UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

INTERPRETAÇÃO DE FALA APLICADA A INTERAÇÃO COM UM SISTEMA DE *STORYTELLING*

TRABALHO DE GRADUAÇÃO

Luiz José Schirmer Silva

Santa Maria, RS, Brasil

2014

INTERPRETAÇÃO DE FALA APLICADA A INTERAÇÃO COM UM SISTEMA DE *STORYTELLING*

por

Luiz José Schirmer Silva

Trabalho de Graduação apresentado ao Curso de Ciência da Computação
da Universidade Federal de Santa Maria (UFSM, RS), como requisito
parcial para a obtenção do grau de
Bacharel em Ciência da Computação

Orientador: Prof. Dr. Cesar Tadeu Pozzer

Co-orientador: Profa. Dra. Lisandra Manzoni Fontoura

Trabalho de Graduação N. 371

Santa Maria, RS, Brasil

2014

Universidade Federal de Santa Maria
Centro de Tecnologia
Curso de Ciência da Computação

A Comissão Examinadora, abaixo assinada,
aprova o Trabalho de Graduação

INTERPRETAÇÃO DE FALA APLICADA A INTERAÇÃO COM UM SISTEMA DE *STORYTELLING*

elaborado por
Luiz José Schirmer Silva

como requisito parcial para obtenção do grau de
Bacharel em Ciência da Computação

COMISSÃO EXAMINADORA:

Lisandra Fontoura
Prof. Dr Lisandra Manzoni Fontoura
(Presidente/Co-orientador)

Prof. Dr. Ana Trindade Winck(UFSM)

Juliana Vizzotto
Prof. Dr. Juliana Kaiser Vizzotto(UFSM)

Santa Maria, 22 de Janeiro de 2014.

AGRADECIMENTOS

Dedico inicialmente meus agradecimentos ao Professor Cesar Pozzer, pela orientação neste trabalho e durante o tempo o qual trabalhamos no Laboratório de Computação Aplicada.

À minha família por todo apoio e incentivo dado, especialmente a meus pais José Luiz e Gilka. Talvez no início tenham ficado confusos em saber que eu iria desistir de ser engenheiro para ingressar no curso de Ciência da Computação, mas nunca desistiram de me apoiar. Muito obrigado a vocês.

À minha namorada Letícia pelo carinho e compreensão. Além disso, sempre me apoiou em todas as minhas ideias por mais malucas que algumas delas possam parecer. Obrigado por tudo, meu amor.

Aos meus amigos e colegas de Laboratório, Schardong, Netto, Mizdal, Diego e Alex pela parceria nos trabalhos, projetos e nas jogatinas.

Às minhas amigas desde a época de escola Aline, Lidi e Laisla. Obrigado pelo apoio e pela parceria.

Por fim gostaria de dedicar também este trabalho a Luiz Eduardo Viegas Flores, o Dudu, amigo e colega de Laboratório que infelizmente veio a falecer nos eventos da boate Kiss.

RESUMO

Trabalho de Graduação
Curso de Ciência da Computação
Universidade Federal de Santa Maria

INTERPRETAÇÃO DE FALA APLICADA A INTERAÇÃO COM UM SISTEMA DE *STORYTELLING*

Autor: Luiz José Schirmer Silva
Orientador: Prof. Dr. Cesar Tadeu Pozzer
Co-orientador: Profa. Dra. Lisandra Manzoni Fontoura
Local e data da defesa: Santa Maria, 22 de Janeiro de 2014.

Atualmente existe uma tendência na qual diferentes sistemas de mídia para entretenimento acabem convergindo, ou seja, adquirindo características comuns entre eles. No caso de *Storytelling*, para a geração de uma história onde o usuário possa interagir de forma adequada, influenciando o desfecho de uma narrativa, este sistema acaba incorporando características que são comuns a jogos digitais dentre outros meios de entretenimento. A interação com o usuário pode se dar através de eventos, assim como em *games* comuns, ou através do processamento de comandos de voz, porém a maioria destes sistemas faz uso de interfaces de usuário pouco atrativas ou fazem uso de comandos de voz específicos. Este trabalho propõe o desenvolvimento de um sistema para interação com sistemas de *storytelling* baseado em processamento de linguagem natural, isto é, um modelo de interação com o usuário o qual utiliza mecanismos de interpretação da fala para interação com a narrativa, onde não é necessário o conhecimento prévio de comandos. O sistema busca interpretar a requisição do usuário baseando-se no contexto da história. Foi implementada uma interface de comunicação entre o usuário e um sistema de *storytelling* que faz uso de gerenciador de enredos IPG, a qual esta é capaz de interpretar comandos de voz enviados pelo usuário de forma natural. Para isso, os comandos dados são transformados em textos e posteriormente processados a fim de gerar sentenças em lógica de primeira ordem. Estas sentenças serão analisadas para gerar eventos em uma história pré-definida. Os eventos criados serão usados para gerar versões alternativas de uma mesma história.

Palavras-chave: Storytelling; Linguagem Natural; Comandos de Voz.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE CÓDIGOS

LISTA DE ABREVIATURAS E SIGLAS

PLN	Processamento de linguagem natural
LaCA	Laboratório de Computação Aplicada
API	Application Programming Interface
IPG	Interactive Plot Generator

SUMÁRIO

1 INTRODUÇÃO

No desenvolvimento de jogos digitais é visível a constante evolução no que se diz respeito a gráficos 3D e uma simulação física apurada. Além disso, há um grande investimento em pesquisas relacionadas a entretenimento digital englobando os mais variados tipos de mídias. Com o advento dos sistemas de televisão digital, muitas características encontradas atualmente em jogos estão sendo empregadas em ambientes onde o espectador não é considerado apenas um agente passivo, mas este mesmo é capaz de interagir com o conteúdo televisivo.

Segundo (FORMAN, 2000), existe uma grande tendência na qual é buscada a convergência entre os principais tipos de entretenimento digital, sejam eles TV, *games* ou cinema, onde é possível que diferentes agentes interajam e gerem experiências colaborativas. Neste contexto, novos jogos e sistemas para televisão digital têm incorporado técnicas que possibilitem que o jogador ou telespectador interaja com o sistema de uma forma aparentemente mais natural. Por exemplo, a possibilidade do usuário poder utilizar a fala para influenciar eventos definidos na narrativa de uma história.

Lima et. al. (2012) propôs um mecanismo que define como os usuários irão interagir com um sistema de *Storytelling*, para televisão ou cinema, utilizando linguagem natural. Para que esta interação possa ser atrativa ao público, ela deve ser considerada simples, intuitiva e principalmente transparente, isto é, o usuário não precisa saber comandos específicos para interagir com a plataforma e, por exemplo, em sistemas de televisão, não devem existir janelas ou qualquer outro tipo de interface que seja capaz de atrapalhar a exibição da história na tela.

Baseando-se nestas premissas, este trabalho apresenta uma implementação de um sistema que utiliza interpretação de fala aplicada à narrativa interativa de histórias (*storytelling*). Em *storytelling*, o usuário deixa de ser um espectador passivo, onde ele tem a

possibilidade de interagir com uma história a fim de guiar ações dos agentes inseridos nela. Sendo assim, é possível criar diferentes versões de uma mesma história. Este trabalho visa à implementação de um sistema que considera os atributos necessários para um sistema transparente. Ela deverá combinar o uso de dispositivos, como um microfone comum, para a captação de áudio. A finalidade desta abordagem é processar as entradas do usuário e transformá-las em texto. Cada entrada em texto é processada e transformada em sentenças de lógica de primeira ordem. Posteriormente elas serão usadas como base para entradas no IPG (*Interactive Plot Generator*) (CIARLINI, 1999), que é um módulo de geração de enredos baseado na linguagem Prolog.

1.1 Objetivos do Trabalho

O presente trabalho tem por objetivo o estudo e desenvolvimento de um *software* para interação com o usuário em um sistema para *storytelling* interativo. O sistema final deve fazer uso da captura da fala do mesmo. Sendo assim, de acordo com os comandos enviados é possível gerar diferentes versões de uma mesma história. Para isso, foram estudados métodos baseados em Processamento de Linguagem Natural(PLN), incluindo o *Stanford parser*.Ele é utilizado para a análise das relações de dependência entre os termos considerados relevantes, ou seja, uma análise das frases geradas pelo usuário. Além disso, um módulo para a geração do enredo em histórias interativas (IPG) e plataformas para o processamento de áudio, como o *Microsoft Speech*, também foram utilizados.

Sendo assim, os objetivos específicos propostos pelo trabalho são:

- Estudo e implementação de técnicas para a captura e interpretação de comandos de voz com a *Microsoft Speech API*;
- Estudo de ténicas para PLN a fim de manipular apropriadamente comandos transformados em textos;
- Integração da camada de interpretação com o módulo para a geração de enredos (IPG);
- Avaliação dos resultados obtidos com o protótipo.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo tem o objetivo de fazer a apresentação dos conceitos teóricos envolvidos no projeto. Foram consideradas técnicas para a geração de conteúdo interativo relacionadas à *Storytelling* e as APIs envolvidas no processamento de comandos de voz realizados pelo usuário.

2.1 A Televisão Interativa e os Sistemas para *Streaming*

Os velhos sistemas de televisão, os quais utilizam modelos de transmissão analógica, vêm sendo substituídos gradualmente por uma moderna implementação de sistemas digitais que englobam desde imagens em alta definição à informações disponibilizadas em tempo real sobre a programação dos canais de TV. A televisão como meio de entretenimento atualmente acaba por agregar diferentes características de mídias anteriormente consideradas distintas a ela como, por exemplo, jogos eletrônicos.

