CONTROLE POR VOZ UTILIZANDO A ENGINE JULIUS COM FALA CONTÍNUA

Aluno: DEIVID GEOVANI SANT'ANA

Orientadora: JOYCE MARTINS

FURB
UNIVERSIDADE DE BLUMENAU

Roteiro

- Introdução
- Objetivos
- Fundamentação Teórica
  - Reconhecimento de voz
  - *Engine* Julius
  - LapsAPI
  - Lego NXT
  - LeJOS
- Trabalhos Correlatos

Roteiro

- Desenvolvimento
  - Requisitos
  - Casos de Uso
  - Robô
  - Linguagem
  - Reconhecimento de voz
  - Resultado e discussões
  - Conclusão
  - Sugestões

Introdução

- Reconhecimento automático de voz
- Fala isolada
- Fala contínua
- Ditado contínuo
- *Engine* Julius

Objetivos

- Controle de voz por fala contínua

- Controlar um robô por voz
- Elaborar um vocabulário
- Enviar para o robô os comandos

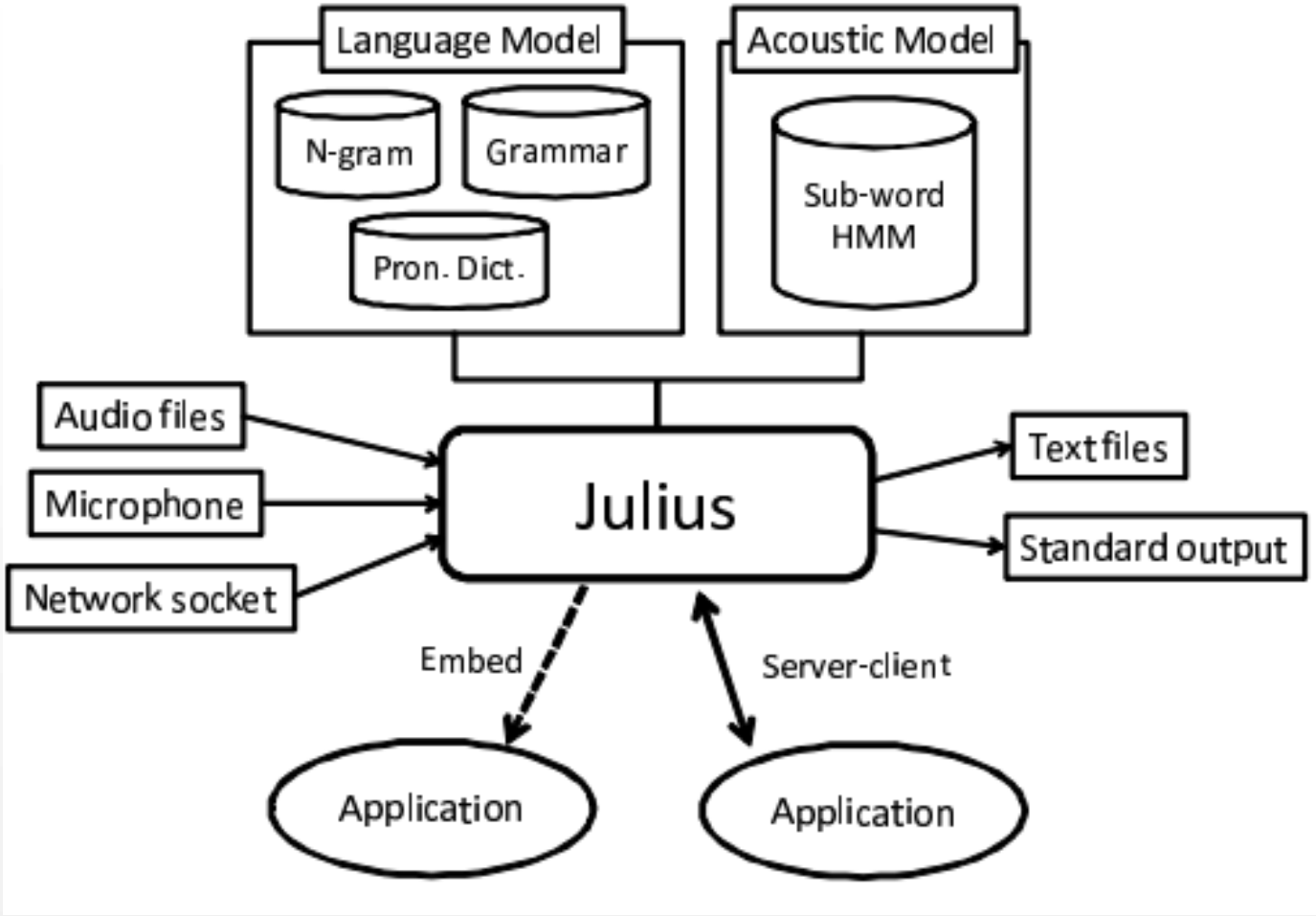
Fundamentação Teórica

- Reconhecimento de voz (problemáticas)
  - Sotaque do locutor
  - Palavras com mesma pronúncia e significados diferentes
  - Variação de pronúncia
  - Relação sinal-ruído

Fundamentação Teórica

- *Engine* Julius (visão geral)
  - Em desenvolvimento desde de 1998
  - Universidade de Kyoto
  - Multi-plataforma
  - Desenvolvimento de pesquisas, comercial e industrial

