CLAITON WERNER KÜSTER
FÁBIO ALEXANDRE IGNÁCIO
FILIPE PAIS LENFERS
LUIZ FABIANO VOIGT GARRETT
SÉRGIO PATRIQUE ZOTTO

EASYFAN

CURITIBA
2006

CLAITON WERNER KÜSTER
FÁBIO ALEXANDRE IGNÁCIO
FILIPE PAIS LENFERS
LUIZ FABIANO VOIGT GARRETT
SÉRGIO PATRIQUE ZOTTO

EASYFAN

Trabalho de Conclusão do Curso de Tecnologia em Informática da Universidade Federal do Paraná, setor Escola Técnica.

Orientador: Dieval Guizelini

Co-Orientador: Roberto Tadeu Raittz

CURITIBA

2006

TERMO DE APROVAÇÃO

CLAITON WERNER KÜSTER
FÁBIO ALEXANDRE IGNÁCIO
FILIPE PAIS LENFERS
LUIZ FABIANO VOIGT GARRETT
SÉRGIO PATRIQUE ZOTTO

EASYFAN

Trabalho de Conclusão de Curso avaliado e aprovado como requisito parcial para obtenção de grau de Graduado no Curso Superior de Tecnologia em Informática, Setor Escola Técnica da Universidade Federal do Paraná, pela seguinte banca examinadora:

Orientador: Professor Dieval Guizelini

Co-orientador: Professor Dr. Roberto Tadeu Raittz

Professor Dr. Mauro José Belli

Professora Msc Jeroniza Nunes Marchaukoski

Professor Luciano Mengarelli

Curitiba, 15 de dezembro de 2006

RESUMO

O Free Associative Neurons – FAN é uma técnica de redes neurais desenvolvida pelo Professor Dr. Roberto Tadeu Raittz. Para demonstrar e aperfeiçoar a técnica o professor desenvolveu o aplicativo LABFAN, originalmente desenvolvido em linguagem C para console, e posteriormente convertido para o Visual C.

Foi desenvolvida uma nova versão deste aplicativo, explorando novos recursos da técnica original e aperfeiçoando o algoritmo, modelado agora conforme os princípios da Orientação a Objeto. Escrito em Java esta nova implementação foi realizada com o apoio de várias pesquisas e simulações. Em sua trajetória de desenvolvimento, a interface foi construída com o intuito de auxiliar o uso da técnica sem a necessidade de conhecê-la, disponibilizando uma biblioteca de apoio que poderá ser incorporada aos aplicativos de terceiros.

O EasyFAN está pronto para ser utilizado por iniciantes da área de Inteligência Artificial, mais precisamente reconhecimento de padrões, bem como por pessoas experientes que desejarem conhecer e explorar os benefícios da técnica FAN. Os iniciantes poderão explorar a interface construída segundo o modelo "wizard" dos aplicativos para MS-Windows. Os mais experientes poderão refinar e avaliar o treinamento explorando o monitoramento eficiente da rede apresentado em forma de tabelas e gráficos.

ABSTRACT

The Free Associative Neurons - FAN is a neural network technique developed for the scientist Dr. Roberto Tadeu Raittz. In order to demonstrate and to perfect the technique the professor developed the software LABFAN. This software was developed in C language for console, and later converted to Visual C.

The new version of this software one was developed, explored new resources of the technique original and perfected the algorithm, shaped now in agreement the principles of the Object Orientation. Written in Java language, this new implementation was made with many searches and simulations. In the development, the inteface was made to help to the use of the thecnic, without to know it, fitting a support library incorporated to another softwares.

The EasyFAN is ready to be used by beginning of the area of Artificial Intelligence, more necessarily recognition of standards, as well as for experienced people whom to desire to know and to explore the benefits of technique FAN. The beginning ones will be able to explore the constructed interface according to model "wizard" of the applicatory ones for MS-Windows. Most experienced they will be able to refine and to evaluate the training exploring the efficient monitoramento of the net presented in form of tables and graphs.

SUMÁRIO

1. INTRODUÇÃO

A Inteligência Artificial é uma área da informática que tem contribuído com importantes avanços em todas as áreas do conhecimento humano, especialmente depois da popularização dos computadores. Ela Tem sido usada em várias aplicações, como planejamento autônomo, jogos, diagnósticos, planejamento logístico, robótica, reconhecimento de linguagem, entre outras funções.

Uma das principais ferramentas utilizadas nos problemas de reconhecimento de padrões (RP) são as redes neurais. O *Free Associative Neurons* – FAN é uma rede neural que utiliza a abordagem neuro-fuzzy que são conjuntos de sistemas híbridos que utilizam mais de uma técnica de identificação de sistemas para a solução de um problema de modelagem.

O FAN foi desenvolvido pelo Professor Dr. Roberto Tadeu Raittz. Durante o desenvolvimento e aperfeiçoamento da técnica, foi criado e implementado o LabFAN, software que permitiu o Dr. Raittz demonstrar sua criação. A primeira versão deste aplicativo foi escrita em C para console, e posteriormente recebeu uma nova implementação em Visual C.

O LabFAN implementa as fórmulas e técnicas criadas pelo Professor Raittz, mas não explora os recursos de interface gráfica, comum nos aplicativos atuais. Ele é limitado quanto à forma de entrada dos dados, fixa parâmetros de treinamento e não permitia a reutilização do algoritmo por outras aplicações. Estas limitações favoreceram a necessidade de se propor uma nova implementação do algoritmo, preservando a técnica e explorando recursos a muito desejado pelo seu autor, mas praticamente inviáveis ao modelo de implementação original.

O objetivo principal deste trabalho é a implementação da técnica FAN utilizando as melhores técnicas de desenvolvimento e criando uma interface que abstraia por completo o funcionamento específico da rede, permitindo sua utilização em qualquer plataforma e por usuários não especializados na área de reconhecimento de padrões.

O objetivo principal pode ser dividido nos seguintes objetivos específicos:

- implementação da rede FAN;
- obtenção de melhores métricas de software, em todos os quesitos, que o LABFAN original;
- obteção uma interface que permita a popularização da técnica.

O capítulo 2 deste trabalho abordará fundamentos teóricos sobre reconhecimento de padrões e o FAN, descrevendo também as tecnologias utilizadas para o desenvolvimento do componente EasyFAN.

No capítulo 3 é demonstrado a solução proposta do trabalho, mostrando detalhadamente os componentes JFAN, EasyFAN e ThingEasyFAN, alem de apresentar detalhes da implementação da API do JFAN e como é realizado o tratamento de eventos no EasyFAN.

Para finalizar os capítulos 4 e 5 mostram toda as especificações técnicas como casos de uso, diagrama de classes, diagrama de seqüência, entre outros, dos componentes JFAN e EasyFAN.

1.1. JUSTIFICATIVA

O Prof. Dr Roberto Tadeu Raittz desenvolveu pesquisas de Inteligência Artificial durante o final dos anos 90 e início dos anos 2000, tendo produzido a dissertação ***Free Associative Neurons – FAN: uma abordagem para reconhecimento de padrões***, e defendida na Universidade Federal de Santa Catarina em 1997, e a tese **FAN 2002: *Um modelo neuro-fuzzy para reconhecimento de padrões***, defendida igualmente na Universidade Federal de Santa Catarina em 2002.

Para demonstrar na prática sua tese, o Prof. Dr Roberto Tadeu Raittz produziu um programa denominado LABFAN, que foi escrito em C, e faz reconhecimento de padrões de arquivos textos armazenados com extensão DAT.

O LabFAN original foi desenvolvido durante a concepção da técnica FAN, sem planejamento específico do software. A preocupação era demonstrar e aperfeiçoar o modelo, e as preocupações com as técnicas algorítmicas ficaram em segundo plano. No EasyFAN invertemos a preocupação, partimos da premissa de que a técnica esta completamente definida e focamos nossos objetivos nos modelos de implementação e conceitos da Orientação Objetos, ganhando assim os recursos disponíveis e aplicáveis da API do JAVA.

2. FUNDAMENTOS TEÓRICOS

2.1. RECONHECIMENTO DE PADRÕES

O reconhecimento de padrões é a ciência que trata da classificação de padrões através de um conjunto de características. Um padrão é uma entidade que pode ser classificada através de suas características.

Humanos conseguem desempenhar a tarefa de reconhecer padrões muito bem, mas existe uma pressão para que se desenvolvam tecnologias que possam fazer essa tarefa de forma mais rápida e barata. Existem também padrões que são difíceis de serem reconhecidos por um humano, como por exemplo, o código de barras.

O reconhecimento de padrões também faz parte do processo de tomada de decisão desempenhado por pessoas, como por exemplo, em um jogo de xadrez a decisão do próximo movimento é geralmente feita se analisando a posição das peças no tabuleiro, ou seja, estudando o atual padrão em que o tabuleiro se encontra.

RIPLEY (2005) comenta alguns usos para as técnicas de reconhecimento de padrões:

- Diagnosticar doenças
- Identificar tipos de carro, avião, etc...
- Identificar impressões digitais, DNA, letras manuscritas, etc...
- Avaliar risco de crédito
- Classificar galáxias por forma
- Escolher uma estratégia ou movimento em um jogo de xadrez

- Reconhecer condições perigosas para dirigir

O reconhecimento de padrões feito por humanos possui uma característica que é o aprendizado. Em geral, nós aprendemos como reconhecer certos padrões, e com as máquinas ocorre da mesma maneira. O aprendizado que é realizado nas máquinas é conhecido como aprendizagem de máquinas, e pode ser estabelecido por algoritmos que melhoram com a prática, ou seja, cujo desempenho em determinada tarefa melhora com a experiência.

Quando o aprendizado é feito em um conjunto de padrões que ainda não possuem classificação e o algoritmo deve sugerir essa classificação, chamamos esse aprendizado de não-supervisionado, em contrapartida quando conhecemos antecipadamente a classificação dos padrões de um conjunto que será usado para o aprendizado, esse conjunto é denominado de treinamento e chamamos essa aprendizagem de supervisionada.

A figura 1 ilustra um sistema típico de reconhecimento de padrões:

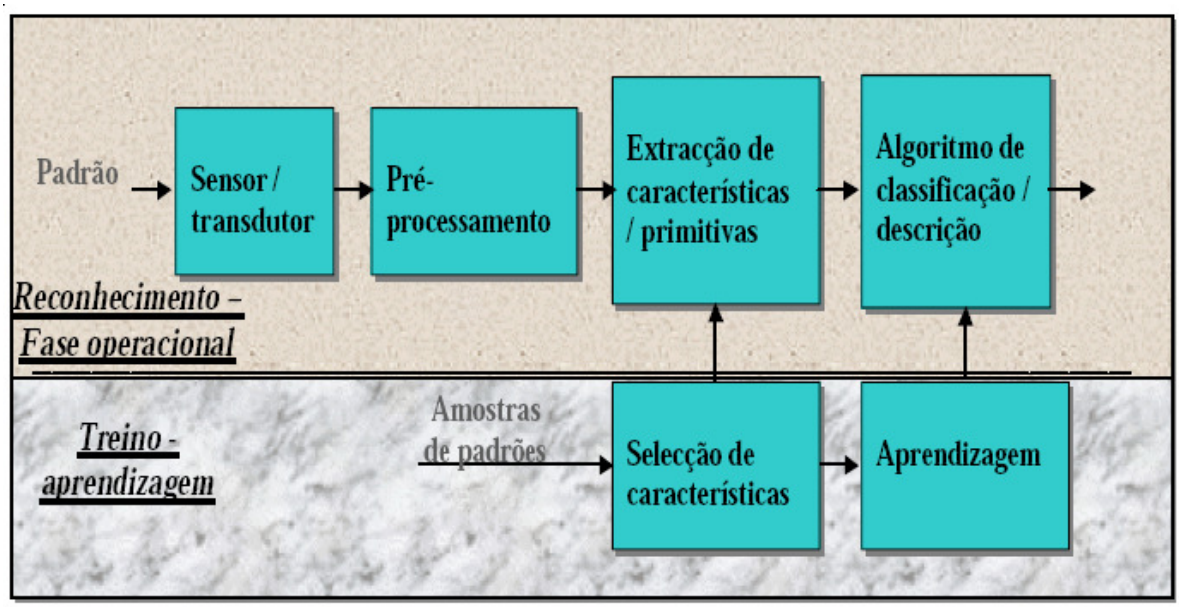

Figura 1 - *Reconhecimento de Padrões.* Disponível em: <http://www.lx.it.pt/~afred/rp-ist/ACETATOS/RP_1_Introducao.PDF>. Acesso em: 5 dez. 2006.

Com base na Figura 1, percebemos que na fase de treino/aprendizagem são submetidos padrões, amostras já pré-selecionadas, dos quais serão escolhidas características, que depois serão usadas na aprendizagem para treinar o algoritmo de reconhecimento de padrões.

Na fase de reconhecimento/operacional temos um padrão (entidade) que será digitalizado por um sensor. O padrão digitalizado sofre o pré-processamento para a preparação dos dados para o algoritmo de reconhecimento de padrões, e os dados poderão ser pré-processados podendo assim se eliminar ruídos, realizar a normalização, criar outros dados combinatórios ou outra função qualquer que seja necessária para premeditar os dados. Após todo esse processo, as características são selecionadas e submetidas ao algoritmo de classificação/descrição que irá determinar o padrão na classe que o algoritmo achar mais adequado.