Segundo Pozzer (2005), a interatividade é sem dúvida a mais beneficiada com o avanço da tecnologia para este meio de entretenimento, onde a utilização de novos *hardwares* para decodificação e transmissão, além das atuais televisões digitais, permite que os usuários interajam com aplicativos que executam localmente. Desta forma, conteúdos voltados especificamente para *vídeo games* ou computadores podem ser disponibilizados através de televisores.

A televisão que era um meio de entretenimento passivo, está começando a dar ao telespectador a possibilidade de escolha perante não apenas ao conteúdo o qual deseja assistir, mas também a forma como ele quer assistí-la. Hoje o telespectador não se limita a apenas escolher seus programas favoritos, mas é possível obter informações adicionais sobre os mesmos, além de gravá-los e em alguns casos interagir com o conteúdo.

No Brasil, este sistema de televisão interativa ainda encontra-se limitado e restringe-

se a exibição de informações adicionais sobre a programação ou a anuncios, porém no futuro será possível utilizar este meio de interatividade para compra de produtos e personalização de conteúdo, por exemplo. Para dar suporte aos sistemas de interatividade na televisão, o Governo Federal estipilou a obrigatoriedade da implantação do *middleware* Ginga (MINISTÉRIO DAS COMUNICAÇÕES, 2013) na produção de televisores de alta definição nacionais. O Ginga é uma especificação de *software* aberta(*open source*) criada por instituições brasileiras que atua como uma camada intermediária de *software*, permitindo que aplicativos transmitidos pelas emissoras de TV sejam executados em diferentes televisores. Conforme esta tecnologia evolui será possível criar histórias e programas para televisivisão onde o telespectador poderá decidir o desfecho de uma história, sendo assim, agindo ativamente, tendo a comunicação com o sistema sendo feito juntamente com o sinal de TV.

Paralelamente ao crescimento de técnicas de interatividade para televisão digital, outro sistema em ascensão é o de jogos e filmes por *streaming*. A possibilidade de processar informações em um servidor na nuvem permite que qualquer dispositivo com acesso a internet, mesmo que com uma configuração modesta, exiba, por exemplo, jogos com gráficos de alto desempenho e filmes em HD de forma eficiente. Este sistema apenas interpreta comandos do usuário e os processa na nuvem retornando em vídeo a execução da aplicação desejada. Apesar de sofrer ainda algumas limitações como, por exemplo, a largura de banda para a conexão, existem casos em que esta implementação funciona de forma eficiente. Um destes exemplos é o sistema de jogos Gaikai (SONY COMPUTER ENTERTAINMENT, 2013), pois é possível acessá-lo através de *smart TVs* ou outros dispositivos com uma configuração simples, assim não limitando-se a um *Hardware* específico. Sendo assim, este modelo de interação através de *streaming* de vídeo mostra-se um meio simples de implementação para um sistema de *Storytelling* quando comparado ao modelo para TV digital.

Estes dois sistemas apresentados podem no futuro contribuir para o desenvolvimento de aplicações voltadas para narrativas interativas, pois é possível desenvolver aplicativos em que filmes ou séries possam deixar de ser lineares e passivos e forçarem o usuários a interagirem com a história influenciando em seu desfecho.

2.2 *Storytelling* Interativo

No que se diz respeito à narração de histórias, *Storytelling* é um modelo de entretenimento digital que faz uso de técnicas e ferramentas que permitem criar e visualisar histórias interativas guiadas por um meio computacional. *Storytelling* pode ser considerado como um sistema que engloba variadas técnicas de conhecimento para o processamento e elaboração de uma história (MATEAS, 1997).

Agentes autônomos que possuem caracterísiticas como personalidade própria, definição de eventos em que as decisões do usuário influenciam no andamento de um história, personagens virtuais, interfaces, dentre outras características comuns a *games* derivam de pesquisas relacionadas a *Storytelling* (SPIERLING et al., 2002). Nota-se que este paradigma concentra um vasto campo de pesquisas, onde em sua grande maioria podem ser relacionadas à técnicas de Inteligência Artificial, mídias digitais e desenvolvimento de jogos. As linhas de pesquisa se dividem normalmente em três: geração , interação e visualização de histórias.

Para a geração de enredos em *Storytelling*, o sistema deve agir sobre algum tipo de dado para representar a história, onde estes dados são um conjunto de regras, cenários e eventos. Um dos grandes desafios para a criação dinâmica de enredos é tornar a trama coerente. Devido a isso, grande parte dos sistemas desenvolvidos depende fortemente de um modelo de história criado pelo programador, ou seja, o sistema já é programado contendo uma história definida. O usuário apenas decide qual o rumo que ela deve tomar de acordo com os eventos acontecidos até então, criando assim, a possibilidade de existir dois ou mais desfechos para a mesma história. Apenas a forma como a história se apresenta ao usuário é de modo interativo, já que somente o autor do sistema tem o poder de alterar os elementos que definem a narrativa. Como dito anteriormente, o usuário limita-se a fornecer parâmetros que serão usados para direcionar a história (SPIERLING et al., 2002).

No que diz respeito à interação, existem vários trabalhos relacionados ao modo de interação com narrativas de histórias, onde a maioria dos projetos encontrados faz uso de interfaces gráficas (GUI) tradicionais (CIARLINI et al., 2005). Para construir um modelo de sistema mais atrativo para o usuário final, pode-se considerar o processamento da fala do mesmo como entrada e transformá-la em texto. Desta forma, não o sistema não limita-se a comandos específicos nem a interfaces que poderiam até comprometer a linguagem

visual da história gerada.

O primeiro sistema a utilizar em larga escala o processamento de linguagem natural foi *Façade* (MATEAS; STERN, 2004), o qual aceita comandos enviados através de texto pelo usuário e processa qual a reação que os personagens devem tomar. O usuário pode criticar atitudes de algum dos personagens digitando frases como, por exemplo, ”ele está mentindo”. O sistema reconhece esta sentença como uma forma de crítica e define qual a reação que o personagem terá ao receber uma crítica. Existem jogos que incorporam caraterísticas semelhantes a destes sistemas, como é o caso de *The Stanley Parable* (GALACTIC CAFE, 2012), no qual a história do jogo é narrada de acordo com as ações realizadas pelo jogador. Este jogo é baseado em ações e respostas através de instruções dadas pelo próprio narrador, porém o jogador não é obrigado a segui-las, o que faz com que a narrativa do *game* mude conforme a experiência do usuário. É como jogar uma história de um livro, na qual o destino do protagonista é influenciado pelas decisões do usuário. O ambiente destes dois sistemas citados pode ser visto na figura 2.1.

(a) (b)

Figura 2.1: a)Facade; b)The Stanley Parable

Existem ainda protótipos para *Storytelling* os quais "fogem"do escopo clássico de interação com filmes ou jogos. No projeto *Paper and Pencil* (LIMA et al., 2012) o usuário é capaz de interagir com uma história através de técnicas de realiadade aumentada, onde através de marcadores em um folha de papel e uma câmera comum, são dispostos os cenários e os personagens no ambiente o qual a câmera está filmando. O sistema permite aos usuários interagir livremente com os personagens virtuais, através de objetos desenhados no papel. O usuário poder afetar indiratamente a história, até chegar ao ponto de seu desfecho ser radicalmente modificado. O sistema pode ser visto conforme a figura 2.2.

Figura 2.2: *Storytelling* e Realidade Aumentada em *Paper and Pencil* (LIMA et al., 2012)

Nas próximas seções deste trabalho será detalhado um modelo de arquitetura para o Processamento de Linguagem Natural aplicada a *Storytelling Interativo*, o qual diferencia-se de alguns modelos tradicionais por não fazer uso de uma interface do usuário (GUI) tradicional, a qual necessita de comandos específicos mas que interprete comandos de voz do usuário.

2.3 O IPG (*Interactive Plot Generator*)

O estudo sobre a geração automática de histórias iniciou através das pesquisas de Vladimir Propp (PROPP, 1968) que observou um padrão na ocorrência de eventos e no encadeamento destes eventos em contos russos. Segundo Pozzer (2005), em geral, os eventos de uma história, não ocorrem aleatoriamente. Eles ocorrem em vista de objetivos os quais fazem os personagens agirem conforme alguma situação. Através da análise de diferentes contos e histórias foi possível constatar que muitas delas seguiam um padrão com eventos e personagens bem definidos onde, por exemplo, definem-se facilmente os papéis de herói e vilão em uma história.

Sendo assim, uma história resulta da interação competitiva ou cooperativa dos personagens na busca de seus objetivos. Com esta análise foi possível definir características comuns a estas histórias, onde as ações dos personagens e suas reações poderiam ser analisadas através de sentenças lógicas. Os textos de um determinado gênero literário podem ser caracterizados pela associação de sentenças lógicas a pequenos trechos da narrativa. Estas sentenças são os eventos comuns ao enredo, onde ele é descrito como uma sequência destas ações. O encadeamento entre os eventos deve respeitar uma lógica onde um evento só ocorre de acordo com algumas condições.