Fundamentação Teórica

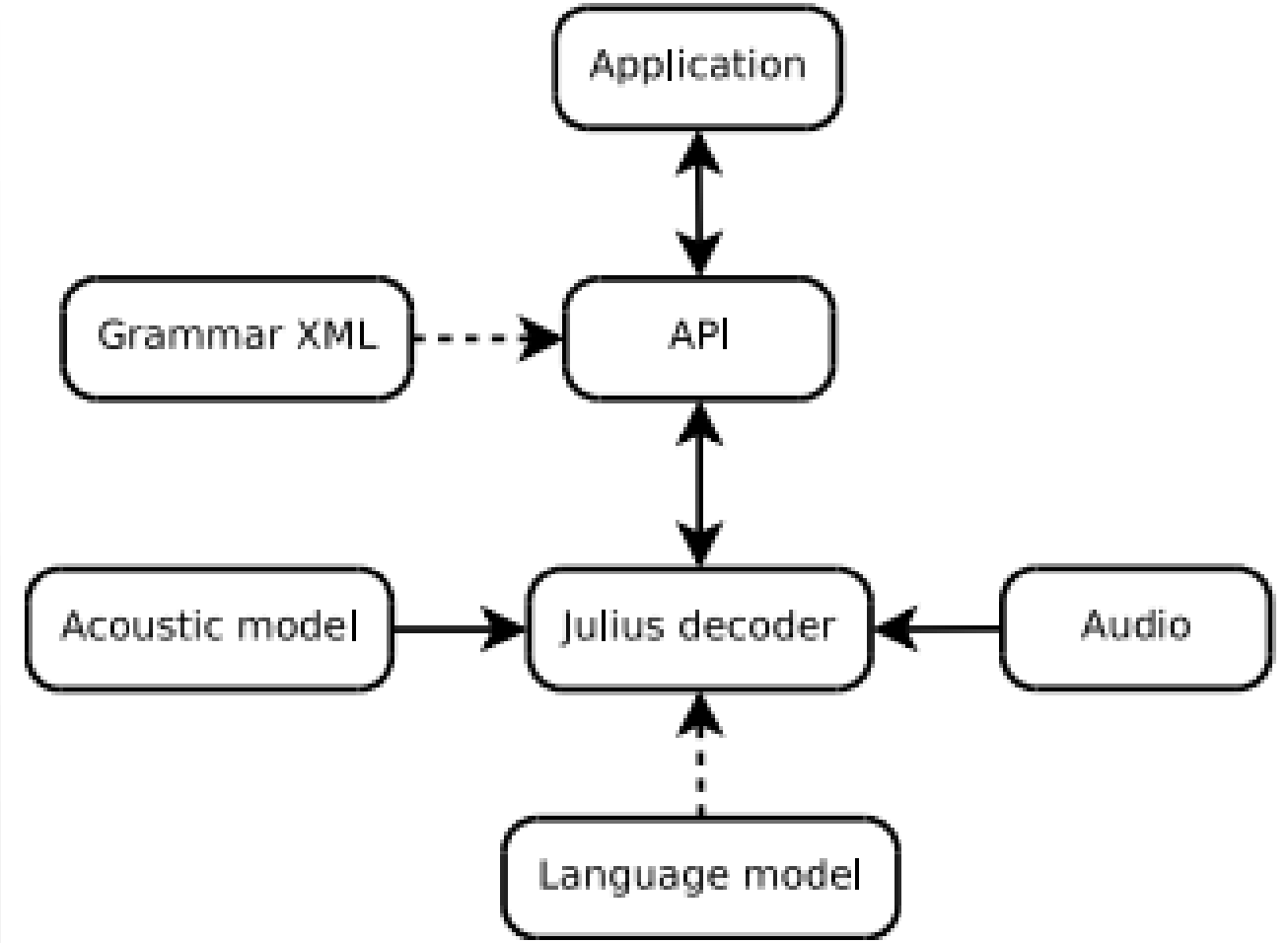


Fonte: Lee e Kawahara (2009, p. 131).

Fundamentação Teórica

- LaspAPI (visão geral)
  - Encapsula a *engine* Julius
  - Abstração no acesso à *engine*

Fundamentação Teórica



Fonte: Klautau et al. (2012).

Fundamentação Teórica

- Lego NXT
  - *Kit* básico de desenvolvimento
  - Unidade central de processamento
  - Sensores
  - Servo-motor
- LeJOS
  - *Firmware* e API
  - Facilidade de implementação

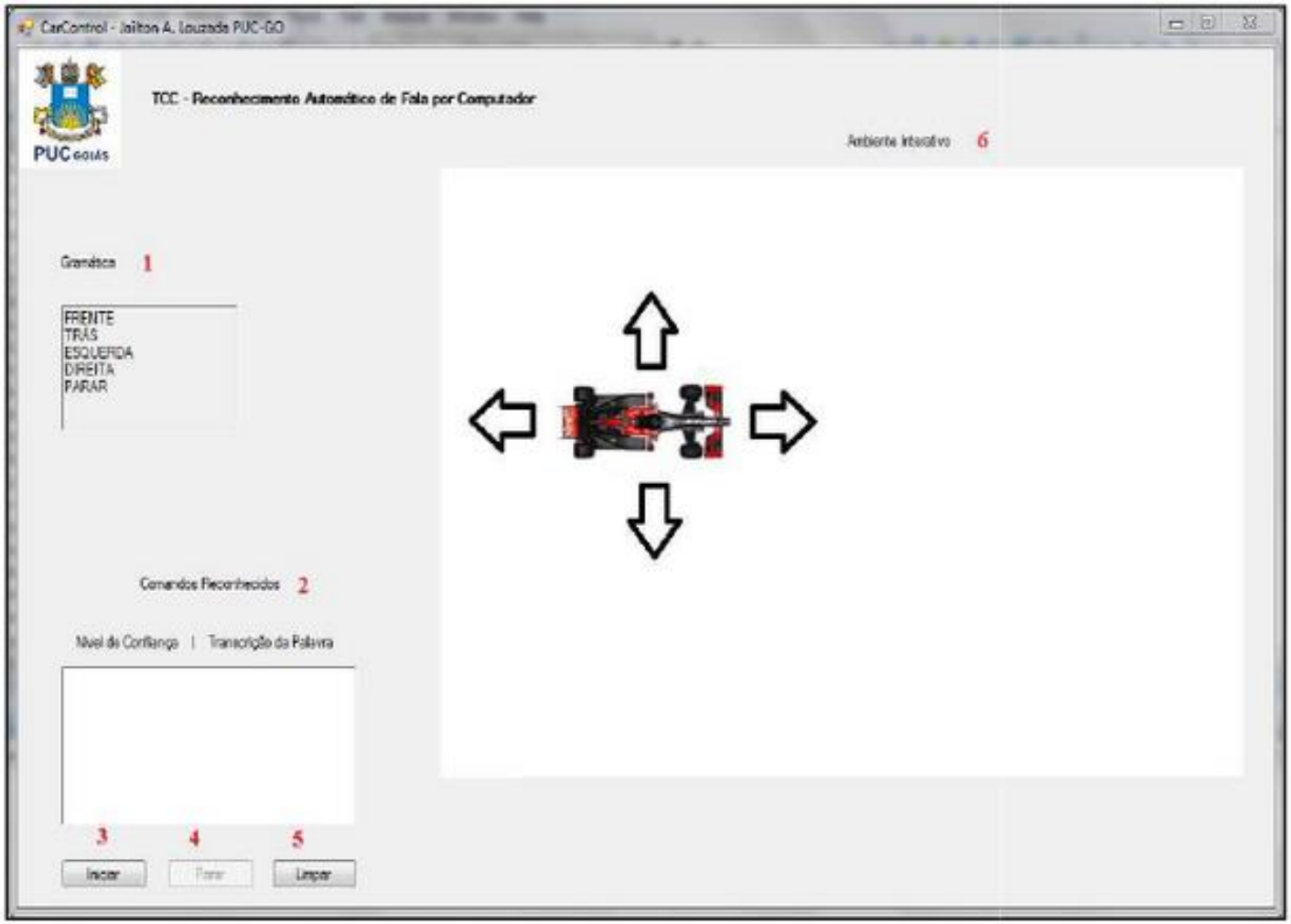
Fundamentação Teórica

Fonte: Lego Group (2013).

