AIML

CATEGORIAS

A unidade básica de conhecimento em AIML é chamada uma categoria.
Cada categoria é composta por:

- **Questão de entrada** , ou estímulo, é chamado o padrão
- **Resposta de saída** , é chamado de modelo
- **Contexto opcional** : dois tipos – "que" e "tema"

A linguagem padrão AIML é simples, consistindo apenas em palavras , espaços e símbolos curinga _ e * . As palavras podem consistir de letras e números, mas não os outros caracteres . O idioma padrão é caso invariante . As palavras são separadas por um único espaço, e os caracteres curinga funcionam como palavras.

As primeiras versões do AIML permitiam somente um caractere curinga por padrão . O AIML 1.01 padrão permite vários curingas em cada padrão , mas a linguagem é projetada para ser o mais simples possível, mais simples até do que expressões regulares. Em sua forma mais simples, o modelo consiste em texto simples, sem identificação.

AIML atualmente suporta duas maneiras para fazer a interface outras linguagens e sistemas. A tag <system> executa qualquer programa acessível como um comando de shell do sistema operacional, e insere os resultados na resposta. Da mesma forma, a tag <javascript> permite scripting arbitrário dentro dos templates. A porção contexto opcional da categoria consiste de duas variantes , chamados <that> e <topic> . A tag <that> aparece dentro da categoria, e seu padrão deve coincidir com a última expressão do robô. Lembrando uma última declaração é importante se o robô faz uma pergunta . A tag <topic> aparece fora da categoria, e recolhe um conjunto de categorias. O tópico pode ser definido dentro de qualquer modelo.

AIML não é exatamente o mesmo que um banco de dados simples de perguntas e respostas. Comparando as consultas, a linguagem é muito mais simples do que algo como SQL. Mas um modelo da categoria podem conter a tag <srai> recursivo , de modo a que a saída não dependa apenas de uma categoria correspondente, mas também de todas as outras de forma recursiva alcançados através do <srai> .

RECURSÃO

AIML implementa recursão com o operador <srai>. Não existe acordo sobre o significado da sigla. O "AI" significa inteligência artificial, mas "SR" pode significar "stimulus-response" (estímulo resposta), "syntactic rewrite" (reescrita sintática), "symbolic reduction" (redução simbólica), "simple recursion" (recursão simples) ou "synonym resolution" (resolução sinônimo). O desacordo sobre a sigla reflete a variedade de aplicações para <srai> em AIML.

(1) Redução simbólica: Reduzir formas gramaticais complexas para as mais simples.
(2) Dividir e Conquistar: Dividir uma entrada em duas ou mais subpartes , e combinar as respostas a cada uma.
(3) Sinônimos: Mapa de maneiras diferentes de dizer a mesma coisa para a mesma resposta.
(4) Ortografia ou correções gramaticais.
(5) Detecção de palavras-chave em qualquer lugar na entrada.
(6) Condicionais: certas formas de ramificação pode ser implementado com <srai>.
(7) Qualquer combinação de (1) - (6)

(1) Redução simbólica refere-se ao processo de simplificação de formas gramaticais complexas em outras mais simples. Normalmente, os padrões atômicos em categorias que armazenam conhecimento robô são indicados em termos mais simples possíveis, por exemplo tendem a preferir padrões como "Quem é SOCRATES " para aqueles como "Você sabe quem Sócrates é" ao armazenar informações biográficas sobre Sócrates.

<category>
<pattern> Sabe quem * é < / pattern>
<template> <srai> quem é <star/> </ srai > </ template>
< / category>

Qualquer que seja a entrada compensada este padrão , a porção ligada a * pode ser inserido na resposta com a marcação <star/>. Esta categoria reduz qualquer entrada do formulário " Você sabe quem é X ? " para "Quem é X ? "

(2) Dividir para conquistar: Muitas frases individuais pode ser reduzida a dois ou mais subsentenças , e a resposta é formada pela combinação das respostas para cada um . A frase que começa com a palavra " Sim ", por exemplo , se ele tem mais de uma palavra , pode ser tratado como a subsentença "Sim" mais o que segue.