O IPG (CIARLINI et al., 2005) é um gerador interativo de histórias baseado no trabalho de Propp, onde foi proposto um modelo formal para a especificação de eventos na

narrativa de histórias através de operações lógicas com pré e pós condições. Ele é implementado em Prolog, estendido por programação em lógica com restrições que, por meio de planejamento e regras de inferência de objetivos, gera conjuntos parcialmente ordenados de eventos (POZZER, 2005). A realização de um enredo leva em conta um processo de simulação onde a situação dos personagens, seus objetivos e as ações que os caracterizam são levadas em conta. O IPG é composto por 2 módulos onde um é responsável pelo planejamento e outro pela inferência de objetivos na narrativa. No planejamento, este módulo é responsável pela ligação entre os eventos de acordo com a lógica dos acontecimentos explicitados por pós e pré-condições. Existem operações realizadas pelo IPG que mudam o estado do mundo e estas descrevem fatos válidos antes e depois da ocorrência de um evento. Uma operação só é processada a partir do momento em que suas pré-condições também são satisfeitas, seja no estado inicial, ou nas pós-condições das operações que as precedem. Neste sentido isto significa que de acordo com os contos analisados por Propp, por exemplo, o herói de uma história só poderia interagir com o vilão caso este já tivesse realizado suas ações que desencadeassem uma reação do herói, como sequestrar outro personagem ou atacar alguém. A tabela 2.1 representa um conjunto de operações baseadas no modelo de Propp, onde a operação 2 só pode ser executada após a 1, e a 7 e 8 após a 5 por exemplo.

0:Init()
1:AbsenceOfYoungerPeople(princess)
2:KidnappingOfaPerson(princess,dragon)
3:CallForHelp(hero,tsar)
4:DepartureOfSeekerHero(hero)
5:FightInAnOpenField(hero,dragon)
6:OutOfEarthReceipt(hero)
7:VictoryInOpenBattle(hero,dragon)
8:Return(hero)
9:Reward(hero)

Tabela 2.1: Ordem das operações.

O processo de simulação é usado para a criação do enredo de determinado gênero, que inicia a partir da descrição dos personagens, o modelo de comportamento de cada e as alternativas que eles possuem para alcançar seus objetivos. O IPG permite que o usuário participe ativamente da construção de histórias através da definição da ordem dos eventos e situações ou pela inserção de novos eventos. Também é possível definir alternativas

para o desfecho de uma história dentro de um conjunto definido de opções. Neste ponto o sistema também avalia as requisições do usuário a fim de validá-las e garantir a integridade da história gerada.

O IPG trabalha através de fatos previamente existentes em sua base de dados. Um fato não pode ser criado durante um processo de simulação de uma história, mas pode ser alterado livremente através de operações, sejam elas dadas pelo usuário. Os fatos são dados através de predicados em Prolog onde são especificados, por exemplo, os papéis de cada personagem, o local onde ele está no momento, dentre outras características. Dado um o contexto de uma história de contos de fadas genéricos, os predicados da base Prolog para este tipo de enredo podem ser vistos através da tabela 2.2 descrita por Pozzer (2005).

Predicados	Descrição
character(CH,KIND)	CH é um personagem do caráter do tipo KIND que pode assumir 2 valores (-1) mal e bom (1)
hero(CH)	CH é um heroi
victim(CH)	CH é uma vítima
villain(CH)	CH é um vilão
place(PL)	PL é um lugar
home(CH,PL)	CH reside em PL
current_place(CH,PL)	CH está no lugar PL
protection(PL,KIND,L)	O lugar PL tem um nível de proteção L do tipo KIND
strength(CH,L)	O personagem CH tem um nível de força L
affection(CH1,CH2,L)	O nível de afeição de CH1 por CH2 é L
alive(CH)	O personagem CH está vivo

Tabela 2.2: Predicados da Base do IPG descritos por Pozzer(2005).

O IPG também possui a especificação das regras de inferência dos objetivos. Estas regras representam os objetivos a serem alcançados pelos personagens e são usadas para a geração de histórias interativas. Cada regra representa um conjunto de fatos os quais cada personagem deseja alcançar, dadas certas situações do enredo. A tabela 2.3 mostra o conjunto de regras utilizadas pelo sistema, dado o contexto das histórias a serem geradas.

Operações	Descrição
go(CH,PL)	CH é um personagem que deve ir até o local PL Geralmente ocorre combinada a outras operações
reduce_protection(PL)	Redução da Proteção do local PL. São reduzidos os números de guardas Cada local possui um nível de proteção associado
attack(CH,PL)	O persongem CH deve atacar o local PL. Atacando o local ele deve matar os guardas reduzindo a protação do local. Para isso existe uma pré-condição de que a natureza do local(boa ou mal) deve ser diferente da do personagem
get_stronger(CH)	O persongem CH recebe poderes que aumentam a sua força
fight(CH1,CH2)	O personagem CH1 e CH2 lutam entre si. Para isso ocorrer, o local da luta deve esta desprotegido e o personagens devem ter naturezas contrárias
kidnap(VIL,VIC)	O personagem vilão sequestra a vítima. Para que isso ocorra a força do vilão deve ser maior do que soma da proteção do local e da força da vítima.
kill(CH1,CH2)	O personagem CH1 mata o CH2. Para isso eles deve ter natureza contrária e um ser mais forte que o outro.
free(HERO,VIC)	O heroi liberta a vítma. Tem como pré-condição a morte do vilão e como pós o aumento da afeição da vítma pelo heroi
Marry(CH1,CH2)	Casamento entre 2 personagens. Para que isso ocorra deve haver o rapto de um deles e o vilão ter sido derrotado por um deles libertando assim o outro.

Tabela 2.3: Regras de Inferência de Objetivos.

Em resumo, a função do IPG é gerar histórias interativas a partir de um determinado contexto de acordo com um conjunto de regras e inferência lógicas, sendo que neste trabalho são considerados contos de fadas genéricos. Além disso, o sistema também permite que o usuário participe ativamente inserindo e removendo eventos na história além de permitir que seja escolhido o desfecho para determinado evento. Neste contexto o IPG busca garantir a integridade da história gerada de acordo com os eventos determinados pelo usuário, verificando suas pré e pós-condições.

Normalmente para a interação com o IPG, uma interface gráfica pode ser implementada para atuar como uma camada intermediária onde esta interpreta os comandos do usuário e os transforma em sentenças lógicas que serão enviadas. No IPG, na base Prolog, são especificados os contextos lógicos dos enredos a serem disponibilizados ao usuário como forma de conteúdo interativo. Além disso, cada enredo é armazenado em um único arquivo que contém informações como os predicados (Personagens, lugares, atributos),

operações e regras de inferência dos objetivos dos personagens.

A interface que realiza a interação com o usuário é chamada Gerenciador de Enredos, a qual foi desenvolvida por Pozzer(2005). Esta camada do sistema garante que os comandos advindos do usuário serão processados e enviados ao IPG, além de permitir também que um módulo para renderização das histórias também possa ser adicionado ao sistema. Este módulo conta com uma interface gráfica para o usuário realizar a escolha de eventos, onde os eventos são representados por caixas retangulares desenhadas na tela e ligados entre si. Estas caixas possuem cores específicas usadas para a orientação do usuário no encadeamento destes eventos na história, onde elas podem possuir sete cores distintas conforme a tabela 2.4.

Cor	Significado
Azul	Representa o ponto de partida para o encadeamento dos eventos
Amarelo claro	Eventos que já estão com a ordenação total concluída
Amarelo Escuro	Eventos onde o usuário pode conectar outros eventos ativos
Verde	Eventos ativos que podem ser conectados ao evento corrente, ou seja, eventos que não violam restrições
Vermelho	Eventos inativos onde não podem ser conectados ao evento corrente pois tem restrições que devem ser executadas antes
Ciano	Eventos usadas para um módulo de renderização da cena. Representao eventos sendo renderizado atualmente e após seu término volta a cor anterior
Verde escuro	Eventos inseridos pelo usuário e ainda não certificados pelo usuário

Tabela 2.4: Cores dos Eventos na Interface descrita por Pozzer (2005).

A inserção de eventos, bem como a ordem deles e a validação deles perante o IPG é feita através de botões e janelas e pode ser vista conforme a figura 2.3.Esta interface necessita da constante interação do usuário para inserção e remoção de eventos na narrativa.

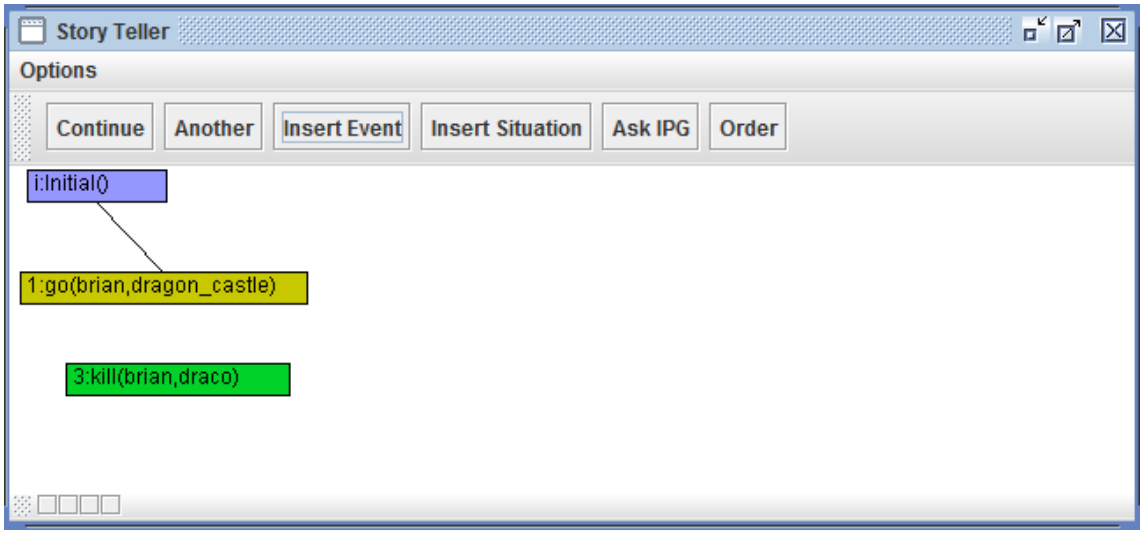

Figura 2.3: Interface principal do gerenciador de enredos criado por Pozzer (2005)

Nota-se que este sistema apresenta alguns problemas com relação a sua utilização para *Storytelling*, pois é necessário fazer uso de janelas para a definição de eventos, além de o usuário ter de interagir com o sistema através de mouse ou teclado. Sendo assim este sistema pode se tornar pouco atraente ao usuário, pois pode atrapalhar o desenvolvimento da história com esta forma de interação "pouco natural". A interface com botões e janelas poderia ser substituída por comandos de voz, o que traria certo dinamismo e naturalidade para a interação como o gerenciador de enredos.

