AMBIENTE DE DESENVOLVIMENTO E DEPURAÇÃO PARA O MICROCONTROLADOR DSPIC30F4011

Aluno: Rafael Angelo Gardini

Orientador: Antonio Carlos Tavares

FURB
UNIVERSIDADE REGIONAL DE BLUMENAU

ROTEIRO

- INTRODUÇÃO
- FUNDAMENTAÇÃO TEÓRICA
- DESENVOLVIMENTO
  - REQUISITOS
  - ESPECIFICAÇÃO
  - IMPLEMENTAÇÃO
  - FERRAMENTAS UTILIZADAS
  - RESULTADOS
- CONCLUSÃO
  - EXTENSÕES
- APRESENTAÇÃO DO PROTÓTIPO

FURB
UNIVERSIDADE REGIONAL DE BLUMENAU

INTRODUÇÃO

- MOTIVAÇÃO
  - Alta demanda por projetos embarcados utilizando microcontroladores;
  - Crescimento contínuo da automação de soluções.
- OBJETIVOS DO TRABALHO
  - Prover um ambiente de desenvolvimento e depuração para o microcontrolador DSPIC30F4011;
  - Gerar, Compilar e Gravar código no microcontrolador.

FURB
UNIVERSIDADE REGIONAL DE BLUMENAU

FUNDAMENTAÇÃO TEÓRICA

- DESENVOLVIMENTO DE FORMA A TORNAR A APLICAÇÃO UM AMBIENTE RAD;

- GERADORES DE CÓDIGO
  - Analisador Léxico;
  - Analisador Sintático;
  - Analisador Semântico;
  - GALS.

FUNDAMENTAÇÃO TEÓRICA 2

- DESENV. PARA MICROCONTROLADORES
  - DSPIC30F4011;
  - LCD 16x2;
  - LED 7 Segmentos;
  - PORTA SERIAL RS232.

- TRABALHOS CORRELATOS
  - ZEXUS++: Ambiente gerador de código C++ para PDAs da empresa ZEXUS;
  - FLOWCHART EDITOR: Editor fluxo programático que, através fluxogramas, realiza a geração de código assembly para o PIC16C84.

FURB
UNIVERSIDADE REGIONAL DE BLUMENAU

REQUISITOS

- Permitir a criação de projetos de implementação para microcontroladores, podendo selecionar os parâmetros e modelos possíveis (RF);
- Possuir um editor visual que possibilite a técnica *drag and drop* (RF);
- Possuir uma paleta de componentes contendo as interfaces de entrada e saída de um microcontrolador (RF);
- Possuir uma linguagem de programação intuitiva e objetiva através da construção de analisadores léxicos, sintáticos e semânticos e possibilitar a interação da mesma com os componentes visuais (RF);

FURB
UNIVERSIDADE REGIONAL DE BLUMENAU

REQUISITOS 2

- Interpretar a linguagem criada previamente e traduzir o código para a linguagem C (RF);
- Compilar o código gerado através da ferramenta Mikroc (RF);
- Gravar o código no microcontrolador através da ferramenta Winpic800 (RF);
- Possuir um módulo de depuração (RF).