Geralmente para se realizar testes no sistema de reconhecimento de padrões, é usado um segundo conjunto de padrões conhecido como conjunto de teste. Esse conjunto visa garantir que o algoritmo de reconhecimento de padrões consiga reconhecer padrões nunca vistos antes em seu aprendizado.

2.2. FAN – FREE ASSOCIATIVE NEURONS

2.2.1. Definição e Contextualização

RAITTZ (2002) define FAN como “uma abordagem baseada em conjuntos difusos e redes neuronais para o reconhecimento de padrões”.

Em FAN o estudo do padrão não considera apenas um ponto, mas toda a região ao redor desse ponto. A esse evento chamamos de “decomposição do padrão”, e é neste processo que é gerada uma vizinhança difusa, ou seja, todo padrão é convertido em um conjunto de vizinhanças difusas.

É pelo treinamento que FAN “aprende” a reconhecer os padrões. O treinamento utiliza-se de dois conjuntos de padrões: um conjunto de treinamento e um conjunto de teste. É pelo conjunto de treinamento que a rede realiza o treino e adquire o aprendizado, mas os resultados do treinamento são medidos testando o conjunto de teste.

A Equação 1 demonstra o conjunto de treinamento, tal que C é igual ao número de característica, e H é igual ao número de padrões. h representa o h-ésimo padrão, e c representa a c-ésima característica. Cada X é um padrão.

$$X_{tc}^{tr}; c = 1, C; h = 1, H;$$

Equação 1- Notação dos conjuntos de treinamento

A equação 2 apresenta a representação do conjunto de teste, sendo que C é igual ao número de característica, e L é igual ao número de padrões. l representa o l-ésimo padrão, e c representa a c-ésima característica. Cada X é um padrão.

$$X_{lc}^{te}; c = 1, C; l = 1, L;$$

Equação 2 - notação dos conjuntos de testes

Conforme definido pelo Dr Raittz, o procedimento de treinamento foi dividivo em nove fases:

1. Normalização
2. Determinação do FAN inical
3. Construção dos conjuntos difusos para os padrões
4. Normalização dos graus de pertinência

5. Determinação da média geométrica do FAN e dos graus de pertinência modificados do padrão
6. Determinação da força de representação do padrão
7. Tomada de decisão
8. Penalização e/ou reforço
9. Teste

Para explorar o recurso de multi-thread do Java, a equipe realizou pesquisas para avaliar quais destes passos poderiam ser executados independentemente e estabelecer a real necessidade da seqüência a ser observada. Está discussão será apresentada posteriormente.

2.2.2. Normalização

Cada característica de cada padrão do conjunto de treinamento e do conjunto de teste é normalizada e recebe um valor compreendido entre 0 e 1 . Abaixo 'X é o valor de X normalizado.

$$'X_{hc}^{tr} = \frac{X_{hc}^{tc}}{\max_{h} X_{hc}^{tc}} \qquad c = 1, C \qquad h = 1, H$$

Equação 3 - Fórmula de normalização

2.2.3 Determinação do FAN inicial

Um neurônio FAN é criado para cada classe distinta em ambos os conjuntos. Cada neurônio representa uma classe e também uma matriz sendo que a primeira dimensão possui o tamanho igual ao número de características dos padrões e a segunda dimensão possui um tamanho definido (geralmente 100), chamado de número de suporte dos conjuntos difusos, e é nesse espaço que o neurônio guardará o seu "conhecimento" (a representação do padrão para a classe que o neurônio representa). Cada posição da matriz é preenchida com um número aleatório que é limitado a ficar entre 0 e 1. Abaixo vemos a representação dos neurônios , tal que cada Y_i representa um neurônio:

Y_{icj}

I=1,I – classes

c=1,C – características

J=1,J – número definido para o suporte dos conjuntos difusos

2.2.4. Construção dos conjuntos difusos para os padrões

Com os conjuntos já normalizados, o valor de cada característica de cada padrão dos conjuntos é multiplicado pelo número de suporte dos conjuntos difusos descrito no passo anterior.

$$''X_{lc}^{tr} = 'X_{lc}^{tr} xJ$$

Equação 4 - Construção dos conjuntos difusos

Para cada característica de cada padrão dos conjuntos é criado um conjunto difuso. Para criação dos conjuntos difusos usa-se um parâmetro chamado de raio difuso. O valor atual de cada característica será considerado o centro do conjunto difuso, para cada lado do centro do conjunto difuso serão considerados um número

de vizinhos igual ao raio difuso subtraído de um pois esses valor irão possuir grau de pertinência diferente de 0. O grau de pertinência de cada posição do conjunto difuso é determinado por uma função de pertinência, geralmente é utilizada uma função de pertinência triangular. Abaixo segue a representação do conjunto difuso:

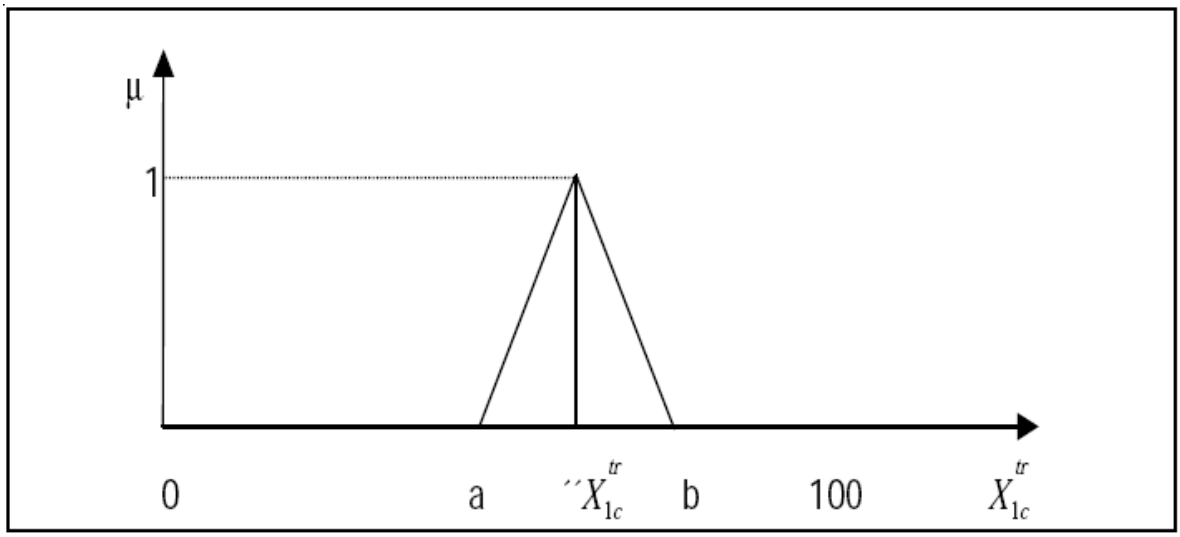

Figura 2 - RAITTZ (2001) – Representação fuzzy de um ponto e sua pertinência

Legenda

M	Grau de pertinência entre 0 e 1
a ... b	Limites
X_{1c}^{tr}	Conjunto de treinamento
$''X_{1c}^{tr}$	Valor normalizado

No eixo μ estão os valores que identificam o grau de pertinência do conjunto (valores compreendidos entre 0 e 1). Os limites da vizinha são determinados pelo intervalo [a,b], o conjunto de treinamento é representado por X_{lc}^{tr}, e o símbolo $''X_{lc}^{tr}$ representa o valor normalizado do padrão na característica observada.

Os valores para os limites a e b são definidos pelas equações 5 e 6 respectivamente.

$$a = \text{int}(''X_{lc}^{tr}) - d + 1$$

Equação 5 - Função para determinação do limite a

$$b = \text{int}(''X_{lc}^{tr}) + d$$

Equação 6 - Função para determinação do limite b

Observa-se que $\text{int}(''X_{lc}^{tr})$ representa somente a parte inteira do valor de $''X_{lc}^{tr}$.

Neste caso, os conjuntos difusos são definidos por uma função de pertinência triangular (representada pela equação 7).

$$\mu(X_{1c}^{tc}) \begin{cases} 0 \quad se\, X_{1c} \le a \quad ou \quad X_{1c} \ge b \\ \dfrac{X_{1c} - a}{''X_{1c}^{tr} - a} \quad se \quad a \le X_{1c} \le ''X_{1c}^{tr} \\ \dfrac{b - X_{1c}}{b - ''X_{1c}^{tr}} \quad se \quad ''X_{1c}^{tr} \le X_{1c} \le b \end{cases}$$

Equação 7 - Função de pertinência triangular

μ_{icj} denota os graus de pertinência.

2.2.5. Normalização dos graus de pertinência

Os valores dos conjuntos difusos (graus de pertinência) são normalizados, e cada valor do conjunto difuso é dividido pela soma de todos os valores do conjunto. Conforme representado pela equação 8.

$$'\mu_{icj|_{h=1}} = \frac{\mu_{icj|_{h=1}}}{\sum_j \mu_{icj|_{h=1}}} \quad para \quad c = 1, C \quad e \quad j = 1, J$$

Equação 8 - Normalização dos conjuntos difusos

2.2.6. Determinação da média geométrica do FAN e dos graus de pertinência modificados do padrão

Para determinar a media geométrica do FAN, primeiro são calculados os valores normalizados dos neurônios. Na equação 9 podemos observar o valor normalizado do neurônio representado por "Y r , e Δ_{ic} representando o total de penalizações que esse neurônio sofreu.

$$''Y_{icj} = \frac{'Y_{icj}}{\sum_j 'Y_{icj} + \Delta_{ic}}$$

Equação 9 - Valor normalizado do neurônio

A equação 10 apresenta a fórmula da média geométrica. Esta média é obtida através da raiz quadrada do produto dos valores dos conjuntos difusos pelos valores normalizados dos neurônios. O símbolo "μ representa o resultado (ou valor obtido) desta média.

$$''\mu_{icj|_{h=1}}^{1} = \sqrt{'\mu_{icj|_{h=1}} X''Y_{icj}} \qquad i = 1, I \quad c = 1, C \quad j = 1, J$$

Equação 10 - média geométrica

2.2.7. Determinação da força de representação do padrão

A força do neurônio é representada pelas equações 11 e 12. O símbolo ' μ_i representa a força do neurônio.

$$''\mu_{ic|_{h=1}}^{u} = 1 - \prod_{j} (\sqrt{1 - ''\mu_{icj|_{h=1}}^{i}}) \qquad i = 1, I \quad c = 1, C$$

Equação 11 - Média geométrica do neurônio

$$''\mu_{i|_{h=1}}^{u} = 1 - \prod_{c} (1 - ''\mu_{ic|_{h=1}}^{u}) \qquad i = 1, I$$

Equação 12 - Força do neurônio

No Quadro 1 é apresentado a implementação (em Java) da média geométrica e da função para determinação da força do neurônio.

```
double determinarForca(IPadrao p) {
             int num_carac;
             double retorno = 1.0;
             double prodSomas;
             CaracteristicaFAN mi;
             double[] conjuntoMi;
             num_carac = p.getQuantasCaracteristicas();
             for (int ii = 1; ii <= num_carac; ii++) {
                    mi = p.getCaracteristicaFAN(ii);
                    conjuntoMi = mi.getConjuntoDifuso();
                    prodSomas = mi.getSomatorio() * this.getSomatorio(ii);
                    int k = 0;
                    double num1 = 0;
                    if (this.getSomatorio(ii) > 0) {
                           int fim = mi.getFim();
                           for (int j = mi.getInicio(); j <= fim; j++) {
                                  num1 = conjuntoMi[k] * this.getValor(ii, j) /
prodSomas;
```

```
                                    num1 = Math.sqrt(num1);
                                    retorno *= (1 - num1);
                                    k++;
                                }
                            }
                        }
                        double expoente = 1 / (double)num_carac;
                        double num2 = Math.pow(retorno, expoente);
                        retorno = (1 - num2);
                        return retorno;
}
```

Quadro 1 - Implementação Java da média geométrica e da força do neurônio

2.2.8. Tomada de decisão

Agora a classe que FAN sugere para o padrão X é a classe i, sendo tomado o i que tem o maior valor de força de representação (' μ_i).

Reforça-se o neurônio que representa classe a qual o padrão X realmente pertence. Se a classe real (correta) do padrão X for diferente da classe sugerida pela rede para o padrão X, o neurônio que representa a classe sugerida por FAN deve ser penalizado.

O Quadro 2 apresenta o código Java desenvolvido para este procedimento.

```
NeuronioFAN classificar(IPadrao p) {
       NeuronioFAN melhorNeuronio = this.neuronios.get(0);
       double maiorForca = melhorNeuronio.determinarForca(p);
       double forcaAtual;
       for (int i= 1;i<this.neuronios.size();i++) {
              forcaAtual = this.neuronios.get(i).determinarForca(p);
              if (maiorForca < forcaAtual) {
                     melhorNeuronio = this.neuronios.get(i);
                     maiorForca = forcaAtual;
              }
       }
       return melhorNeuronio;
}
```

Quadro 2 - Implementação em Java da função de tomada de decisão

2.2.9. Penalização e/ou reforço

A penalização e/ou reforço só ocorre usando os padrões do conjunto de treinamento.

