CDD: 006.35

UM EXEMPLO DE USO DO PADRÃO XML NA DEFINIÇÃO DE UMA LINGUAGEM ESPECIALIZADA PARA A INTELIGÊNCIA ARTIFICIAL

AN EXAMPLE OF THE USE OF THE XML STANDARD IN THE DEFINITION OF A SPECIALIZED LANGUAGE FOR ARTIFICIAL INTELLIGENCE

Ewerton José Wantroba, Juliano Ratusznei, Lucélia de Souza, Sandra Mara Guse Scós Venske

Departamento de Ciência da Computação - Universidade Estadual do Centro-Oeste - UNICENTRO, Caixa Postal 3010, CEP 85015-430, Guarapuava-PR, Brasil; e-mails: ewerton.irati, julianojules, luceliadsz, sandravenske@gmail.com

Recebido para publicação em 19/07/2008
Aceito para publicação em 10/10/2008

RESUMO

O artigo descreve um exemplo de uso do padrão XML na definição de uma linguagem especializada para a Inteligência Artificial, a AIML, Artificial Intelligence Markup Language. Também relata um estudo de caso envolvendo o desenvolvimento de um protótipo de um agente conversacional em linguagem natural, com o objetivo de atuar como um sistema de ajuda durante a interação de usuários em sistemas colaborativos.

Palavras-chave: Chatterbot. AIML. Processamento linguagem natural. Editores AIML. Interpretadores AIML.

ABSTRACT

This article describes an example in the use of standards XML in the definition of a language specialized for Artificial Intelligence Markup Language. Also described by a study envolving the development of a prototype of a conversation agent in natural language with the objective to act like a system to help during the interaction of users in collaborated systems.

Keywords: chatterbot, AIML. Processing of natural language. AIML editors. AIML interpreters.

1 Introdução

A necessidade de interação usuário-sistema por meio da comunicação em linguagem natural usando interfaces computacionais vem crescendo como alvo de pesquisas (TORRES, 2004); (LEONHARDT, 2005); (NETO, 2006); (DA SILVA, 2007); (PRIMO et al, 2008); (CONPET, 2008).

Nesse contexto, a forma como as informações estão estruturadas torna-se essencial à fluência natural de uma conversa, assim como para o acesso ágil às categorias de informações que necessitam ser criadas.

Atualmente o desenvolvimento da Inteligência Artificial (IA) permite a criação de *bots* de conversação, conhecidos como *chatbots* ou *chatterbots* (*chat* = conversa, *bot* = robô), os quais são programas de computador que usam técnicas de IA, com o propósito de simular a habilidade de conversação de um agente computacional com um ser humano (WALLACE, 2005).

Os *bots* são baseados em rotinas de "estímulo-resposta", ou seja, quando o usuário pergunta, ele responde com base nos termos utilizados durante a conversação.

Estes programas são utilizados em ambientes onde a interação em linguagem natural facilita a comunicação usuário-sistema. As aplicações envolvendo *bots* incluem: educação à distância, como o *chatterbot* Júnior, que é voltado para o ensino de crianças e adolescentes e apresentando uma linguagem e interface voltada para este público (PRIMO et al, 2008); turismo, como o *chatterbot* Balbinu's, cuja função é fornecer informações e responder perguntas sobre o litoral alagoano (DA SILVA, 2007); bibliotecas digitais, por exemplo o *chatterbot*, que é utilizado para o serviço de referência digital de uma biblioteca, mais precisamente a Biblioteca da Procuradoria Regional do Trabalho da 13ª Região (NETO, 2006); entretenimento, como o *chatterbot*, cujo objetivo é informar detalhes de um ambiente virtual, no qual a interação ocorre por meio de comandos de texto (TORRES, 2004); meio ambiente, como o *chatterbot* Ed, que conversa sobre a preservação de energia e de outros recursos naturais (CONPET, 2008).