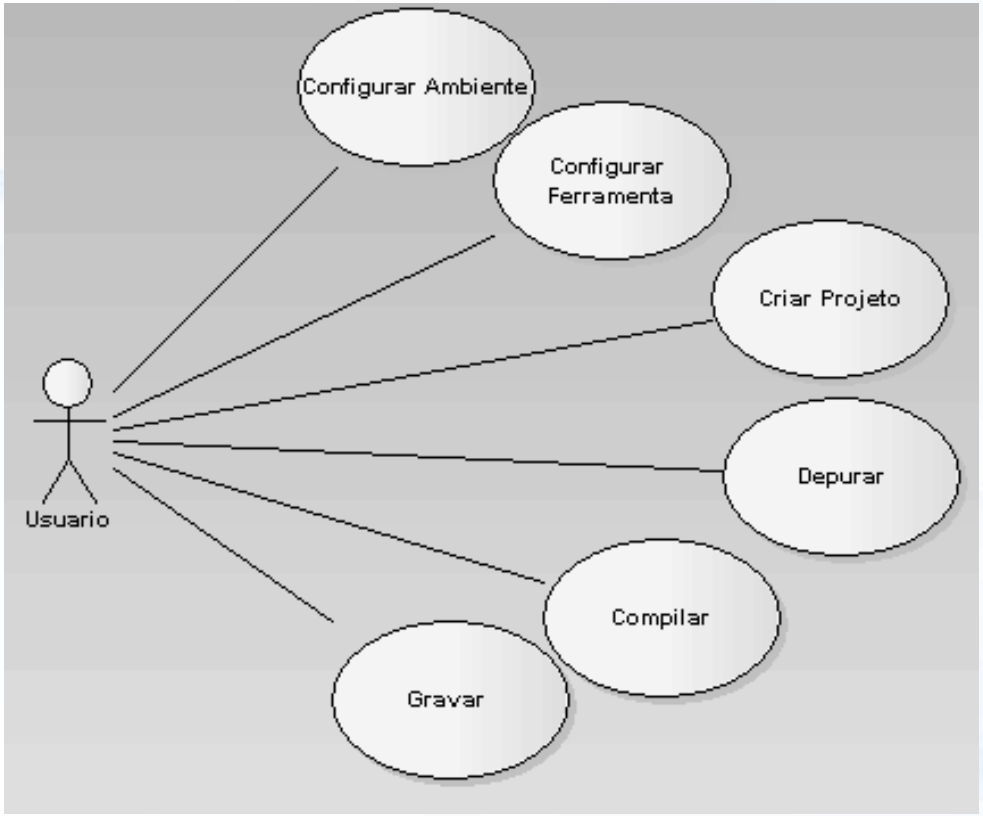
FURB
UNIVERSIDADE REGIONAL DE BLUMENAU

ESPECIFICAÇÃO
Ferramentas Utilizadas

- Utilização da ferramenta Enterprise Architect 7 na criação dos diagramas de casos de uso, seqüência e de classes;

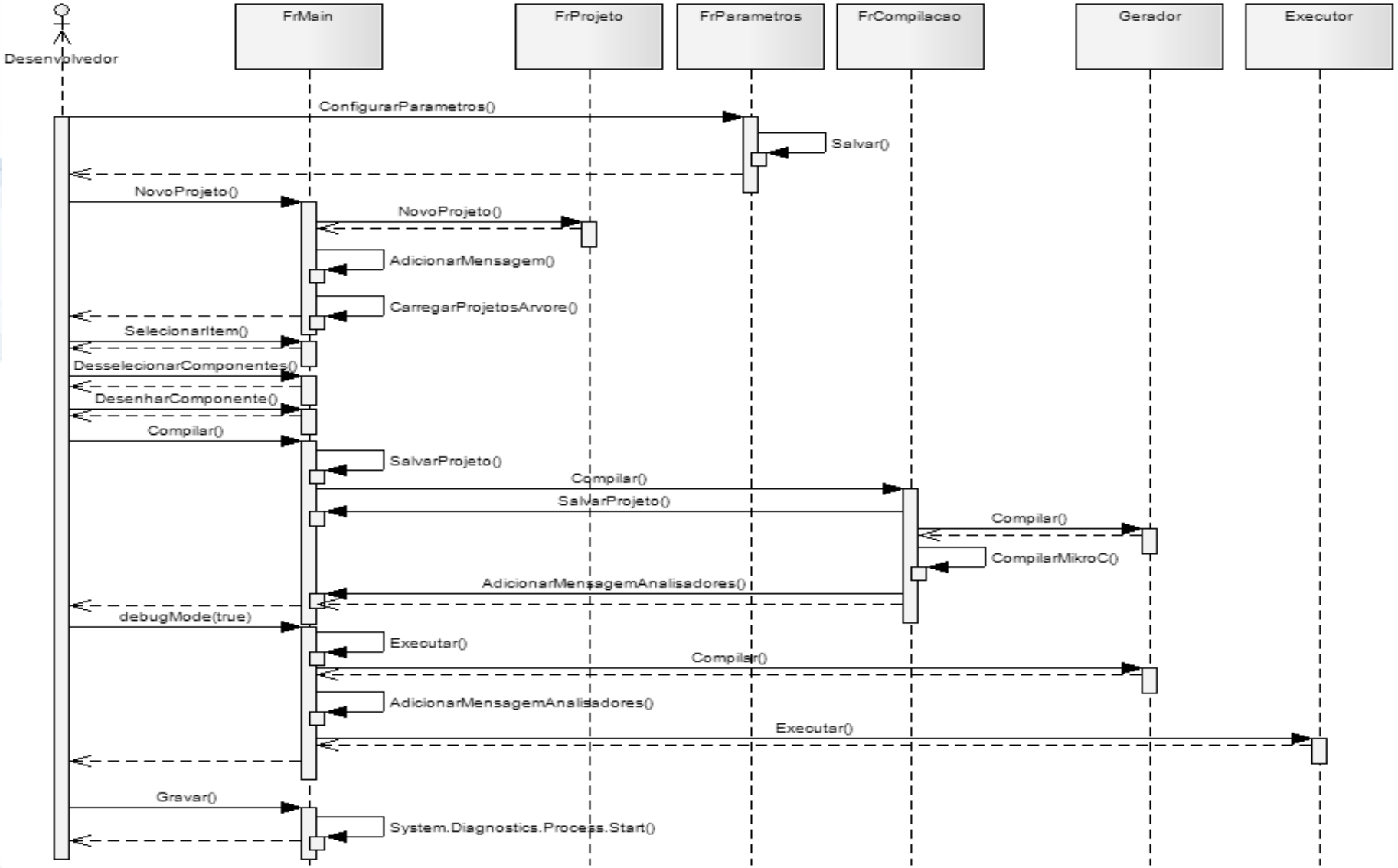
- Utilização da ferramenta GALS na criação das regras BNF para criação da linguagem.