Para o cálculo do reforço, representado pela equação 13 , o padrão é identificado pelo símbolo X_{1c}^{tr} e a classe real é representada por p_i.

$$'Y_{icj}^{R} = '\mu_{icj} + 'Y_{icj}, \quad para \quad i = p_i, j = 1, J, c = 1, C$$

Equação 13 - fórmula do reforço

O quadro 3 apresenta a implementação deste recurso.

```
reforcar(IPadrao p) {
	CaracteristicaFAN caracFan;
	int k, fim;
	NeuronioFAN neuronio = this.neuronios.get(p.getClasse());
	int carateristicas = neuronio.quantasCaracteristicas();
	for (int i = 1; i < carateristicas; i++) {
		k = 0;
		caracFan = p.getCaracteristicaFAN(i);
		fim = caracFan.getFim();
		for(int j = caracFan.getInicio(); j <= fim; j++) {

	neuronio.adicionaValor(i,j,(caracFan.getConjuntoDifuso()[k]));
			k++;
		}
	}
}
```

Quadro 3 - Implementação em Java da função de reforço

A penalização ocorre apenas nos casos de erro da rede, ou seja, quando a classe sugerida é diferente da classe real do padrão.

O valor da penalização é calculado considerando o padrão X_{1c}^{tr}, a classe sugerida pela rede como $'p_i$ e o parâmetro α que determina a intensidade da penalização. A penalização é melhor representada na equação 15.

$$'\mu_{icj}^{P} = 1 - a * ('\mu_{icj}), \quad i = p'_i, \quad j = 1, J, \quad c = 1, C$$

Equação 14 – Fator de penalização em função da força de representação

$$'Y_{icj}^{P} = '\mu_{icj}^{P} * 'Y_{icj}$$

Equação 15 - fórmula da penalização

O quadro 4 apresenta a implementação da penalização.

```
penalizar(NeuronioFAN neuronio, IPadrao p) {
	double novoValor;
	int fim, k;
	CaracteristicaFAN caracFan;
	int carateristicas = neuronio.quantasCaracteristicas();
	for (int i = 1; i <= carateristicas; i++) {
		k = 0;
		caracFan = p.getCaracteristicaFAN(i);
		fim = caracFan.getFim();
		for(int j = caracFan.getInicio(); j <= fim; j++) {
			novoValor = neuronio.getValor(i,j);
			if (random.nextInt(100)>10) {
				novoValor                                                    *=
Math.pow(caracFan.getConjuntoDifuso()[k],0.00001);
			} else {
				novoValor -=  caracFan.getConjuntoDifuso()[k];
			}
			synchronized (this) {
				if(this.tempera != null) {
					novoValor *= tempera.getValor();
				}
			}
			if (novoValor<0) novoValor = 0;
				neuronio.setValor(i,j,novoValor);
			k++;
		}
	}
}
```

Quadro 4 - Função penalizar da biblioteca jFAN

2.2.10. Teste

Nesse passo é medida a taxa de erro da rede, tal que apenas o conjunto de teste é usado para gerar a taxa de erro. Considerando p_i como a classe real do padrão, $'p_i$ como a classe sugerida da rede, l como um padrão do conjunto de teste, L como a quantidade de padrões do conjunto de teste e t_e como a taxa de erro, temos uma equação de taxa de erro apresentada na equação16:

$$t_e = \frac{L - \sum_e U(l)}{L} * 100\%$$

Equação 16 - Taxa de erro

$$onde \quad U(l) = \begin{cases} 1 \quad se \quad P_i = P'_i \\ 0 \quad se \quad P_i \neq P'_i \end{cases}$$

Para taxas de erros menores, obtém-se maiores capacidades de reconhecimento pela rede FAN.

Os procedimentos descritos são executados continuamente e cada ciclo de execução denominamos de “época” de treinamento. Para cada época todos os passos de treinamento e teste são realizados novamente.

2.2.11. As Implementações de FAN

Inúmeros testes e implementações foram realizadas sobre conjuntos de dados utilizando-se a técnica FAN. RAITTZ (2002) apresenta os mais significativos:

> *“1996 - Primeira implementação em Borland-C com parâmetro de combinação de características H=1, com resultado obtido sobre o conjunto de testes de 91,0 % de acertos.*
> *1997 - Implementação em Clipper Summer 87, com parâmetro de combinação de características H=2, com acertos de 94,6 %, o mesmo obtido por Herington (TODESCO, 1995; RAITTZ, 1997).*
> *1997 - Implementação modificada em Borland-C, com taxa de acerto de 95,25% de acertos.*
> *1997 - Primeira implementação em Delphi visando a obter um produto genérico para o uso de FAN.*
> *1998, 1999, 2000 - Novas alterações no Programa em C com alterações na forma de penalização e eliminação do parâmetro H, em que foi obtido um percentual de acerto superior a 96% (igualando a TODESCO).*
> *2000 - Nova implementação em DELPHI, realizada no laboratório de jogos de empresas da UFSC nos mesmos moldes.*
> *2001 – Obtenção, pela alteração da forma de treinamento, usando-se, para cada época, padrões escolhidos aleatoriamente no conjunto*

> *de treinamento com uso de têmpera simulada, a melhor taxa de acertos: 96,36% com erro de 3,64%. A têmpera simulada, nesse contexto, significa a diminuição gradativa da penalização conforme o avanço do número de épocas.*
> *As alterações realizadas em relação às primeiras implementações na forma de treinamento adotada, especialmente no que se refere à penalização, permitem ao método fugir de mínimos locais da taxa de erro." (RAITTZ, 2002)*

2.2.12. FAN aplicado ao problema XOR

Em 1969, Minsky & Papert constataram que um neurônio do tipo *Perceptron* só é capaz de resolver problemas com dados de classes linearmente separáveis. Logo não conseguiriam resolver problemas do tipo XOR (ou exclusivo). Após uma desmotivação inicial, estes problemas foram solucionados por uma rede neural.

A solução é apresentada na tabela 1, onde cada linha representa um padrão, as colunas X1 e X2 são as características observadas e a coluna C possui a classe do padrão.

X1	X2	C
0	0	0
0	1	1
1	0	1
1	1	0

Tabela 1 - Representação do problema do XOR

No caso do problema XOR, RAITTZ (2002) descreve: "Ocorre que essas características não são independentes entre si, o que a estrutura do algoritmo de FAN prevê a princípio". Então, FAN no seu modelo atual não consegue resolver o problema XOR, mas se utilizarmos uma característica extra que é feita pela combinação das outras características, o problema XOR pode ser resolvido por FAN.

Esta nova característica denominará X3 e os valores serão obtidos pela combinação das características X1 e X2 conforme demonstrado na equação 17.

$$X3 = \sqrt{\frac{X1X2}{2}}$$

Equação 17 - cálculo da característica X3 para o problema do XOR

X1	X2	X3	C
0	0	0	0
0	1	0	1
1	0	0	1
1	1	0,7	0

Tabela 2 - Dados do problema XOR, incluído a nova característica

Com a inclusão da característica X3 ao conjunto das características originais de cada padrão, a rede FAN consegue resolver o problema XOR facilmente.

Portanto, a interface da aplicação de treinamento da rede FAN deve permitir preparar os padrões a serem analisados, combinando as características existentes. Este recurso é relevante ao algoritmo FAN.

2.3. TECNOLOGIAS UTILIZADAS

2.3.1. Java

A linguagem Java[1] tem sua origem nos anos 90, com a empresa Sun Microsystems, sendo uma de suas principais características ser uma linguagem de

[1] **Java:** Disponível em: <http://java.sun.com>. Acesso em: 3 abr. 2006.

programação orientada a objetos. Isto significa que se procura identificar qual é o melhor conjunto de objetos que vão descrever um sistema computacional, havendo o relacionamento e a troca de mensagens entre estes objetos.

O que difere a linguagem Java de outras linguagens é que a linguagem Java é compilada para o que se denomina *bytecode*, sendo posteriormente executada numa máquina virtual, enquanto que outras linguagens são compiladas para um código nativo, que é dependente de determinada plataforma.

Pode-se dizer que a linguagem Java foi projetada para ter as seguintes características:

- Ter orientação a objeto, sendo baseada no modelo Smaltalk e Simula67.
- Ter portabilidade, ou seja, independência de plataforma, podendo rodar em qualquer sistema operacional: "write once run anywhere".
- Possuir recursos de rede, com extensa biblioteca de rotina que podem facilitar a cooperação com protocolos TCP/IP, como http e ftp.
- Atenção à segurança, pois pode executar programas via rede com restrições para execução.
- Sintaxe parecida com a da linguagem C/C++.
- Facilidade de internacionalização, pois suporta caracteres unicode[2].
- Simplicidade na especificação, na linguagem e no ambiente de execução da JVM.
- Há a distribuição de um grande conjunto de bibliotecas, chamadas API.
- Podem ser criados programas distribuídos e multitarefa, ou seja, múltiplas linhas de execução num mesmo programa.

---

[2] **Unicode**: padrão em que cada código representa somente um símbolo, independente da localidade. Padronizada pelo Unicode-Consortium. (http://www.unicode.org)

- Deslocamento de memória automática, por um processo de coletor de lixo.
- Carga dinâmica de código, pois os programas são formados por uma coleção de classes armazenadas independentemente e que são carregadas quando são utilizadas.

2.3.2. Eclipse

O Eclipse[3] é o resultado do trabalho realizado por uma comunidade de software livre que tem o seu foco na construção de uma plataforma de desenvolvimento com *frameworks*, ferramentas e *runtimes* que vão possibilitar a construção, o desenvolvimento e gerenciamento do programa durante o seu ciclo de vida.

A plataforma do Eclipse está sendo desenvolvida, complementada e tem o suporte por parte de um grande grupo de vendedores de tecnologia, inovadores, universidades, instituições de pesquisa e indivíduos.

2.3.3. Thinlet

Thinlet[4] é uma ferramenta para a construção de uma interface com o usuário, construída em uma classe Java simples, que analisa a hierarquia e as propriedades da interface, controlando a interação com o usuário e chamando o serviço de lógica.

---

[3] **Eclipse**: Disponível em: <http://www.eclipse.org>. Acesso em: 3 abr. 2006.

[4] **Thinlet:** Disponível em: <http://www.thinlet.com>. Acesso em: 3 abr. 2006.

Uma das vantagens, é que a apresentação gráfica é descrita em um arquivo XML, separada dos métodos de aplicações, escritos em código Java.

Outra vantagem do Thinlet é que seu tamanho é extremamente reduzido, sendo comprimido ao total de apenas 39 KB.

Sobre o licenciamento, o Thinlet se integra à categoria LGPL (Lesser General Public License)

O requisito mínimo para rodar o Thinlet é ter o Java 1.1 na máquina virtual.

2.3.4. Thing

O Thing[5] é um editor para interfaces que é escrito em XML e que usa o Thinlet.

Para que o Thing seja instalado e funcione adequadamente é necessário que se tenha no computador a ser usado uma Máquina Virtual Java.

Sua licença é GLP (General Public License) e o Thing usa o próprio Thinlet para funcionar.

2.3.5. JfreeChart

[5] **Thing**: Disponível em: <http://thing.sourceforge.net>. Acesso em: 3 abr. 2006.

JfreeChart[6] é uma biblioteca livre escrita em Java, que permite o desenvolvimento profissional de gráficos e assemelhados de forma fácil e sem maiores dificuldades, que serão exibidos em aplicações.

As características principais JfreeChart são:

- Uma API consistente e bem documentada, possibilitando uma grande gama de tipos de gráficos;
- Um projeto flexível que é fácil de estender e aplicações que atingem tanto servidores quanto clientes;
- Suporte a diversos tipos de saídas, incluindo componentes em Swing, arquivos de imagens tipo raster[7] (entre outras, como por exemplo arquivos de imagens png e jpeg), e formatos de arquivos gráficos vetoriais (pdf, eps e svg).
- A biblioteca JFreeChart é distribuída nos termos da LGPL.

2.3.6. JavaHelp

Trata-se de um sistema de ajuda extensiva com uma feição própria, independente de plataforma que possibilita incorporar ajudas *on-line* em *applets*, componentes, aplicações, sistemas operacionais e dispositivos. O JavaHelp[8] pode ser usado para distribuição da documentação on-line pela internet ou intranet.

---

[6] **JfreeChart:** Disponível em: <http://www.jfree.org/jfreechart>. Acesso em: 6 set. 2006.

[7] **Raster**: Imagens que contém a descrição de cada pixel, em oposição aos gráficos vetorias.

[8] **JavaHelp**: Disponível em: <http://java.sun.com/products/javahelp>. Acesso em: 6 set. 2006.

Por ter sido escrito na linguagem de programação Java, o sistema de JavaHelp pode rodar em qualquer plataforma ou navegador que suporta o Java *Runtime Environment*.

O sistema é implementado usando os componentes de software do Java Foundation Classes (JFC), estipulando tanto a flexibilidade quanto a facilidade para desenvolver interfaces direcionadas ao usuário ou cliente e também para a funcionalidade. Por exemplo, o sistema do JavaHelp pode ser visualizado na sua própria janela ou unido em uma aplicação. Podem ser também adicionados controles de navegação de acordo com a vontade do usuário.

A janela de ajuda consiste dos seguintes componentes:

- *toolbar*
- *content pane*
- *navigation pane*

O *content pane* usa HTML 3.2 como formato para visualizar os tópicos. Componentes *Lightweight* Java também podem ser usados para adicionar funcionalidades.