Outros exemplos de *bots* envolvem conversas sobre um tema específico, como o *chatterbot* utilizado para fornecer informações sobre o estado atual de uma rede de computadores para o treinamento de profissionais (LEONHARDT, 2005). Também podem atuar fazendo propagandas, como é o caso do *bot* Sete Zoom da marca *Close Up* (SETE ZOOM, 2008), e até mesmo podem falar sobre horóscopo (DA SILVA, 2002).

Como característica, um *chatterbot* pode inclusive apresentar personalidade, como é o caso do Vitor-P, que pode interromper um diálogo, caso alguém o aborreça em uma conversação (CANUTO, 2005).

Existem na literatura três gerações de *bots* que utilizam diferentes técnicas quanto à implementação. A primeira geração é representada pelo software Eliza (anos 60), que foi utilizado por psicólogos para auxiliar no diagnóstico de seus pacientes. Sua implementação consistiu do uso de casamento de padrões e regras gramaticais (CANUTO, 2005).

A segunda geração de *bots* é marcada pelo *chatterbot* Júlia, projetado utilizando redes neurais (também chamadas redes conversacionais), apresentando um sistema que simula o raciocínio do cérebro humano para estabelecer um diálogo mais próximo da linguagem natural (TORRES, 2004).

O *bot* Alice marca a terceira geração, com um diálogo mais estimulante que seus antecessores, utilizando a linguagem AIML (*Artificial Intelligence Markup Language*, Linguagem de Marcação da Inteligência Artificial), derivada da XML (*eXtensible Markup Language*, Linguagem de Marcação Extensível) (TITTEL, 2008). A escolha da sintaxe XML permite integração com ferramentas já existentes para este fim, tais como editores XML.

Como forma de investigar como as aplicações utilizam a linguagem AIML, o presente trabalho apresenta um protótipo de *bot*, denominado *HelpBot*, com a intenção de apoiar a interação de usuários em sistemas colaborativos. Nesse sentido, além da AIML, outras tecnologias também são utilizadas para o desenvolvimento do *bot*, como o Program D (BUSH, 2008), que é o programa para edição das categorias AIML, e o GaitoBot (SPRINGWALD, 2008), para interpretação das *tags* criadas.

Este artigo está assim organizado: a segunda seção apresenta a linguagem AIML e suas aplicações. A terceira seção apresenta as principais ferramentas disponíveis para o desenvolvimento (tanto para a edição, mais especificamente sobre o GaitoBot,

quanto para as ferramentas para interpretação, como o Program D). Na quarta seção, o protótipo desenvolvido é apresentado. Finalmente na última seção, são apresentadas algumas conclusões e propostas de trabalhos futuros.

2 AIML: conceitos principais

Existem várias *tags* AIML cuja finalidade é permitir que a conversa com o *bot* seja a mais próxima de uma conversa real, fazendo com que o humano que está interagindo tenha a impressão de estar conversando com outra pessoa. Alan Turing (TURING, 1936) propôs o que ficou conhecido *Teste de Turing*, cujo objetivo foi verificar se uma pessoa seria capaz de dizer se as respostas às suas perguntas estão sendo dadas por programa computacional ou por um ser humano.

Um arquivo AIML possui o formato segundo o Quadro 1. Os principais elementos da AIML segundo Wallace (2005) são:

<aiml>: endereça o início e fim do documento.

<category>: denota uma unidade na base de conhecimento.

<pattern>: armazena o que um usuário pode dizer ou escrever para o *bot* (destacado no Quadro 1 em letras maiúsculas).

<template>: contém a resposta para uma entrada do usuário (destacado no Quadro 1 em letras minúsculas).