<category>
<pattern> SIM * </ pattern>
<template> <srai> SIM </ srai > <sr/> </ template>
< / category>

A marcação <sr/> é simplesmente uma abreviação para <srai> <star/> </ srai > .

```
<?xml version="1.0" encoding="UTF-8" ?>
<aiml version="1.0">
     <category>
     <pattern>
     olá
</pattern>
     <template>
     Oi!
     </template>
</category>
<category>
     <pattern>
     *
     </pattern>
     <template>
     <random>
     <li>
     Fale alguma coisa…
</li>
     <li>
     Se você quer conversar sobre nada, você está no lugar certo!
</li>
     <li>
     Espero que esteja gostando do papo.
```

```
</li>
     <li>
     Estou apaixonado por Julia Roboberts
</li>
     </random>
</template>
     </category>
</aiml>
```

4 **ARTIFICIAL INTELLIGENCE MARKUP LANGUAGE (AIML)**

No período de 1995 a 2000, *Alicebot* desenvolveu o AIML, tendo como padrão gramatical o *Extensible Markup Language* (XML). Optou-se por este por ser de fácil aprendizado, muito semelhante ao HTML. É formado por uma série de *tags* com comandos claros, facilitando a implementação da base de conhecimento (LEONHARDT, 2005, p. 5).

Conforme Leonhardt (2005, p. 5), "[...] o AIML é baseado em padrões de entrada do usuário, conhecidos como categorias. Uma frase escrita por um usuário é comparada aos padrões descritos na linguagem, e com base neste processo, são selecionadas ou construídas as respostas." Ainda, segundo o autor, o AIML tem como principais *tags*:<*aiml*> inicia e termina um bloco programado em AIML;

a) <*category*> identifica uma "unidade de conhecimento" na base de conhecimento;b) <*pattern*> identifica um padrão de mensagem simples frequentemente utilizado por usuários; c) <*template*> contém a resposta para uma mensagem do usuário.

A base de conhecimento do *chatterbots* é formada pelas *tags* da linguagem AIML, ou seja, todas as possíveis perguntas e respostas do usuário ficam armazenadas nessa base, conforme observa-se no Quadro 1:

Quadro 1: Exemplificação do código da base de conhecimento

Neste exemplo percebe-se que se o usuário perguntar "O que é um *chatterbot*?" o agente de conversação (*chatterbot*) responderá "Um *chatterbot* é um agente ou robô de conversação".

Dessa forma, a possível pergunta do usuário fica entre as *tags pattern* e as respostas dadas pelo *chatterbot* ficam entre as *tags template*. Além disso, com a utilização do AIML, podem-se ser feitas definições de múltiplas respostas para uma única pergunta e até implementar condição para a escolha de uma resposta. Com isso, o *chatterbot* torna-se mais abrangente e flexível em suas respostas (LEONHARDT, 2005, p. 5). Assim, o *chatterbot* faz a consulta da resposta para a pergunta na base de conhecimento; esse processo é feito pela máquina de inferência.

A máquina de inferência (Esquema 1) é responsável pela tradução da escrita humana para um entendimento do *chatterbot*, identificando e retornando com entendimento à altura. A primeira fase consiste em identificar caracteres especiais ou abreviações inseridas no texto e

substituí-los por outras que o *bot* identifica como padrão. A segunda consiste na separação de assunto que o usuário entrou, considerando a pontuação. Por último, são removidas as pontuações e aplicado um efeito de *uppercase*, ou seja, o texto escrito é convertido em maiúsculo (LEONHARDT, 2005, p. 60).

Esquema 1: Arquitetura da ALICE Fonte: Teixeira et al. (2005).

Conforme o Esquema 1, a pessoa encaminha a pergunta ao *chatterbot* e a máquina de inferência procura na base de conhecimento a resposta adequada para informar ao usuário e mostra essa resposta por meio de uma mensagem na tela do computador.