A navegação na ajuda possui:

- Conteúdo (Table-of-contents): suporta um visual de tópicos que pode ser expandido, com ilimitado número de níveis hierárquicos.
- Índice (Index): que suporta a fusão de vários índices.
- Palavras chave (Full-text search): foi projetado para ser flexíveis e configuráveis. A máquina de busca pode estar instalada ou no cliente ou no servidor.

2.4. JUSTIFICATIVA PARA A ESCOLHA DA TECNOLOGIA ADOTADA

> *It is an old observation that the best writers sometimes disregard the rules of rhetoric. When they do so, however, the reader will usually find in the sentence some compensating merit, attained at the cost of the violation. Unless he is certain of doing as well, he will probably do best to follow the rules.*
> *William Strunk e E. B. White, The Elements of Style (1918)*
> *(http://www.bartleby.com/141/strunk1.html)*

Conforme orientação do professor responsável, todo recurso extra utilizado em substituição dos recursos previstos na API da linguagem Java, deve possuir justificativa técnica.

A seguir apresentamos a justificativa pela utilização do Thinlet, recurso alternativo a API padrão Swing.

Foram feitos testes de memória com o NetBeans Profile usando a interface do Thinlet e a interface Swing. Observou-se que a interface que utilizou o Thinlet consome menos memória do que a com Swing.

O Thinlet usa o xml (eXtensible Markup Language) para definir a interface, dessa forma a definição da interface fica separada da programação permitindo se alterar mais facilmente a interface. A forma de xml usada pelo Thinlet para definir a interface é muito simples e próxima do html, facilitando e agilizando o desenvolvimento das janelas e diálogos.

Só a título de comparação, o objeto denominado char[] aloca 12.592B no Thinlet e 47.680B no Swing.

Ao se criar a mesma interface utilizando thinlet e swing, o profile do Netbeans apresenta a ocupação de memória descrita na tabela 3.

Class Name – Allocated Objects	Bytes Allocated				Objects Allocated			
	Thinlet		Swing		Thinlet		Swing	
char[]	12.592B	(15,3%)	47.680B	(29,6%)	2.278	(13,7%)	4.158	(17,9%)
byte[]	8.872B	(10,8%)	15.744B	(9,8%)	213	(1,3%)	63	(0,3%)
java.lang.Object[]	7.824B	(9,5%)	9.032B	(5,6%)	3.574	(21,5%)	2.645	(11,4%)
int[]	6.328B	(7,7%)	1.416B	(0,9%)	1.263	(7,6%)	126	(0,5%)
java.lang.String	4.608B	(5,6%)	6.072B	(3,8%)	1.851	(11,1%)	2.430	(10,4%)
sun.java2d.loops.GraphicsPrimitive[]	4.544B	(5,5%)	3.792B	(2,4%)	43	(0,3%)	43	(0,2%)
java.util.HashMap$Entry	2.808B	(3,4%)	4.728B	(2,9%)	1.068	(6,4%)	1.797	(7,7%)
java.util.HashMap$Entry[]	2.608B	(3,2%)	2.880B	(1,8%)	89	(0,5%)	183	(0,8%)
javax.swing.Jlabel			2.208B	(1,4%)			60	(0,3%)
javax.swing.Jbutton			2.200B	(1,4%)			38	(0,2%)
java.lang.StringBuilder			2.064B	(1,3%)			1.216	(5,2%)
float[]	2.056B	(2,5%)			298	(1,8%)		
java.nio.HeapByteBuffer	1.680B	(2%)			333	(2%)		
java.beans.PropertyChangeEvent			1.664B	(1%)			489	(2,1%)
java.awt.Rectangle	1.656B	(2%)	600B	(0,4%)	638	(3,8%)	237	(1%)
java.nio.HeapCharBuffer	1.632B	(2%)			306	(1,8%)		
java.awt.Dimension	1.536B	(1,9%)	832B	(0,5%)	930	(5,6%)	512	(2,2%)
java.lang.Short[]	1.040B	(1,3%)	1.040B	(0,6%)	2	(0%)	2	(0%)
java.lang.Integer[]	1.040B	(1,3%)	1.040B	(0,6%)	1	(0%)	1	(0%)
java.util.StringTokenizer			1.000B	(0,6%)			232	(1%)
easyfan.Principal			944B	(0,6%)			1	(0%)
javax.swing.plaf.metal.MetalScrollButton			928B	(0,6%)			16	(0,1%)
java.lang.Integer	864B	(1,1%)	944B	(0,6%)	523	(3,1%)	555	(2,4%)
java.awt.Component[]			824B	(0,5%)			365	(1,6%)
java.awt.geom.RectIterator	728B	(0,9%)			117	(0,7%)		
java.lang.Object[][]			728B	(0,5%)			7	(0%)
double[]	704B	(0,9%)			110	(0,7%)		
javax.swing.JScroolPan$ScrollBar			704B	(0,4%)			8	(0%)
javax.swing.JSaparator			656B	(0,4%)			9	(0%)
javax.swing.Jpanel			640B	(0,4%)			16	(0,1%)
java.lang.String[]	568B	(0,7%)			89	(0,5%)		
int[]	552B	(0,7%)			110	(0,7%)		
long[]	472B	(0,6%)			11	(0,1%)		
javax.lang.Short	432B	(0,5%)			256	(1,5%)		
EasyFan	400B	(0,5%)			1	(0%)		
javax.lang.ref.SoftReference	384B	(0,5%)			95	(0,6%)		
java.util.Hashtable$Entry	384B	(0,5%)	3.168B	(2%)	153	(0,9%)	1.231	(5,3%)
thinlet.FrameLauncher	352B	(0,4%)			1	(0%)		
javax.util.Hashtable$Entry[]	328B	(0,4%)	6.648B	(4,1%)	39	(0,2%)	141	(0,6%)

Tabela 3 - Comparação do uso de memória entre as API Swing e Thinlet

2.5. O APLICATIVO LABFAN

O LabFAN tem por característica, a junção da rede original desenvolvida por Roberto Tadeu Raittz e o laboratório desenvolvido por André Luis Biaggi em um Projeto da empresa Ciashop – Soluções para Comércio Eletrônico e é um Software implementado em linguagem C que encapsula o modelo FAN e é dependente de plataforma.

A versão 1.0 possui as seguintes funcionalidades:

- Carregador de arquivo com suporte somente a arquivos .dat.
- Possibilita a gravação de redes já treinadas.
- Executor e gerenciador de treinamento, teste.
- Gerador de matriz de confusão.
- Possibilita a configuração de alguns poucos parâmetros de treinamento.

2.5.1. Opções do Software

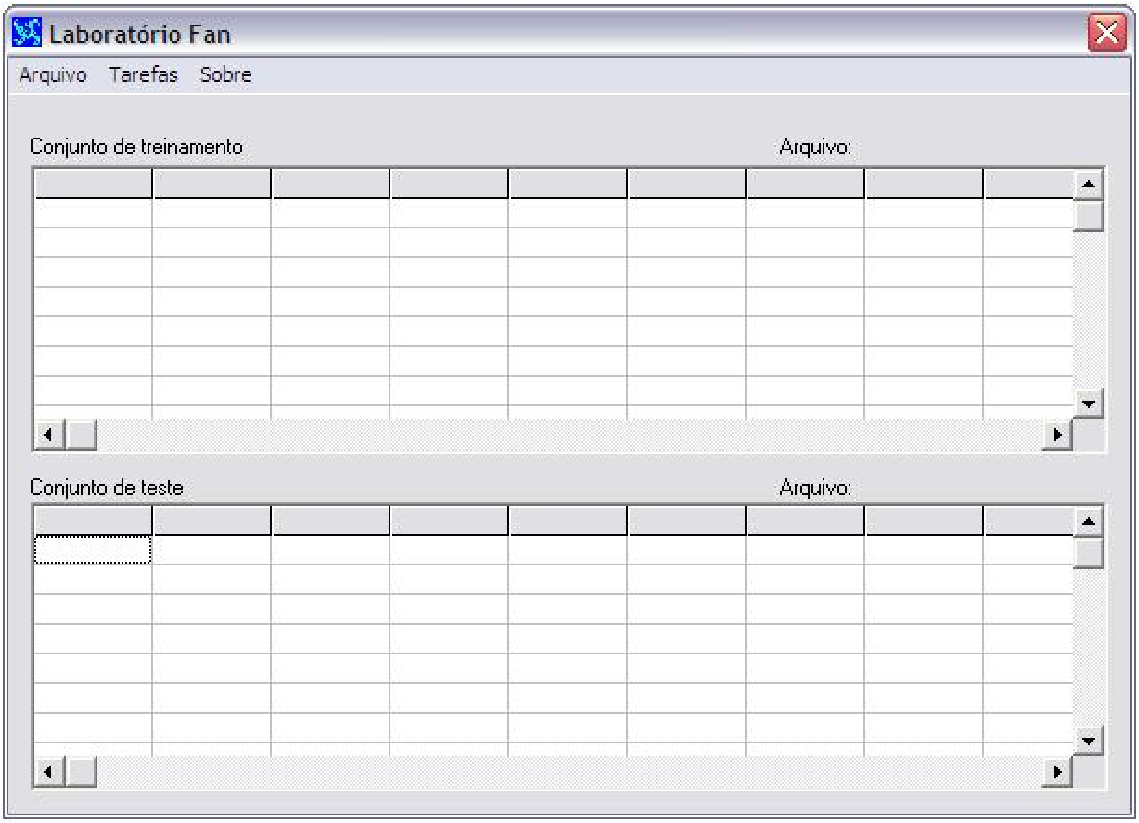

Figura 3 - Tela inicial do LabFAN

A partir da opção Arquivo, é possível carregar um conjunto de padrões dat para treinamento e um para teste.

A opção Sobre mostra um breve comentário sobre o software.

Depois de carregados os conjuntos, a opção Tarefas>Treinamento acessará a seguinte interface demonstrada na figura 4.

Figura 4 - Tela do LabFAN (dialogo de treinamento)

A figura 4 representa a tela de treinamento do LabFAN, nela pode-se observar a presença do menu de opções. Ao escolher o menu Arquivo, encontramos as opções: salvar uma rede e carregar uma rede já treinada.

Nessa tela são mostrados os erros e acertos totais no tempo transcorrido, assim como o número de épocas treinadas e testadas até o momento.

São mostradas também as taxas de melhor e atual máximo do mínimo e média harmônica.

No item Ações pode-se iniciar o treinamento, assim como no botão Iniciar/parar treinamento, pode-se também treinar com a tática de treinar só uma época, ou testar com a tática de testar só uma época, e na última sub-opção pode-se reiniciar uma rede já treinada.

Na opção Configurações, tem-se a possibilidade de se definir os parâmetros de treinamento, como indicar o parâmetro de parada do treinamento que pode ser por tempo, número de épocas ou valores de médias absoluta ou harmônica.

Existe a possibilidade de editar parâmetros específicos como o tamanho do Range, do Raio difuso e o peso de cada classe.

O item Característica, do menu de configurações, possibilita a escolha das características que se quer usar no treinamento.

Existe também a sub-opção de reiniciar os valores dos parâmetros para padrão, que são os valores default do software.

A opção Resultados possui a ferramenta Matriz de confusão que mostra a estatística de erros e acertos do treinamento, na qual é indicada a ocorrência de quantos padrões foram indicados como sendo da classe correspondente e quantos padrões foram indicados como sendo de outras classes não correspondentes.

Além desses resultados são mostrados os acertos e erros totais, e a taxa de erro (Figura 5).

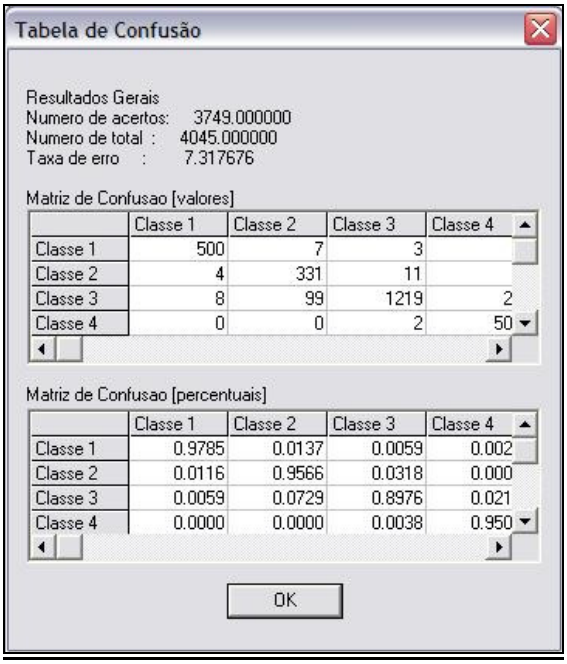

Figura 5 - Tela de resultados do LabFAN

3. SOLUÇÃO PROPOSTA

A solução proposta pela equipe consiste em dois componentes:

- A biblioteca JFAN
- O aplicativo EasyFAN

A biblioteca foi desenvolvida para que a rede pudesse ser utilizada por terceiros sem que tenha que se “reescrever” o FAN, ampliando o potencial de uso da rede treinada pelo ambiente EasyFAN.

O ambiente EasyFAN é uma aplicação multiplataforma que implementa vários recursos de importação de dados, acompanhamento do treinamento e de testes da rede. O EasyFAN tem o propósito de ser fácil, por esta razão foi implementado um modo “Wizard”. Neste modo, o usuário pode carregar um conjunto de padrões, treinar a rede FAN com estes padrões e realizar testes sem a necessidade de conhecimentos mais aprofundados sobre a área de Reconhecimento de Padrões.