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<aiml version="1.0.1">
<category>
     <pattern>
          COMO VAI VOCE
     </pattern>
     <template>
          Eu vou bem, obrigado. E você, como vai?
     </template>
</category>
</aiml>
```

Quadro 1 - Exemplo de uso da linguagem AIML.

O arquivo inicia com um cabeçalho que indica as versões XML e AIML e é dividido em *tags* ou categorias <category>, em que cada categoria possui padrões <pattern>, que armazenam a entrada que um humano digitará em uma conversa com o *bot.* A *tag* <template> armazena a resposta para a entrada, ou seja, registra o que o *bot* responderia, caso lhe fosse feita essa pergunta.

Uma das *tags* mais utilizadas no desenvolvimento de um *bot* é *tag* <that>, cuja finalidade é dar continuidade à conversação. Por exemplo, considerando o Quadro 1 em sua base, o *bot*, além de responder a pergunta, pode fazer outra pergunta para a pessoa com quem está conversando, como "E você, como vai?", e, quando a mesma responder, por exemplo, "Eu vou bem", deve-se prever como o *bot* irá se comportar, já que ele só consegue entender apenas uma frase por vez. Sabendo que o *bot* armazena a última sentença respondida em uma variável chamada "topic", a *tag* <that> consulta essa variável e compara com o que está contido na *tag*, conforme o Quadro 2, supondo que o *bot* já possui a categoria do Quadro 1 na sua base de conhecimento.

<category>
  <pattern>
  EU VOU BEM
  </pattern>
  <that>
  Eu vou bem, obrigado. E você, como vai?
  </that>
  <template>
  Que bom que você também está bem.
  </template>
</category>
	A conversa ficaria assim:
	
	humano> Como vai você?
	
	bot> Eu vou bem, obrigado. E você, como vai?
	
	humano> Eu vou bem.
	
	bot> Que bom que você também está bem.

Quadro 2 - Exemplo de uso da *tag* <that>.

Outros aspectos que se percebem durante uma conversa são as muitas maneiras da mesma pergunta ser formulada. Por exemplo, a pergunta "Como vai você?" já foi definida no Quadro 1. No sentido da mesma resposta ser utilizada para a pergunta em formatos diferentes ("Como você está?"), seria necessário responder novamente algo que já foi definido.

Para resolver este problema, utiliza-se a *tag* <srai>, que evita a redundância de informações, sendo responsável por redirecionar a resposta de uma determinada questão à categoria que a respondeu (Quadro 3).

```
<category>
        <pattern>
                COMO VOCE ESTÁ
        </pattern>
        <template>
                <srai>
                      COMO VAI VOCE
                </srai>
        </template>
</category>
```

Quadro 3 - Exemplo de uso da *tag* <srai>.

A AIML provê a substituição de qualquer cadeia de caracteres fazendo uso do símbolo asterisco (*), que, para alguns interpretadores, pode ser substituído pela *tag* <star> (Quadro 4).

```
<category>
        <pattern>
                EU ME CHAMO *
        </pattern>
        <template>
                Prazer em conhecê-lo *! <!-- onde * substitui o nome
                                            da pessoa -->
        </template>
</category>
```

Quadro 4 - Exemplo de uso do asterisco * .

Caso seja necessário armazenar o nome do usuário para uso futuro na conversa, a linguagem AIML permite a criação de variáveis utilizando as *tags* <set> (para armazenar ou alterar o valor de uma variável) e <get> (para ler o valor de uma variável). Essa operação deve ser transparente ao usuário. Para tanto, a *tag* <think> é responsável por realizar as operações internas do *bot* (Quadro 5).

```
<category>
   <pattern>
       EU ME CHAMO *
   </pattern>
   <template>
       <think> <set="nome">*</set> </think>
       Muito prazer <get="nome">.
   </template>
</category>
```

A conversa ficaria assim:

pessoa > Eu me chamo Pedro

bot > Muito prazer Pedro.

Quadro 5 - Exemplo de uso das *tags* <set> e <get>.

A *tag* <random> busca uma expressão aleatória dentre as especificadas para responder ao usuário, conforme o Quadro 6.

