INCLUSÃO DO ALGORITMO DE TRANSFORMAÇÃO DE UM AUTÔMATO FINITO EM EXPRESSÃO REGULAR NO AMBIENTE "EDITOR DE AUTÔMATOS FINITOS"

Acadêmico: Fernando Rafael Piccini

Orientador: José Roque Voltolini da Silva

Roteiro da Apresentação

- Introdução
- Fundamentação Teórica
- Desenvolvimento do Trabalho
- Conclusão

Introdução

- **Editor de Autômatos Finitos (MORASTONI, 2002).**

- **Objetivo principal:**

  - inclusão do algoritmo de transformação de um Autômato Finito (AF) em Expressão Regular (ER) no Editor de Autômatos Finitos.

- **Objetivos específicos:**

  - apresentar uma ER correspondente ao AF desenhado no Editor de Autômatos Finitos (EAF), utilizando o algoritmo de transformação proposto por Silva (2006);

  - apresentar uma tabela de transição correspondente ao AF desenhado na ferramenta EAF;

  - disponibilizar as funções para abrir e salvar a estrutura (grafo) de um AF na ferramenta EAF.

Fundamentação Teórica

- Expressões Regulares
- Autômato Finitos
- Equivalências
- Algoritmo de transformação de um AF em ER
- Protótipo que implementa o algoritmo de transformação de uma ER para um AF (GLATZ, 2000)

Expressão Regular

- ∅ = para denotar a linguagem vazia;
- ε = palavra vazia;
- (x + y) = união da linguagem X com a linguagem Y;
- (xy) = concatenação da linguagem X com a linguagem Y;
- (x*) = fechamento da linguagem X.

Expressão Regular

expressão regular	linguagem denotada
aa	somente a palavra aa
(aa | bb)	palavra com aa ou bb palavras que iniciam com a,
a(b)*	concatenado com zero ou mais ocorrências de b
$(a \mid b)^+$	todas as palavras sobre { a, b }

Alfabeto composto pelas letras a e b (Σ = {a, b}).

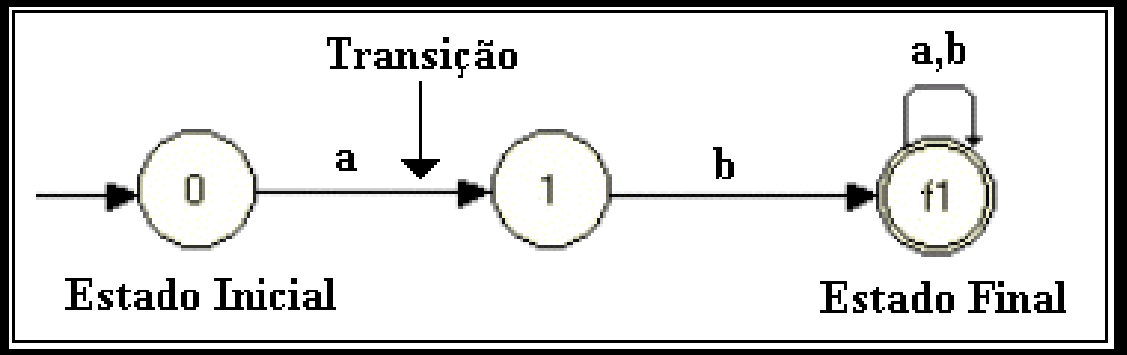
Autômatos Finitos

$M = (\Sigma, Q, \delta, q_0, F)$, onde:

a) Σ = alfabeto de símbolos de entrada;
b) Q = conjunto finito de estados possíveis ;
c) δ = função de transição de estados;
d) q_0 = estado inicial (elemento de Q);
e) F = conjunto de estados finais (F está contido em Q).

Representação dos Autômatos Finitos

- Diagrama de transição



- Tabela de transição

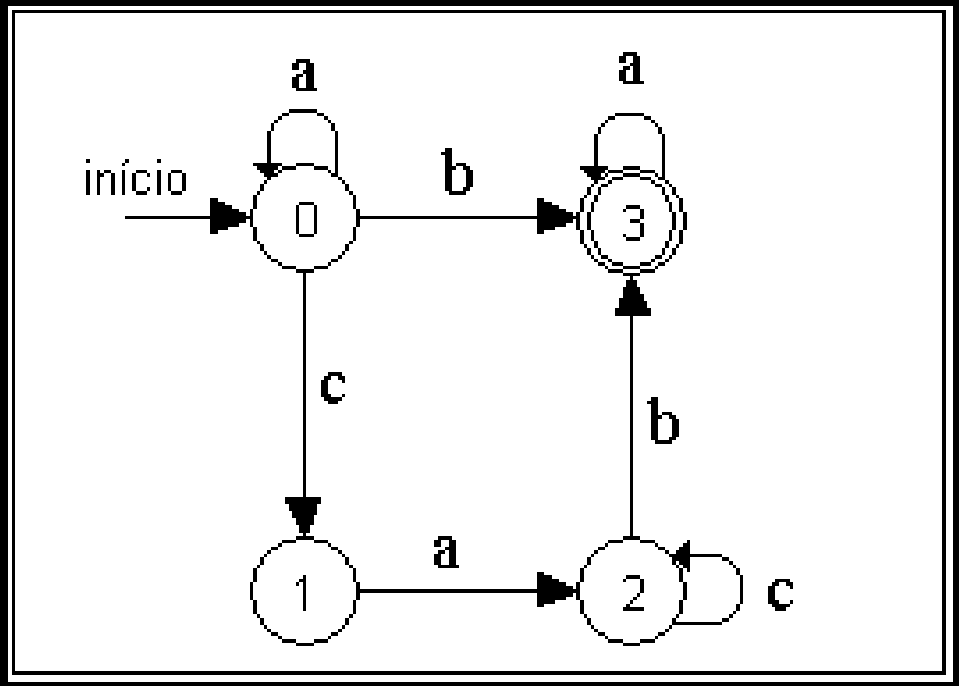
	a	b
→ 0	1	—
1	—	f1
* f1	f1	f1

Classificação dos Autômatos Finitos

- Autômatos Finitos Determinísticos (AFD)
- Autômatos Finitos Não-Determinísticos (AFN)
- Autômatos Finitos com Movimento Vazio (ε-AFN)

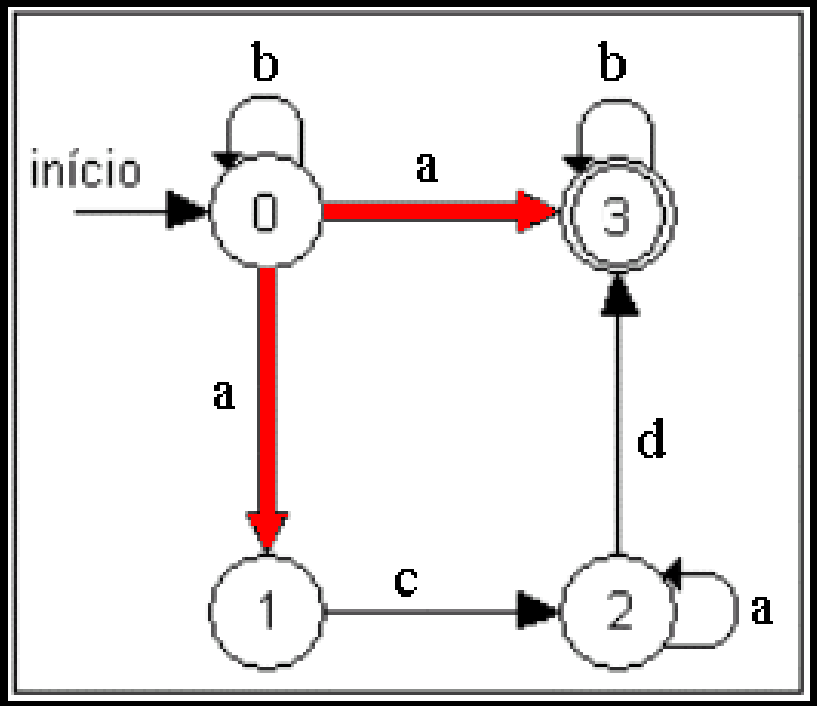
Autômato Finito Determinístico (AFD)

- Ao processar um símbolo da entrada a partir do estado corrente, um AFD pode assumir um único estado.



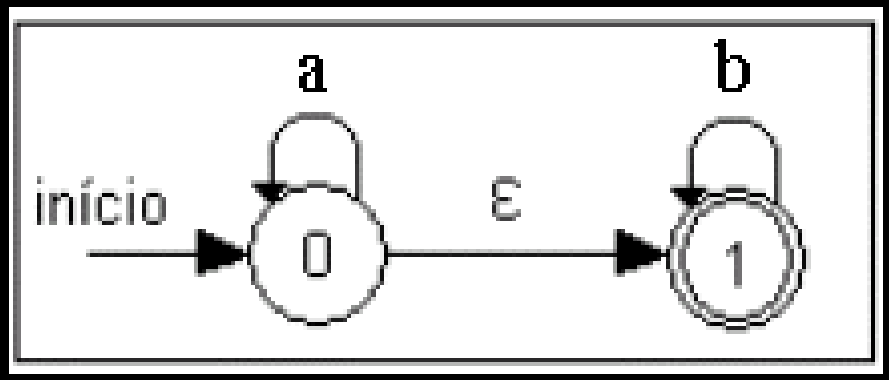
Autômato Finito Não-Determinístico (AFN)

- Ao processar um símbolo de entrada a partir do estado corrente, um AFN pode ter como resultado um conjunto de novos estados.



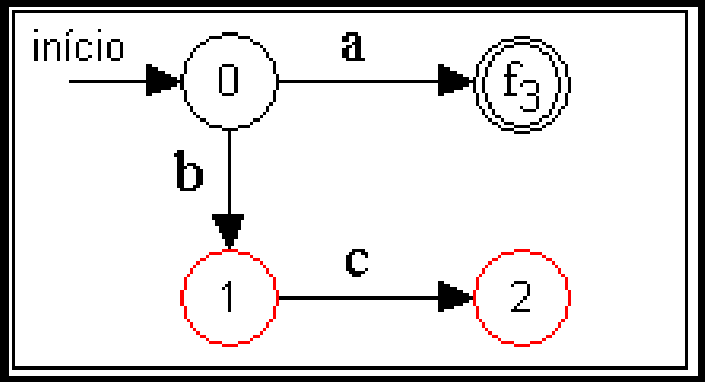
Autômatos Finitos com Movimento Vazio (ε-AFN)

- Um movimento vazio é apenas uma transição sem leitura de símbolo algum e pode ser interpretado como um não determinismo.