3.1. O COMPONENTE JFAN

A rede neural FAN foi implementada para ser uma biblioteca a ser utilizada por terceiros. Esta biblioteca consiste de uma rica API (Application Program Interface) que abstrai completamente a lógica de implementação da técnica FAN.

Com o jFAN vários softwares podem usar a rede FAN como quiserem, sem se preocupar com as questões de inteligência artificial. Como o jFAN foi desenvolvido em java, ele ser como biblioteca somente para a linguagem java ou linguagem que consigam se utilizar de API's feitas em java.

Durante o período de desenvolvimento deste trabalho, o jFAN subsidiou o desenvolvimento de um segundo trabalho de conclusão de curso. A equipe composta pela Amanda Campos Costa, Marcelo Preiss e Jonath Rodrigues, orientados pelo Professor Dr. Mauro José Belli, estão desenvolvendo um sistema que utilizará a rede FAN para fazer projeções futuras nas cotações da bolsa de valores.

Esta utilização permitiu demonstrar o completo desacoplamento da biblioteca jFAN do aplicativo EasyFAN.

As duas principais classes que serão utilizadas por alguém que queira usar o jFAN são a MonitorFAN e GerenciadorPadroes. A classe GerenciadorPadroes concentra as funções para gerenciar os padrões para teste / treino / validação / classificação. A classe MonitorFAN concentra as funções e estratégias necessárias para a utilização da rede.

O próprio jFAN já possui vários recursos opcionais para alterar a estratégia de treinamento (randomização da posição dos padrões no conjunto, o uso de têmperas simuladas, normalização de neurônios, tipos de normalização de padrões e a porcentagem de padrões do conjunto de treinamento que será usada).

A versão 1.0 da API jFan oferece as seguintes funcionalidades:

- Carregador de arquivo com suporte a arquivos .xml, .xls (Microsoft Excel), .dat (LabFAN) e .txt.
- Persistor de dados para redes e padrões para os formatos .xml e .dat.
- Gerenciador de padrões.
- Mapeador de padrões textuais.
- Geradores de eventos com estatísticas.
- Gerador de exceções.

- Executor e gerenciador de treinamento, teste, validação e classificação.
- Normalizadores e calculadores de características.
- Recursos de uso de temperas.
- Indexadores de classes.
- Calculador de funções de pertinência.
- Gerador de matriz de confusão, e outras taxas de erro/acerto.

3.2. O COMPONENTE EASYFAN

O software EasyFAN é uma implementação para uso genérico da rede neuro-fuzzy FAN e se diferencia de seu predecessor (LabFAN) em muitas funções adicionais, como por exemplo, possui diversas estratégias de treinamento, visualizações gráficas de resultados de treinamento, visualização dinâmica da evolução de treinamentos e testes, mas sua principal diferença são as funções de Validação, que tem por função validar os resultados de teste, e a função de Classificação, que é o ponto chave do software, no qual a partir de um conjunto de padrões que se desconhece a classe a que pertencem e que se deseje saber, é realizada uma rotina de classificação de tais padrões, onde é indicando uma classe a cada padrão, considerando as característica que possuem maior grau de pertinência.

Durante o desenvolvimento do Projeto EasyFAN, o mesmo foi utilizado por Wladimir Bastos, estudante do curso de Sistemas da Informação, na Faculdade Hélio Rocha em Salvador, Bahia. Seu trabalho trata da utilização de algoritmos FAN em sistemas de reconhecimento automático de locutor em modo dependente de

texto. O objetivo é comprovar a possibilidade do uso da arquitetura FAN como reconhecedor de padrões para o reconhecimento de indivíduos. Ele está sendo orientado pelo professor Ernesto Massa.

O EasyFAN faz uso do componente JFAN.

3.3. O COMPONENTE THINGEASYFAN

Trata-se de um componente desenvolvido pela equipe, que agrega ao EasyFAN as potencialidades do componete Thinlet, uma ferramente de interfaces gráficas GUI. Com o Thinlet, que é o programa de mapeamento de arquivos com extensões xml, permite-se a visualização das interfaces elaboradas aproveitando-se os recursos do Java.

Thinlet é um componente (ferramenta) GUI, que analisa gramaticalmente a hierarquia e as propriedades do GUI, segura a interação do usuário, e interage com a lógica de programação. O Thinlet separa a apresentação gráfica (descrita em um arquivo XML) e os métodos da aplicação (escritos com o código Java). Seu tamanho comprimido é 39KB, e a licença é LGPL .

Para elaboração das interfaces gráficas do sistema, foi utilizado o Thing que é um editor do GUI de XML para Thinlet, que facilita o desenvolvimento dos componentes Thinlet e ajuda na interação com as classes Java.

Através da utilização das classes do ThinletCommons se é possível setar configurações gráficas, cores, fontes , mensagens de diálogos e utilizar os seus grandes componentes que são o FileChooser que interage com a máquina em uso,

possibilitando se encontrar, abrir e salvar arquivos e através do ExtensionFileFilter procurar documentos por extensões previamente estabelecidas.

Abaixo são citadas as classes do ThinletCommons e Thing utilizadas em nosso projeto:

ThinletDTD

AntiAliasedThinlet

ColorChooser

DirChooser

ExtensionFileFilter

FileChooser

FileFilter

FontChooser

LoggingThinlet

MessageDialog

ThinletDialog

ThinletTester

3.4. DETALHES DA IMPLEMENTAÇÃO DA API JFAN

Estrutura dos pacotes da API (figura 6):

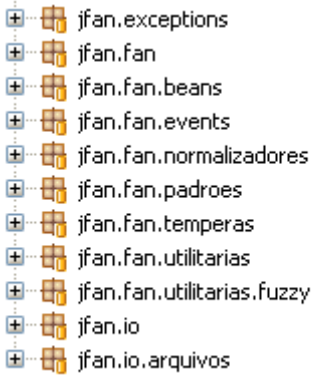

Figura 6 - Organização dos pacotes no Eclipse

3.4.1. O pacote jfan.exceptions

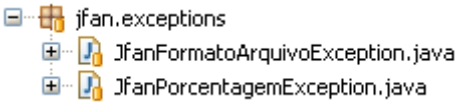

Figura 7 - Classes do pacote jfan.exceptions

Neste pacote são encontradas as classes que encapsulam as mensagens de erro da API jFan. As classes, pertencentes deste pacote, para tratamento de erro são:

1.1) JfanFormatoArquivoException: esta exceção é disparada quando há uma tentativa de carregamento de um formato de arquivo não reconhecido, com extensão diferente a xls,dat, txt e epd.

1.2) JfanPorcentagemExcpetion: esta exceção é disparada quando um valor não se encontra no intervalo de porcentagem definido entre 0 de 100, inclusive.

3.4.2. O pacote jfan.fan

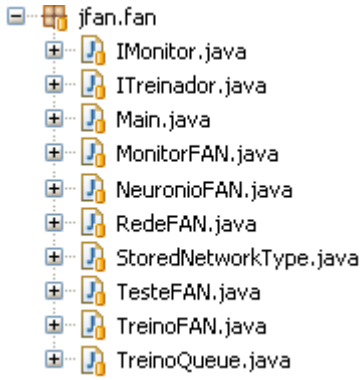

Figura 8 – Classes do pacote jfan.fan

2.1) IMonitor: interface que implementa métodos que se dizem respeito ao estado da rede, indicando se deve iniciar a execução da rede, se a rede deve ser parada ou se ela está em execução.

2.2) ITreinador: interface que implementa métodos que introduzem a função de especificar listas de neurônios e uma lista de padrões para treinamento.

2.3) Main: classes para mero uso de testes.

2.4) MonitorFAN: esta classe implementa a interface IMonitor, e tem por objetivo controlar a execução da rede. Seu método execute roda a rede em uma *thread*. Esse monitor gera diversos eventos com estatísticas, médias harmônicas, aritméticas, máximos e mínimos e matrizes de confusão. Tem por finalidade, dar suporte ao software EasyFAN. Todo software que use o jfan, usará extensivamente essa classe, pois nela estão concentradas as atividades de treinamento, teste, validação e classificação.

2.5) NeuronioFAN: está classe representa um neurônio em FAN, contém uma matriz neural com os pesos para cada característica e seus respectivos somatórios, possui propriedades de representação da força de um padrão, obtenção e especificação de pesos de penalização.

2.6) RedeFAN: a classe RedeFAN faz todo papel de gerenciar o treinamento, teste, validação e classificação. Essa classe mantém a rede atual do treinamento e também a melhor rede treinada, além de manter a taxa de acerto da melhor rede.

2.7) StoredNetworkType: classe enumerada que contempla os tipos de redes com melhores índices, e estão classificadas em: melhor harmônica, melhor média aritmética, melhor máximo do mínimo e rede atual.

2.8) TesteFAN: esta classe é responsável pelos testes da rede FAN, basicamente testa um padrão e emite o neurônio que a rede julga ser.

2.9) TreinoFAN: esta classe tem funções de treinamento da rede, aplica penalização e reforço a um conjunto de padrões.

2.10) TreinoQueue: extende a classe TreinoFAN e usa o princípio de filas para fazer o treinamento da rede.

3.4.3. O pacote jfan.fan.beans

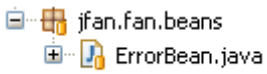

Figura 9 - Classes do pacote jfan.fan.beans

3.1) ErrorBean: esta classe encapsula uma matriz de confusão, e outras medidas de erro que ajudem na geração de estatísticas.

3.4.4. O pacote jfan.fan.events

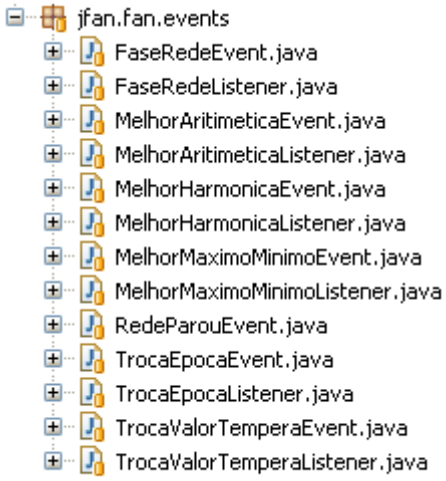

Figura 10 - Classes do pacote jfan.fan.events

4.1) FaseRedeEvent: classe de evento onde se especifica a fase em que a rede se encontra e se obtém sua fase.

4.2) FaseRedeListener: interface que fornece métodos que indicam qual fase a rede se encontra e se a rede parou.

4.3) MelhorAritimeticaEvent: classe de evento que encapsula métodos para especificação de uma média aritmética e uma matriz de confusão bem como sua obtenção.

4.4) MelhorAritimeticaListener: interface que fornece um método que indica que a média aritmética foi suprimida por uma melhor.

4.5) MelhorHarmonicaEvent: classe de evento que encapsula métodos para especificação de uma média harmônica e uma matriz de confusão bem como sua obtenção.

4.6) MelhorHarmonicaListener: interface que fornece um método que indica que a média harmônica foi suprimida por uma melhor.

4.7) MelhorMaximoMinimoEvent: classe de evento que encapsula métodos para especificação da média máxima do mínimo e uma matriz de confusão bem como sua obtenção.

4.8) MelhorMaximoMinimoListener: interface que fornece um método que indica que a média máxima do mínimo foi suprimida por uma melhor.

4.9) RedeParouEvent: classe de evento que encapsula métodos para especificação da época da rede bem como sua obtenção.

4.10) TrocaEpocaEvent: classe de evento que encapsula métodos para especificação da melhor média máxima do mínimo, melhor média aritmética, melhor média harmônica, uma matriz de confusão e a época bem como sua obtenção.

4.11) TrocaEpocaListener: interface que fornece um método que indica que a época mudou.

4.12) TrocaValorTemperaEvent: classe de evento que encapsula métodos para especificação de um valor para Tempera.

4.13) TrocaValorTemperaListener: interface que fornece um método que indica que o valor da Tempera mudou.

3.4.5. O pacote jfan.fan.normalizadores

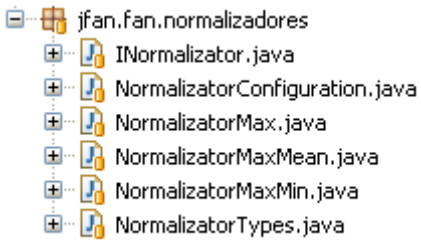

Figura 11 - Classes do pacote jfan.fan.normalizadores

5.1) INormalizator: a classe que implementar esta interface terá métodos para normalizar listas ou um padrão único, com opções de todas as características ou uma característica específica de um padrão. Também implementa a especificação

dos valores de máximos, mínimos e médias para as características que serão usadas para normalizá-las.

5.2) NormalizatorConfiguration: esta classe engloba todas as configurações de normalização dos mínimos, máximos e médias, possui métodos para obtenção e especificação das referidas normalizações.

5.3) NormalizatorMax: normaliza as características com base no máximo que foi configurado para a característica em questão, prove opções de normalização com uso de processamento paralelo.

5.4) NormalizatorMaxMean: normaliza as características com base na média que foi configurada para a característica em questão, prove opções de normalização com uso de processamento paralelo.

5.5) NormalizatorMaxMin: normaliza as características com base no mínimo que foi configurado para a característica em questão, prove opções de normalização com uso de processamento paralelo.

5.6) NormalizatorTypes: classe enumerada que encapsula os tipos possíveis de normalização.