```
<category>
        <pattern>
                COMO VAI VOCE
        </pattern>
        <template>
                <random>
                        <li>Eu vou bem, obrigado.</li>
                        <li>Hoje não estou muito bem.</li>
                        <li>Tenho estado bem ultimamente.</li>
                        <li>Tudo certo comigo.</li>
                </random>
        </template>
</category>
```

Quadro 6 - Exemplo de uso da *tag* <random>.

A criação de arquivos AIML por *bot-masters* (pessoas que gerenciam *bots*), em editores textuais puros, dificulta a inclusão e o gerenciamento das inúmeras *tags* necessárias para o andamento de uma conversa que exige um conhecimento razoável. Para realização desta tarefa, existem editores gráficos específicos para AIML, entretanto, dentre os pesquisados, destaca-se o editor GaitoBot (SPRINGWALD, 2008), conforme descrito na próxima seção.

3 Ferramentas para o desenvolvimento do protótipo

Esta seção divide-se na apresentação de alguns editores AIML, seguida da descrição de alguns programas interpretadores.

3.1 Editores AIML

Para o gerenciamento de categorias do *bot*, alguns editores podem ser citados: *AutoAIML* (http://aiml.harrybailey.com/beta/); *MakeAIML* (http://makeaiml.aihub.org/); *TSL AIML Parser/Chatbot editor* (http://www.nabble.com/TSL-AIML-Parser-Chatbot-editor-to3733275.html); *CHAT4D* (http://www.toolbox.free.fr/cariboost2/crbst_3.html); *GaitoBot* (http://www.gaitobot.de/).

Dentre as ferramentas supra-citadas, para o desenvolvimento deste projeto, foi utilizado o editor GaitoBot, dada a documentação escassa disponível referente aos outros editores.

O GaitoBot foi desenvolvido pela *Springwald Software* e, apesar de ser um editor comercial, disponibiliza uma primeira versão *beta* pública (*freeware*), que permite criar as *tags* AIML em uma interface gráfica amigável, armazenando as *tags* de forma organizada, conforme a Figura 1.

É baseado na plataforma *.NET* da *Microsoft* e é disponibilizado para testes em *Windows 2000* e *XP* (SPRINGWALD, 2008). Possui seu próprio interpretador AIML, gerando, na versão servidor, arquivos de *log* de todos os diálogos, o que facilita a depuração no caso de erros. Possui um *workflow view* que permite ter uma visão ampla, até mesmo para grandes projetos. Como desvantagem, o GaitoBot ainda não suporta todas as *tags* AIML, mas consiste em uma plataforma que está em desenvolvimento (WALLACE, 2005).

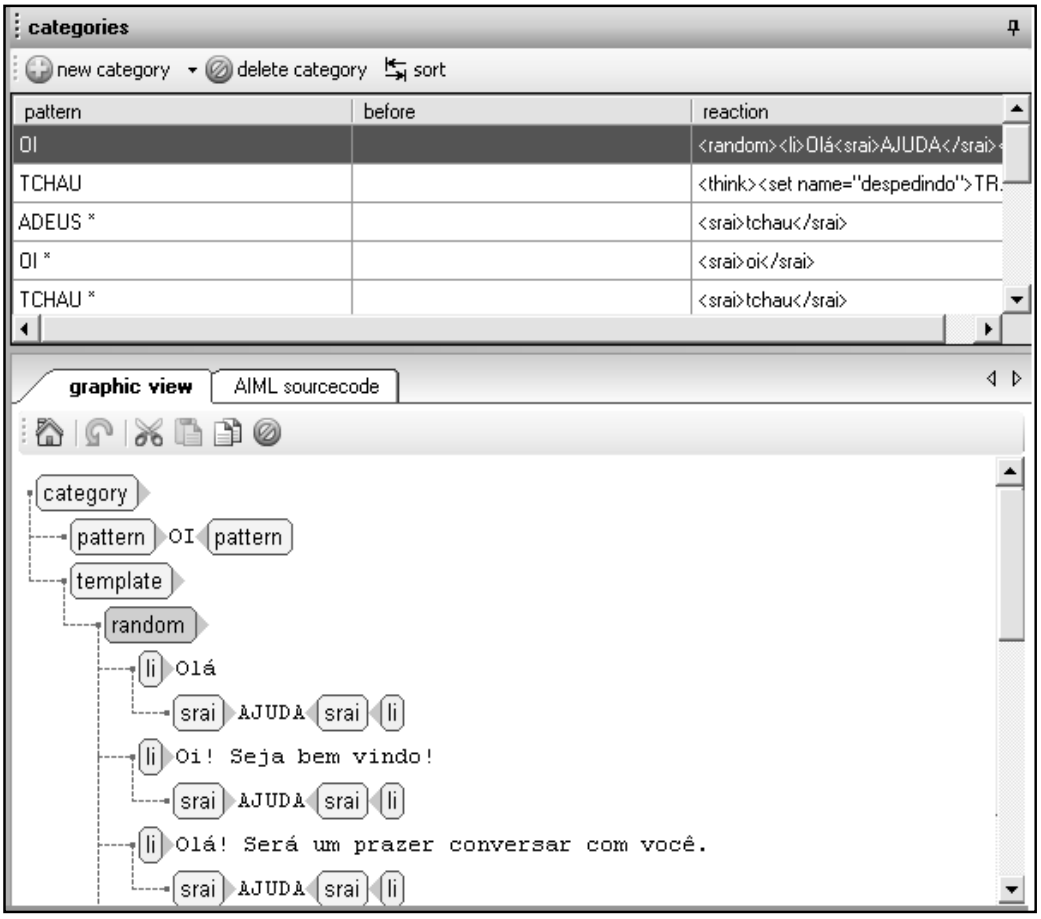

Figura 1 - Tela inicial do editor AIML GaitoBot.

3.2 Interpretadores AIML