Além do módulo para gerenciar enredos, um módulo para renderizar as histórias pode ser implementado, porém este não será abordado neste trabalho, pois o foco é a interação do usuário com o sistema responsável por gerenciar as histórias.

2.4 Reconhecimento de Voz

O termo reconhecimento de voz pode ser relacionado com diferentes tipos de aplicações, dentre elas, síntese de voz, sistemas para autenticação e, como no caso deste trabalho, a interpretação de linguagem natural. Técnicas para reconhecimento de voz podem ser utilizadas para o processamento de fala humana com o objetivo de extrair algum significado semântico. Um software para reconhecimento de voz pode ser capaz de reconhecer fonemas que são pronunciados pelo usuário a fim de realizar o processamento para reconhecê-las baseando-se em um dicionário contido no sistema (GUILHOTO; ROSA, 2001). Contudo, existem problemas ainda relacionados a esta técnica, pois o sistema pode confundir as palavras que possuem fonemas parecidos, o que justifica o fato de se

restringir o número de palavras ditas devido a este problema.

De forma análoga a isso, ruídos no ambiente também podem influenciar o mau funcionamento de um programa para reconhecimento de voz da mesma forma que no problema anterior, causando certa confusão nas palavras identificadas ou simplesmente não conseguindo interpretá-las. Apesar dos problemas explicitados, existem APIs eficientes para a utilização em sistemas como estes, como é o caso da Microsoft Speech (MICRSOFT, 2013a), a qual faz uso de microfones comuns para a captação do som, possuindo também suporte ao sensor Kinect e além disso possui uma gama abrangente de recursos tanto na identificação de expressões como na síntese de voz.

2.5 Processamento de Linguagem Natural

Há tempos a ideia de se desenvolver sistemas capazes de interpretar a linguagem humana é alvo de pesquisas na área de Inteligência Artificial. Normalmente a área de Processamento de Linguagem Natural (PLN) trabalha com problemas os quais envolvem a interpretação da linguagem humana em tradução de textos, recuperação de informação, criação de dicionários, análise semântica, processamento de corpora (coleções de textos), dentre outros. Um sistema de PLN normalmente é analisado segundo o conhecimento morfológico, sintático e semântico. O conhecimento morfológico diz respeito a como as palavras são construídas incluindo seus radicais, sufixos e as diferentes variantes. Análise sintática diz respeito a como as palavras relacionam-se entre si, ou seja, a interpretação de substantivos, verbos, adjetivos, dentre outras classes de palavras (JURAFSKY; MARTIN, 2000). Possuindo-se uma classificação dos diferentes tipos de palavras, a análise sintática é feita utilizando-se um *parser* para verificar a posição de cada palavra em uma frase. A sintaxe da linguagem define a posição delas para que devam estar organizadas de forma correta em uma frase, sendo a validação, tarefa da análise sintática. Por fim a análise semântica diz respeito a um processo de mapeamento das sentenças a fim de extrair o seu significado, baseando-se na análise sintática (ABRAHÃO; LIMA, 1996).

Todo este processo para interpretar uma frase inicia-se através da identificação dos tipos de palavras, por exemplo considerando a frase ”Brian should kill Draco”, esta poderia ser classificada através de um *parser* da seguinte forma: Brian/Subst, should/auxiliar, kill/verbo, Draco/objeto. Uma abordagem comum para implementação de *parsers* sintáticos em frases é a probabilística. Esta implementação utiliza cadeias de Markov para iden-

tificar a provável sequência de palavras e, dentro deste contexto, a técnica que tem sido mais utilizada em *parsers* sintáticos probabilísticos é a de n-gramas. Esta técnica consiste em estabelecer uma estatística a cada n palavras, onde normalmente implementa-se um n= 3, sendo que são analisadas 2 palavras para a previsão de uma terceira e estabelecendo assim uma estatística para a construção de frases (MÜLLER, 2006).

Desta forma, é possível descobrir qual é a palavra da combinação mais provável, permitindo a verificação e correção da construção das frases. Porém, o processo de identificação das classes de palavras não é suficiente para a análise sintática, pois existem situações onde dependendo da posição de uma palavra em uma frase ela pode assumir funções diferentes. Por exemplo, "como"pode ser considerado um verbo ou uma conjunção. É preciso realizar uma análise da construção gramatical, onde a linguagem é modelada por gramáticas livres de contexto que divide a frase em sintagmas nominal e verbal, e assim é possível verificar qual classe se encaixa em cada sintagma. Em outras palavras, constrói-se uma árvore representando a estrutura da frase e as relações de dependências entre os termos. O *parser* pode gerar diferentes árvores com diferentes combinações de termos, porém a estrutura das árvores geradas, é comparada com a estrutura da frase, utilizando técnicas de busca em árvore para determinar a correção da construção das frases. Para a sentença ”*Brian Should Kill Draco*”, por exemplo temos a árvore da Fig 2, representado a sua estrutura.

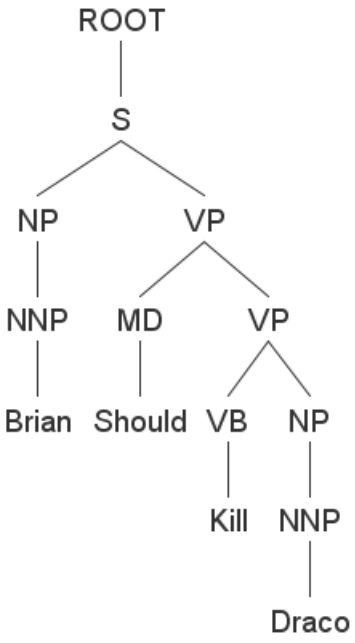

Figura 2.4: Ilustração da árvore sintática de ”*Brian Should Kill Draco*”.

Nota-se que esta subárea de conhecimento da Inteligência Artificial pode facilmente estar relacionada às mais variadas aplicações como, por exemplo, sistemas para banco de dados, robótica e a geração de conteúdo interativo. Com relação a isso, uma abordagem a qual faz do uso da PLN baseada em recomendação de conteúdos para televisão interativa foi sugerida em um trabalho realizado anteriormente (SCHARDONG et al., 2013). Isso é, para aqueles programas que sejam potencialmente atrativos para um determinado telespectador, é possível construir um trailer com as cenas que tendem a chamar mais atenção do espectador para o qual se está direcionando a recomendação. Ao invés de utilizar a interpretação de comandos, este sistema faz uso de técnicas de agrupamento para dados de legendas de séries televisivas.

Este modelo busca encontrar termos relevantes para uma determinada série e, posteriormente, retorna os resultados ao usuário de acordo com os termos buscados por ele. Não é necessário que o usuário seja forçado a realizar buscas relacionadas a estilos de programas específicos. A busca é relacionada a termos e frases, onde as palavras que são consideradas relevantes são processadas de modo que dado um conjunto de legendas para determinada série televisa, os termos avaliados podem ou não ser associadas a este conteúdo (SCHARDONG et al., 2013).Esta proposta mostrou-se eficiente para recomendação de conteúdo, porém ao ser analisada para o ponto de vista de Storytelling, ela não pode ser adaptada. Este modelo não interpreta as sentenças enviadas pelo usuário, mas sim apenas compara a ocorrência delas em um conjunto de textos. Devido a isso, para a interação com Storytelling, uma análise sintática e semântica se faz necessária.

3 METODOLOGIA

O trabalho desenvolvido possui um caráter experimental e exploratório, pois inicialmente foram analisadas técnicas para a solução do problema relacionado a reconhecimento de voz e como extrair o significado das entradas do usuário. Além disso, o caráter experimental se dá ao fato de o sistema ter sido testado com diferentes tipos de dispositivos multimídia para a captura de áudio e de utilizar um modelo de interação alternativo quando comparado aos modelos tradicionais. Este sistema pode ainda ser considerado genérico, pois ele cria a possiblidade de poder ser "portado"futuramente para diferentes plataformas.

Este protótipo foi testado em diferentes computadores, fazendo o uso de microfones comuns e ainda também foi usado através do sensor Kinect do Xbox 360 (MICRSOFT, 2013b). Além disso, o sistema foi concebido de forma a operar como uma arquitetura cliente-servidor, onde o cliente apenas fica encarregado de transcrever as entradas em áudio para texto, que posteriormente serão enviadas ao servidor o qual ficará responsável pelo processamento junto ao IPG. Isto faz com que o cliente possa ser implementado em diferentes arquiteturas, pois a interação como o IPG não é realizada de forma direta, ficando a cargo dele apenas a captura de áudio. Neste trabalho o cliente foi implementado para ser executado em sistemas Windows através de uma API em C#.