ESPECIFICAÇÃO
Casos de Uso



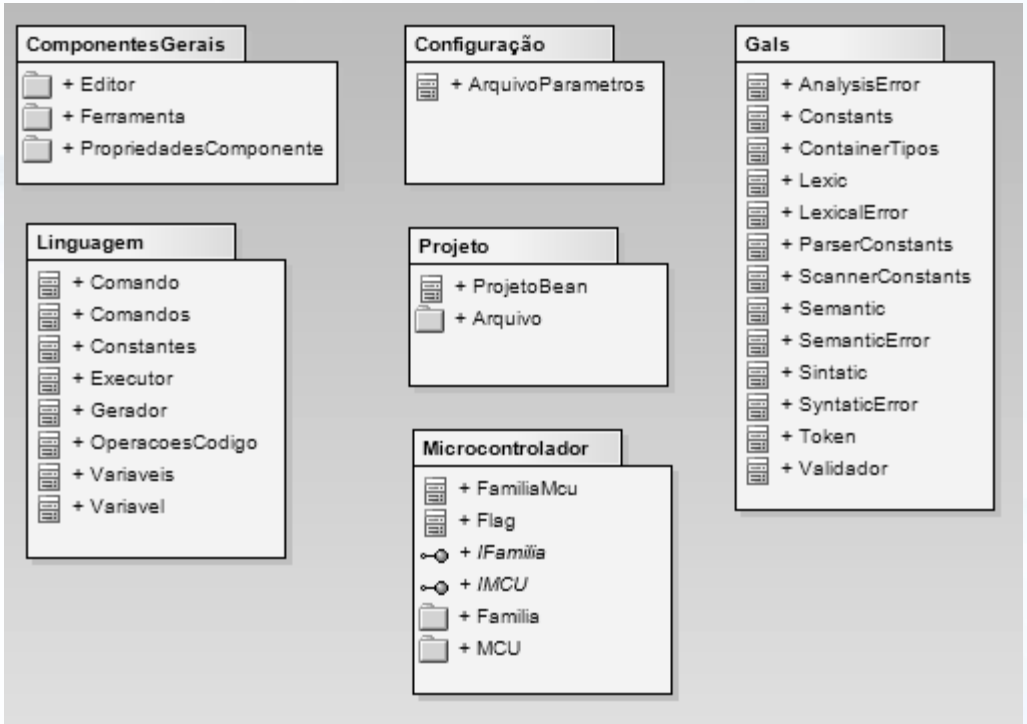
ESPECIFICAÇÃO

Diagrama de Seqüência



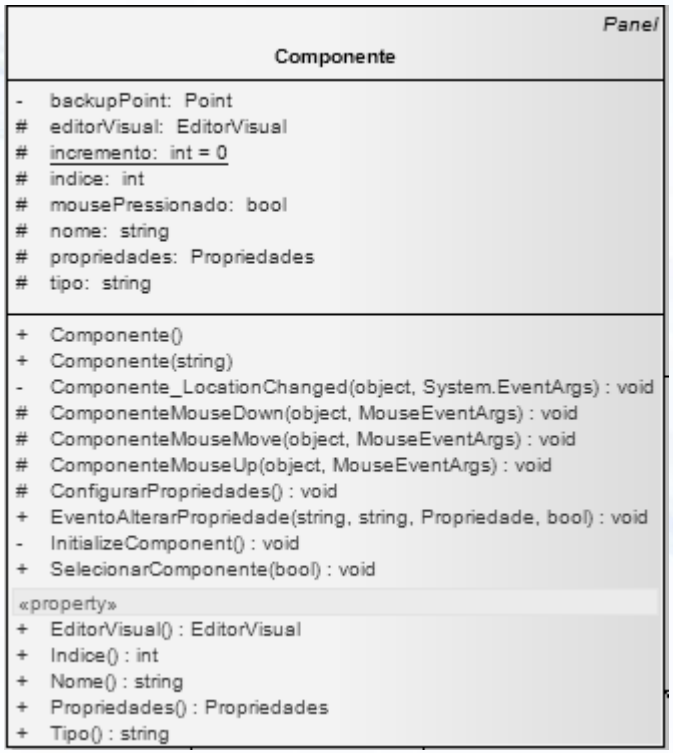
ESPECIFICAÇÃO

Diagrama de Classes (Pacotes Principais)