3.4.6. O pacote jfan.fan.padrões

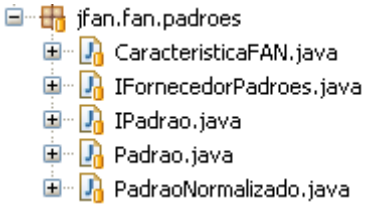

Figura 12 - As classes do pacote jfan.fan.padroes

6.1) CaracteristicaFAN: esta classe representa um conjunto difuso de pertinências, cada característica possui um intervalo que possui valores de pertinência definido por um inicio e um fim.

6.2) IFornecedorPadroes: esta interface prove uma padronização com métodos para obtenção de listas de padrões de treinamento, teste, validação, classificação bem como características relacionadas a lista como número de classes, características e padrões.

6.3) IPadrao: esta interface implementa métodos para obtenção das características relacionadas a um padrão.

6.4) Padrao: esta classe implementa a interface IPadrao e estende a classe PadraoNormalizado, representa um padrão do modelo FAN com suas características e classe associada.

6.5) PadraoNormalizado: encapsula objetos da classe CaracteristicaFAN, representa um padrão do modelo FAN com suas características e classe associada.

3.4.7. O pacote jfan.fan.temperas

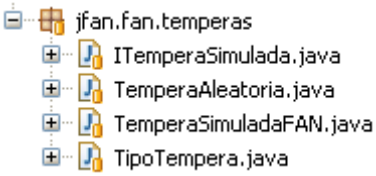

Figura 13 - Classes do pacote jfan.fan.temperas

7.1) ITemperaSimulada: esta interface implementa métodos para manipulação da tempera, a tempera aplica penalização diferenciada com o passar das épocas melhorando a performance no reconhecimento de padrões.

7.2) TemperaAleatoria: este principio foi utilizado em uma das implementações originais de FAN, esta classe implementa estas características, foi adaptada para ser mais genérica que o modelo original proposto. Os valores iniciais para mínimo e máximo são, respectivamente, 0.7943 e 1.0000.

7.3) TemperaSimuladaFAN: esta classe representa a tempera simulada (simulated annealing) para a rede FAN. A TemperaSimuladaFAN inicia com seus limites definidos de 0 a 1 (inclusive) com incremento (step) de 0.1.

7.4) TipoTempera: classe enumerada que contempla os dois tipos de tempera utilizada na API jFan respectivamente a aleatória e a simulada.

3.4.8. O pacote jfan.fan.utilitárias

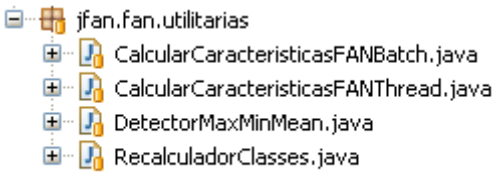

Figura 14 - As classes do pacote jfan.fan.utilitarias

8.1) CalcularCaracteristicasFANBatch: calcula todas as características FAN para um conjunto de padrões fornecidos, o calculo é feito de modo seqüencial.

8.2) CalcularCaracteristicasFANThread: utiliza a classe CalcularCaracteristicasFANBatch, mas neste caso o cálculo da característica FAN é realizado em paralelismo.

8.3) DetectorMaxMinMean: esta classe abstrata fornece métodos estáticos para procura de máximos, mínimos e médias ao fornecer uma lista com padrões.

8.4) RecalculadorClasses: esta classe tem por finalidade substituir as classes originais fornecidas pelo usuário substituindo seus valores pelos seus respectivos índices dentro de um array.

3.4.9. O pacote jfan.fan.utilitarias.fuzzy

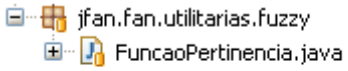

Figura 15 - As classes do pacote jfan.fan.utilitarias.fuzzy

9.1) FuncaoPertinencia: esta classe contém classes internas que fornecem métodos para cálculo de pertinência para as características, um valor tem pertinências variáveis associadas dentro do limite da função. A classe provê funções matemáticas do tipo triangular, gaussiana, trapezoidal, senoidal, simétrica fechada e sino. Por padrão a API jFan utiliza a função triangular.

3.4.10. O pacote jfan.io

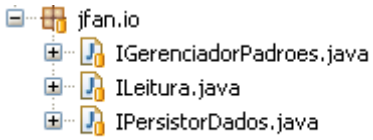

Figura 16 - As classes do pacote jfan.io

10.1) IGerenciadorPadroes: esta interface provê métodos para adição e obtenção de padrões sejam eles de treinamento, teste, validação ou classificação, além disso fornece métodos para trabalhos com mapeamentos de tipos textuais.

10.2) Ileitura: esta interface provê métodos que fornecem dados sobre os padrões nos arquivos persistidos em unidade física.

10.3) IPersistorDados: esta interface delega métodos que definem como as assinaturas de métodos para persistência de padrões e melhores redes treinadas devem ser feitas.

3.4.11. O pacote jfan.io.arquivos

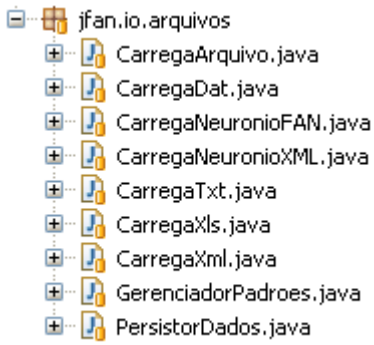

Figura 17 - As classes do pacote jfan.io.arquivos

11.1) CarregaArquivo: esta classe utiliza as classes CarregaDat e CarregaTxt para leitura de dados persistidos em unidade física que contenham padrões.

11.2) CarregaDat: esta classe faz a leitura de padrões persistidos em unidade física, sua sua principal função é a de fornecer compatibilidade com o formato de arquivos implementado anteriormente no LabFAN.

11.3) CarregaNeuronioFAN: esta classe faz a leitura de uma rede persistida em unidade física, sua principal função é a de retornar o estado de aprendizado da

rede FAN obtido anteriormente. Esta classe é usada para carregar redes salvas no LabFAN.

11.4) CarregaNeuronioXML: esta classe tem as mesmas características que a classe acima, porém utiliza XML descrevendo mais claramente os elementos de uma rede FAN. É por padrão o formato utilizado pelo EasyFAN (*.enn – EasyFan Neural Network).

11.5) CarregaTxt: esta classe faz a leitura de padrões persistidos em unidade física, provê funcionalidades de especificação de delimitadores de linhas, colunas e separadores de colunas, sua principal função é agregar facilidades ao uso do EasyFAN.

11.6) CarregaXls: esta classe faz a leitura de padrões persistidos em unidade física, provê a funcionalidade de carregar dados de uma planilha do Excel, permite delimitar colunas e linhas.

11.7) CarregaXml: esta classe é o padrão para leitura de arquivos no EasyFan, possui elementos que identificam todos os dados de um padrão na extenção *.epd (EasyFAN Pattern Data).

11.8) GerenciadorPadroes: esta classe agrega os padrões de treino, teste, classificação e validação, efetua o mapeamento de dados textuais e provê funcionalidades para obtenção e gerenciamento de dados sobre os referidos padrões.

11.9) PersistorDados: esta classe possui métodos para persistir dados de padrões e redes treinadas pelo EasyFAN. Os formatos possíveis para gravação são os formatos ENN padrão do EasyFAN e FAN formato padrão do LabFAN.

3.5. O TRATAMENTO DOS EVENTOS NO EASYFAN

O tratamento de eventos que vem da API jFAN é feito de forma assíncrona para garantir que a rede FAN rode livremente sem ter seu desempenho prejudicado pelo tratamento do evento que o EasyFAN irá fazer.

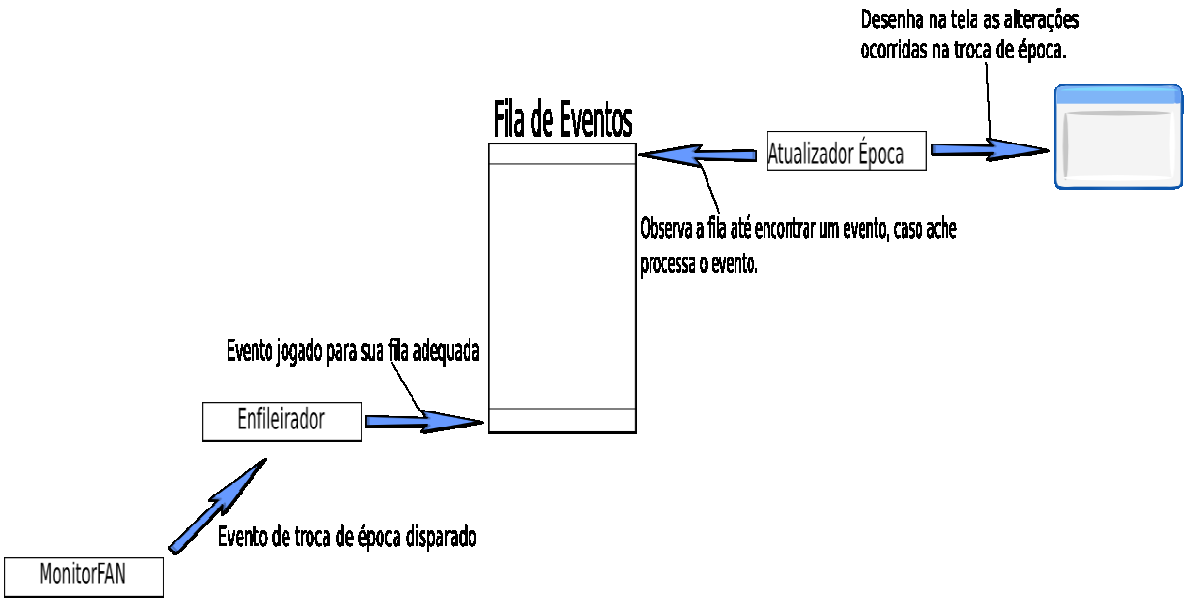

Figura 18 - Tratamento de eventos do EasyFAN

Cada evento gerado pelo jFAN é colocado em uma fila (first in – first out). Existe um observador que fica monitorando essa fila, e caso ache algo na fila realiza o evento. A figura 18 ilustra como é feito o tratamento assíncrono usando como exemplo o evento de troca de época que o monitorFAN da api jFAN dispara a cada troca de época.

Cada evento possui uma fila separada e um observador separado para cada fila, os observadores rodam como threads para fazer uso do processamento em paralelo, o que pode ser uma vantagem em computadores que possam rodar mais de uma thread simultaneamente, como os processadores de núcleo duplo.

O enfileirador serve como intermediário, pois ele vai definir em qual fila será alocado o evento e se deve respeitar algum tamanho máximo de fila.

4. Especificação Técnica JFan

4.1. CASOS DE USOS

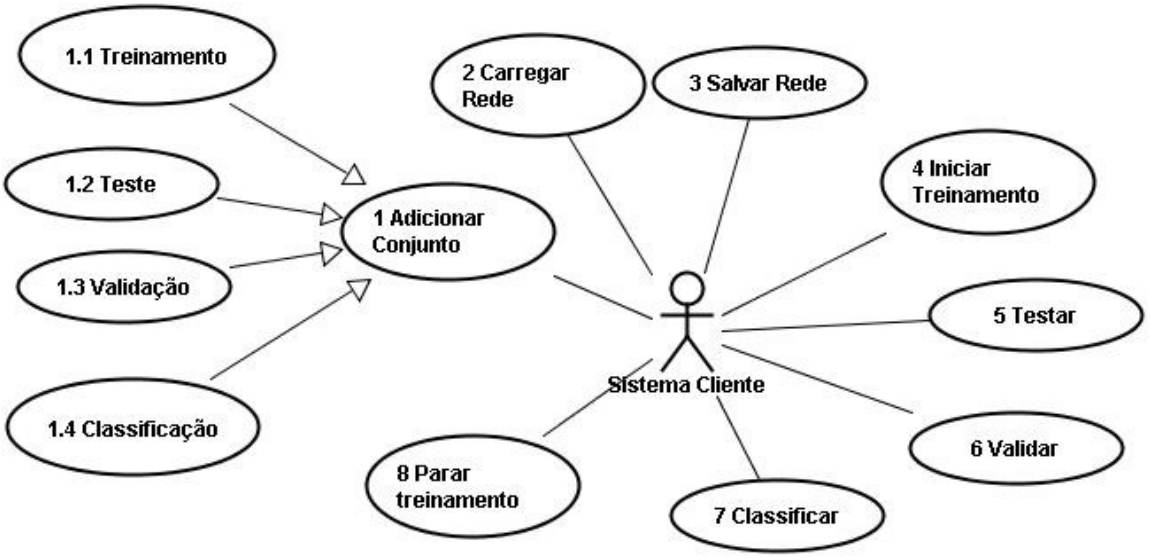

Figura 19 - Caso de uso JFAN

1 Adicionar Conjunto

Prioridade: 3 (Média)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Sistema Cliente

Propósito: Realizar o carregamento do conjunto de arquivos que podem ser carregados em quatro modos diferentes: Treinamento, Teste, Validação e Classificação.
Tipo: Opcional

Fluxo de Eventos:

1. Sistema Cliente solicita Carregamento de Conjunto
2. Sistema Cliente Carrega Conjunto

1.1 Adicionar Conjunto: Treinamento

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Sistema Cliente

Propósito: Carregar conjunto de arquivos na fase de Treinamento devendo o conjunto possuir uma das seguintes extensões: XLS, EPD, TXT, DAT.
Tipo: Opcional