Dentre os vários interpretadores AIML existentes, algumas ferramentas podem ser citadas: o *Program E*:, escrito em *PHP* (*Hypertext Preprocessor*), utiliza como gerenciador de banco de dados *MySQL* e apresenta algumas interfaces de *chat*: *HTML* (*HyperText Markup Language*, Linguagem de Marcação de Hipertexto), *Flash* e *XML-R* (*XML Reader*) (http://sourceforge.net/projects/programe/); o *Program R*:, implementado com base no trabalho do Dr Wallace (WALLACE, 2005) e escrito na linguagem *Ruby* (http://aiml-programr.rubyforge.org/); o *Program P*: caracteriza-se como um interpretador AIML *free*, escrito em *Delphi* (http://www.sweb.cz/alicebot/introduction.html).

Dentre as ferramentas pesquisadas, para o desenvolvimento deste projeto, foi utilizado o interpretador Program D, que atualmente encontra-se na versão 4.6, caracterizando-se como o interpretador recomendado para o trabalho com AIML, pois preserva funcionali-dades, relata um número considerável de *bugs*, trazendo melhorias em sua arquitetura.

Utiliza a extensão de servidor da aplicação *web* `.war` e pode interagir com um cliente *web* usando tecnologias *Ajax*. Muitas otimizações têm aumentado o número de *bots* que o Program D pode tratar, aumentando taxas de respostas e reduzindo o uso da memória. Para *bots* que compartilham arquivos AIML, o Program D ainda pode detectar isto, além de usar um atalho para adicionar rapidamente uma associação com arquivos AIML para o novo *bot*, sem necessitar fazer uma entrada de *re-parsing* no arquivo AIML. A interface *shell* apresenta novos comandos: `/memory`, que fornece estatísticas de uso da memória e `/category`, que mostra o número de categorias adicionadas.

Portanto, para o desenvolvimento do *bot* foram utilizados, além da AIML, o editor de *tags* GaitoBot e o interpretador Program D, por considerar estes os mais viáveis atualmente para uso em conjunto, levando em consideração a documentação disponível.

4 Protótipo do *HelpBot*

Após levantamento bibliográfico, observou-se a dificuldade de se disponibilizar atualmente um *bot* que não faça uso da linguagem AIML, visto que o trabalho despendido para a estruturação da conversa, além de demandar muito tempo, é monótono.

Dessa forma, o objetivo do presente projeto é a especificação e o desenvolvimento de um protótipo de um *bot* conversacional, utilizando linguagem natural por meio da criação/tradução de determinadas categorias AIML. Definiu-se como foco de atuação sistemas de ajuda, visto que até o momento não foram encontrados relatos de uso de *bots* para esse fim.