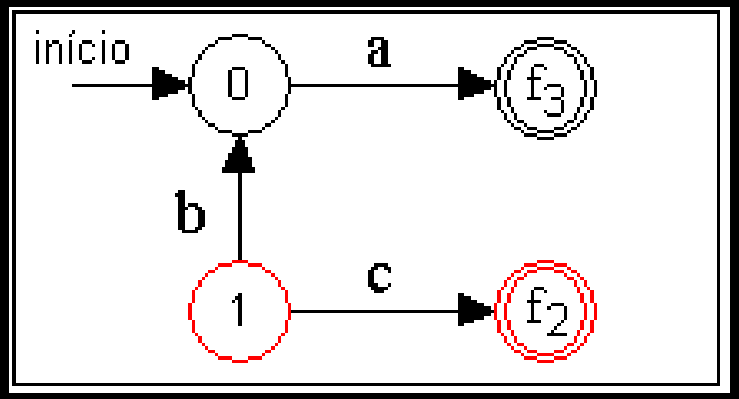


Propriedades dos Autômatos Finitos

- Estados mortos

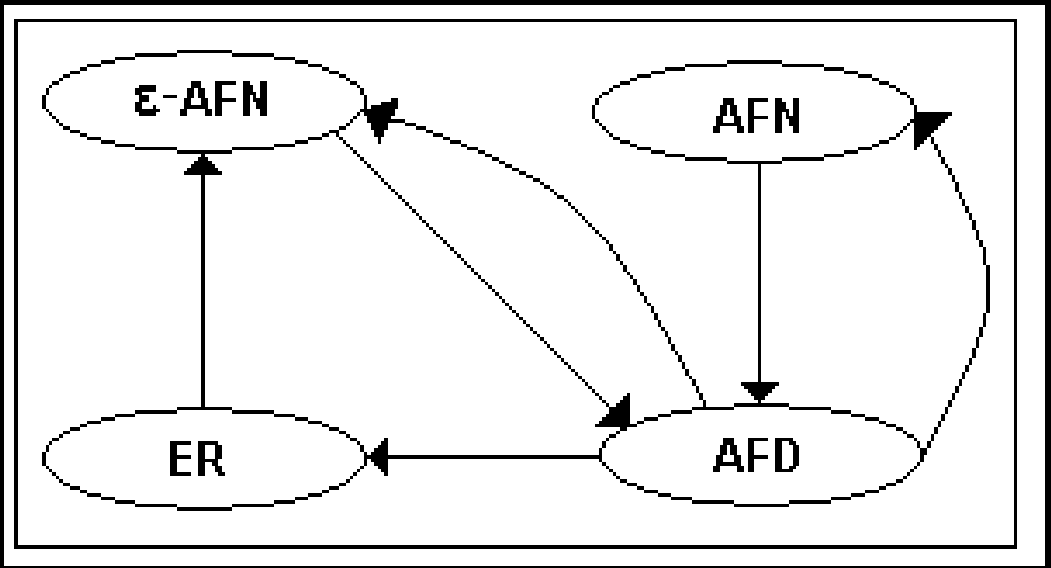


- Estados inalcançáveis



Equivalências

- Equivalência entre as quatro notações para linguagens regulares



Fonte: Hopcroft, Ullman e Motwani (2002, p. 98)

Algoritmo de transformação de um AF em uma ER proposto em Silva (2006)

- Identificação dos estados (ID)
- Novos estados e transições
  - Estado inicial
  - Estados finais
- Processo de transformação
  - União
  - Concatenação
  - Fechamento
  - Desdobramento

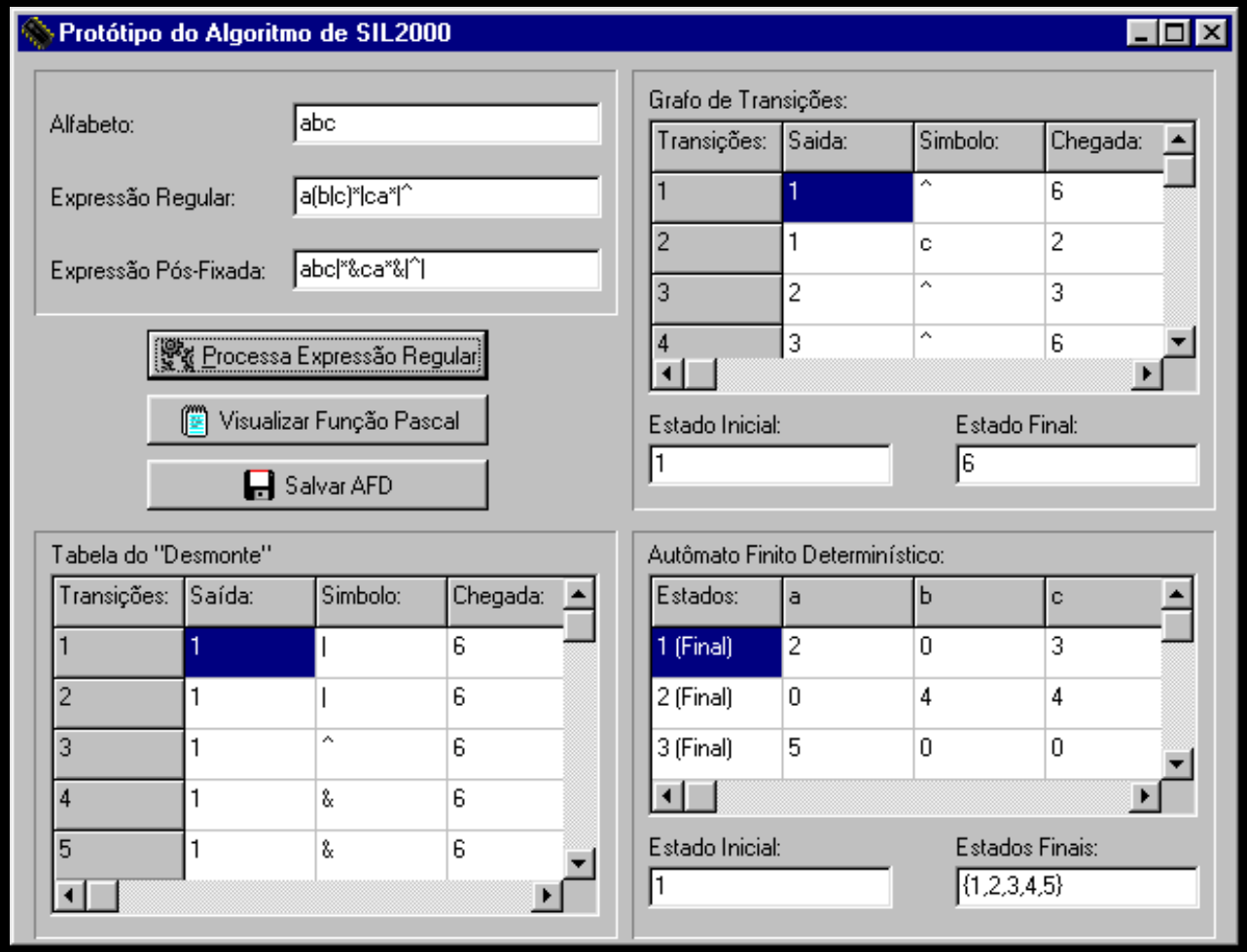
Algoritmo de transformação de um AF em uma ER proposto em Silva (2006)

Modelo de tabela para efetuar a transformação de um AF em ER

(X) linha excluída	NÓ DE ORIGEM (X)	EXPRESSÃO REGULAR (er)	NÓ DE DESTINO (Y)

Fonte: adaptado de Silva (2006)

Protótipo que implementa o algoritmo de transformação de uma ER para um AF



Tela principal do protótipo (Glatz, 2000)

Desenvolvimento do Trabalho

- Requisitos
- Especificação
- Implementação
- Operacionalidade

Principais Requisitos

- permitir abrir e salvar um arquivo contendo a estrutura (grafo) de um AF em arquivo;
- possuir um módulo para apresentação de tabela de transição;
- gerar uma ER a partir de um AF;
- adicionar ao projeto a biblioteca RxLib para aprimoramento da interface.

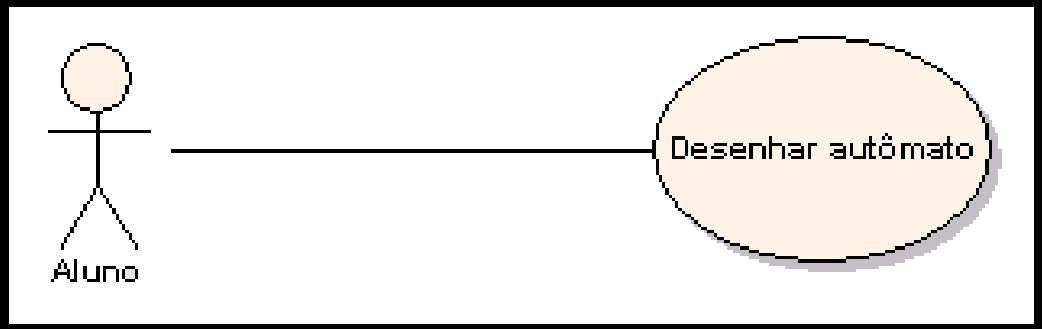
Especificação

- **Orientação a objetos representado através dos:**

  - diagramas de casos de uso;
  - diagramas de atividades;
  - diagrama de classes.

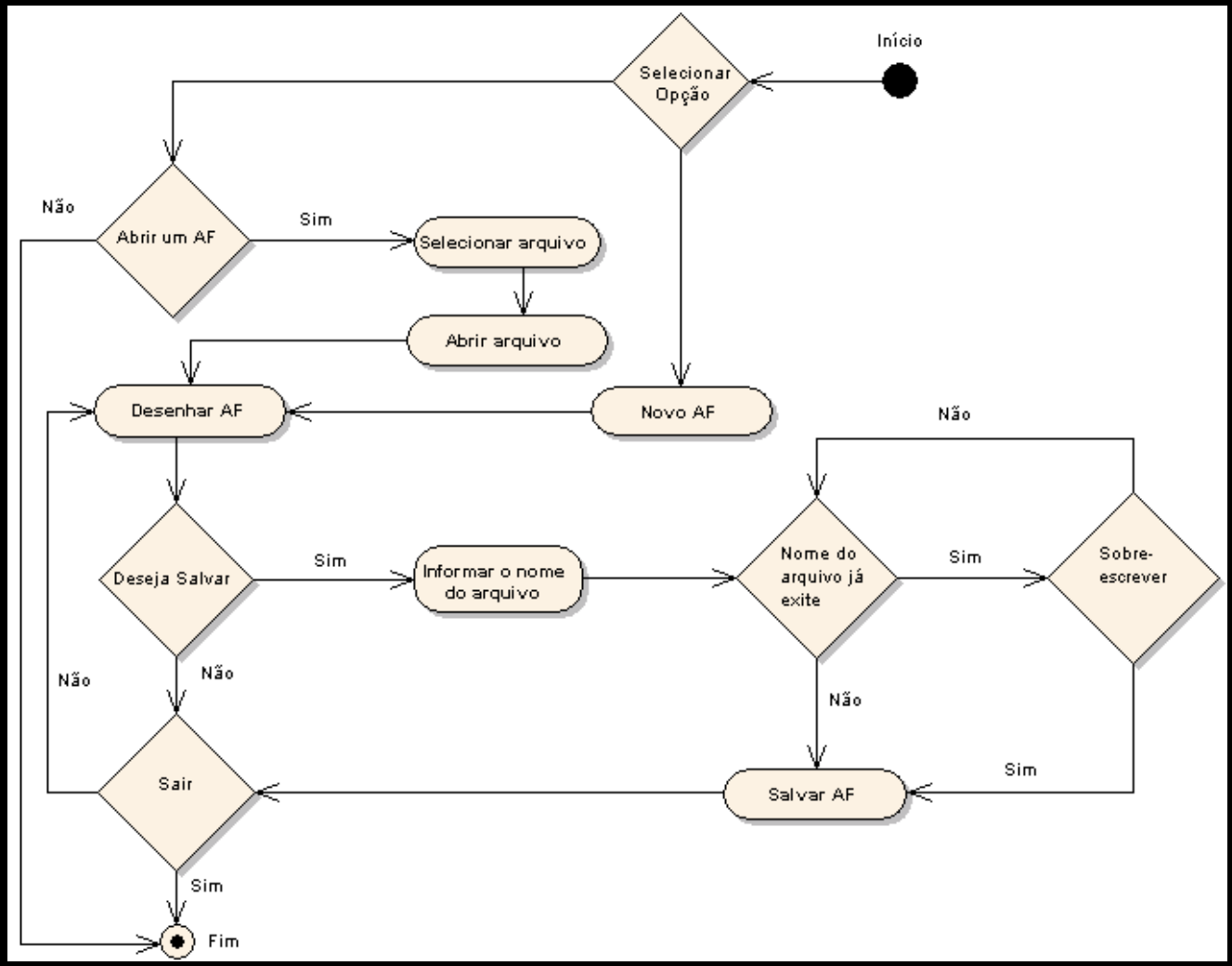
- ***Enterprise Architect***

Caso de Uso: Desenhar AF

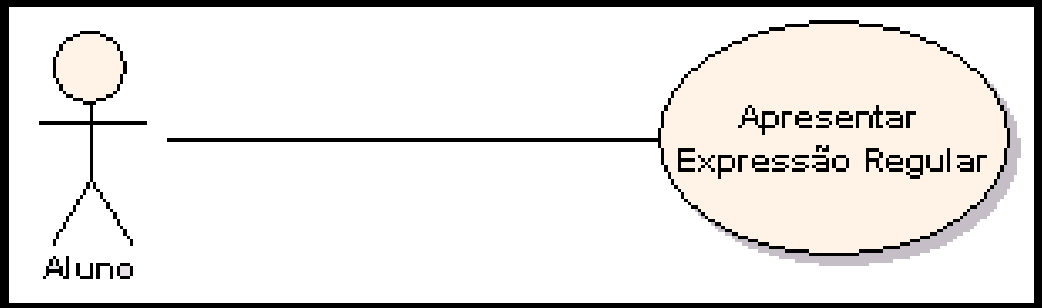


- Abrir um nova janela de edição no EAF;
- Criar estado inicial;
- Criar outros estados;
- Definir estados finais;
- Criar transições entre os estados.