Trabalhos Correlatos

- Reconhecimento automático de fala por computador (LOUZADA, 2010)
  - Utiliza a *engine* Julius
  - Reconhecimento de palavra isolada
  - Utiliza XML para modelo de linguagem
  - Vocabulário restrito (5 comandos)
  - Locomoção de objeto gráfico

Trabalhos Correlatos

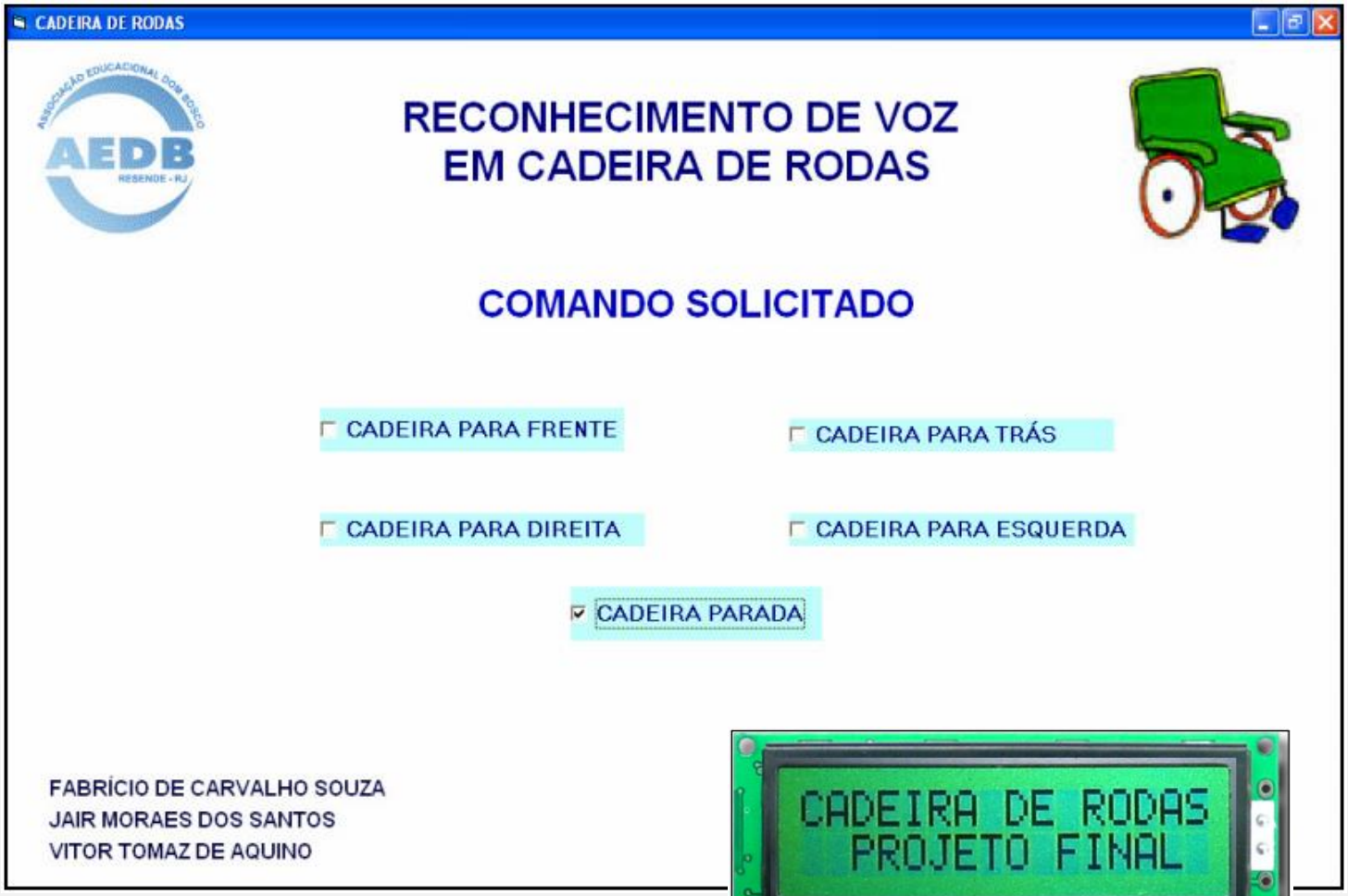


Fonte: Louzada (2010, p. 32).

Trabalhos Correlatos

- Reconhecimento de voz para aplicação em cadeira de rodas (BARCELOS et al., 2008)
  - Utiliza ViaVoice
  - Vocabulário restrito(5 comandos)
  - Locomoção de uma cadeira de rodas

Trabalhos Correlatos



Fonte: Barcelos et al. (2008).

Trabalhos Correlatos

- Reconhecimento de voz através de reconhecimento de padrões (OLIVEIRA, 2002)
  - Utiliza reconhecimento de padrões
  - Utiliza fase de treinamento
  - Vocabulário restrito(7 comandos)
  - Abertura de programas específicos