Fluxo de Eventos:
1. Sistema Cliente solicita Carregamento do Conjunto de Treinamento
2. Sistema Cliente Adiciona Conjunto de Treinamento

Fluxo alternativo para o passo 1, caso o arquivo a ser adicionado não possua nenhuma das extensões estabelecidas:
1.1 Sistema exibe mensagem “Arquivo incompatível com o Sistema”
1.2 Encerra Caso de Uso

1.2 Adicionar Conjunto: Teste

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Sistema Cliente

Propósito: Carregar conjunto de arquivos na fase de Teste devendo o conjunto possuir uma das seguintes extensões: XLS, EPD, TXT, DAT.
Tipo: Opcional

Fluxo de Eventos:
1. Sistema Cliente solicita Carregamento do Conjunto de Teste
2. Sistema Cliente Adiciona Conjunto de Teste

Fluxo alternativo para o passo 1, caso o arquivo a ser adicionado não possua nenhuma das extensões estabelecidas:
1.1 Sistema exibe mensagem “Arquivo incompatível com o Sistema”
1.2 Encerra Caso de Uso

1.3 Adicionar Conjunto: Validação

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Sistema Cliente

Propósito: Carregar conjunto de arquivos na fase de Validação devendo o conjunto possuir uma das seguintes extensões: XLS, EPD, TXT, DAT.
Tipo: Opcional

Fluxo de Eventos:
1. Sistema Cliente solicita Carregamento do Conjunto de Validação
2. Sistema Cliente Adiciona Conjunto de Validação

Fluxo alternativo para o passo 1, caso o arquivo a ser adicionado não possua nenhuma das extensões estabelecidas:

1.1 Sistema exibe mensagem “Arquivo incompatível com o Sistema”
1.2 Encerra Caso de Uso

1.4 Adicionar Conjunto: Classificação

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Sistema Cliente

Propósito: Carregar conjunto de arquivos na fase de Classificação devendo o conjunto possuir uma das seguintes extensões: XLS, EPD, TXT, DAT.
Tipo: Opcional

Fluxo de Eventos:
3. Sistema Cliente solicita Carregamento do Conjunto de Classificação
4. Sistema Cliente Adiciona Conjunto de Classificação

Fluxo alternativo para o passo 1, caso o arquivo a ser adicionado não possua nenhuma das extensões estabelecidas:
1.1 Sistema exibe mensagem “Arquivo incompatível com o Sistema”
1.2 Encerra Caso de Uso

2 Carregar Rede

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Sistema Cliente

Propósito: Sistema Cliente realiza o carregamento do arquivo de rede que deve possuir uma das seguintes extensões: ENN, FAN.

Fluxo de Eventos:
1. Sistema Cliente solicita Carregamento do Arquivo de Rede (Neurônio)
2. Sistema Cliente Carrega Arquivo de Rede

Fluxo alternativo para o passo 1, caso o arquivo a ser adicionado não possua nenhuma das extensões estabelecidas:
1.1 Sistema exibe mensagem “Arquivo incompatível com o Sistema”
1.2 Encerra Caso de Uso

3 Salvar Rede

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Sistema Cliente

Propósito: Após treinar uma rede, o sistema deve escolher qual tipo de rede que deseja arquivar (Atual, Melhor Máximo do Mínimo, Melhor Média harmônica e Melhor Média Aritmética) e qual extensão que a rede vai ter (*.ENN, *.FAN).

Tipo: Primário

Fluxo de Eventos:
1. Sistema Cliente seleciona opção “Salvar Rede”
2. Sistema solicita local de armazenamento, nome e extensão da rede a salvar
3. Sistema Cliente escolhe local de armazenamento, nome e extensão da rede
4. Sistema salva rede

4 Iniciar Treinamento

Prioridade: 5 (Alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Sistema Cliente

Tipo: Primário

Fluxo de Eventos:
1. Sistema Cliente seleciona opção “Iniciar Treinamento”
2. Sistema inicia treinamento

Fluxo alternativo para o passo 1, caso não exista arquivo de treinamento carregado
1.1. Sistema desabilita opção de Treinamento, até que se tenha um arquivo de treino carreado

Fluxo alternativo para o passo 1, caso não exista arquivo de teste carregado
1.1 Sistema desabilita opção Treinamento, até que se tenha um arquivo de teste carregado

5 Testar

Prioridade: 5 (Alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Sistema Cliente

Propósito: Sistema escolhe qual tipo de rede que deseja testar (Carregada, Atual, Melhor Média Harmônica, Melhor Máximo do Mínimo e Melhor Média Aritmética) .

Tipo: Primário

Fluxo de Eventos:
1. Sistema Cliente seleciona opção “Testar rede”
2. Sistema inicia teste

Fluxo alternativo para o passo 1, caso não tenha nenhuma rede treinada ou carregada
1.1. Sistema Exibe Mensagem “Não há nenhuma rede de treinamento carregada, para se testar uma rede é necessário que se carregue uma rede ou se treine uma nova rede”.
1.2. Encerra Caso de Uso

Fluxo alternativo para o passo 1, caso não exista arquivo de teste carregado
1.1. Sistema desabilita opção Teste de Rede

6 Validar

Prioridade: 5 (Alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Sistema Cliente

Propósito: Sistema Cliente escolhe qual tipo de rede que deseja validar (Carregada, Atual, Melhor Média Harmônica, Melhor Máximo do Mínimo e Melhor Média Aritmética) .

Tipo: Primário

Fluxo de Eventos:
1 Sistema CLiente seleciona opção “Validar rede”
2 Sistema escolhe o tipo de rede a ser validada.
3 Sistema inicia Validação.

Fluxo alternativo para o passo 1, caso não tenha nenhuma rede treinada ou carregada
1.1 Sistema exibe Mensagem “Não há nenhuma rede de treinamento carregada, para se validar uma rede é necessário que se carregue uma rede ou se treine uma nova rede”.
1.2 Encerra Caso de Uso

Fluxo alternativo para o passo 1, caso não exista arquivo de validação carregado
1.1. Sistema desabilita opção de Validação de Rede

7 Classificar

Prioridade: 5 (Alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Sistema Cliente

Propósito: Sistema realiza classificação do conjunto de arquivos de dados de classificação carregado

Tipo: Primário

Fluxo de Eventos:
1 Sistema Cliente seleciona opção “Classificar Conjunto”
2 Sistema inicia classificação

Fluxo alternativo para o passo 1, caso não se tenha nenhuma rede de classificação carregada
1.1. Sistema exibe Mensagem “Não há nenhuma rede de classificação carregada, para se classificar um conjunto é necessário que se carregue uma rede uma rede de classificação”.
1.2 Encerra Caso de Uso

Fluxo alternativo para o passo 1, caso não exista arquivo de dados classificação carregado
1.1. Sistema desabilita opção de Classificação de Rede

8 Parar Treinamento

Prioridade: 4 (Média - Alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Sistema Cliente

Propósito: Sistema realiza interrupção do treinamento

Tipo: Primário

Fluxo de Eventos:
1 Sistema Cliente seleciona opção “Parar Treinamento”
2 Sistema Para o Treinamento