Diagrama de Atividade: Desenhar AF



Caso de Uso: Apresentar ER



- Escolher opção "Expressão Regular";
- Ferramenta executa o algoritmo de transformação de um AF para ER;
- EAF apresenta uma nova janela com a ER correspondente ao AF desenhado na tela de edição.

Diagrama de Atividade: Apresentar ER

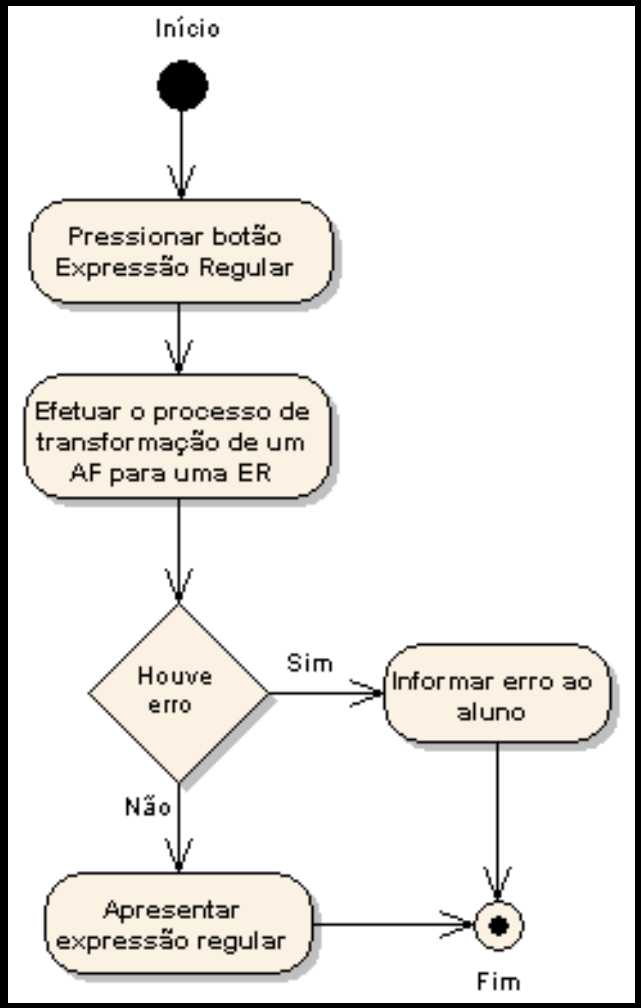
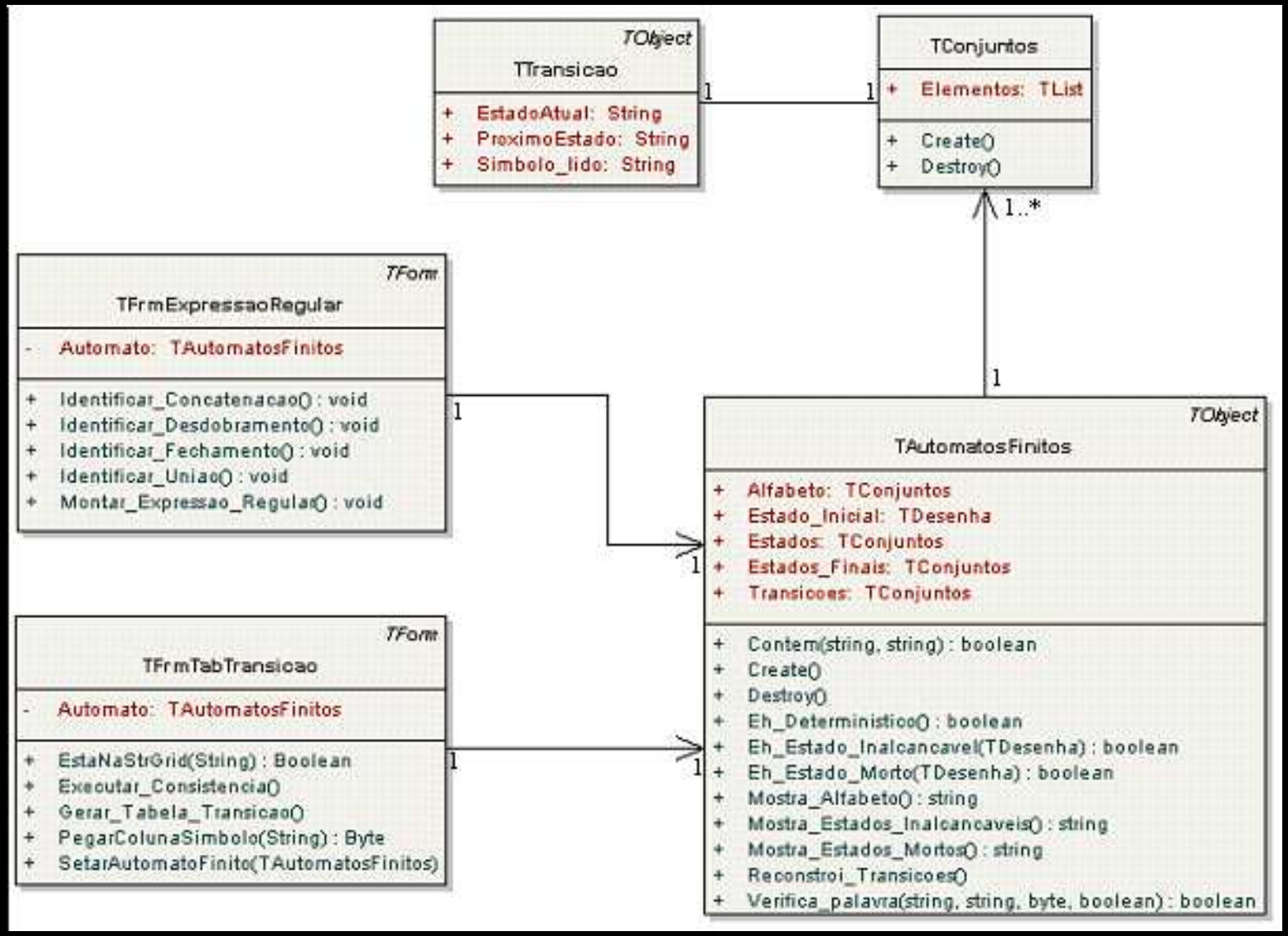


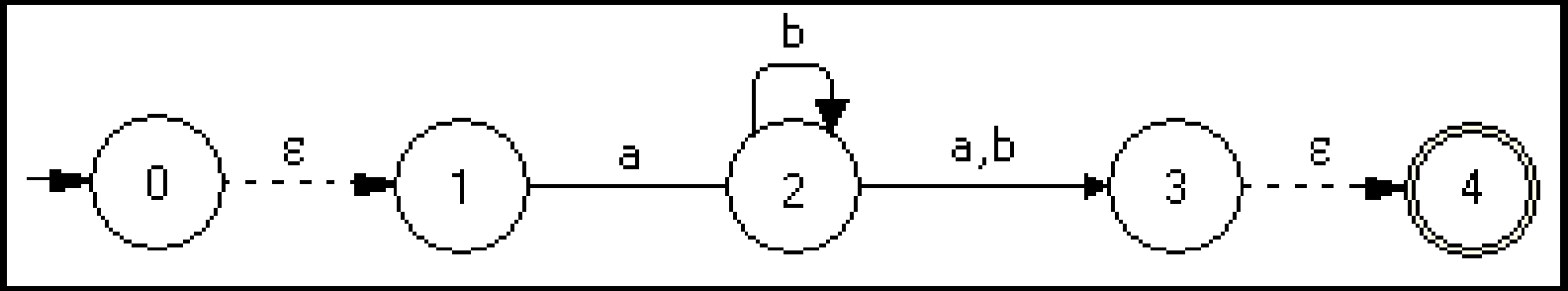
Diagrama de Classe



Implementação

- **Ambiente**
  - **Borland Delphi 7**
- **Componentes**
  - **TDesenha e TDesenhaLinha**
- **Biblioteca**
  - **RxLib**

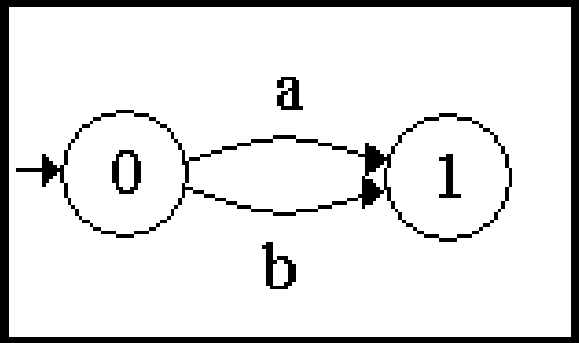
Algoritmo de transformação de um AF em uma ER proposto em Silva (2006)



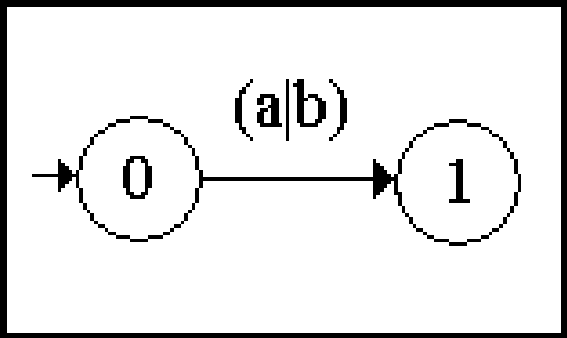
	(LE)	(X)	(ER)	(Y)	
	F	1	a	2	
	F	2	b	2	
	F	2	a	3	
	F	2	b	3	
	F	0	ε	1	
	F	3	ε	4	