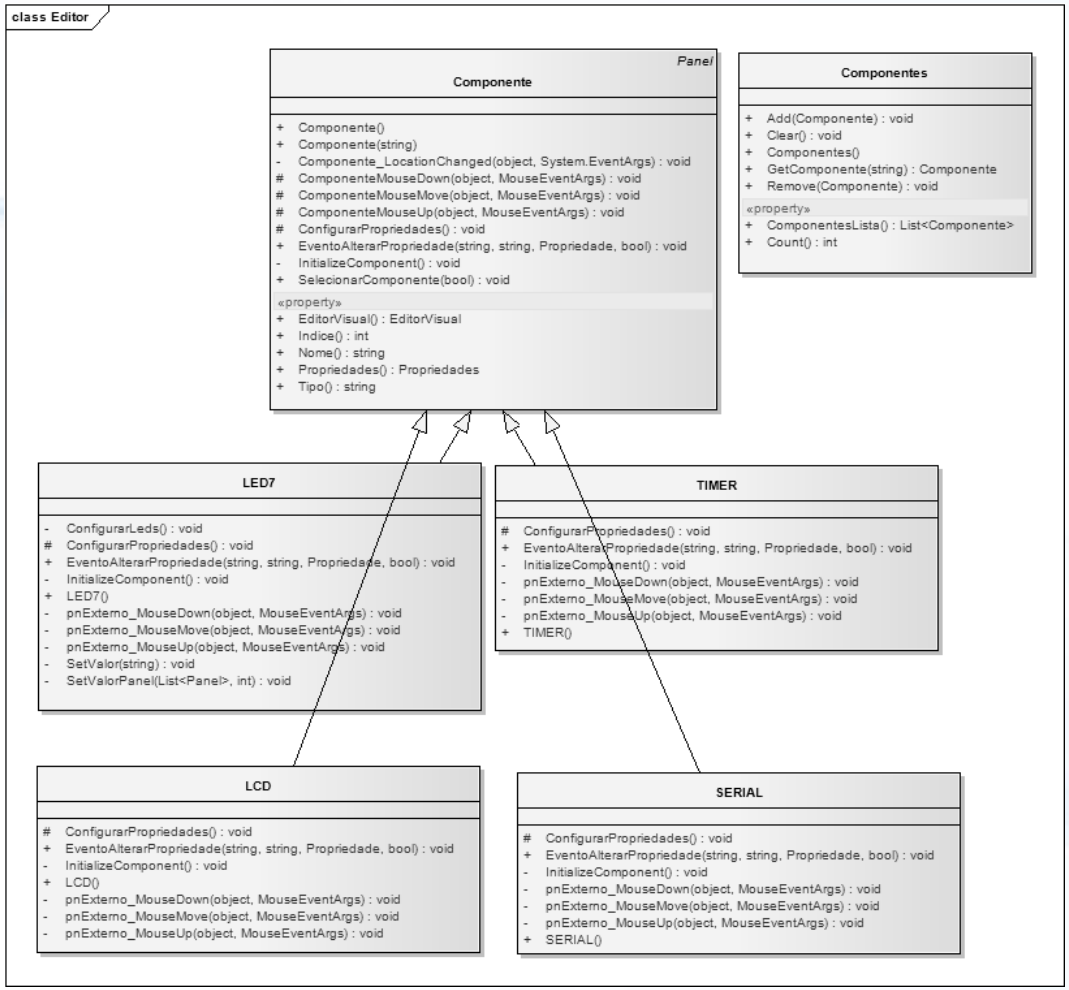
ESPECIFICAÇÃO
Diagrama de Classes (Classe Componente)

- Cria a padronização dos componentes visuais.