3.1 Captura de áudio

Existem diversas bibliotecas para o desenvolvimento de uma aplicação para reconhecimento de fala, onde neste trabalho foi utilizada a API Speech (MICRSOFT, 2013a) devido a sua facilidade e dinamismo. Esta API é responsável pelo sistema de reconhecimento de voz do Windows, além de possuir compatibilidade com o Kinect SDK (MICRSOFT, 2013b). Através dela é possível criar aplicativos para controlar o computador

e também utilizar o recurso TTS do Windows para ditar textos em aplicativos do Office como Word, OneNote, PowerPoint dentre outros.

Com relação à utilização do Kinect para reconhecimento de voz, o Kit de desenvolvimento para este sensor (Kinect SDK), trouxe uma nova versão da biblioteca para reconhecimento de voz usada no Windows. As duas versões são compatíveis, porém a nova é totalmente otimizada para o uso com o Kinect. Além disso, a nova versão não tem suporte para o modelo ditado. Isto significa que o programador precisa especificar, dado um dicionário de palavras, quais valores poderão compor uma gramática para reconhecimento no sistema. A versão anterior possuía o método *DictationGrammar* o qual retornava automaticamente uma instância da classe para a gramática padrão fornecida pela tecnologia de fala do Windows Desktop. Em resumo, era retornada uma instância para a gramática que utiliza uma língua padrão como, por exemplo, inglês.

O módulo do sistema responsável para a identificação de comandos de voz está localizado no lado cliente do sistema, onde com a utilização da API Speech, a fala é transformada em texto e posteriormente enviada ao servidor. O diagrama 3.1 a seguir específica a organização das classes que compõem o cliente:

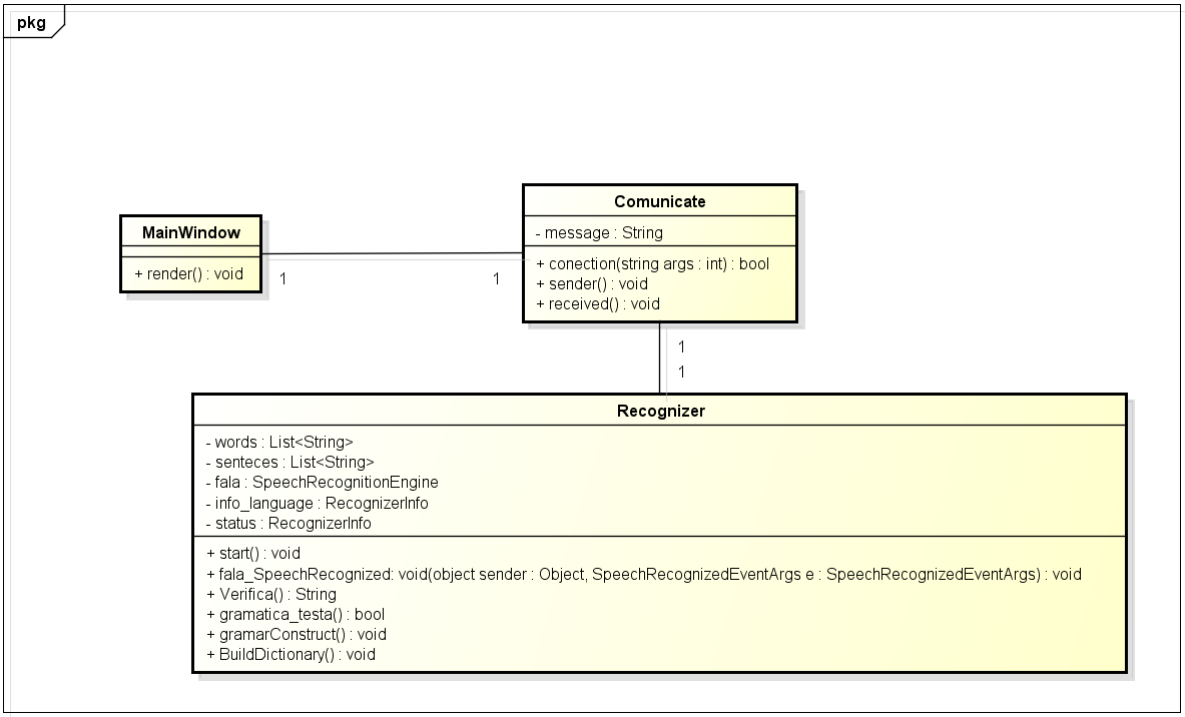

Figura 3.1: Diagrama de classes do módulo para reconhecimento de voz

3.2 Processamento das entradas

O módulo responsável por processar as entradas do usuário está localizado no servidor da aplicação. Esta parte do sistema tem como principal objetivo extrair o significado das sentenças enviadas, transcrevendo-as em uma lógica a qual o IPG seja capaz de compreender. Devido a isso foi preciso utilizar técnicas as quais fazem o uso de PLN para analisar as requisições e, sendo assim, foram implementadas classes para análise sintática e semântica dos textos recebidos. Neste contexto, a análise gramatical das sentenças recebidas foi feita através de um *parser* probabilístico usados por uma destas classes, onde através dele foi possível indentificar e processar a estrutura das frases, relacionando as ações de verbos com sujeitos, dentre outras estruturas frasais. Esta etapa do processo é de suma importância, pois através é a partir dela que podemos comparar as expressões linguísticas para posteriormente extrair os seus significados.

Para esta análise foi utilizado o Stanford Parser NPL (STANFORD, 2012), onde através dele foi possível ver a relação de dependência entre os termos das frases. Também foi produzida uma árvore da estrutura de cada frase, como a que foi representada na figura 2.4. Este *parser* foi desenvolvido pelo grupo de pesquisa em processamento de linguagem natural da Universidade de Stanford e é disponibilizado através de uma API Java. Após a análise de dependência entre os termos da frase, foram criados métodos os quais possuem o objetivo de extrair algum significado para as sentenças. Os termos recebidos são analisados na tentativa de obter situações ou eventos desejados pelo usuário.

A partir do significado obtido das frases, são geradas sentenças em lógica de primeira ordem, as quais serão usadas como entrada para o IPG. A comunicação com o módulo em Prolog é feita através do SICstus Prolog, que é um interpretador multi-plataforma para a linguagem Prolog. O SICstus disponibiliza uma interface em Java chamada Jasper que permite que uma aplicação Java possa se comunicar com a base Prolog. Sobre o Jasper foi utilizado o Componente de Consulta (POZZER, 2005) que tem como objetivo converter os dados gerados pelas consultas ao IPG em dados que possam ser interpretados pelo Gerenciador em Java. Este componente ainda é responsável por iniciar o IPG, com as características da história a ser gerada, além de facilitar as consulta à base do IPG. Uma visão geral do módulo responsável por tratar as requisições do usuário pode ser vista na figura 3.2.

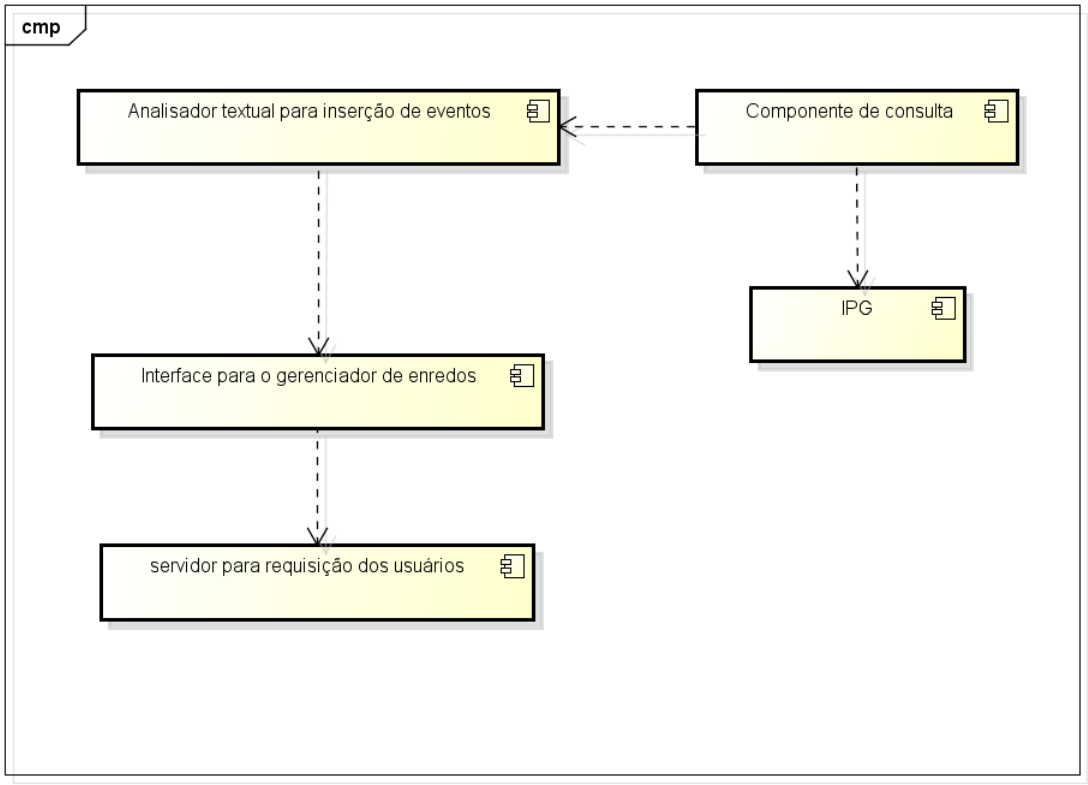

Figura 3.2: Diagrama de componentes do sistema implementado

Todo este módulo foi implementado na linguagem Java e baseado no trabalho desenvolvido anteriormente por Pozzer (2005). A interface principal de comunicação com o usuário foi refeita, eliminando assim botões e janelas desnecessárias. As sentenças lógicas são geradas e enviadas ao IPG que busca garantir a integridade do enredo. Apesar de muitos dos recursos que fazem uso de modelos de interação tradicional terem sido removidos ou reformulados, alguns deles foram mantidos. Por exemplo, o modo de exibição de eventos ao usuário na forma de retângulos coloridos foi mantido, conforme pode ser visto na figura 2.3. Este protótipo tem como principal objetivo ser facilmente adaptado a sistemas para televisão digital. Através das modificações feitas, a interação com o usuário pode ser facilitada e mais adequada a esta plataforma do que o meio de interação tradicional. Nas próximas seções serão apresentados os detalhes da implementação do sistema.