Modelo de tabela para efetuar a transformação de um AF em ER

União



união

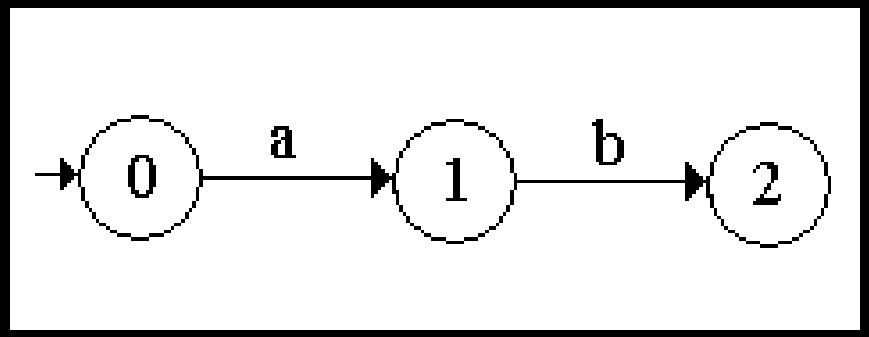


(LE)	(X)	(ER)	(Y)
F	0	a	1
F	0	b	1

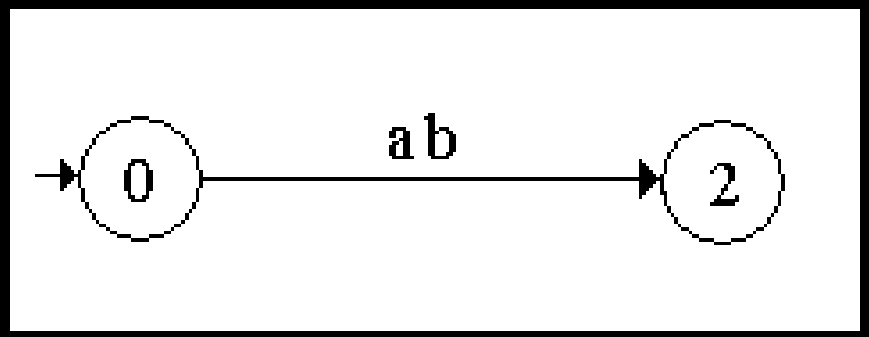
união

(LE)	(X)	(ER)	(Y)
~~T~~	~~0~~	~~a~~	~~1~~
~~T~~	~~0~~	~~b~~	~~1~~
F	0	(a|b)	1

Concatenação

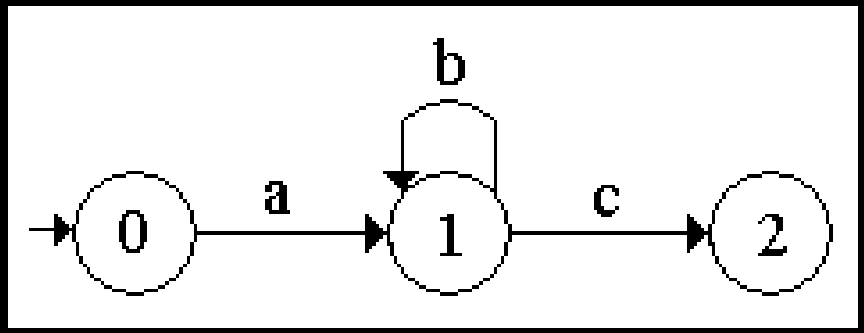


(LE)	(X)	(ER)	(Y)
F	0	a	1
F	1	b	2

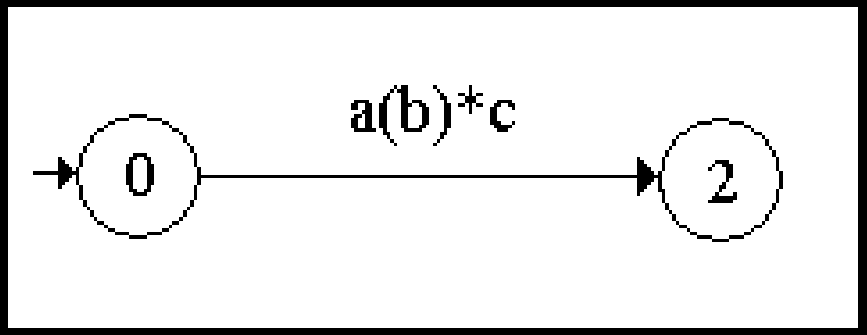


(LE)	(X)	(ER)	(Y)
~~T~~	~~0~~	~~a~~	~~1~~
~~T~~	~~1~~	~~b~~	~~2~~
F	0	ab	2

Fechamento

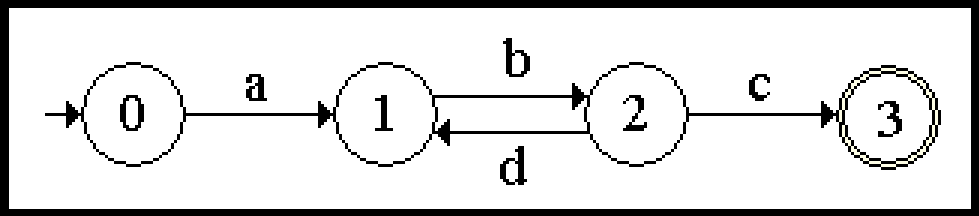


(LE)	(X)	(ER)	(Y)
F	0	a	1
F	1	b	1
F	1	c	2

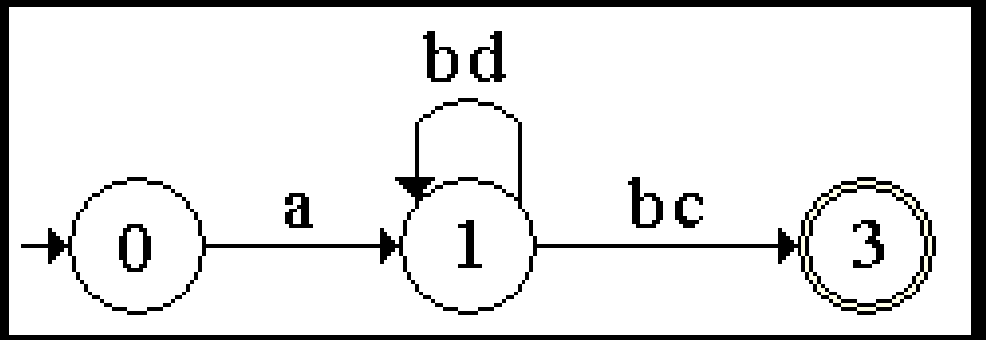


(LE)	(X)	(ER)	(Y)
~~T~~	~~0~~	~~a~~	~~1~~
~~T~~	~~1~~	~~b~~	~~1~~
~~T~~	~~1~~	~~c~~	~~2~~
F	0	a(b)*c	2

Desdobramento



(LE)	(X)	(ER)	(Y)
F	0	a	1
F	1	b	2
F	2	c	3
F	2	d	1



N1 = 1b

(LE)	(X)	(ER)	(Y)	
F	0	a	1	
T	1	b	2	A
F	~~2~~ 1	bc	3	B
F	~~2~~ 1	bd	1	B

Operacionalidade

- Tabela de Transição
- Expressão Regular

Tabela de Transição

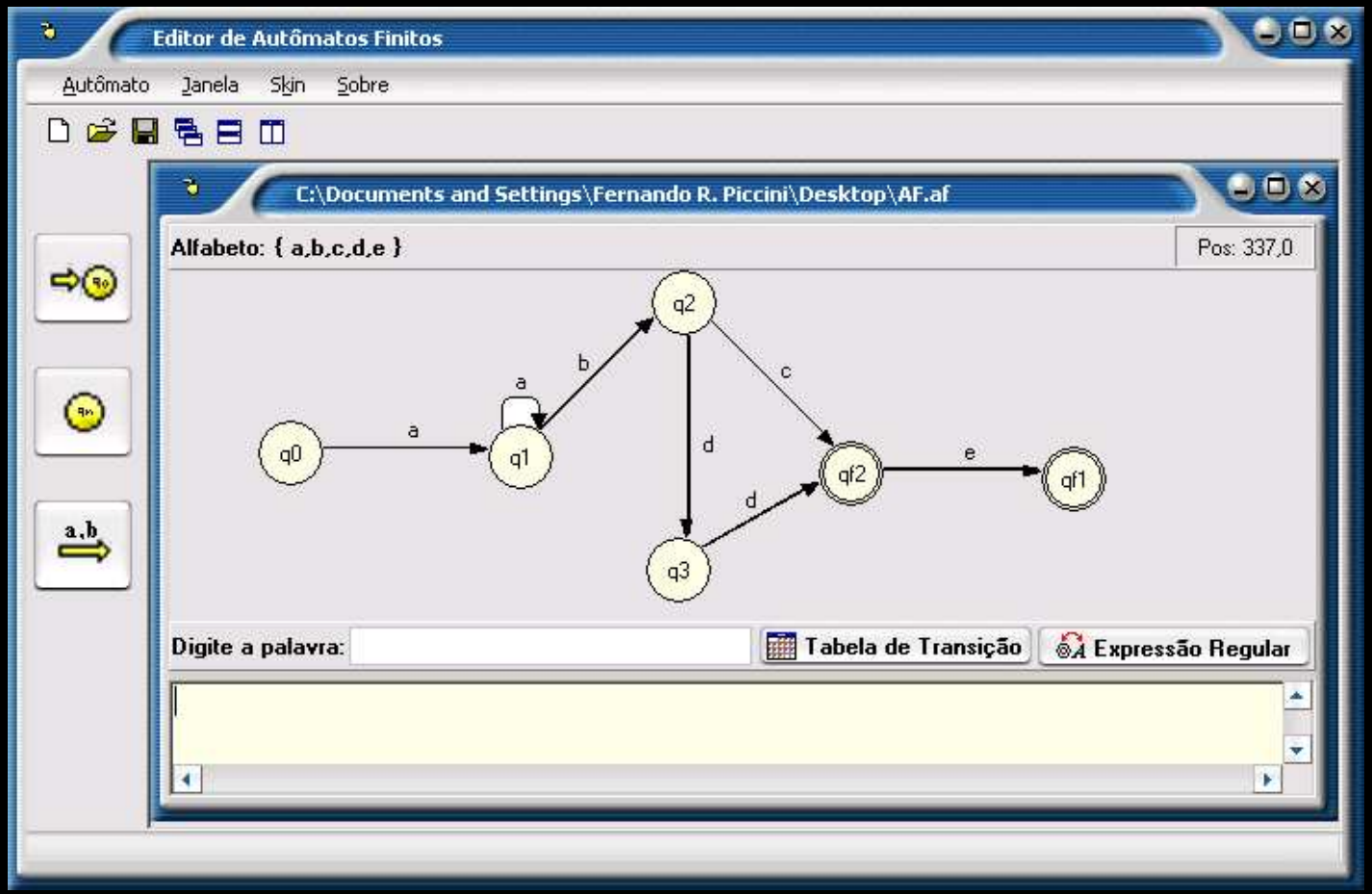


Tabela de Transição

	a	b	c	d	e
-> q0	q1	-	-	-	-
q1	q1	q2	-	-	-
q2	-	-	qf2	q3	-
q3	-	-	-	qf2	-
* qf1	-	-	-	-	-
* qf2	-	-	-	-	qf1