ESPECIFICAÇÃO

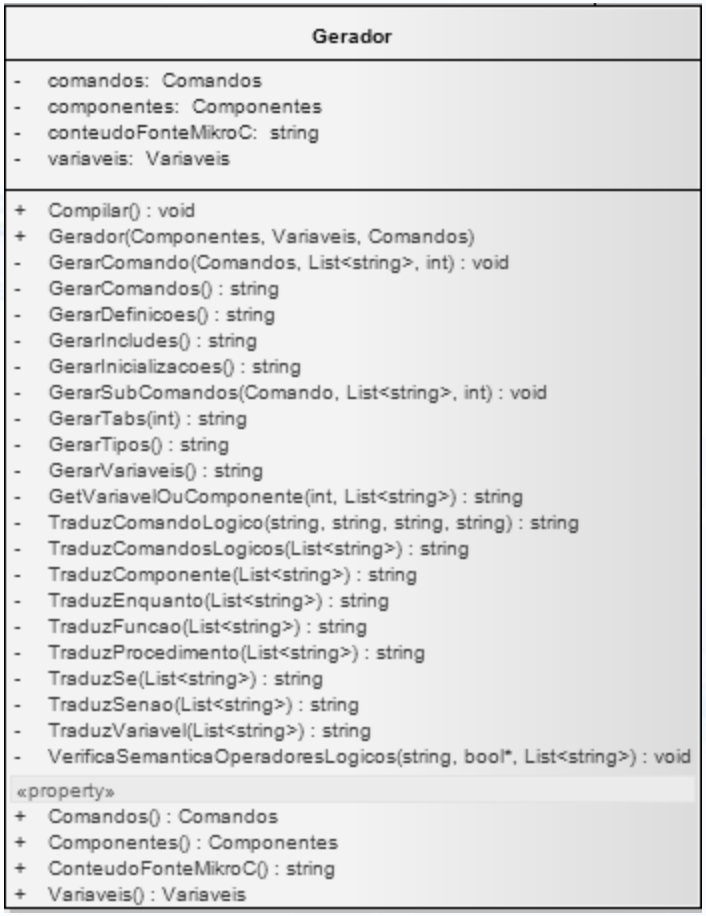
Diagrama de Classes (Classe Componente Extensão)



FURB
UNIVERSIDADE REGIONAL DE BLUMENAU

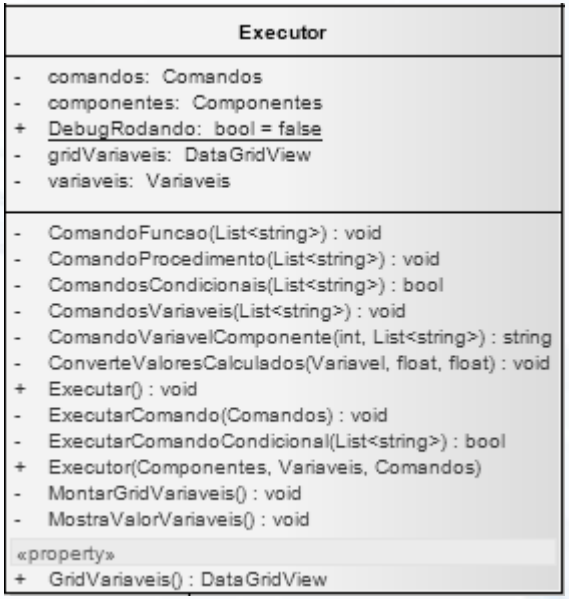
ESPECIFICAÇÃO

Diagrama de Classes (Classe Gerador)