4 IMPLEMENTAÇÃO

Esta pesquisa pode ser considerada como do tipo exploratória, onde foram abordadas algumas das diferentes técnicas para o Processamento de Linguagem Natural e interação com o usuário. Apesar de existirem diferentes bibliotecas e APIs para interpretação de fala, neste trabalho foram utilizadas as APIs Microsoft Speech (MICRSOFT, 2013a) e o Stanford Parser (STANFORD, 2012). A primeira foi utilizada para o reconhecimento de voz e a segunda para analisar sintaticamente as sentenças recebidas do usuário. Posteriormente através de uma análise semântica, a qual foi implementada, transformou-se os comandos recebidos em sentenças de lógica de primeira ordem, as quais forma analisadas pelo IPG. O sistema divide-se basicamente em dois módulos, onde o primeiro é responsável por tratar as entradas do usuário e convertê-las em textos e o segundo analisa estas entradas a fim de validá-las de acordo com o enredo. A história é baseada em contos de fadas comuns, onde a princesa Marian é raptada pelo Dragão Draco e o heroi Brian deve salvá-la. De acordo com os comandos do usuário, diferentes versões desta história podem ser geradas.

4.1 O Módulo para Reconhecimento de voz

Para interpretar as frases ditas pelo usuário, foi implementado um módulo em c# onde são identificadas as palavras-chave dado o contexto da história definida pelo sistema. Para tanto foi utilizada a API Speech da Microsoft (MICRSOFT, 2013a). Para a utilização deste módulo foi criado um "dicionário"contendo palavras que podem ser consideradas pelo sistema. Este dicionário contém aproximadamente 100 palavras consideradas sensíveis ao contexto da história a ser gerada além de possuir outras como pronomes, verbos e inclui ainda uma pequena lista de sinônimos. Para a identificação de voz, foi preciso criar e carregar uma gramática que indica o método usado para o reconhecimento. A API Spe-

ech possui uma classe denominada *grammar* que permite carregar uma gramática livre de contexto especificada no formato Microsoft Speech API (SAPI) XML. As palavras definidas no dicionário alimentam esta gramática utilizada pelo programa no reconhecimento de termos relevantes, onde são consideradas juntamente com uma "*language culture*"que é o padrão de língua utilizado, o qual neste caso é o inglês. Desta forma, é possível transformar frases ditas pelo usuário em conjuntos de frases em formato de texto quando os comandos identificados são comparados com as palavras contidas no dicionário.

Nos trechos de código 4.1.1 e 4.1.2 temos parte do método responsável pela inicialização dos componentes e pela criação da gramática usada.

```
try
   {
      Console.WriteLine("oi");
      foreach (RecognizerInfo status in
                SpeechRecognitionEngine.InstalledRecognizers())
      {
          if (status.Culture.ToString() == "en-US")
          {
            fala = new SpeechRecognitionEngine(status);
            info_language = status;
            break;
          }
          if (fala == null)
          {
             MessageBox.Show("Tipo de linguagem não encotrada
               (en-US) " +
               SpeechRecognitionEngine.InstalledRecognizers()[0].
                                                 Culture.ToString()
                + " esta é a linguagem padrão");
             fala = new SpeechRecognitionEngine(
                         SpeechRecognitionEngine.
                                    InstalledRecognizers()[0]);
          }
        }
          fala.SpeechRecognized +=
                 new EventHandler<SpeechRecognizedEventArgs>(
                                        fala_SpeechRecognized);
          gramatica();
          fala.SetInputToDefaultAudioDevice();
               // multiplas falas
          fala.RecognizeAsync(RecognizeMode.Multiple);
 }
 catch (Exception e)
  {
     MessageBox.Show(e.Message, "Opa! Voz não detectada");
  }
```

Código 4.1.1: Parte do método responsável pela inicialização dos componentes

```
private void GramarConstruct()
    {
        String path = "C:\\Users\\Luiz
                      \\storyteller\\voice_hue\\words2.txt";
        Choices texts = new Choices();
        try{
            StreamReader arq = File.OpenText(path);
            string s = arq.ReadLine();
            while(s!=null){
                texts.Add(s);
                words.Add(s);
                Console.WriteLine(s);
                s = arq.ReadLine();
            }

            var gb = new GrammarBuilder {
                     Culture = info_language.Culture };
            gb.Append(texts);

            Grammar wordsList = new Grammar(gb);
            fala.LoadGrammar(wordsList);

        }
        catch(Exception e){
           MessageBox.Show(
                    "erro no arquivo de gramatica "
                                      + e.Message);
        }
    }
```

Código 4.1.2: Método da classe voice usado para gerar a gramática

Para interagir como o sistema, o usuário pode inserir um ou mais eventos em uma requisição. Este módulo interpreta frases ditas pelo usuário e as transforma em texto onde o seu envio para o módulo servidor é feito através de um comando específico que é o "continue". Não apenas eventos podem ser podem ser inseridos, mas também a ordem como eles ocorrem pode ser definida ligando-os conforme mostra a figura 2.3. Isso é feito, por exemplo, através do seguinte comando: "link 1 to 2"onde o primeiro evento é ligado ao segundo bastando, o usuário falar os identificadores numéricos para estes eventos. Para remover um evento, da mesma forma o usuário deve falar o identificador numérico como, por exemplo , para remover o primeiro evento, o comando especificado é *"remove 1*.

Caso o usuário queira gerar alguma história sem que haja uma constante interação sua como sistema, o IPG permite criar histórias automaticamente por meio dos comandos *"continue"* e *"another"*. O *"continue"* faz com que o IPG infira um novo objetivo e na sequência, através de algoritmos de planejamento, faz com que encontre um conjunto de eventos que satisfaça tal objetivo. Caso o usuário não esteja satisfeito como a história gerada ele pode requisitar ao IPG uma nova através do comando *"another"*. Vale salientar que os comandos *"continue"* e *"another"* existiam na versão anterior desenvolvida por Pozzer (POZZER, 2005). Apesar disso, neste trabalho toda a interface de comunicação

do usuário e o sistema para gerar história foi refeita eliminando o uso de botões e interfaces que demandavam uma constante interação através de mouse ou teclado. Apenas alguns comandos de mouse para manipular as caixas que representam os eventos foram mantidos. Para fins de verificação uma pequena janela contendo as requisições do usuário é apresentada no canto inferior direito da tela conforme a figura 4.1.

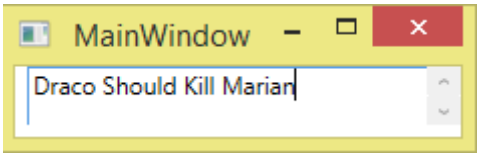

Figura 4.1: Janela do cliente

Todos esses comandos ditos não são executados pelo cliente e sim enviados em formato texto ao servidor. Após identificados os comandos enviados pelo usuário é necessário processá-los a fim de enviar requisições ao módulo gerador de histórias. Como neste sistema proposto para *Storytelling*, a comunicação entre o módulo para reconhecimento de voz e o gerador de histórias é feita através de uma arquitetura cliente-servidor, isto possibilita que como trabalho futuro, diferentes clientes possam gerar várias versões de uma mesma história. As requisições são validadas pelo servidor que fica encarregado de transformar os comandos do usuário em uma lógica capaz de ser interpretada pelo IPG e retornar um resultado válido ao usuário.

4.2 O Modulo gerador de histórias

Para o sistema proposto é preciso realizar uma análise das sentenças enviadas pelo usuário. Este módulo foi implementado na linguagem Java e funciona na forma de um servidor que recebe os comandos vindos do cliente, onde estes são analisados a fim de serem validados pelo IPG. Para isso foi realizado um processamento dos textos enviados, sendo realizada uma análise sintática com a construção de uma árvore sintática e a análise da dependência entre os termos das sentenças enviadas, além de uma análise semântica para extrair o significado das frases. Para a análise sintática foi utilizado o Stanford Parser (STANFORD, 2012), como um *parser* probabilístico a fim de analisar a relação de dependência entre os termos das frases e produzir também uma árvore da estrutura de cada frase. Para exemplificar considerando a sentença *Brian Should Attack Draco* , a figura 4.2 representa a árvore sintática gerada para esta frase e as relações de dependência são

listadas no trecho de código 4.2.1 gerado como saída do processamento do *parser*.