Tabela de Transição

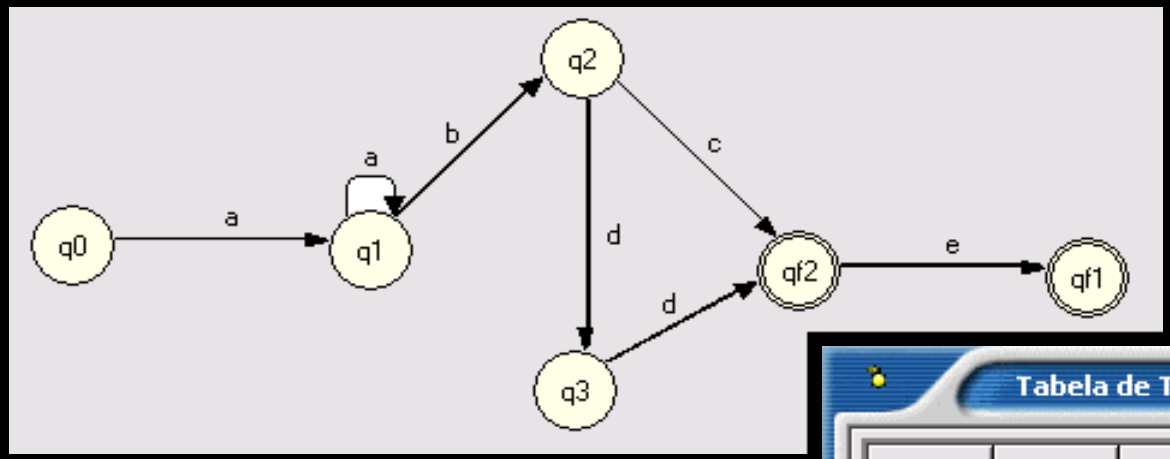
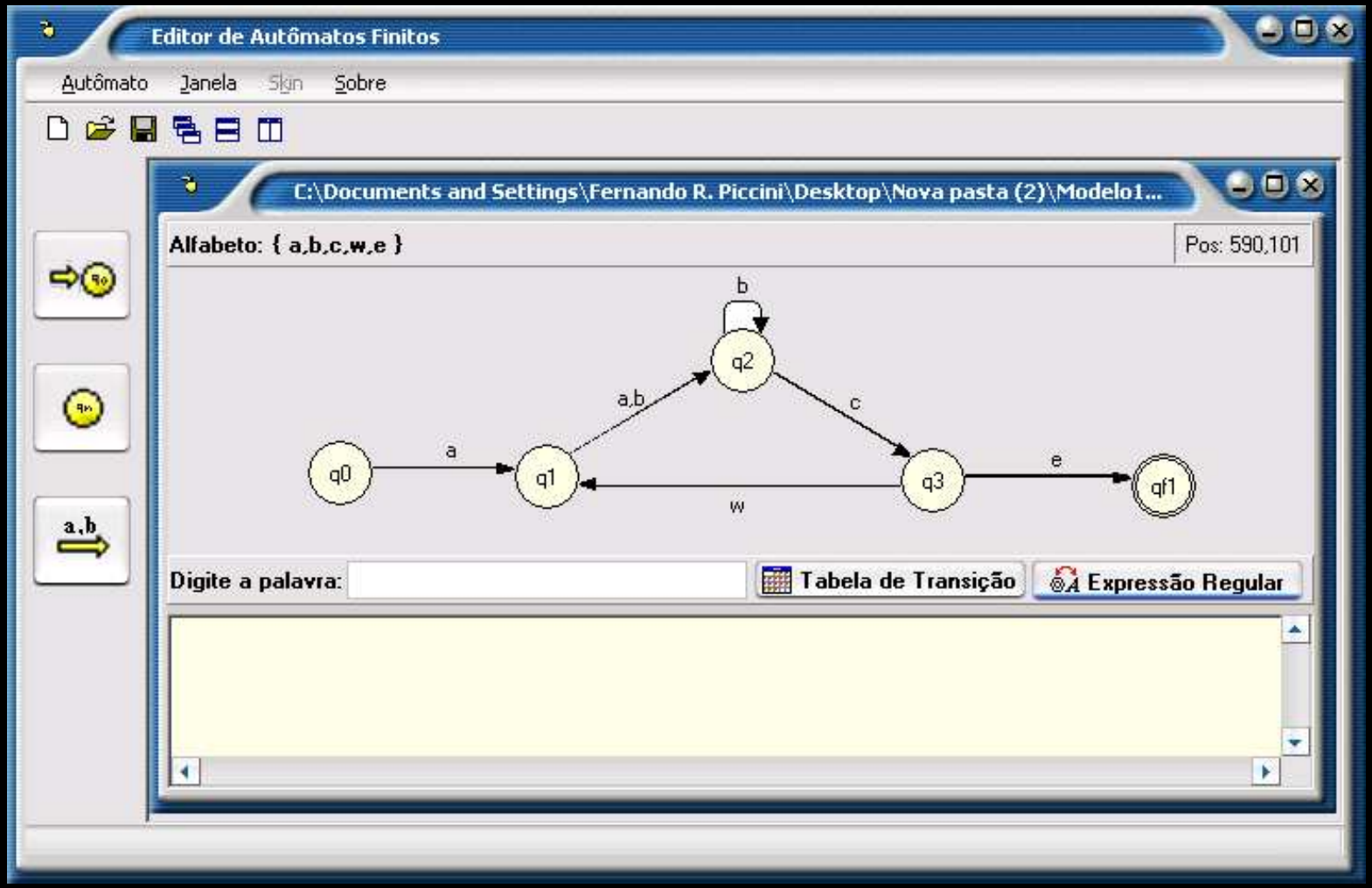


Tabela de Transição

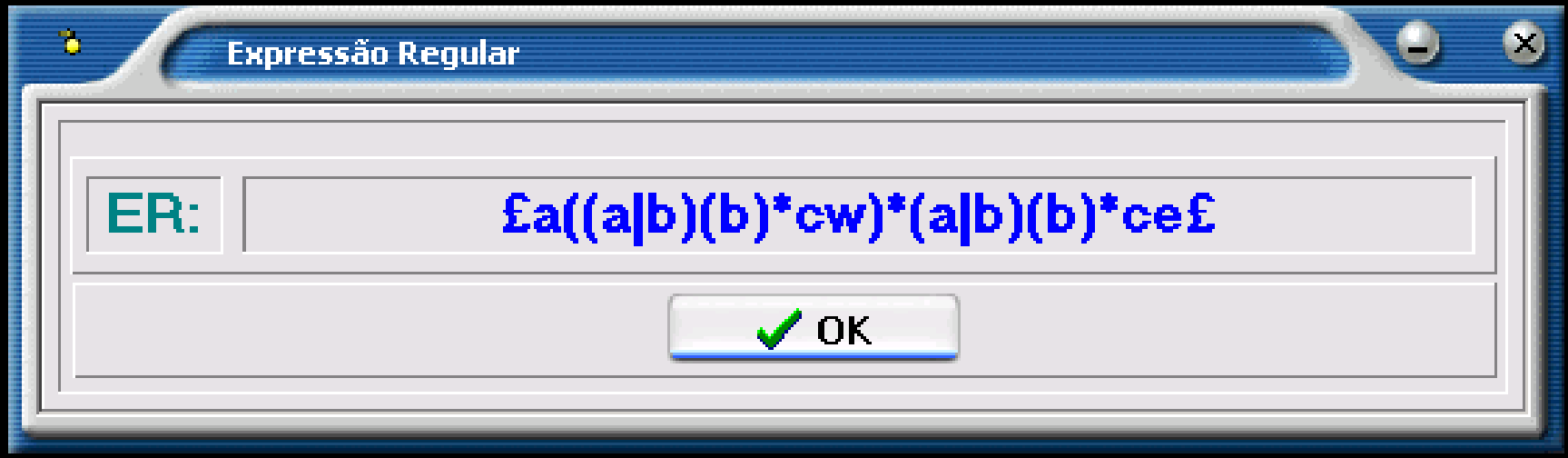
	a	b	c	d	e
-> q0	q1	-	-	-	-
q1	q1	q2	-	-	-
q2	-	-	qf2	q3	-
q3	-	-	-	qf2	-
* qf1	-	-	-	-	-
* qf2	-	-	-	-	qf1