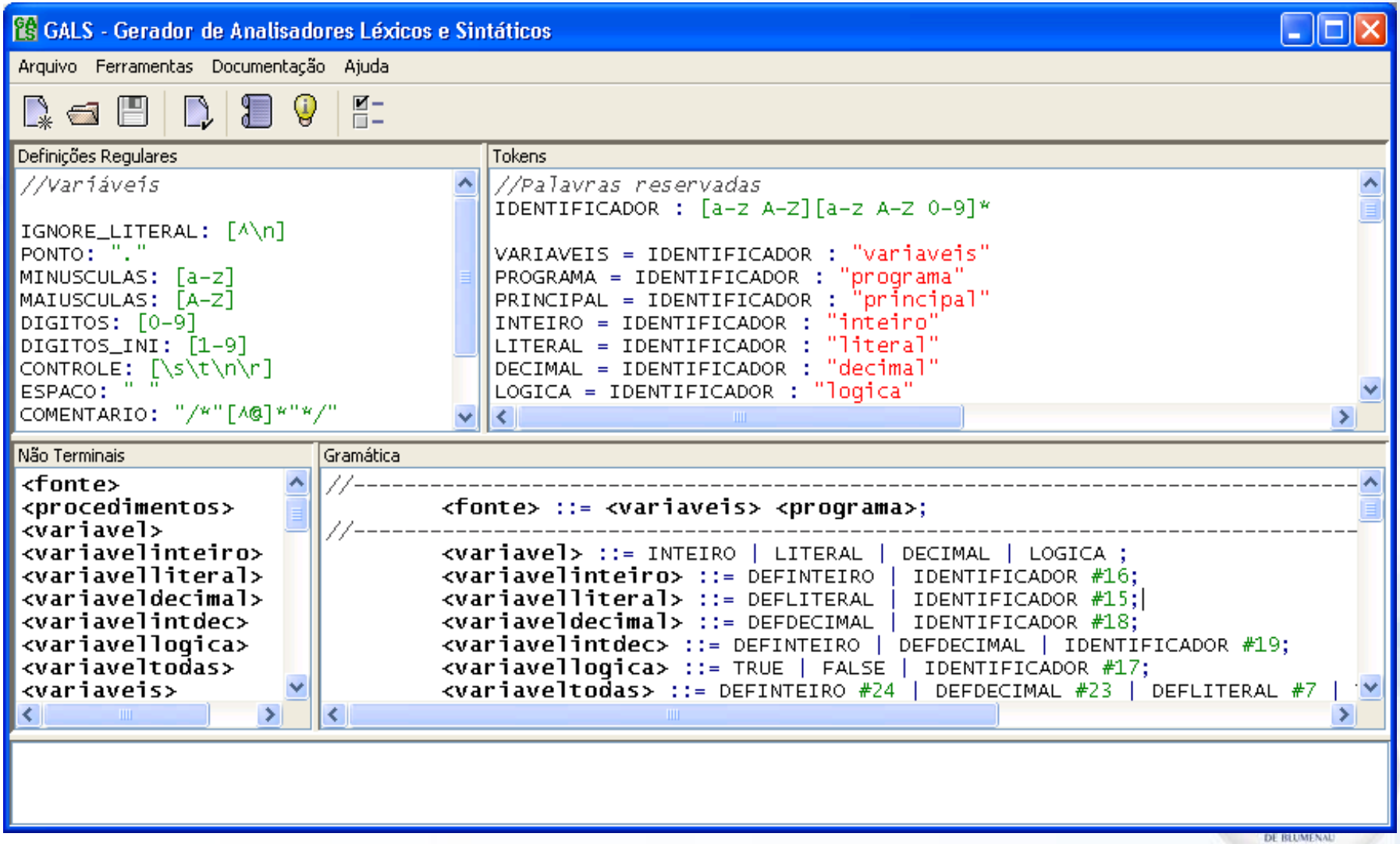
FURB
UNIVERSIDADE REGIONAL DE BLUMENAU

ESPECIFICAÇÃO
Diagrama de Classes (Classe Executor)