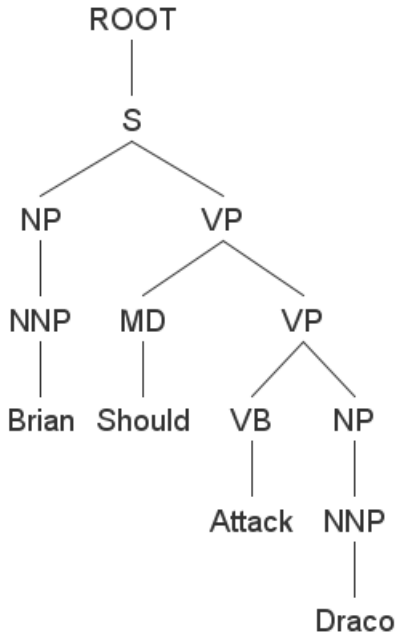

Figura 4.2: Árvore sintática de *Brian Should Attack Draco*

```
nsubj(attack-3,Brian-1)
aux(attack-3,should-2)
root(ROOT-0,attack-3)
dobj(attack-3,Draco-4)
```

Código 4.2.1: Relações de dependência da frase *Brian Should Attack Draco*

Nota-se que as saídas são relações binárias, que definem quem está realizando ou sofrendo uma ação. No caso de *Brian Should Attack Draco*, a relação *nsubj* refere-se ao sujeito "Brian"e ao verbo "attack", bem como *dobj* refere-se ao mesmo verbo e ao objeto "Draco". Desta forma é possível extrair a seguinte sentença em lógica de primeira ordem: *Attack(Brian,Draco)*. O sistema ainda permite que sentenças mais elaboradas também possam ser interpretadas como, por exemplo, *"The princess has been kidnapped so Brian must go to Dragon's castle"*. Nota-se que esta frase pode ser facilmente mapeada para a setença lógica *go(Brian,Dragon's castle)*. A verificação feita pelo sistema identifica uma ação anterior realizada pelo vilão e considera-se apenas a reação do personagem *Brian*. É possível também definir situações ou estados para os personagens com estas regras. Desta forma pode-se definir as afeições de um pelo outro além de definir estados iniciais na história, como local onde se encontram (castelo, igreja), por exemplo.

Parte do código utilizado para extrair o significado dos comandos recebidos pode ser visto nos trechos de código 4.2.2 , 4.2.3 e 4.2.4 apresentados a seguir, onde são mostradas a análise através do *parser* usado e a inserção dos eventos descritos pelo usuário na ação da história.

```
  public void API(LexicalizedParser lp, String received) {
  String sent2 = received;
  TokenizerFactory<CoreLabel> tokens =
      PTBTokenizer.factory(new CoreLabelTokenFactory(),"");
  List<CoreLabel> palavras =
      tokens.getTokenizer
                      (new StringReader(sent2)).tokenize();
  Tree parse = lp.apply(palavras);
  TreebankLanguagePack bla = new PennTreebankLanguagePack();
  GrammaticalStructureFactory var =
                           bla.grammaticalStructureFactory();
  GrammaticalStructure struct=var.newGrammaticalStructure(parse);
  List<TypedDependency>out=struct.typedDependenciesCCprocessed();
  teste = new semantica(wow);
  teste.analise(out.toArray());
 }
```

Código 4.2.2: Método da classe semantica usado para geração das árvores sintáticas e análise das frases

```
public void conecte(String received){
         String[] wow = received.split(" ");
         int index1,index2;
         index1 = Integer.parseInt(wow[1]);
         index2 = Integer.parseInt(wow[3]);
         Event op1,op2;
         op1 = (Event)program.getOperations().elementAt(index1);
         op2 = (Event)program.getOperations().elementAt(index2);
         int p1, p2;
         p1 = program.getPlotPosition( op1.getID() );
         p2 = program.getPlotPosition( op2.getID() );
         if( p2 == -1 )

         if( program.checkDependenciesAdd(p1,op2) == true )
         {
            program.plotSequenceChanged = true;

            program.plotSequence.add(p1+1, op2);
            program.updateRenderedSequence(p1+1);
            program.setActiveOperation();
         }
      }
   }
```

Código 4.2.3: Método da classe semantica usado para conectar eventos

```
public void analise(Object testa[]){
    String saida;
    ArrayList<String> sentences;
    sentences = new ArrayList<>();
    for(int i=0;i < testa.length;i++){
       String bla = testa[i].toString();
       if(bla.contains("nsub") || bla.contains("dobj")){
           sentences.add(bla);
       }
    }
   for(String op : operations){
    if(sentences.get(0).contains(op)){
      if( op.equals("fight") ||
      op.equals("kidnap") || op.equals("free")
        || op.equals("kill") || op.equals("marry")){
       saida = op +"(";
       String[] bla = sentences.get(0).split(",");
       saida += bla[1].substring(0,bla[1].length()-3)+",";
       bla = sentences.get(1).split(",");
       saida += bla[1].substring(0,bla[1].length()-3)+")";
       System.out.println(saida);
       program.addUserOperation(saida);
      }
      if(op.equals("go")){
          saida = op+"(";
          String[] bla = sentences.get(0).split(",");
          saida += bla[1].substring(0,bla[1].length()-3)+",";
          bla = sentences.get(1).split(",");
          for(int i=0; i<places.length;i++){
              if(bla[i].contains(places[i])){
                  saida+= places[i] + ")";
              }
          }
      }
      if(op.equals("attack")){
          saida = op+"(";
          String[] bla = sentences.get(0).split(",");
          saida += bla[1].substring(0,bla[1].length()-3)+",";
          bla = sentences.get(1).split(",");
          for(int i=0; i<places.length;i++){
              if(bla[i].contains(places[i])){
                  saida+= places[i] + ")";
              }
          }
       }
    }
```

Código 4.2.4: Método da semantica usado para gerar a análise das senteças

Com a análise realizada, foi possível transcrever a fala do usuário em uma lógica capaz de o IPG compreender. Os comandos foram enviados ao componente de consulta e analisados pelo IPG a fim de apresentar graficamente quais podem ser apresentados dada a sequência da história. Desta forma é possível analisar os comandos do usuário e enviá-los ao IPG como o objetivo de validá-los considerando suas pré e pós-condições.

5 RESULTADOS

O sistema foi testado em diferentes computadores com microfones comuns e também com o Kinect. Apesar das limitações decorrentes dos ruídos no ambiente, o identificador de comandos de voz funcionou de forma eficiente. A maioria dos conjuntos de requisições foi identificada sem maiores problemas. Porém vale salientar a dificuldade a qual a biblioteca Speech possui em diferenciar palavras com fonemas parecidos como, por exemplo, "will"e "kill". Em alguns casos, o sistema não conseguiu compreender o comando dito pelo usuário, o que pode atrapalhar a execução da aplicação.

O sistema implementado também não possui um modo o qual seja possível diferenciar timbres de voz. Em uma sala a qual estejam mais pessoas conversando além do usuário, o sistema poderá interpretar qualquer comando vindo de qualquer pessoa. Isto pode atrapalhar a utilização deste sistema principalmente se for usado junto ao kinect. Outra limitação do sistema diz respeito às palavras que podem ser identificadas. Apesar de um dicionário contendo aproximadamente 100 palavras ter sido criado, ainda há palavras as quais não puderam ser identificadas. Porém todas as palavras que são consideradas relevantes ao sistema estão presentes neste dicionário incluindo alguns sinônimos para elas. O fato de algumas palavras não terem sido identificadas pelo sistema pode gerar certa estranheza ao usuário, pois, desta forma, a aplicação pode perder um pouco de seu aspecto de naturalidade do ponto de vista do utilizador.

Diferentemente do módulo responsável pela interpretação da fala, os resultados obtidos da interação com gerenciador de enredos e o IPG, contidos no servidor, não apresentaram problemas significativos. Praticamente todas as requisições puderam ser atendidas e processadas pelo sistema sendo que puderam ser visualizadas na interface gráfica. Para o usuário são disponibilizados que os eventos inseridos na forma de retângulos coloridos ligados um ao outro e desenhados na tela. O fato deste módulo não ter apresentados pro-

blemas maiores problemas com relação à transformação das requisições em texto e com a validação dos eventos no IPG, permite que seja a ele facilmente incorporado um módulo para renderizar as histórias geradas, porém isto não foi explorado neste trabalho, pois o foco principal é o modo de interação com o usuário. Foram realizados vários testes e através da interação com usuário foi possível gerar diferentes histórias como o exemplo a seguir:

O vilão Draco deseja raptar a princesa Marian. Para isso ele vai até o castelo da princesa e o ataca, diminuindo a sua proteção. Draco luta com a princesa e acaba sequestrando-a. Sabendo disso o Herói Brian vai até o castelo de Draco e o ataca. Brian luta com Draco, acaba matando-o e libertando a princesa, que por sua vez retorna ao seu castelo.