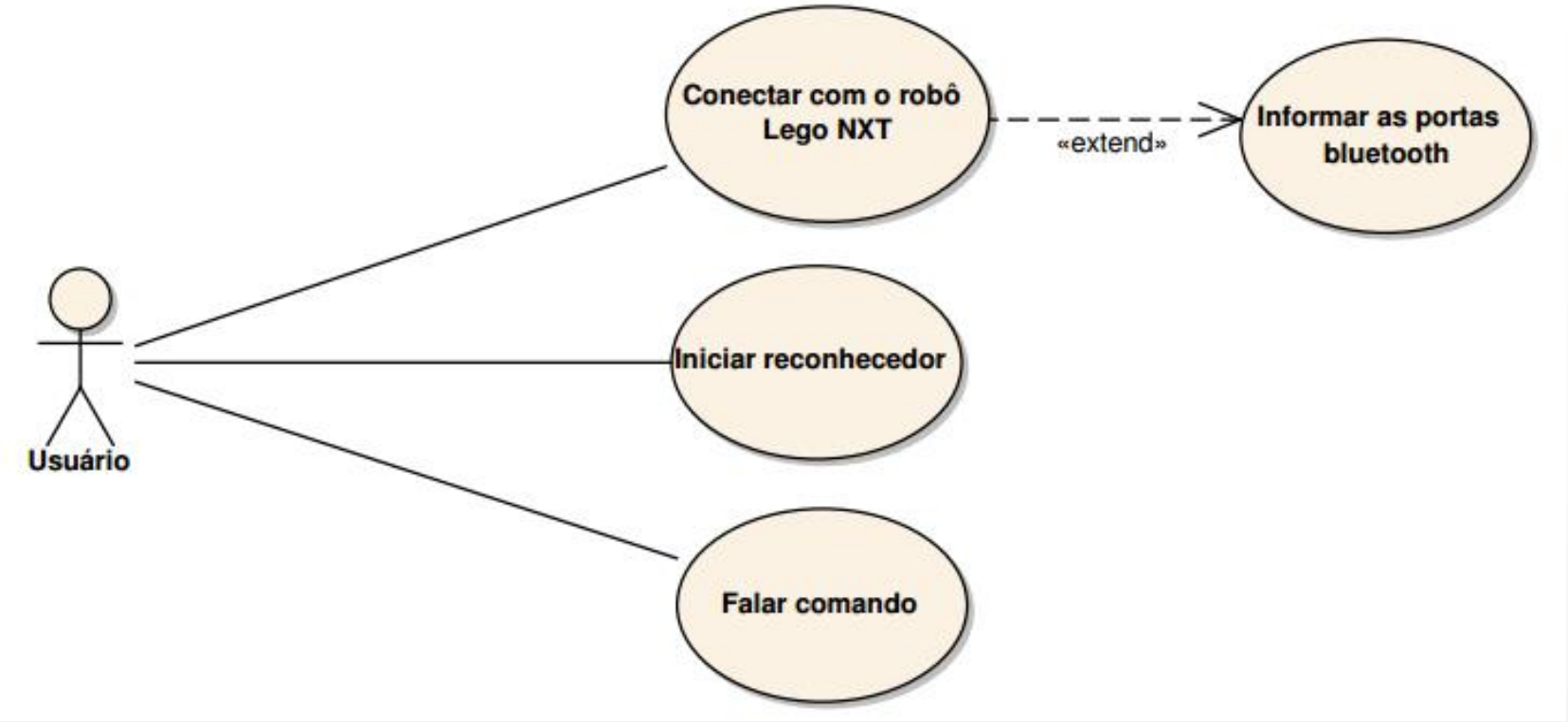
Desenvolvimento: requisitos

- Requisitos funcionais
  - Captar a voz
  - Possuir gramática de comandos
  - Gerar comando de controle para robô

Desenvolvimento: requisitos

- Requisitos não funcionais
  - Utilizar *engine* Julius encapsulada na LapsAPI
  - Usar robô Lego
  - Transmitir comandos por *bluetooth*
  - Implementar a inteligência do robô em Java
  - Implementar a interface de captação de voz em C#

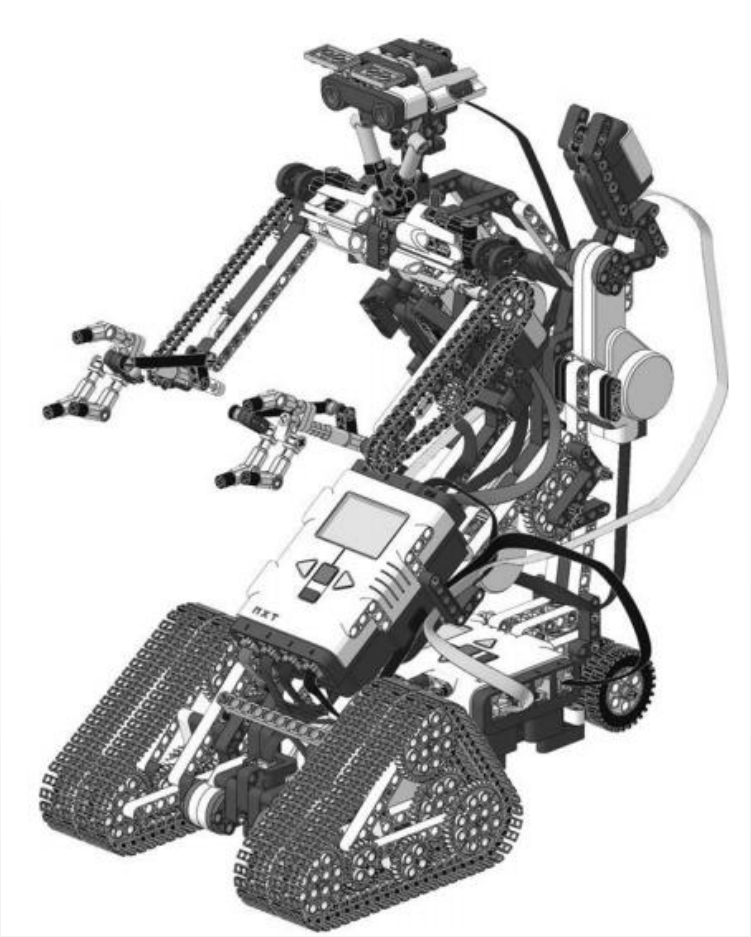
Casos de uso


Conectar com o robô Lego NXT
«extend»
Informar as portas bluetooth
Iniciar reconhecedor
Falar comando
Usuário

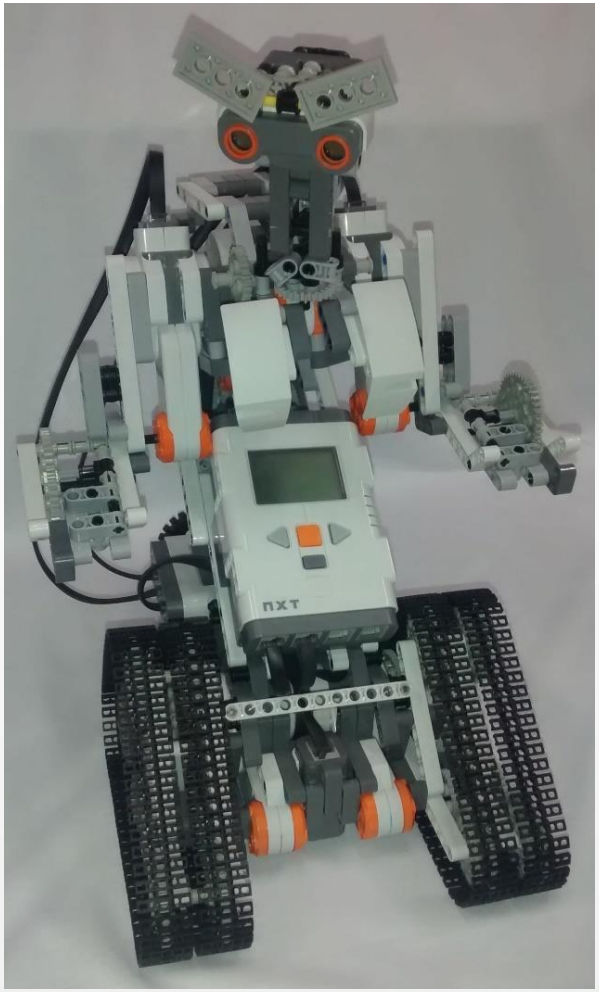
Montagem: robô

- Lego Digital Designer para a elaboração de um modelo virtual do robô
- Modelo inspirado em Benedettelli (2008)
- Dificuldade na montagem e aquisição de peças