Considerando o contexto de sistemas colaborativos *web*, apesar da demanda de utilização, muitos usuários podem ser apoiados por um *bot* conversacional que os direcionem para uma correta interação a fim de realizarem suas tarefas.

Um exemplo que pode ser citado são as ferramentas de *e-group*, em que usuários com conhecimento básico em Informática podem não usufruir de todas as suas funcionalidades. Assim, o usuário poderia solicitar ajuda on-line ao *bot*, que o instruiria passo a passo sobre como completar a tarefa.

A Figura 2 apresenta o protótipo do *HelpBot* (*Chatterbot* de Ajuda) em desenvolvimento, visto que, para a criação de sua interface, será estudado a linguagem VHML (*Virtual Human Markup Language*) (http://www.vhml.org/), também derivada da XML, que é uma linguagem para tratar os vários aspectos da Interação Humano-Computador, considerando animação facial e corporal, produção de texto em fala, representação emocional, entre outras.

A especificação até o momento contempla o arquivo `saudações.aiml` do *bot* (Figura 3) e outros que se relacionam com a execução de tarefas em sistemas colaborativos (a serem ampliadas).

Para especificação das categorias de `saudações.aiml`, foram utilizadas variáveis para aprimoramento da conversa, tornando-a mais personalizada. Por exemplo, em uma saudação em que o *bot* responde "Boa noite [nome-usuário]", a variável `nome-usuário` armazena o nome de quem está conversando.

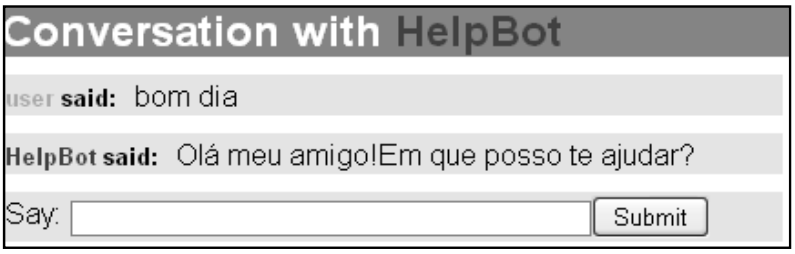

Figura 2 - Tela do protótipo do HelpBot em desenvolvimento com o Program-D.

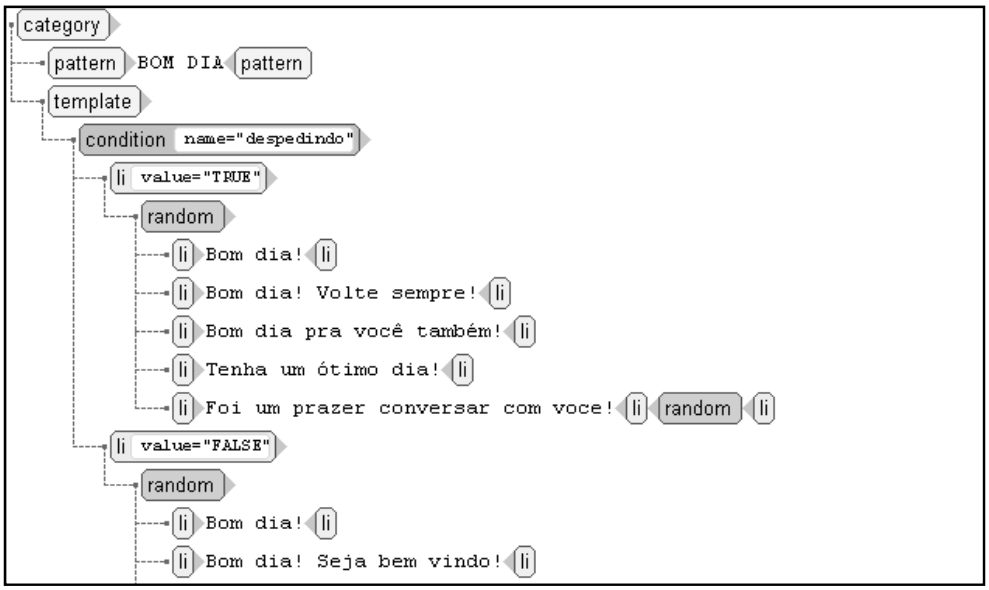

Figura 3 - Uma categoria do arquivo Saudações do *HelpBot*, visualizada no editor.

Outra situação é quando o *bot* responde "Boa noite", tratando o caso de ser no início da conversa ou no final, por meio da criação de uma variável utilizada dentro de uma categoria acessada com a *tag* `<condition>` no início da conversa.

A Figura 3 apresenta parte do conhecimento do protótipo *HelpBot,* mais precisamente uma categoria de saudação editada no GaitoBot. Após cada saudação, o *bot* toma a iniciativa de perguntar qual é a dúvida do usuário através da *tag* `<srai>AJUDA</srai>`, que redireciona à categoria AJUDA, que, por sua vez, contém várias formas de se fazer essa pergunta através de uma *tag* `<random>`.

Faz-se uma outra observação quanto ao entendimento de algumas *tags*, como `<that>` e `<condition>`, que apresentam variações na especificação entre editor e interpretador AIML.

A seguir serão apresentadas algumas conclusões referentes ao trabalho e propostas de trabalhos futuros.

5 Conclusões