Os eventos da história foram criados a partir de comandos vindos do usuário como, por exemplo, *"Draco must go to marian's castle"*, o qual foi analisado e transformado na setença *"go(Draco,princess_castle)"* a qual o IPG é capaz de interpretar. A história gerada pode ser vista na figura 5.1 a seguir:

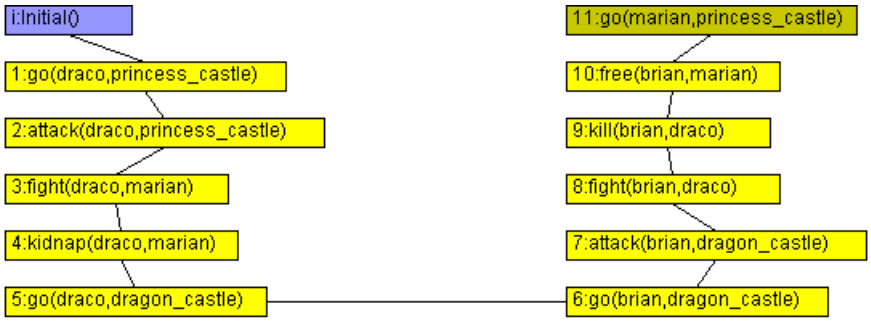

Figura 5.1: Exemplo de história gerada pelas requisições.

A interface final com as requisições aparentemente não sofreu muitas alterações, porém foram removidos botões desnecessários ao novo modo de interação. A nova Interface pode ser vista na figura 5.2 a seguir.

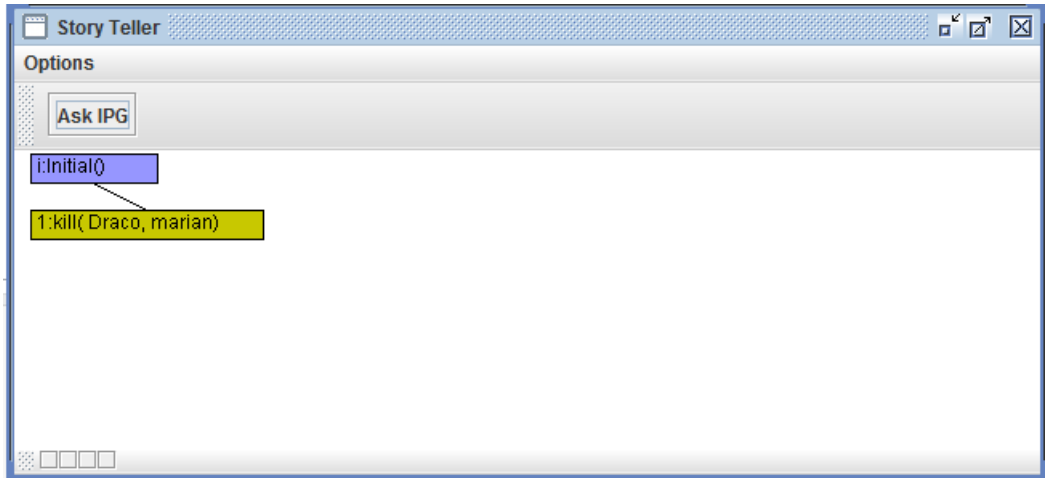

Figura 5.2: interface final

O sistema desenvolvido não apresentou maiores problemas na interpretação dos comandos do usuário. A maioria dos erros identificados concentra-se no módulo responsável por interpretação da fala, porém isso se dá devido às limitações da API utilizada. Em geral o sistema apresentou resultados satisfatórios, onde o usuário pode interagir facilmente com uma história gerenciada pelo IPG. Espera-se que este protótipo possa servir como base para futuros sistemas para televisão interativa.

6 CONCLUSÃO

Foram alcançados os resultados esperados com o protótipo desenvolvido. Apesar de possuir algumas limitações devido à biblioteca para processamento de comandos de voz, não foram detectados maiores problemas com relação ao processamento das requisições do utilizador. O processamento das entradas do usuário faz parte da etapa principal do processo de geração de histórias interativas. O processo para transcrever a fala do usuário em uma lógica capaz de o IPG utilizar, representa a maior contribuição que este trabalho apresenta. Esta é uma tarefa complexa e que ainda não é totalmente eficiente, pois existe ainda uma dificuldade em processar sinônimos e em algumas estruturas frasais.

O sistema proposto visa eliminar caixas de diálogo e a constante interação dos usuários através de comandos de mouse ou teclado. Com a utilização de comandos de voz para a geração de enredos, o modo como a pessoa utiliza o sistema torna-se aparentemente mais natural e transparente, onde o usuário não precisa saber comandos específicos para dar uma possível ordem a um personagem e influenciar a história.

Em sistemas para televisão interativa ou para *streaming*, uma interface que utiliza um conjunto de botões ou janelas pode atrapalhar o modo como a história é exibida na tela e ainda tornar o sistema pouco atrativo ao espectador. O espaço em tela necessário para apresentar uma GUI tradicional poderia competir com o espaço necessário para renderizar as cenas da história, atrapalhando a experiência deste tipo de sistema, onde devido ao espaço limitado, certos detalhes da história apresentada poderiam passar despercebidos. Desta forma este sistema pode transformar a geração de histórias interativas em uma experiência mais agradável do que os métodos tradicionais.

6.1 Trabalhos Futuros

Apesar de neste trabalho o foco tenha sido a identificação dos comandos de voz, como trabalho futuro pretende-se implementar um módulo para renderização de histórias em 3D, onde cada ação da história poderá ser disponibilizada ao usuário através de *frames* de vídeo. Além disso, o fato de este projeto ter sido concebido como uma arquitetura cliente-servidor permite que outros tipos de dispositivos possam interagir como o sistema. Desta forma pretende-se implementar modelos para o cliente que possam ser executados em dispositivos móveis e outras plataformas para computadores. Não apenas comandos de voz poderiam ser interpretados pelo sistema, mas ainda diretamente um conjunto de textos digitados pelo espectador. Pretende-se ainda realizar testes onde mais de um dispositivo possa interagir ao mesmo tempo com a história gerada, permitindo que mais de um espectador em um mesmo ambiente possa influenciar no andamento do enredo da história. Desta forma será possível criar uma plataforma eficaz para *Storytelling* interativo e ainda consolidar a arquitetura proposta.

REFERÊNCIAS

ABRAHÃO, P. R.; LIMA, V. L. S. Um estudo preliminar de metodologias de análise semântica da linguagem natural. **II encontro para para o processamento cumputacional de português escrito e falado**, [S.l.], 1996.

CIARLINI, A. **Geração interativa de enredos**. PhD thesis,Departamento de Informática, PUC-Rio, Rio de Janeiro.

CIARLINI, A.; POZZER, C. T.; FURTADO, A. L.; FEIJó, B. A logic-based tool for interactive generation and dramatization of stories. **Proceedings of the International Conference on Advances in Computer Entertainment Technology, Valencia**, [S.l.], p.133–140, 2005.

FORMAN, P. The Future of digital entertainment/creating convergence. **Scientific American**, [S.l.], 2000.

GALACTIC CAFE. **The Stanley Parable**. disponível em :http://www.stanleyparable.com/.

GUILHOTO, P. J. S.; ROSA, S. P. C. S. **Reconhecimento de voz**. 2001.

JURAFSKY, D.; MARTIN, J. H. **Speech and Language Processing**: an introduction to natural language processing, computational linguistics, and speech recognation. [S.l.]: Prentice Hall, 2000.

LIMA, E. S.; FEIJO, B.; BARBOSA, S. D. J.; CIARLINI, A. E. M.; POZZER, C. T.; FURTADO, A. L. Draw Your Own Story: paper and pencil interactive storytelling. **Proceedings of the 10th International Conference on Entertainment Computing (ICEC 2011)**, [S.l.], 2012.

MATEAS, M. An oz-centric review of interactive drama and believable agents. **Tehcnical report,School of Computer Science, Carnegie Mellon University**, [S.l.], 1997.

MATEAS, M.; STERN, A. Natural Language Understanding in Façade: surface-text processing. **Proceedings of Technologies for Interactive Digital Storytelling and Entertainment**, [S.l.], 2004.

MICRSOFT. **Microsoft Speech API 5.4**. Available at: http://msdn.microsoft.com/en-us/library/ee125663(v=vs.85).aspx.

MICRSOFT. **Microsoft Kinect SDK**. Available at:http:http://www.microsoft.com/en-us/kinectforwindows/.

MINISTÉRIO DAS COMUNICAÇÕES. **Ginga Middleware**. disponível em: http://www.mc.gov.br/acoes-e-programas/conteudos-digitais-criativos/programa-ginga-brasil.

MÜLLER, D. N. **COMFALA - Modelo Computacional do Processo de Compreensão da Fala**. Tese de Doutorado,Instituto de Informática,UFRGS.

POZZER, C. T. **Um sistema para geração, interação e visualização 3D de histórias para tv interativa**. Tese de Doutorado,Departamento de Informática, PUC-Rio, Rio de Janeiro.

PROPP, V. **Morphology of Folktale (Laurece Scott, Trans.)**. 2.ed. [S.l.]: University of Texas Press, 1968.

SCHARDONG, G.; SILVA, L. J. S.; WINCK, A. T.; POZZER, C. T. Agrupamento de Dados Baseado em Mean Shift Aplicado a Legendas de Séries Televisivas. **Proceedings of the 1º Symposium on Knowledge Discovery, Mining and Learning, São Carlos-SP**, [S.l.], 2013.

SONY COMPUTER ENTERTAINMENT. **Gaikai**. Available at:www.gaikai.com.

SPIERLING, U.; BRAUN, N.; IURGEL, I.; GRABSON, D. Setting the Scene: playing digital director in interactive storytelling and creation. **Computer Graphics**, [S.l.], 2002.

STANFORD. **The Stanford NLP (Natural Language Processing) Group**. disponível em:http://nlp.stanford.edu/downloads/lex-parser.shtml.