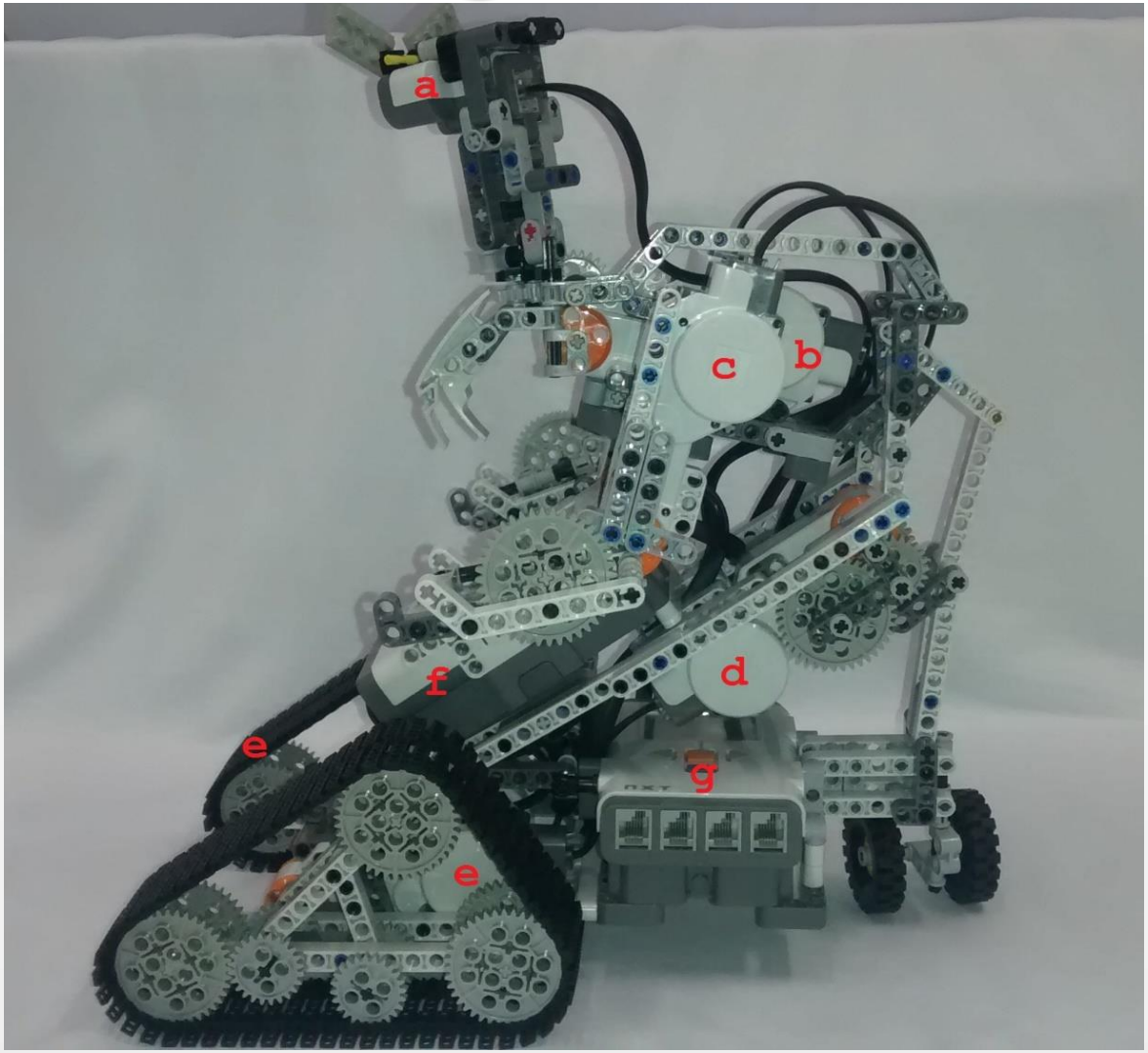
Montagem: robô



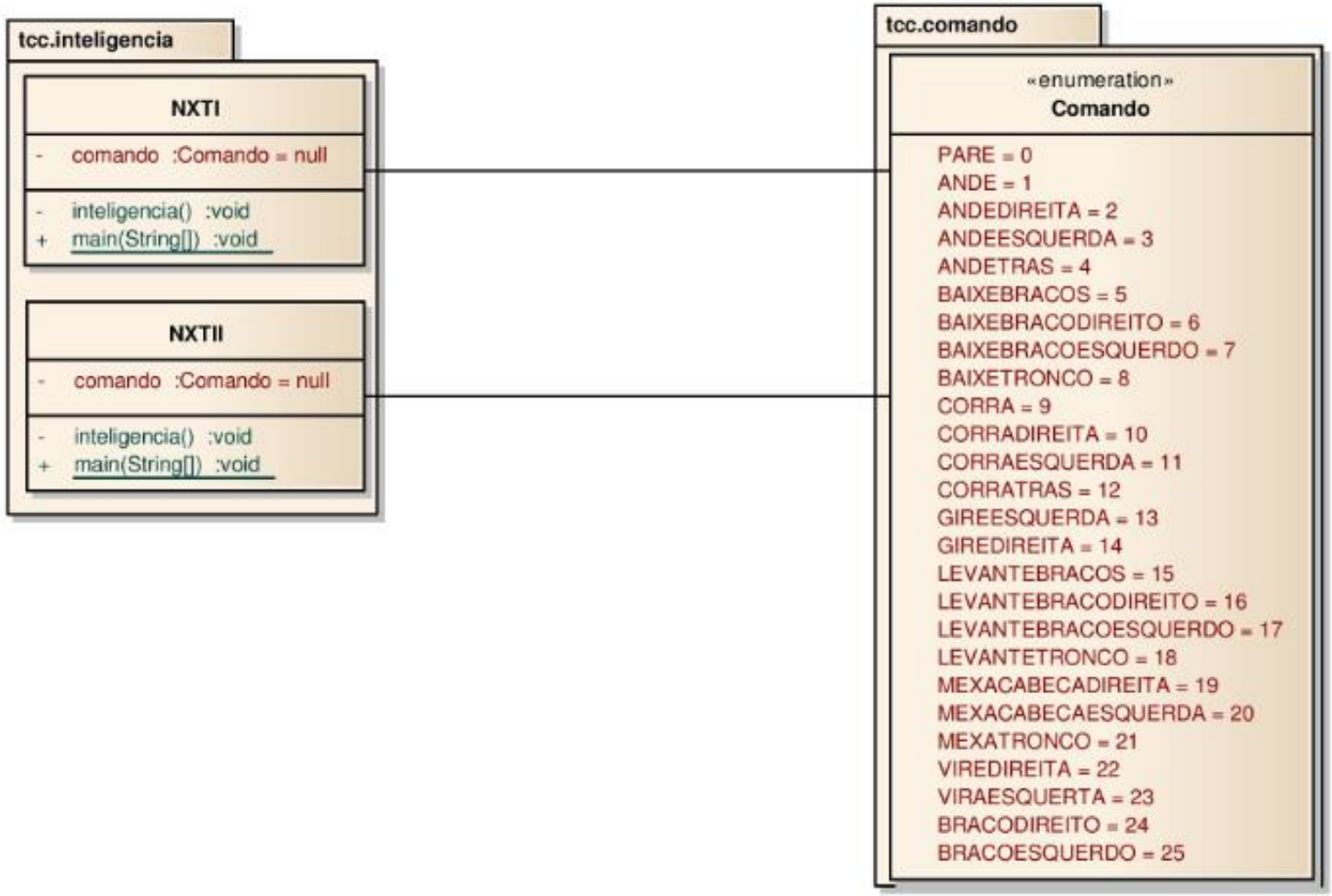
Fonte: Benedettelli (2008, p. 516).