OK

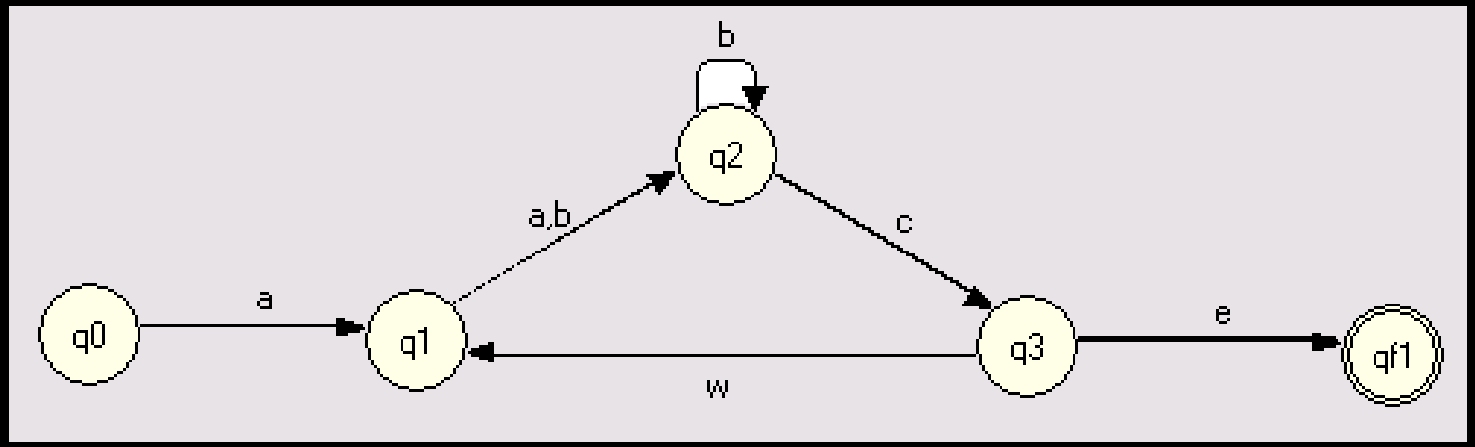
Expressão Regular

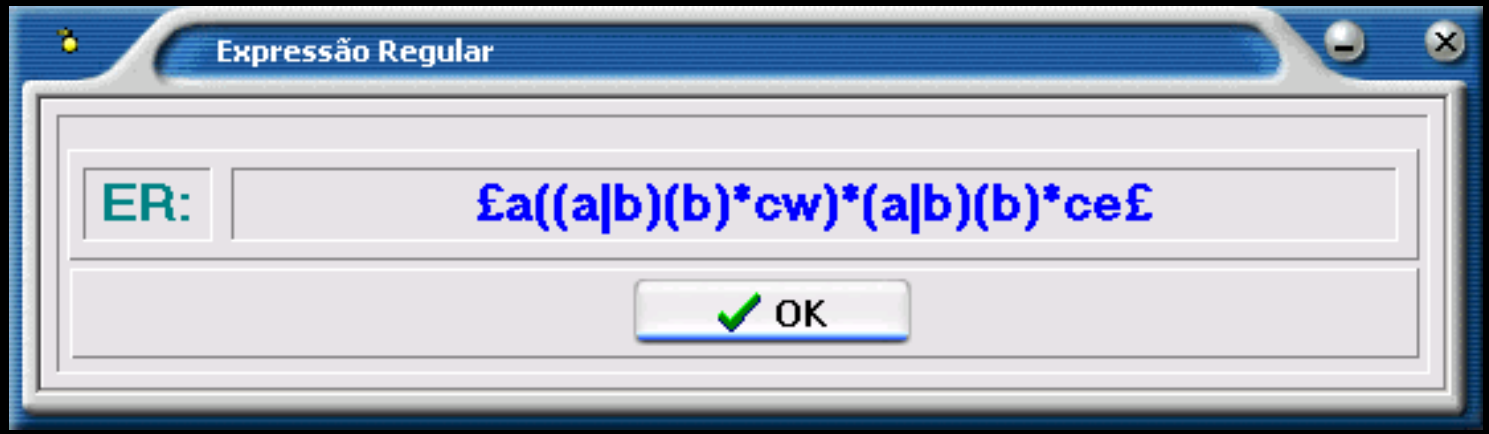


Expressão Regular



Expressão Regular



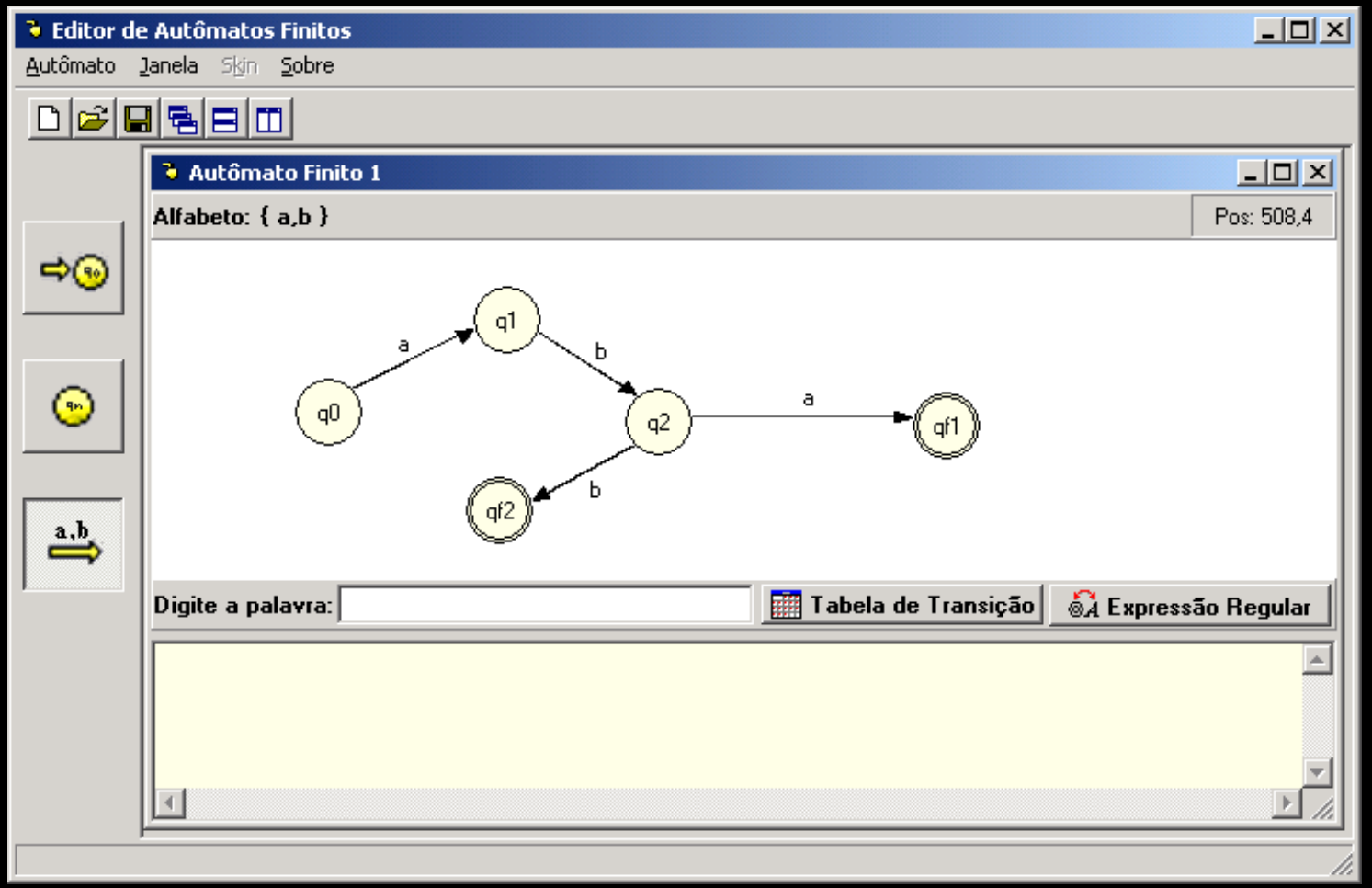


Resultados e Discussão

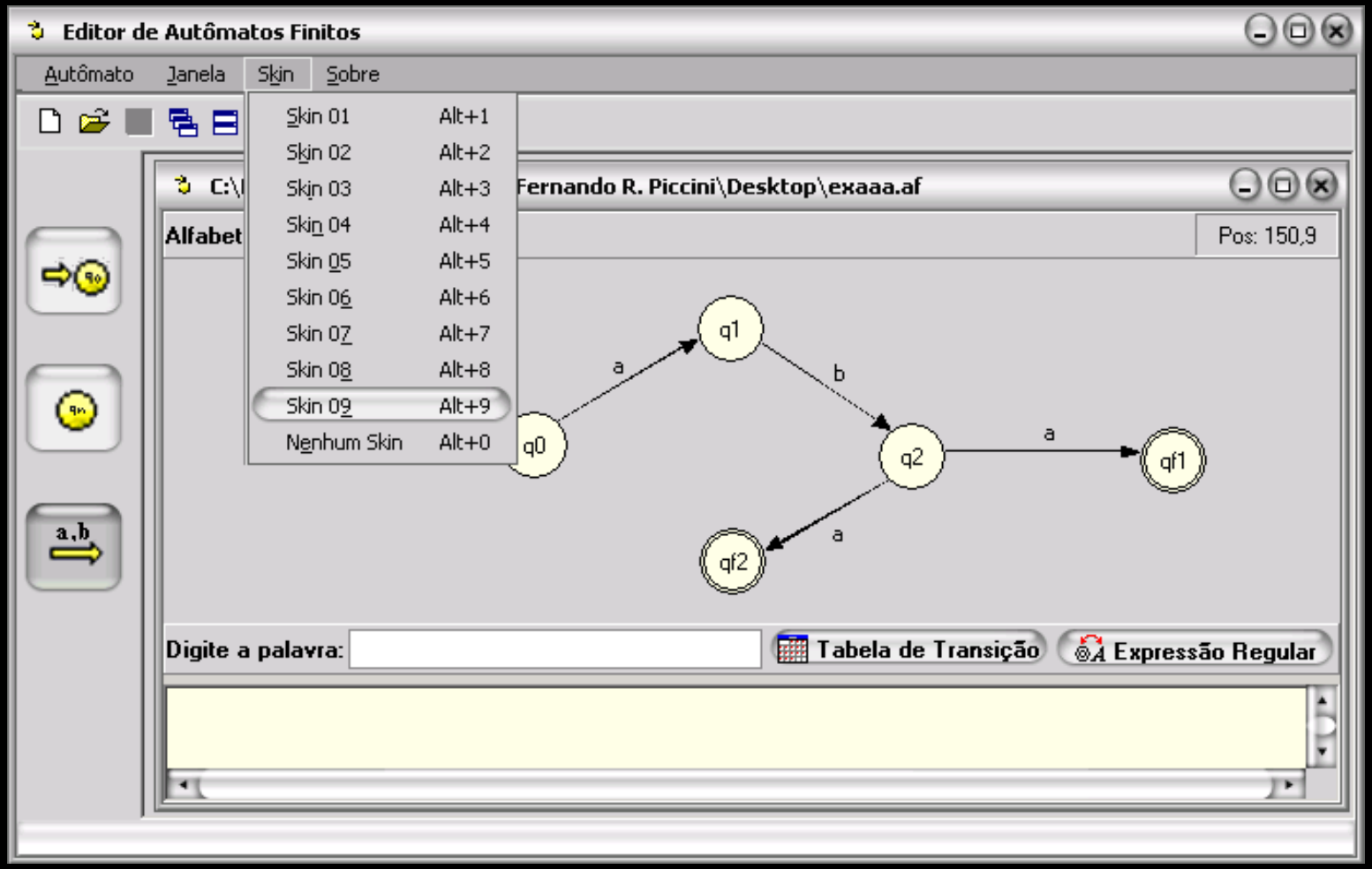
- RxLib (Interface):
  - cores
  - padrões
  - formatos

- Validação do Algoritmo de Transformação de um AF para uma ER

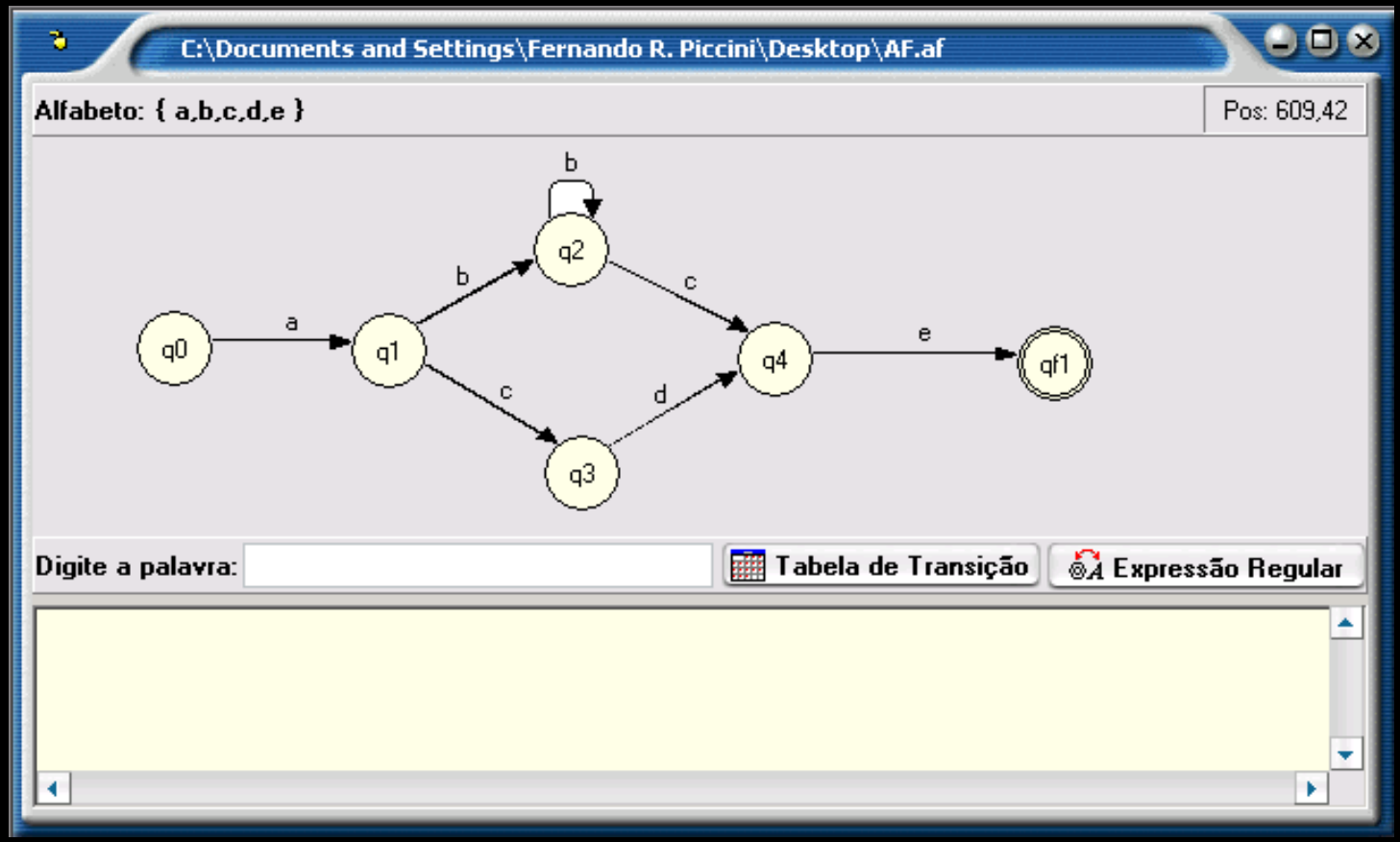
Interface antes da utilização da biblioteca RxLib