O trabalho objetivou demonstrar um exemplo de uso do padrão XML na definição da linguagem AIML, por meio da especificação de um protótipo de um *bot*, com a intenção de atuar como sistema de ajuda durante a interação de usuários em sistemas colaborativos.

Dentre as ferramentas de edição e interpretação citadas para o desenvolvimento, definiu-se o GaitoBot devido às características de interface gráfica e amigável, agilidade no gerenciamento das *tags* e execução de testes.

Por outro lado, verificou-se que, se um interpretador aceitar *tags* extra não-padrão, isto somente serve para o incentivo da criação de *tags* não portáveis em AIML, o que não acontece com o Program D, escolhido neste projeto.

Outro ponto considerado também foi a dificuldade de instalação, configuração de versões e compatibilidade entre editor-interpretador, que foi bastante facilitada pela escolha do Program D.

Dentre as dificuldades encontradas na especificação e desenvolvimento das categorias AIML, destacam-se:

- Apesar das vantagens proporcionadas pelo uso da AIML, o processamento em linguagem natural ainda é uma tarefa complexa, dada a ocorrência frequente de ambiguidades na gramática nativa.
- A literatura escassa a respeito da criação de novas *tags*, como, por exemplo, para a criação de *links web* a serem disponibilizados pelo *bot* durante a conversação.
- A existência de diferença nos testes da especificação de uma categoria entre a execução com o editor e o interpretador AIML.
- A necessidade da inclusão e/ou tradução de um número excessivo de categorias (AliceBot tem aproximadamente 41.000 categorias) na base de conhecimento do *bot* para tratar a linguagem natural.
- A relação entre editor e interpretador AIML. Muitas vezes, o teste de conversação acontece no editor GaitoBot, mas não é executado pelo interpretador Program D. Um exemplo é a *tag* `<img>`, tratada pelo editor, mas não presente no interpretador Program D.

Dentre as vantagens de uso da linguagem AIML, cita-se que é uma linguagem eficiente para a criação de *bots*, fato que se justifica pela utilização no desenvolvimento da maioria dos *bots* atuais.

Como trabalhos futuros, pretende-se aumentar o conhecimento do *bot* por meio da continuidade da especificação das *tags* e desenvolver uma interface gráfica para controlar a animação facial e corporal do *bot*. Para tanto, pretende-se fazer um estudo de viabilidade de utilização da linguagem VHML (*Virtual Human Markup Language*) (VHML, 2008).

Agradecimentos

Os autores agradecem à bolsa institucional BIC/UNICENTRO.

REFERÊNCIAS

BUSH, N. **Program D**. Disponível em: <http://aitools.org/Program_D_Release_Notes# What.27s_New>. Acesso em: maio 2008.