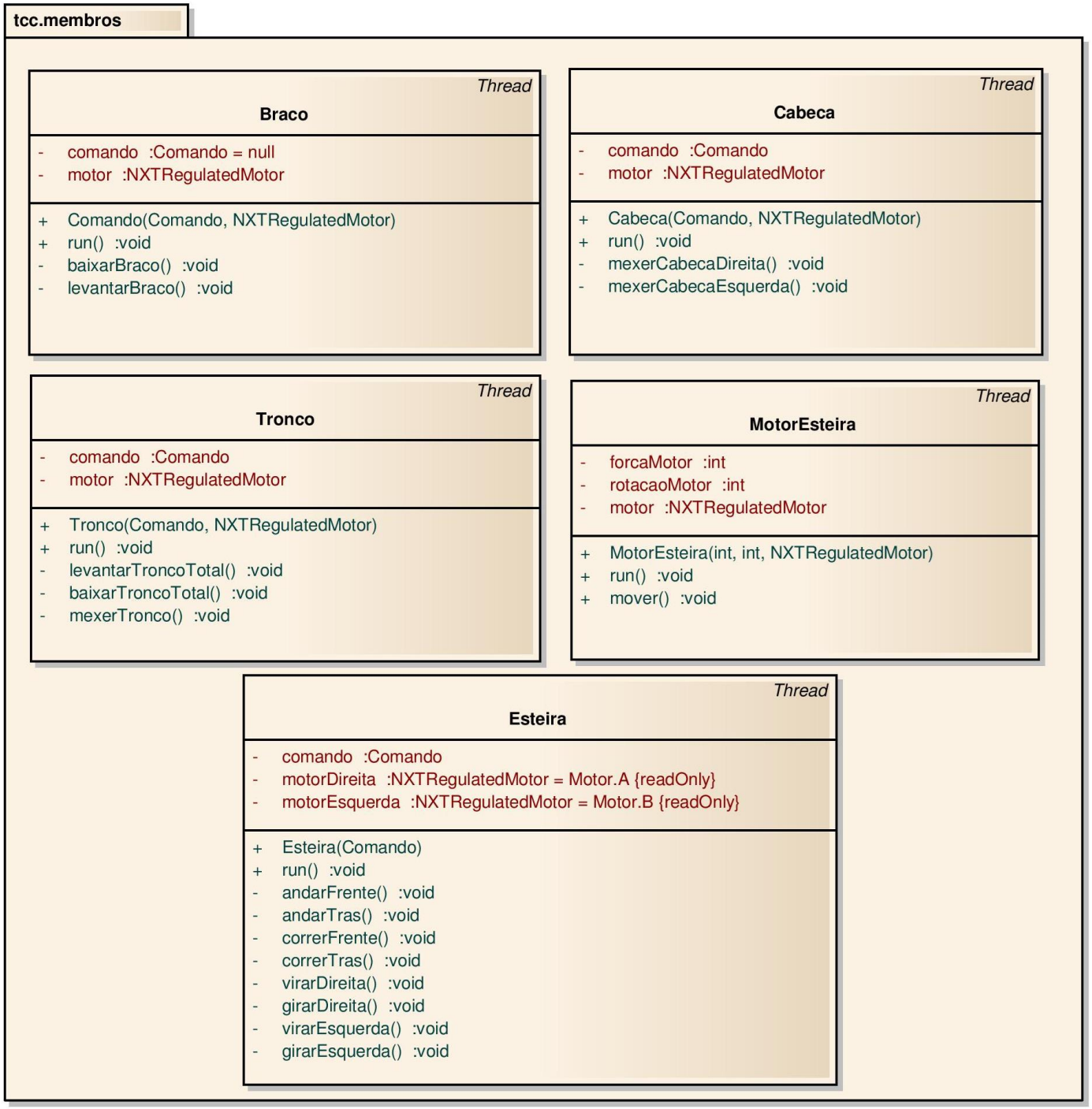


Montagem: robô



Especificação: robô





Implementação: robô

- Conexão via *bluetooth*

```
1.  NXTConnection conexao = Bluetooth.waitForConnection();
2.  conexao.setIOMode(NXTConnection.PACKET);
3.  DataInputStream input = conexao.openDataInputStream();
```

Implementação: robô

- Obtenção do comando falado

```
1.  while (comando != Comando.PARE) {
2.          try {
3.                  for (int i = 0; i < 9; i++) {
4.                          comandoRecebido = (int)
5.                          (input.readByte());
6.                          if ((i == 4)) {
7.                                  comando = Comando.values()
8.                                  [comandoRecebido];
9.                          }
10.                 }
11.         }
12. }
```

Implementação: robô

- Execução do comando recebido

```
1.   switch (comando) {
2.          case ANDE:
3.                 new Esteira(Comando.ANDE).start();
4.                 break;
5.   ...
6.                 default: break;
7.   }
```

Implementação: robô

- Movimentação de um servo-motor

```
1.  NXTRegulatedMotor motor = Motor.A;
2.  motor.waitComplete();
3.  motor.setSpeed(this.forcaMotor);
4.  motor.rotate(this.rotacaoMotor);
```

Especificação: linguagem

- Similar a BNF
- Gera arquivos para o modelo de linguagem
- Restringe o vocabulário, agilizando o reconhecimento e tornando-o mais confiável