Interface após a utilização da biblioteca RxLib

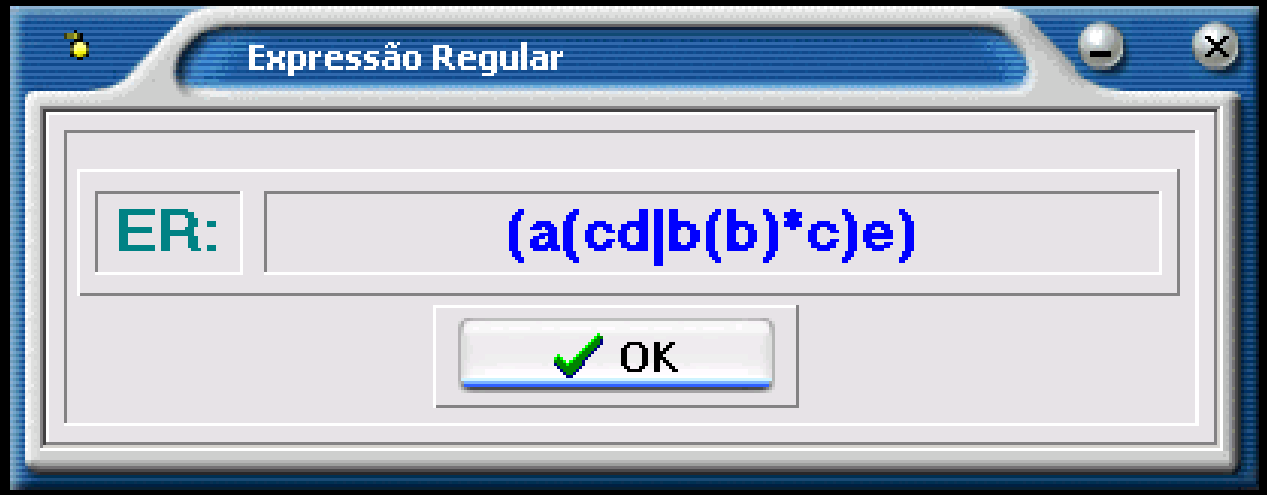


Validação do Algoritmo de Transformação de um AF para uma ER



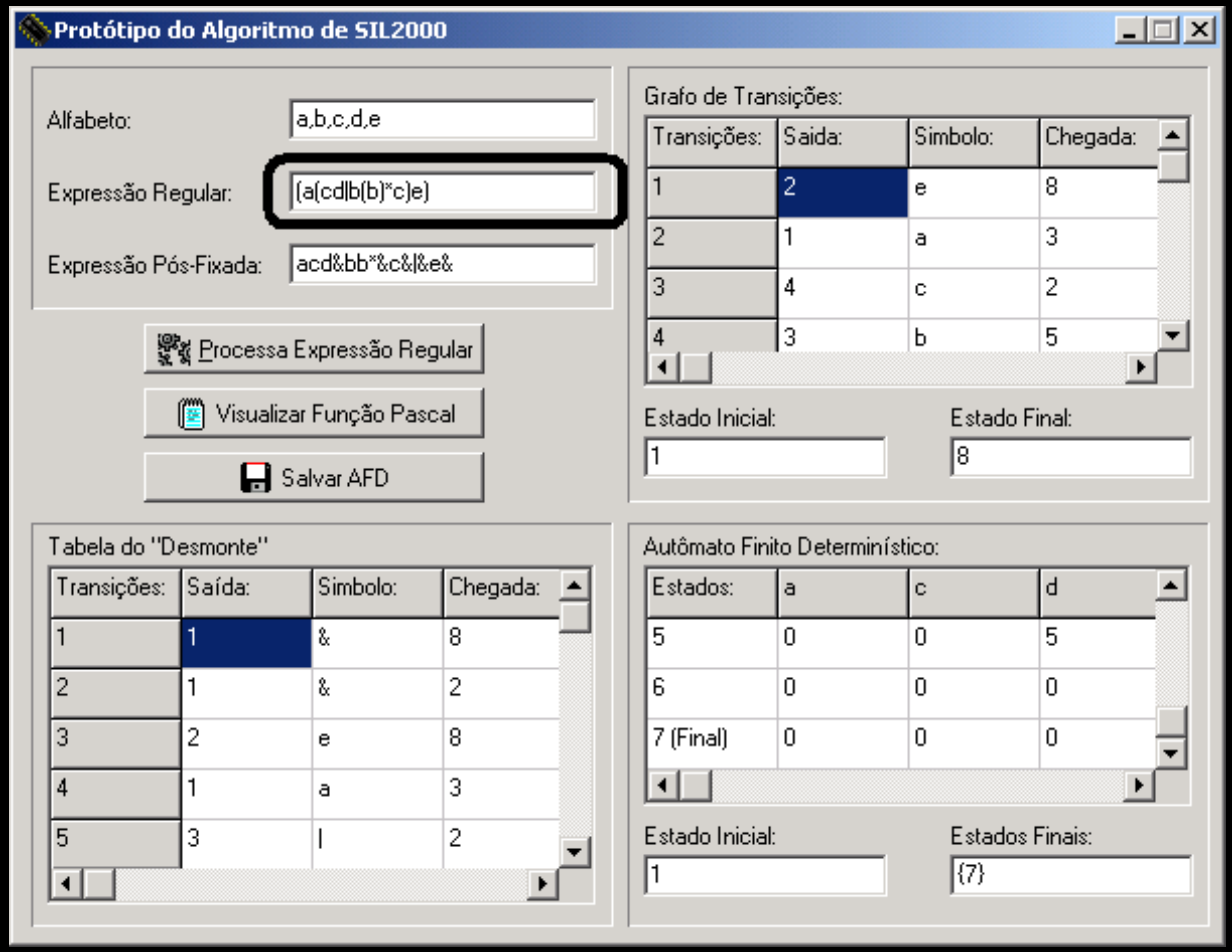
AF_1 submetido para apresentação de uma expressão regular no EAF

Validação do Algoritmo de Transformação de um AF para uma ER



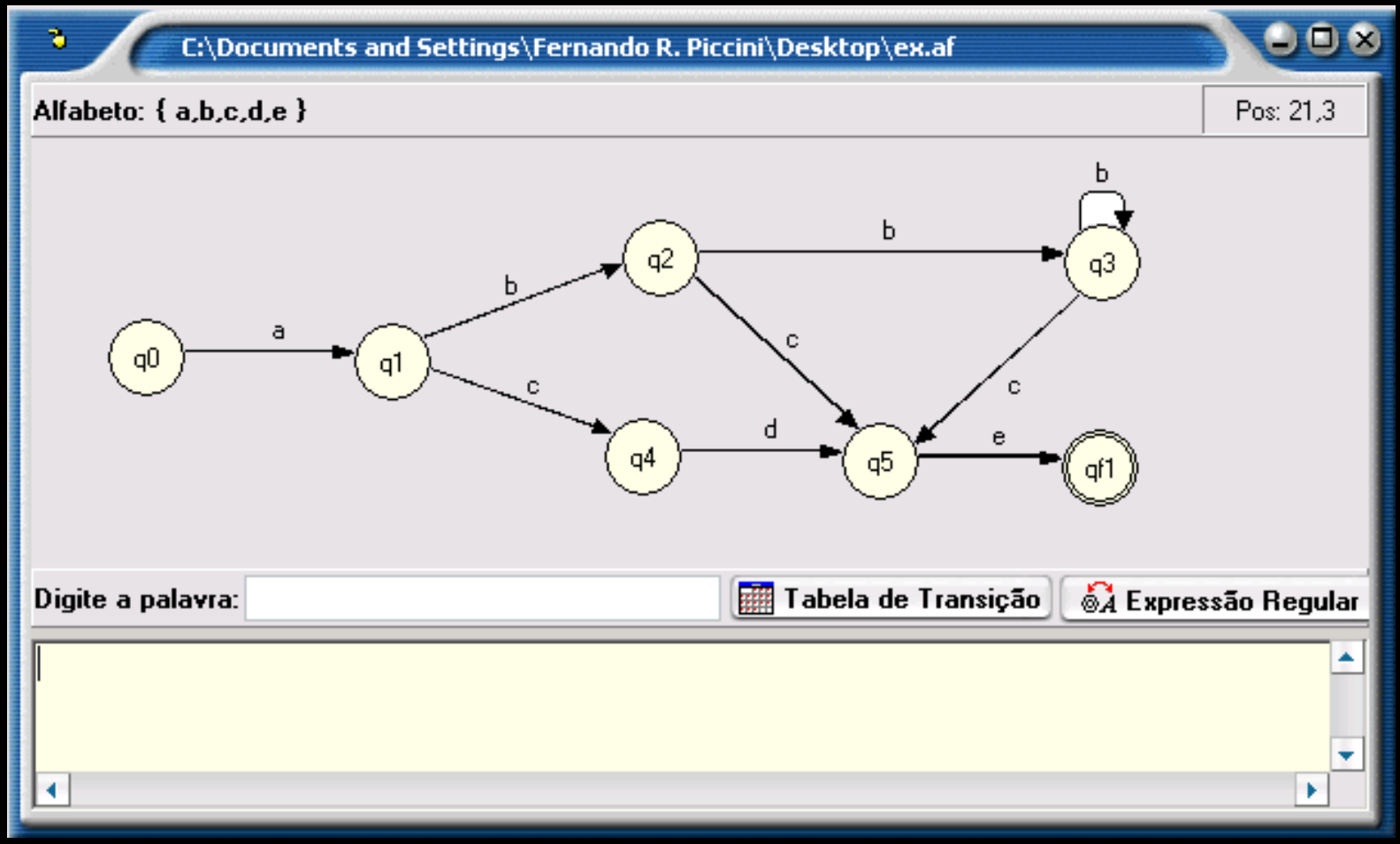
ER_1 equivalente ao AF_1

Validação do Algoritmo de Transformação de um AF para uma ER



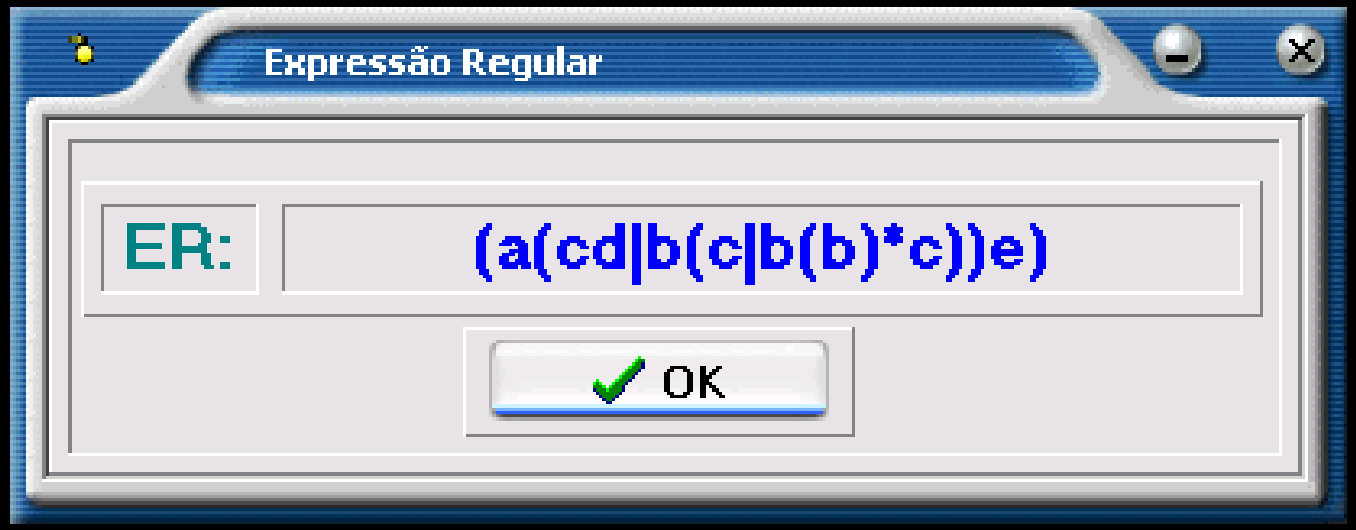
ER_1 submetida há geração do AF_2 na ferramenta de Glatz (2000)