ESPECIFICAÇÃO

BNF

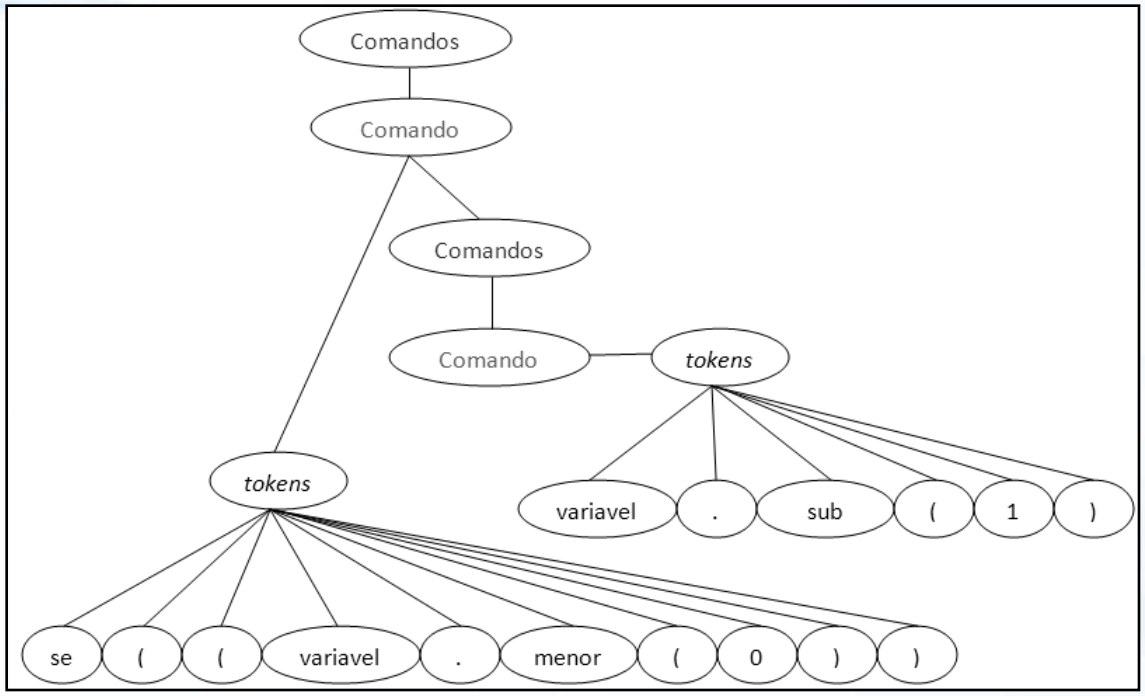


IMPLEMENTAÇÃO
Ferramentas e técnicas utilizadas

- Utilização da ferramenta Microsoft Visual Studio 2008 Express na codificação da ferramenta;
- Utilização da ferramenta GALS na geração dos analisadores léxicos sintáticos em linguagem Java;
- Tradução do código gerado pela ferramenta GALS para linguagem C#;
- Criação de APIs auxiliares em linguagem C de forma a cobrir as limitações do software MikroC.

IMPLEMENTAÇÃO
Geração de código

- Através de uma estrutura de árvore, os *tokens* são organizados e traduzidos.



IMPLEMENTAÇÃO
Geração de código

- Trecho de tradução.

```
case "igual":
{
   switch (tipo)
   {
      case Variavel.LITERAL:
      {
         comando += "strcmp(" + variavelToken + ", " +
                                parametroToken + ") == 0";
      }
      break;
      default:
         comando += variavelToken + " == " + parametroToken;
      break;
   }
      }
```

IMPLEMENTAÇÃO
Interpretação de código

- Segue os mesmos padrões de estrutura em árvore.

```
switch (tokens[2])
{
   case "igual":
   {
      return variavel.Valor == parametro;
   }
   case "naoigual":
   {
      return variavel.Valor != parametro;
   }
   case "menor":
   {
       return double.Parse(variavel.Valor) < double.Parse(parametro);
   }
   case "maior":
   {
       return double.Parse(variavel.Valor) > double.Parse(parametro);
   }
      }
```

IMPLEMENTAÇÃO
Bibliotecas prontas

- Permitem a utilização de funções que não estão contidas na API do MikroC.

Função	Descrição
`Trim`	Função que retira os espaços à direita e a esquerda da cadeia de caracteres.
`Substring`	Captura um subtexto dentro de outro texto passado através de um índice inicial e o tamanho do subtexto desejado.
`indexOf`	Busca o índice de um subtexto dentro de um texto passado.
`lastIndexOf`	Busca o ultimo índice de um subtexto dentro de um texto passado.
`charAt`	Busca um caractere através de um índice presente dentro do texto passado.

```
typedef struct
{
            //propriedades do componente
} COMPONENTE;
void installCOMPONENTE(COMPONENTE componente)
{
            //função que realiza a instalação do componente
}
void handleCOMPONENTE(COMPONENTE componente)
{
            //função que invoca a chamada das propriedades
}
```

IMPLEMENTAÇÃO
Linguagem da ferramenta

- Objetiva reduzir a quantidade de código implementado.

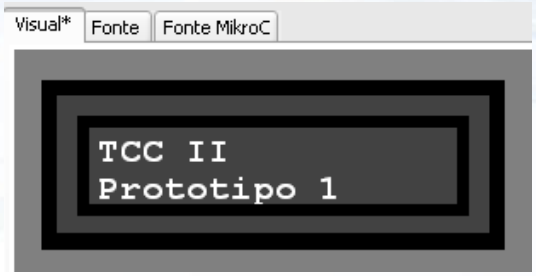
```
variaveis
{
	// declaração de variáveis
	texto : literal;
}
programa principal
{
	// armazena o conteúdo na variavel
texto
	texto = "Teste TCC II";
	// escreve a variavel texto na
	// linha 1 do LCD
	lcd1.Linha1 = texto;
}
```

```
// arquivos incluídos no código fonte
#include "Utils.h"
#include "LCD.h"
#include "LED7.h"
#include "SERIAL.h"
#include "TIMER.h"
// criação dos tipos boolean e string
typedef unsigned short boolean;
typedef char string[20];
// criação dos valores true e false não presentes na linguagem C
#define true 0
#define false 1
// declaração do component LCD
LCD lcd1;
// declaração da variável texto
string texto;
void main()
{
	// instalação do componente LCD
	installLCD(lcd1);
	// inicialização dos dados do componente LCD
	strcpy(lcd1.Linha1, "");
	strcpy(lcd1.Linha2, "");
	lcd1.Cursor = false;
	// instalação do componente LCD
	handleLCD(lcd1);
	// atribui o conteúdo a variável texto
	strcpy(texto, "Teste TCC II");

	// copia o conteúdo da variável para a propriedade
	strcpy(lcd1.Linha1, texto);
	// escreve o conteúdo das propriedades no componente LCD
	handleLCD(lcd1);
}
```