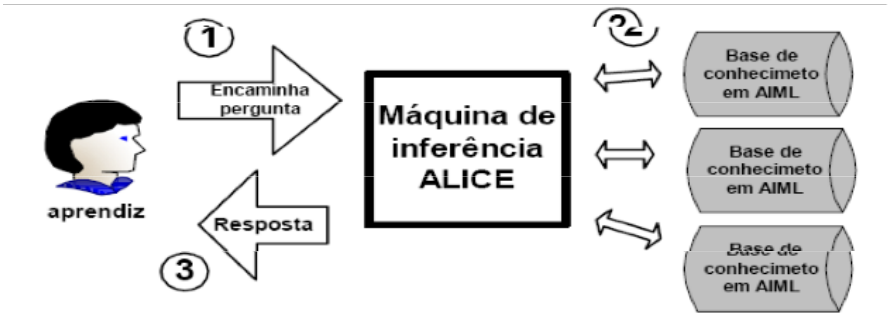
Para identificar a entrada do usuário na base de conhecimento e retornar uma resposta, utiliza- se o interpretador AIML, chamado de Program D. É uma ferramenta *free* (livre) desenvolvida em Java. Pode-se dizer que o Program D é a máquina de inferência. Esse programa possui uma ferramenta que mostra as perguntas dos usuários e as respostas dadas pelo *chatterbot*. Além disso, por intermédio dele é possível configurar diversos *chatterbots* rodando ao mesmo tempo (AIRES, 2008).

<aiml> <category>

*<pattern>*O que é um *chatterbot*?*</pattern> <template>*

Um *chatterbot* é um agente ou robô de conversação. *</template>*

</category> </aiml>



A busca na base de conhecimento é filtrada conforme a entrada feita, ou seja, mediante a pergunta do usuário a máquina efetuará efetuar essa busca nos arquivos da base de conhecimento, retornando a melhor resposta possível. Além disso, é possível atribuir um padrão de retorno no caso da máquina de a inferência não encontrar a resposta correta, ou se o usuário não conseguiu se expressar bem em sua dúvida; isso o ajudará a melhor formular sua pergunta.

```
<category>
    <pattern>qual o seu nome</pattern>
```

```
    <template>
      <random>
      <li>me chamo Tecnobot.</li>
      <li>Fui batizado com o nome Tecnobot</li>
      <li> Geralmente me chamam de Tecnobot</li>
      </random>
    </template>
</category>
```

```
<category>
<pattern>olá</pattern>
<template>qual o seu nome</template>
</category>
```

Seqüência do diálogo

Usuário: olá

Robô: qual o seu nome

```
<category>
<pattern>meu nome é *</pattern>
<template>Muito prazer, <set name="nome"><star/></set></template>
</category>
```

```
<category>
<pattern>odeio *</pattern>
<template>o ódio é ruim para a qualidade de vida</template>
</category>
```

Seqüência do diálogo

Usuário: odeio chuva

Robô: o ódio é ruim para a qualidade de vida

```
<category>
<pattern>odeio *</pattern>
<template>o ódio é ruim para a qualidade de vida</template>
</category>
```

```
<category>
  <pattern>eu tenho * anos</pattern>
      <template>Um dia eu também farei <set name="idade"><star/></set> anos</template>
</category>
<category>
  <pattern>Você lembra a minha idade</pattern>
       <template>Lembro sim, você tem <get name="idade"/> anos</template>
</category>
```

Usuário: eu tenho 18 anos

Robô: Um dia eu também farei 18 anos

```
<category>
  <pattern>Eu nasci em *</pattern>
  <template>Obrigado pela informação.
        <think><set name=”naturalde”><star/></set></think>
   </template>
</category>
```

```
<category>
<pattern>oi</pattern>
<template><srai>saudacao</srai></template>
</category>

<category>
<pattern>e ai</pattern>
<template> <srai>saudacao</srai></template>
</category>

<category>
<pattern>tem alguém ai</pattern>
<template><srai>saudacao</srai></template>
</category>
```

Seqüência do diálogo

Usuário: oi

Robô: que bom, alguém para conversar

```
<category>
    <pattern>saudacao</pattern>
    <template>
      <random>
      <li>como vai</li>
      <li>que bom, alguém para conversar</li>
      <li>já estava me sentindo sozinho</li>
      <li>sobre o que vamos teclar</li>
      </random>
    </template>
</category>
```

```
<category>
  <pattern>Vamos falar sobre celulares</pattern>
  <template>
      Ok. O que lhe interessa nos <setname=”topic”>celulares</set>
  </template>
</category>

<topic name=”celulares”>
<category>
```

Seqüência do diálogo

Usuário: Vamos falar sobre celulares

Robô: Ok. O que lhe interessa nos celulares

Usuário: eu gosto das telas por toque

<pattern>eu gosto das telas por toque</pattern>
<template>Não consigo usar, me faltam os desdos, hehehe.</template>
</category>
</topic>

Robô: Não consigo usar, me faltam os desdos, hehehe