Validação do Algoritmo de Transformação de um AF para uma ER



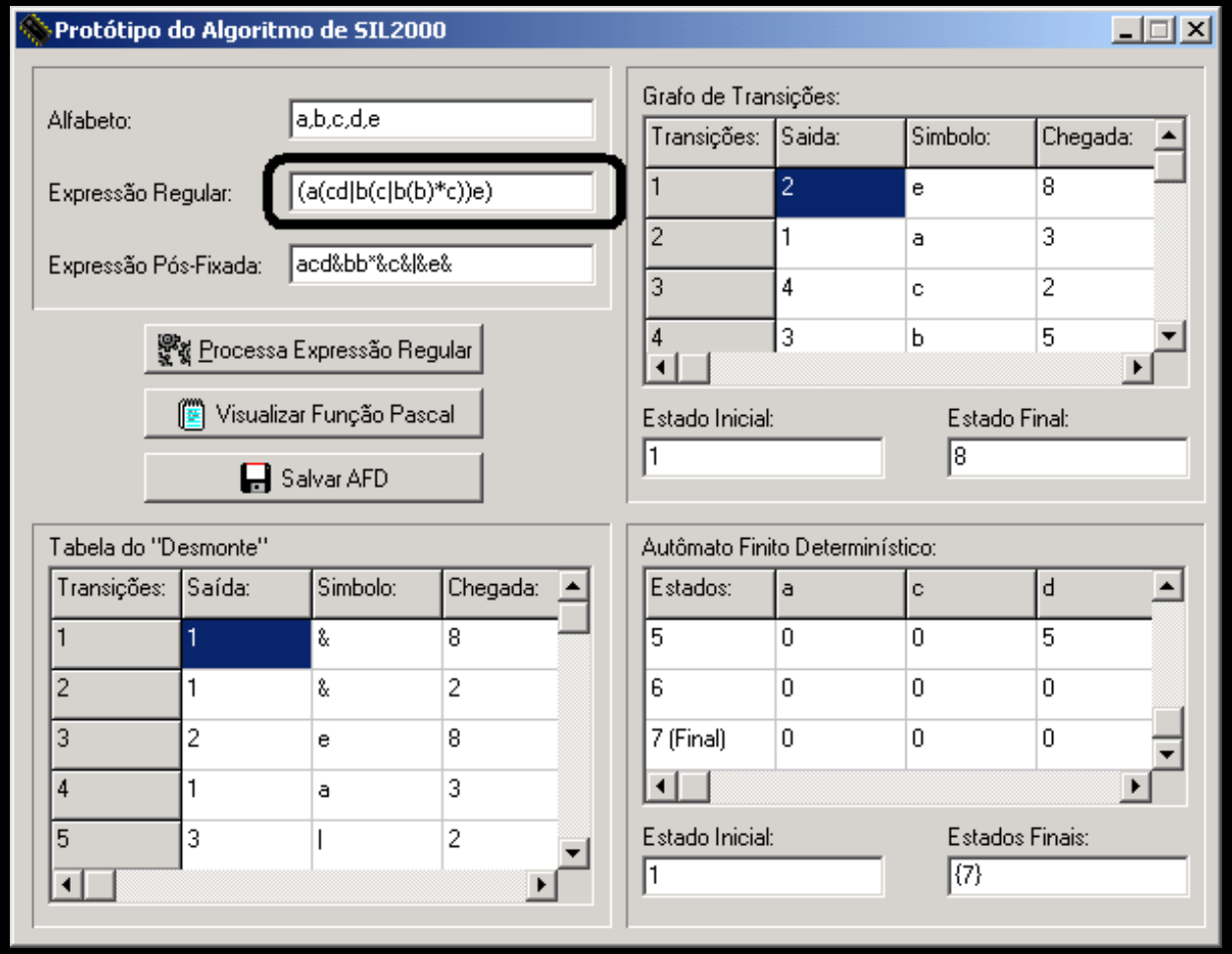
Apresentação do AF_2 gerado a partir da ER_1 "(a(cd|b(b)*c)e)"

Validação do Algoritmo de Transformação de um AF para uma ER



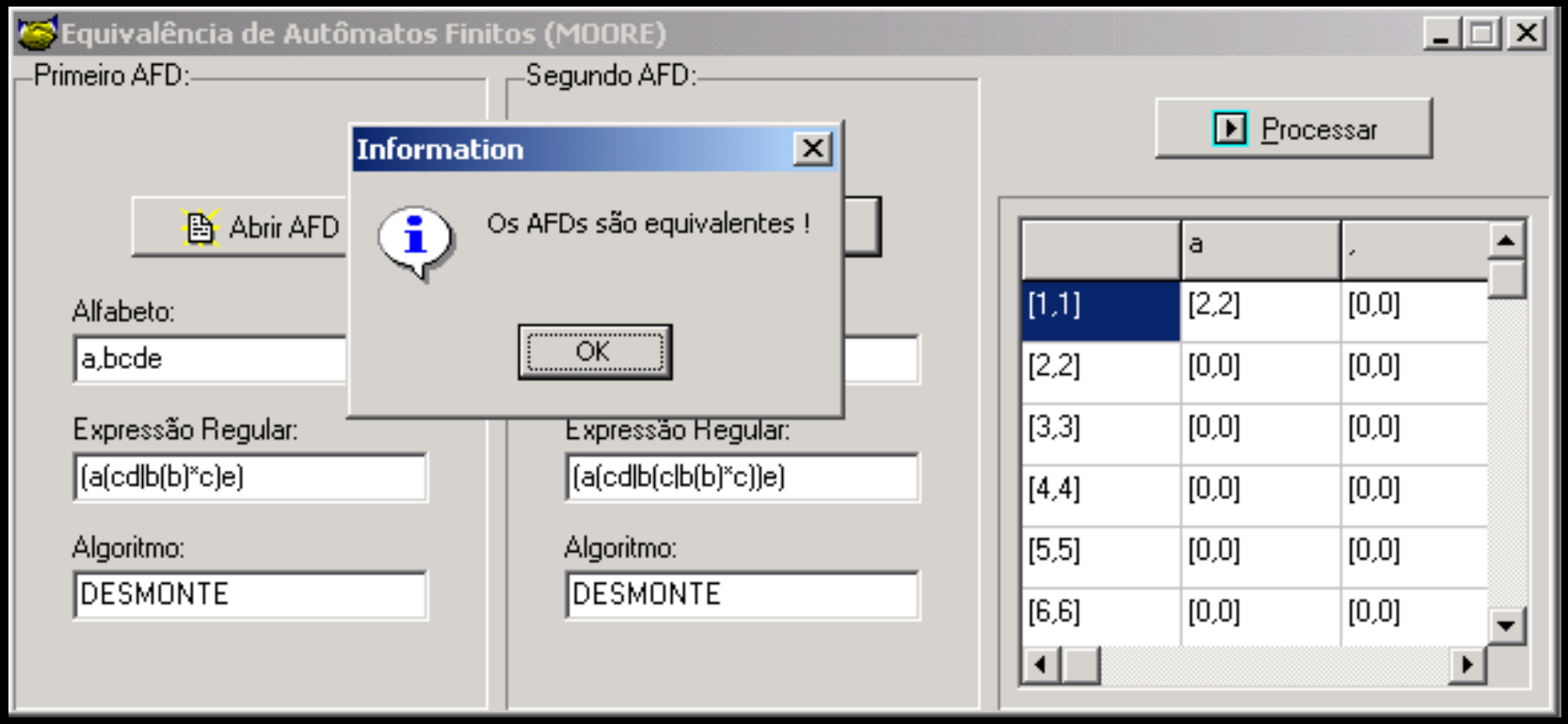
ER_2 equivalente ao AF_2 apresentado

Validação do Algoritmo de Transformação de um AF para uma ER



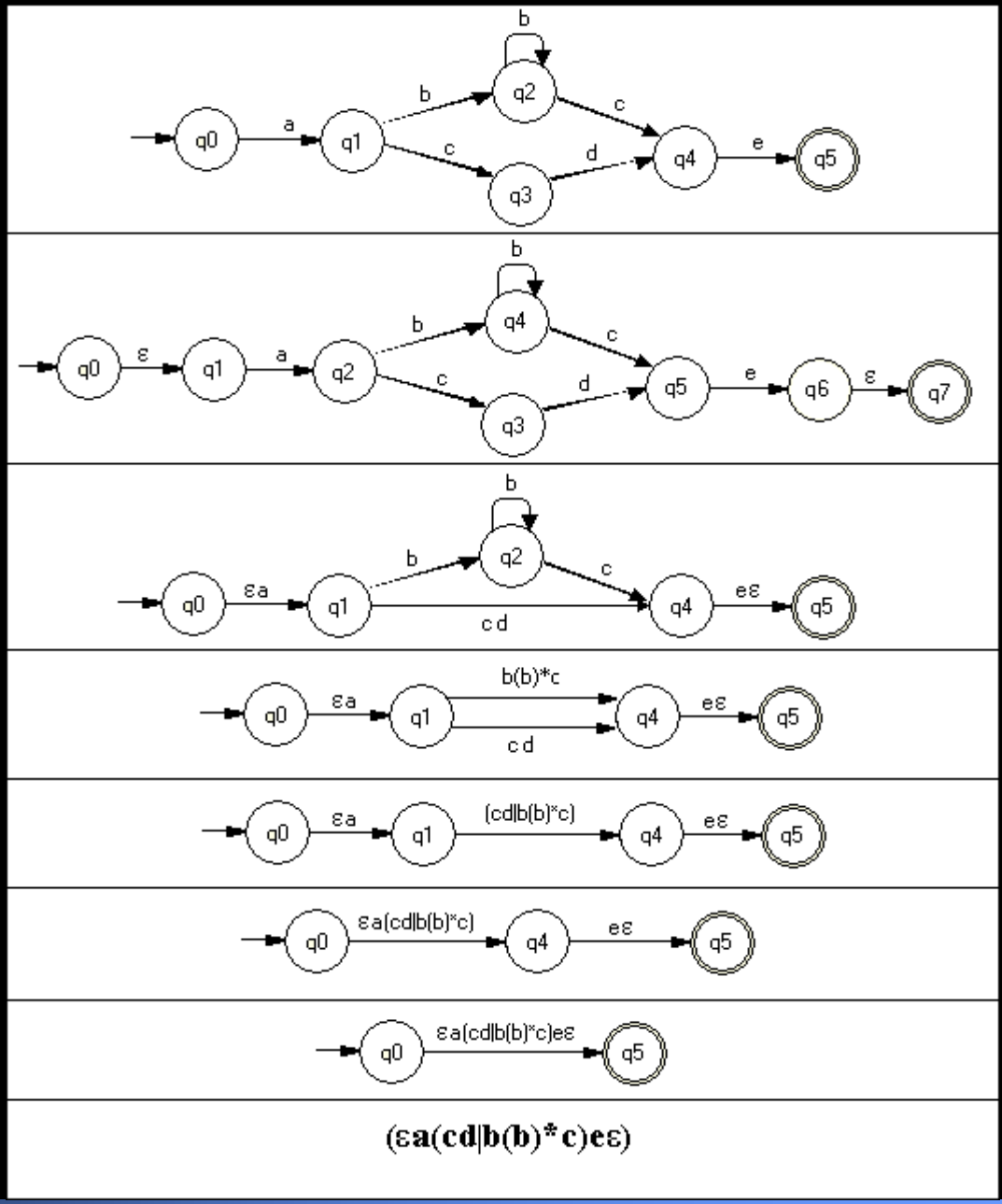
ER_2 submetida há geração do AF_3 na ferramenta de Glatz (2000)

Validação do Algoritmo de Transformação de um AF para uma ER



Validação do AF_2 e AF_3 através da equivalência de Moore

Processo de Transformação AF para ER



Conclusão

- Foram atingidos os objetivos propostos:

  - permitir abrir e salvar a estrutura (grafo) de um AF em arquivo;

  - apresentar tabela de transição correspondente a um AF especificado na tela de edição do EAF;
  - adicionar biblioteca RxLib;

  - validar o algoritmo de transformação através de exemplos de AFs editados na ferramenta.

- Observa-se que os componentes TDesenha e TDesenhaLinha não foram desenvolvidos nesse trabalho, porém o código fonte destes encontra-se disponível junto a aplicação.

Extensões

- aplicar algoritmos de minimização em AFDs;
- implementar algoritmo para minimizar a expressão regular, gerada pelo algoritmo proposto em Silva (2006);
- identificar os estados que geram o não-determinismo;
- determinar a complexidade do algoritmo proposto por Silva (2006);
- comparar o algoritmo proposto por Silva (2006) com outro algoritmo, objetivando verificar qual deles é mais eficiente.