FURB
UNIVERSIDADE REGIONAL DE BLUMENAU

IMPLEMENTAÇÃO
Protótipo




```
variaveis
{
            texto: literal;
}
programa principal
{
            texto = lcd1.Linha1;
            texto.concatenar(" LCD");
            lcd1.Linha1 = texto;
}
```

FURB
UNIVERSIDADE REGIONAL DE BLUMENAU

IMPLEMENTAÇÃO
Comparativo

O comparativo é realizado utilizando as características da ferramenta do presente estudo e os trabalhos correlatos.

Funcionalidade	ADDM30F	Zexus++	Flowchart Editor
Editor visual	Sim	Sim	Sim
Edição de código fonte	**Sim**	**Sim**	**Não**
Paleta de componentes	Sim	Sim	Sim
Editor de propriedades	**Sim**	**Sim**	**Não**
Lista de projetos	**Sim**	**Não**	**Não**
Tradução de código	Sim	Sim	Sim
Compilação	**Sim**	**Sim**	**Não**
Gravação no dispositivo	**Sim**	**Não**	**Não**
Endentação de código	**Sim**	**Sim**	**Não**
Sintaxe colorida	**Sim**	**Não**	**Não**

CONCLUSÃO

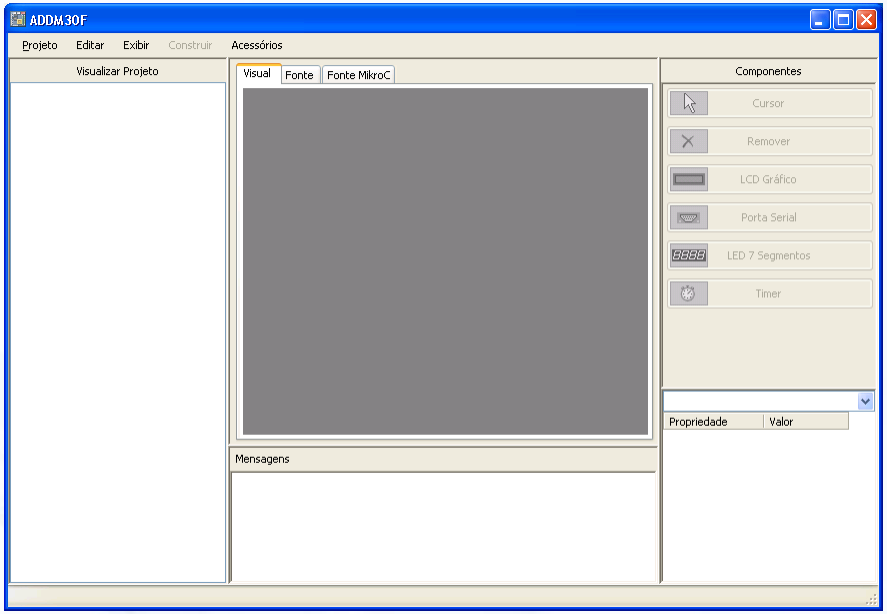
- Cumpriu o que se esperava em relação ao que foi inicialmente proposto;
- A ferramenta Mikroc dificultou o desenvolvimento em alguns aspectos, como a falta de algumas funções auxiliares que tiveram de ser criadas, bem como a falta da funcionalidade de chamada externa através de parâmetros;
- A ferramenta Winpic800 atendeu de forma excelente, possuindo uma chamada externa simples de ser executada;
- A ferramenta GALS foi de suma importância no desenvolvimento deste trabalho, por tornar a especificação dos analisadores léxicos e sintáticos muito simples e intuitiva;
- Possui uma linguagem de programação limitada contando apenas com poucas funções de manipulação e um conjunto de componentes pequeno.

FURB
UNIVERSIDADE REGIONAL DE BLUMENAU

CONCLUSÃO
Extensões

- Desenvolver um número maior de componentes de entrada e saída do microcontrolador;
- Criação de mais funções auxiliares na linguagem de programação e a implementação da depuração com a utilização de *breakpoints;*
- Atualização da ferramenta de compilação Mikroc por não disponibilizar chamada externa.

DEMONSTRAÇÃO



FURB
UNIVERSIDADE REGIONAL DE BLUMENAU