a sentença que será reconhecida, sendo ela formada por uma ou mais regras. Neste caso é formada por um silêncio no início e no fim e entre eles uma outra regra é definida, no caso a regra SENT que recebe outra regra a ACTION. Assim, a regra ACTION é atribuída à regra SENT, ou seja, as mesmas palavras farão parte de ambas.

No arquivo de vocabulário, .voca, cada regra apresenta as palavras que a constituem e que por sua vez deverão ser retornadas como resultado do reconhecimento. Além disso, ao lado de cada palavra deve aparecer sua respectiva representação fonética. Para isso foi utilizado um conversor grafema para fonema conforme descrito em [Siravenha et al. 2008]. Alguns exemplos do vocabulário são listados a seguir:

```
% NS_B
<s>   sil
% NS_E
</s>  sil
% ACTION
ler    l e X
deletar    d e l e t a X
fechar    f e S  a X
sair    s a j X
descartar    d e s k a X t a X
procurar    p r o k u r a X
gravar    g r a v a X
pára  p a r a
```

4. Resultados Experimentais

Nos experimentos realizados, a base de dados LaPSMailBenchMark foi utilizada para as simulações. Estas ocorreram nos dois modos: gramática e ditado. Para o modo de

gramática utilizou-se os arquivos conforme descrito na seção anterior.

A Tabela 1 mostra os valores da *Word Error Rate* (WER), que é a taxa de erro por palavra. Fica evidente que a implementação do modo de gramática possibilitou a construção de um sistema mais eficiente, pois apresenta uma taxa muito menor. Além disso, o modo de gramática também apresenta um valor menor de *real-time factor* (xRT), que consiste na razão entre o tempo investido no reconhecimento e a duração da respectiva sentença). Isso ocorre pelo fato do decodificar possuir um número menor de possibilidades de palavras a considerar no momento da busca pela sentença correta.

Tabela 1. Comparativo entre os modos de gramática e modelo de linguagem

Modo	WER	xRT
Gramática	5.76%	0.49
Modelo de Linguagem	64.6%	0.91

5. Conclusões e Trabalhos Futuros

Os testes mostraram que a aplicação deve ser desenvolvida usando uma gramática controlada, pois a taxa de erro é muito menor em relação ao modo de ditado, além de possuir ainda menor tempo de resposta.

Atualmente, a JLaPSAPI trabalha somente em modo ditado, mas a implementação das funcionalidades que permitem a inclusão de uma gramática estão em andamento. Um objetivo foi dar suporte a arquivos no formato reconhecido pela JSAPI, que utilizam o *Java Speech Format Grammar* (JSFG), e então a API fará a conversão deste padrão para o utilizado pelo Julius. Esta fase encontra-se concluída e permite processar um arquivo no formato da JSAPI, e gerar então um arquivo .voca e .grammar. Agora, através da Java Native Interface (JNI) deve-se invocar os métodos do Julius, que é escrito em C, que por sua vez permite o reconhecimento usando uma gramática livre de contexto.

A fase final do trabalho são testes sistemáticos para verificar a usabilidade do software, de forma a compreender quantitativamente o eventual ganho de produtividade do usuário ao adicionar a extensão em seu gerenciador de emails Thunderbird.

Referências

Stephen Barley, Debra Meyerson e Stine Grodal (2010). E-mail as source and symbol of stress. *Articles in Advance*.

John Canny (2006). The future of human-computer interaction. *Queue - HCI*, 4:24–36.

W. D. Colen e P. Batista (2010). Veja mamãe, sem as mãos! SpeechOO, uma extensão de ditado para o BrOffice.org. *11th Fórum Internacional Software Livre*.

Email Marketing Reports (2011). http://www.email-marketing-reports.com/metrics/email-statistics.htm.

X. Huang, A. Acero e H. Hon (2001). *Spoken Language Processing*. Prentice-Hall.

Jgram (2009). http://julius.sourceforge.jp/en_index.php?q=en_grammar.html.

JLaPSAPI (2011). http://www.laps.ufpa.br/falabrasil/jlapsapi.

JSAPI (2011). java.sun.com/products/java-media/speech/.

Akinobu Lee, Tatsuya Kawahara e Kiyoshiro Shikano (2001). Julius - an open source real-time large vocabulary recognition engine. *Proc. European Conference on Speech Communication and Technology*, páginas 1691–1694.

Liveconnect (2011). https://developer.mozilla.org/en/java_in_firefox_ extensions.

Nelson Neto, Carlos Patrick, Aldebaro Klautau e Isabel Trancoso (2010). Free tools and resources for brazilian portuguese speech recognition. *The Brazilian Computer Society*, 16:53–68.

Rafael Oliveira, Pedro Batista, Nelson Neto e Aldebaro Klautau (2011). Recursos para desenvolvimento de aplicativos com suporte a reconhecimento de voz para desktop e sistemas embarcados. *12º Fórum Internacional de Software Livre*.

Outlook (2011). http://office.microsoft.com/pt-pt/outlook-help/acerca-do-reconhecimento-de-voz-hp003084099.aspx.

SAPI (2011). www.microsoft.com/speech/.

Patrick Silva, Pedro Batista, Nelson Neto e Aldebaro Klautau (2010). An open-source speech recognizer for Brazilian Portuguese with a windows programming interface. *The International Conference on Computational Processing of Portuguese (PROPOR)*.

Ana Siravenha, Nelson Neto, Valquíria Macedo e Aldebaro Klautau (2008). Uso de regras fonológicas com determinação de vogal técnica para conversão grafema-fone em português brasileiro. *7th International Information and Telecommunication Technologies Symposium*.

Paul Taylor (2009). *Text-To-Speech Synthesis*. Cambridge University Press.

Thunderbird (2011). http://www.spreadthunderbird.com/content/ thunderbird-3-faq.