CANUTO, E. P. **Victor-P***:* um CVA-chatterbot com personalidade. Trabalho de Graduação. Or.: Flávia de A. B. Universidade Federal de Pernambuco, 2005. Disponível em: <www.cin.ufpe.br/~tg/2005-1/epc.doc>. Acesso em: jun. 2008.

CONPET Petrobrás. **Bot Ed.** Disponível em: <http://www.inbot.com.br/ed/>. Acesso em: jun. 2008.

DA SILVA, A. B. **Um chatterbot em AIML plus que conversa sobre horóscopo**. Trabalho de Graduação, 2002. Or. profª Flávia Almeida Barros. Universidade Federal de Pernambuco.

DA SILVA, A.; COSTA, E. Barros. **Um chatterbot aplicado ao turismo**: um estudo de caso no litoral alagoano e na cidade de Maceió. Escola Regional de Informática, ERI-PR, 14. Guarapuava, 2007. Disponível em: <http://www.sbc.org.br/ bibliotecadigital/download.php?paper=698>. Acesso em: maio 2008.

LEONHARDT, M. D. **DOROTY:** um chatterbot para treinamento de profissionais atuantes no gerenciamento de redes de computadores. Dissertação de Mestrado. Porto Alegre: PPGC da UFRGS, 2005. Acesso em: abr. 2008.

NETO, G. H. N. **Chatbots no serviço de referência online:** uma ferramenta para a gestão da biblioteca da PRT 13ª Região. Monografia. Curso de Especialização em Gestão de Unidades de Informação do Centro de Ciências Sociais Aplicadas. Or. Dr. Guilherme A. Dias. Universidade Federal da Paraíba, 2006. Disponível em: <http://eprints.rclis.org/archive/00008853/01/mono_gustavo_corrigida.pdf>. Acesso em jun. 2008.

PRIMO et al. **Júnior, um chatterbot para educação a distância**. Simpósio Internacional de Informática Educativa, 10. RIBIe, 2008. Disponível em: <http://lsm.dei.uc.pt/ribie/docfiles/txt200372912710J%C3%BAnior.pdf>. Acesso em: maio 2008.

SETE ZOOM. **Sete zoom bot.** Disponível em: <http://www.inbot.com.br/sete/>. Acesso em mar/2008.

SPRINGWALD Software. **Gaito bot.** Disponível em: <http://www.gaito.de>. Acesso em: maio 2008.

TITTEL, Ed. **Teoria e problemas de XML**. Porto Alegre: Bookman – Coleção Schaum, 2003.

TORRES, D. G. **Um chatterbot em XBotML para MUDs.** Trabalho de graduação. Or. Flávia A. Barros, 2004. Disponível em: <www.cin.ufpe.br/~dgt/TG/dissertacaoTG_DGT_final.doc>. Acesso em: jul. 2008.

TURING, A. **On computable numbers, with an application to the entscheidungsproblem,** Proceedings of the London Mathematical Society, Series 2, Volume 42, 1936.

VHML. **Virtual human markup language**. Disponível em: <www.vhml.org/>. Acesso em: jul. 2008.

WALLACE, R. **Alice:** artificial intelligence foundation. Disponível em: <http://www.pandorabots.com/pandora/talk?botid=f5d922d97e345aa1>. Acesso em: fev. 2008.