Especificação: linguagem

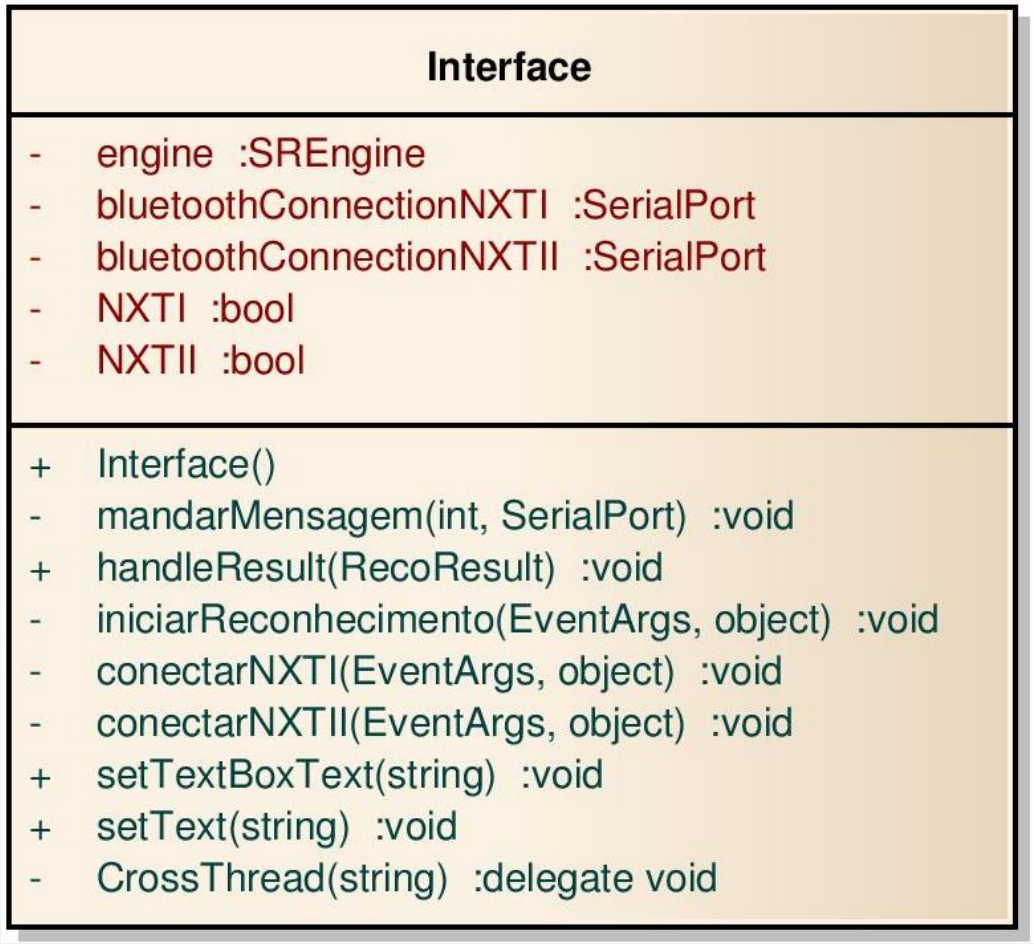
```
 1.   INICIO: silencio_inicio COMANDO silencio_fim
 2.   COMANDO: locomover
 3.   COMANDO: locomover para direcao
 4.
 5.   COMANDO: mover os membros
 5.   COMANDO: mover o membro lado
 6.   COMANDO: mover o tronco
 7.
 8.   COMANDO: membro lado
 9.   COMANDO: cabeca para ONDE
10.
11.   COMANDO: mexer a cabeca para ONDE
12.   COMANDO: mexer o tronco
13.
14.   COMANDO: virar para ONDE
15.
16.   ONDE: direita
17.   ONDE: esquerda
18.
19    COMANDO: parar
```

```
%silencio_inicio
    <s>       sil
%silencio_fim
    </s>      sil
%locomover
    ande      a~  dZ i
    corra     k o R  a
%mover
    baixe     b a j S i
    levante   l e v a~  tS i
%virar
    gire      Z i r i
    vire      v i r i
%mexer
    mexa      m e S  a
%parar
    pare      p a r i
%cabeca
    cabeça    k a b e s a
%tronco
    tronco    t r o~  k u
%membro
    braço     b r a s u
%membros
        braços b r a s u s
%direita
    direita   dZ i r e j ta
%esquerda
    esquerda  e s k  e R da
%direcao
    direita   dZ i r e j ta
    esquerda  e s k  e R da
    trás      t r a j s
    frente    f r e~  tS i
%lado
    direito   dZ i r e j tu
    esquerdo e s k  e R d u
%a
    a         a
%o
    o         u
%os
    os        u s
%para
    para      p a r a
```

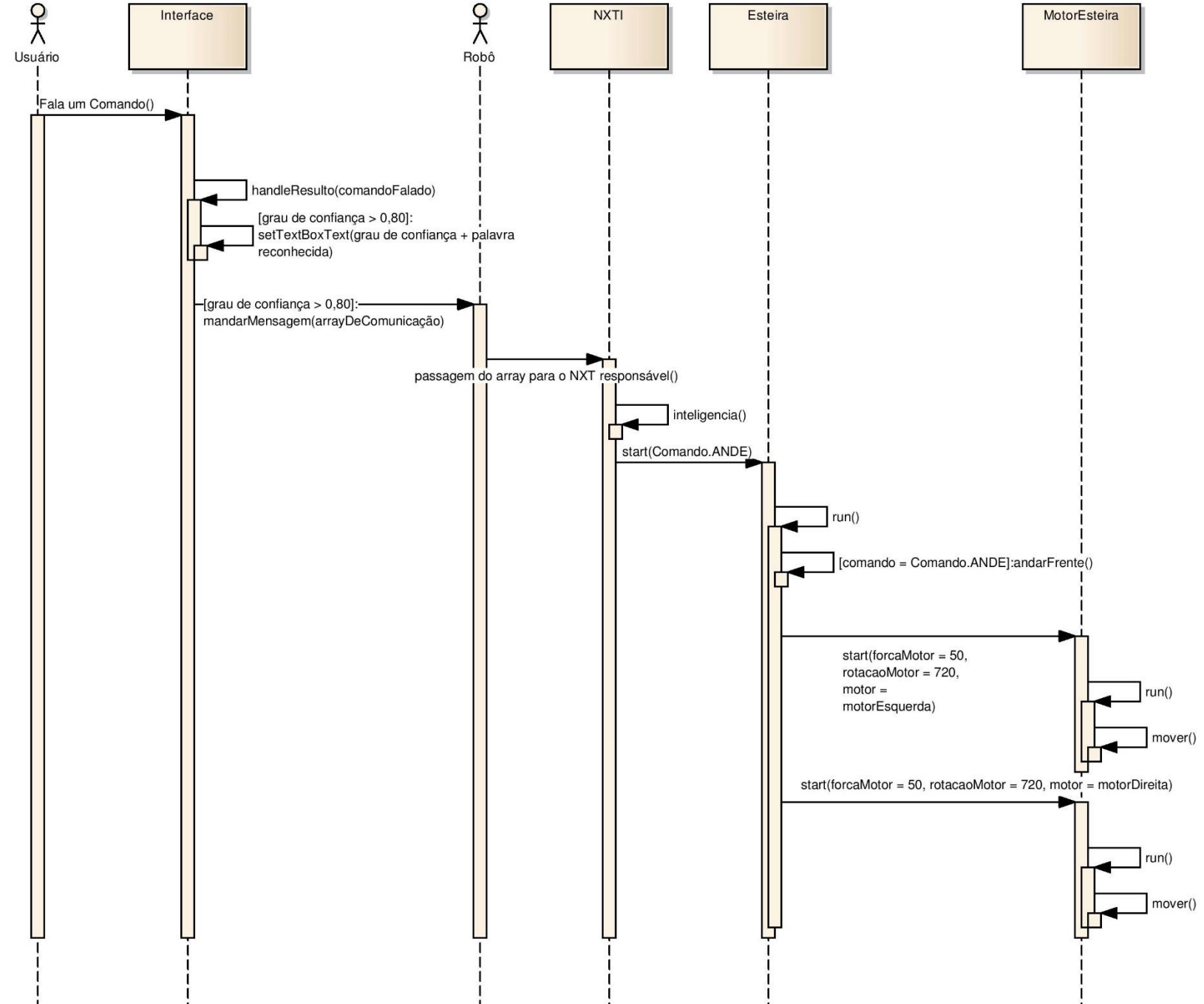
Compilação: linguagem

```
C:\Users\Deivid\Desktop\TCC\Julius\julius-4.3.1-win32bin\bin>mkdfa.pl comandos
comandos.grammar has 14 rules
comandos.voca    has 19 categories and 26 words
---
Warning: dfa_minimize not found in the same place as mkdfa.pl
Warning: no minimization performed
Now parsing grammar file
Now modifying grammar to minimize states[1]
Now parsing vocabulary file
Now making nondeterministic finite automaton[17/17]
Now making deterministic finite automaton[17/17]
Now making triplet list[17/17]
---
generated: comandos.dfa comandos.term comandos.dict
```

Especificação: reconhecimento de voz



Executando o comando Ande



Implementação: reconhecimento de voz

- Inicialização da *engine* Julius

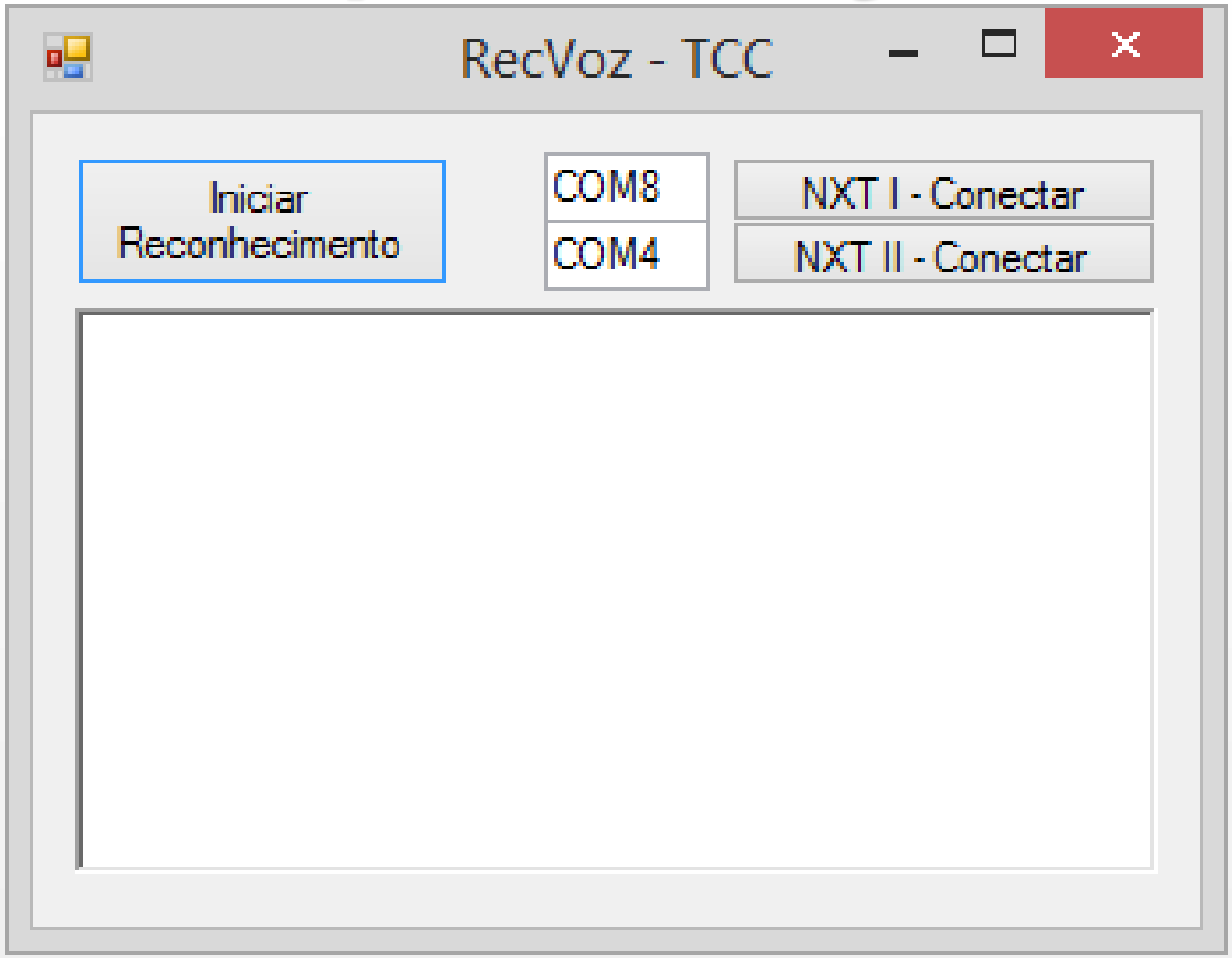
```
1.    private SREngine engine = null;
2.    ...
3.    engine = new SREngine(@".\LaPSAM\LaPSAM1.5.jconf");
4.    engine.startRecognition();
```

Implementação: reconhecimento de voz

- Passagem do comando reconhecido para o robô

```
1.    if (result.getConfidence() > 0.80)
2.    {
3.          setTextBoxText(result.getConfidence()
4.          + " | "
5.          + result.getUterrance() + "\n");
6.          switch (result.getUterrance())
7.          {
8.                case @" <s> pare </s>":
9.                      mandarMenssagem(
10.                     bluetoothConnectionNXTI, 0);
11.                     mandarMenssagem(
12.                     bluetoothConnectionNXTII, 0);
13.                     break;
14.                     ....
15.               default: break;
16.         }
17.   }
```

Operacionalidade da Implementação



Resultados e Discussões

- Elaboração da linguagem para controle do robô
- Funcionamento do programa
- Grau de confiança na sentença
- Montagem do robô

Resultados e Discussões

Característica/ferramenta	LOUZADA (2010)	BARCELOS et al. (2008)	OLIVEIRA (2002)	trabalho desenvolvido
reconhecimento usando engine Julius	X			X
reconhecimento usando IBM ViaVoice		X		
reconhecimento nativo			X	
necessita de treinamento			X	
independente de locutor	X		X	X
interage com meio físico		X		X
linguagem restrita	X	X	X	X
reconhecimento de fala contínua				X

Conclusões

- Elaboração de aplicação para reconhecimento de voz
- Controle de robô por voz
- Elaboração da linguagem para controle do robô
- Envio de comandos para o robô

Sugestões

- Aumentar o vocabulário, para permitir a passagem de parâmetro
- Utilizar os sensores do Lego NXT
- Portar o reconhecimento de voz para o Lego NXT
- Desenvolver a aplicação para a comunicação com Arduino

Sugestões

- Criar linguagem para outros tipos de robô, como bípedes e rastejadores
- Criar um ambiente para ensinar programação utilizando o Lego NXT e controle por voz

Demonstração