4.2. DIAGRAMA DE CLASSES

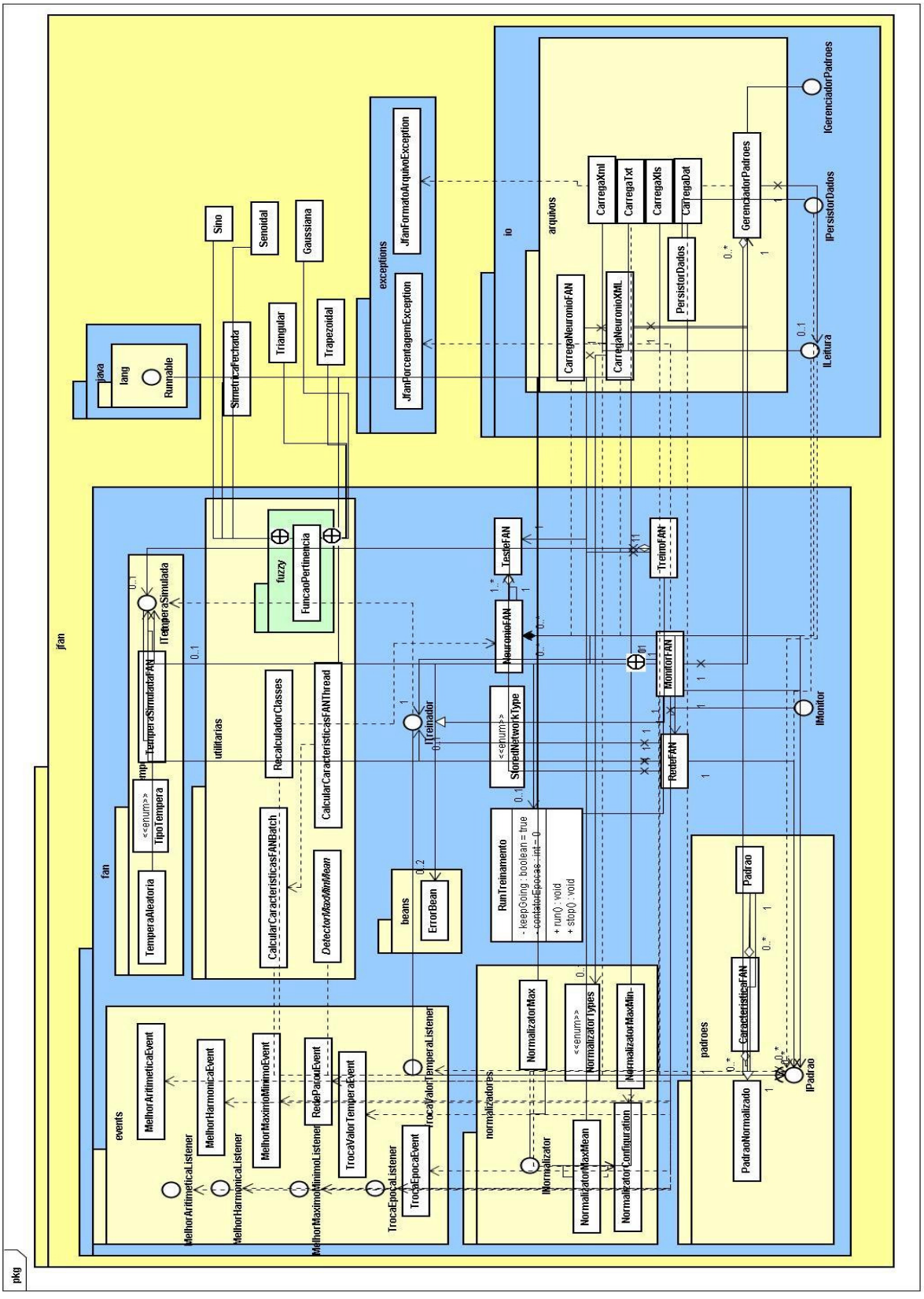

Figura 20 – Diagrama de Classes JFAN

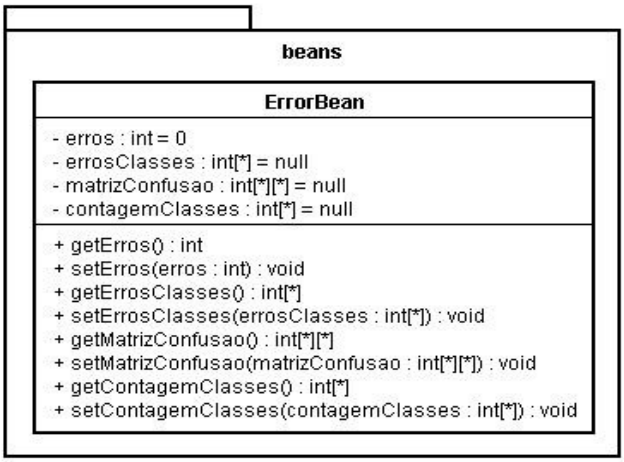

Figura 21 – Diagrama de Classes JFAN – Pacote beans

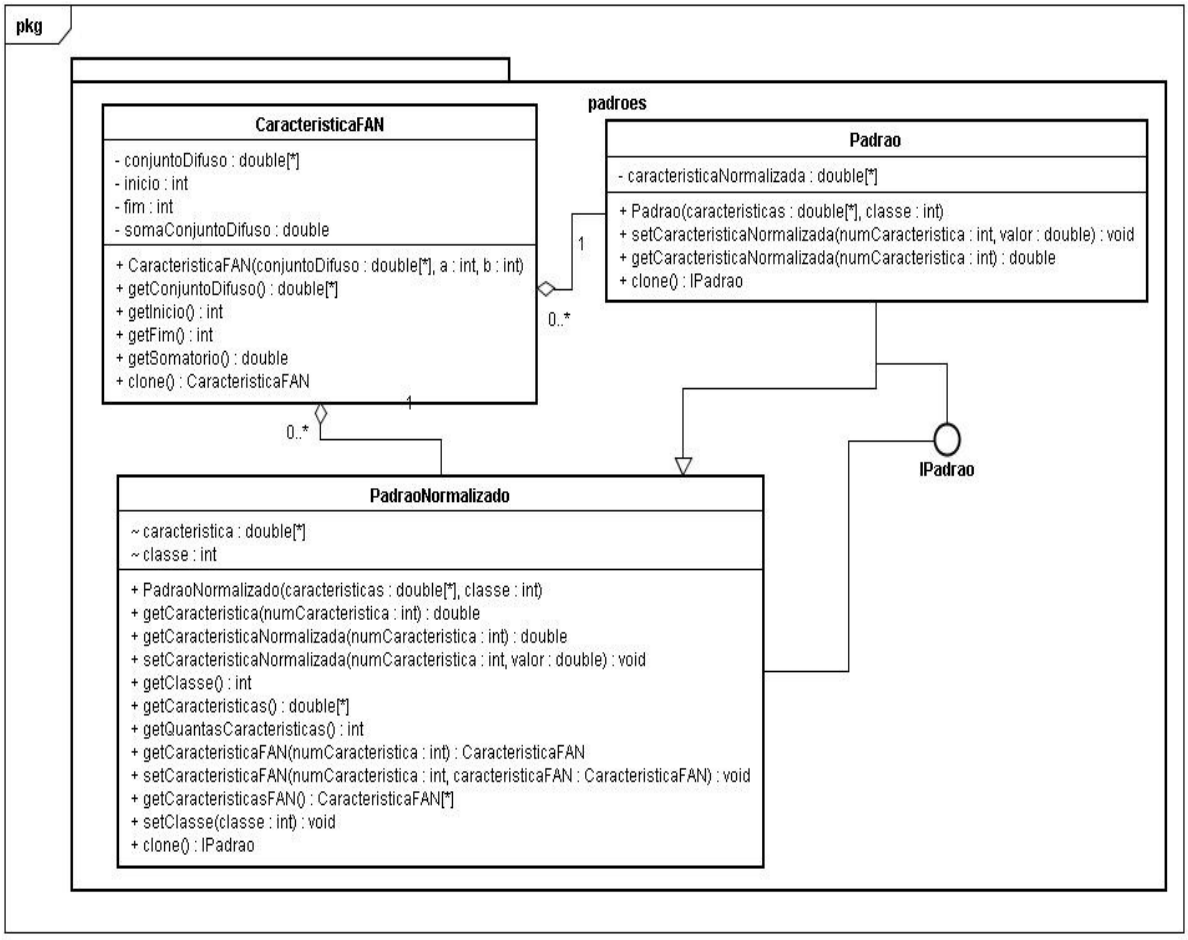

Figura 22 – Diagrama de Classes JFAN – Pacote padrões

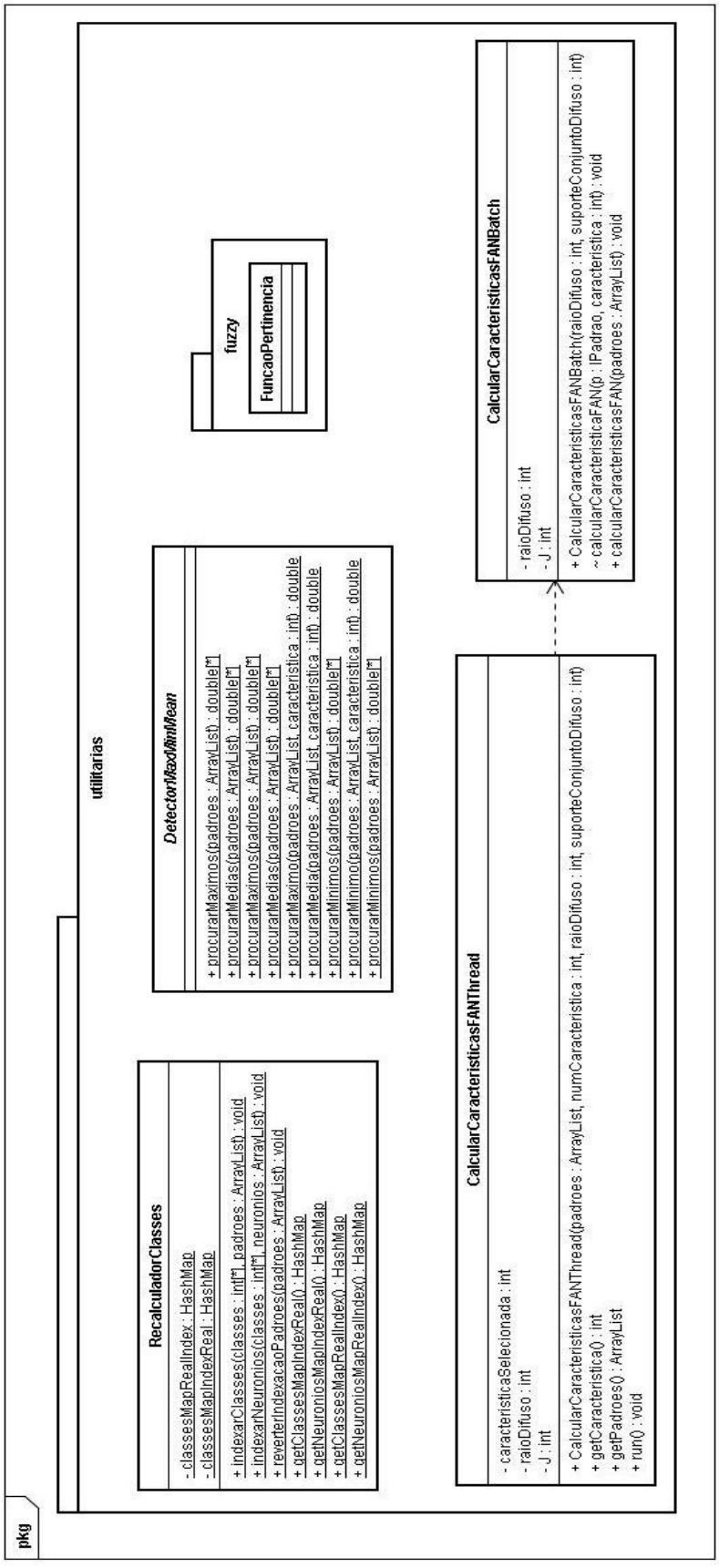

Figura 23 – Diagrama de Classes JFAN – Pacote utiliárias

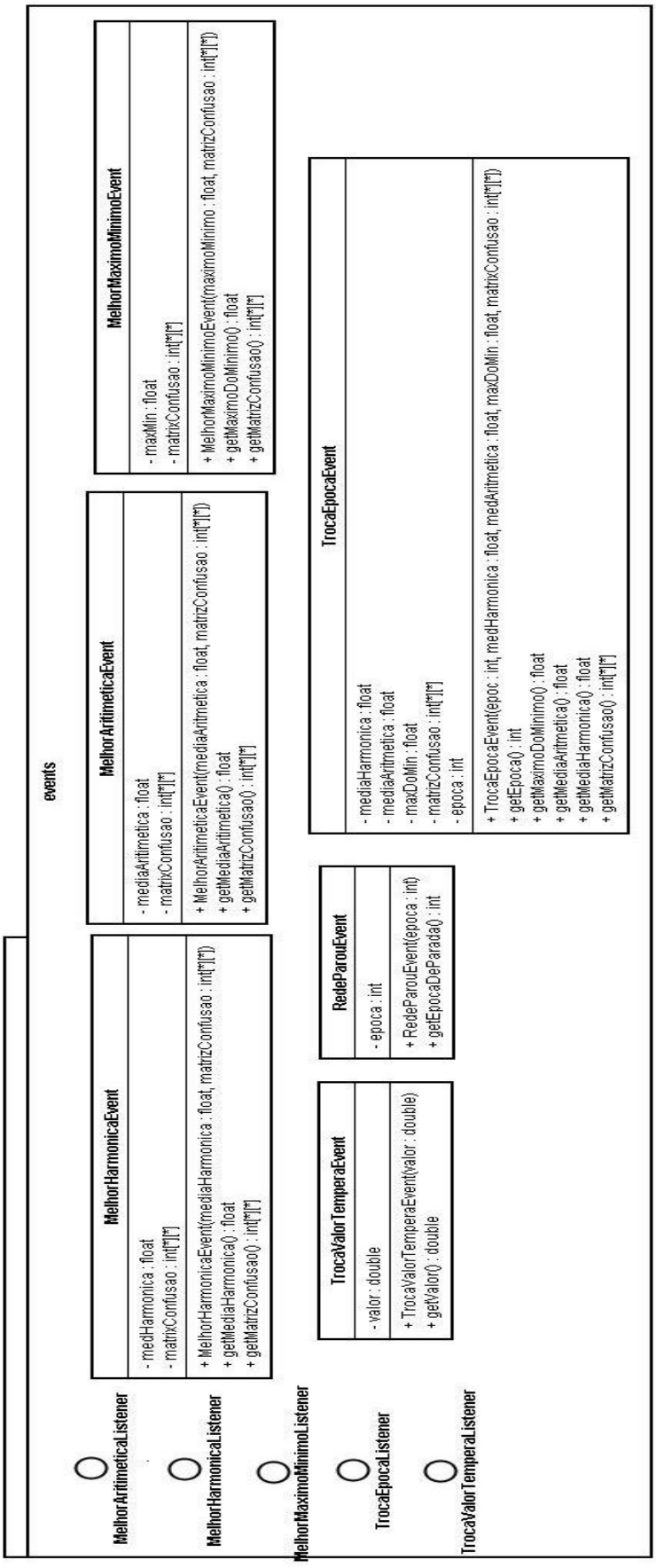

Figura 24 – Diagrama de Classes JFAN – Pacote events

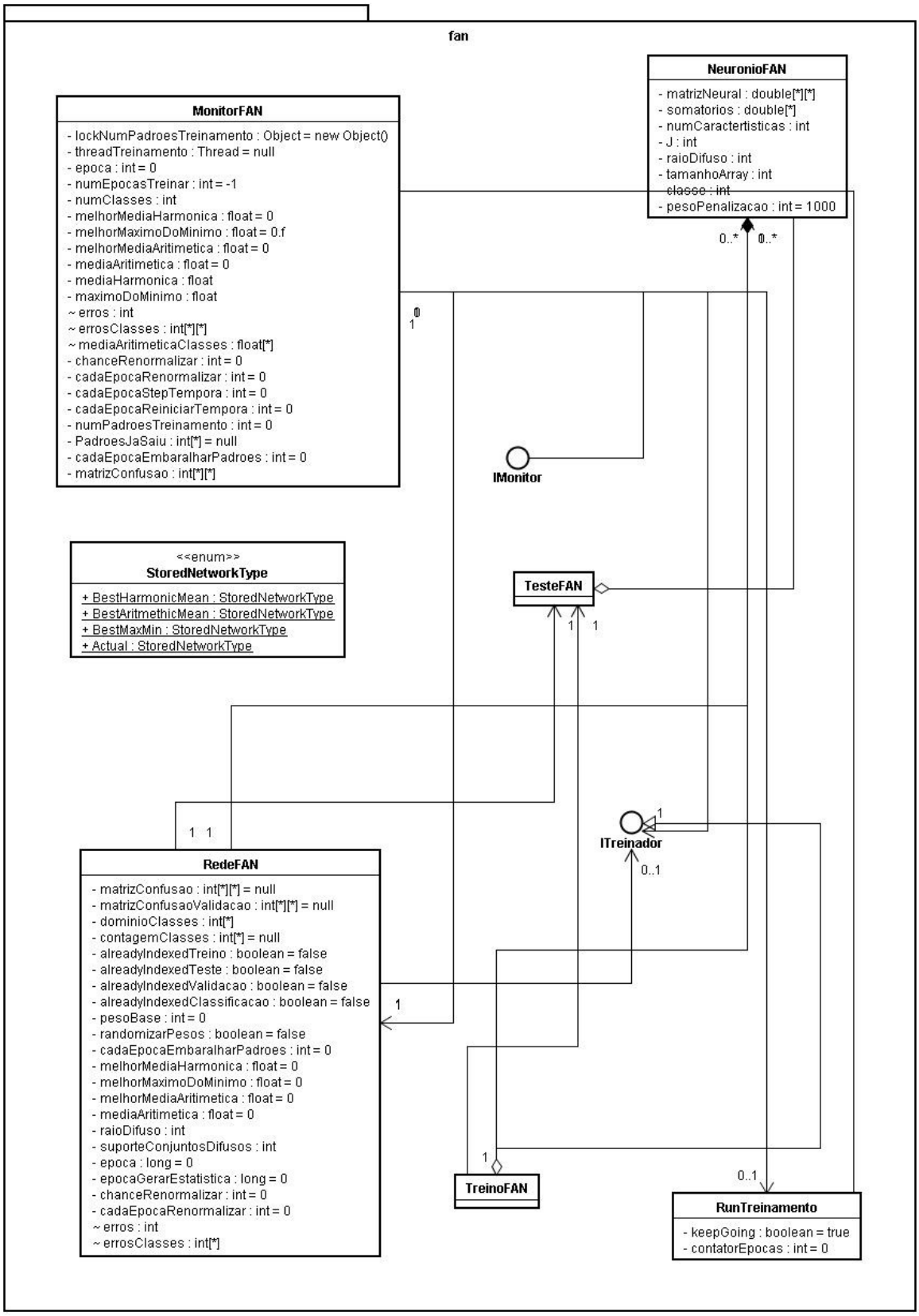

Figura 25 – Diagrama de Classes JFAN – Pacote fan 1

Figura 26 – Diagrama de Classes JFAN – Pacote fan 2

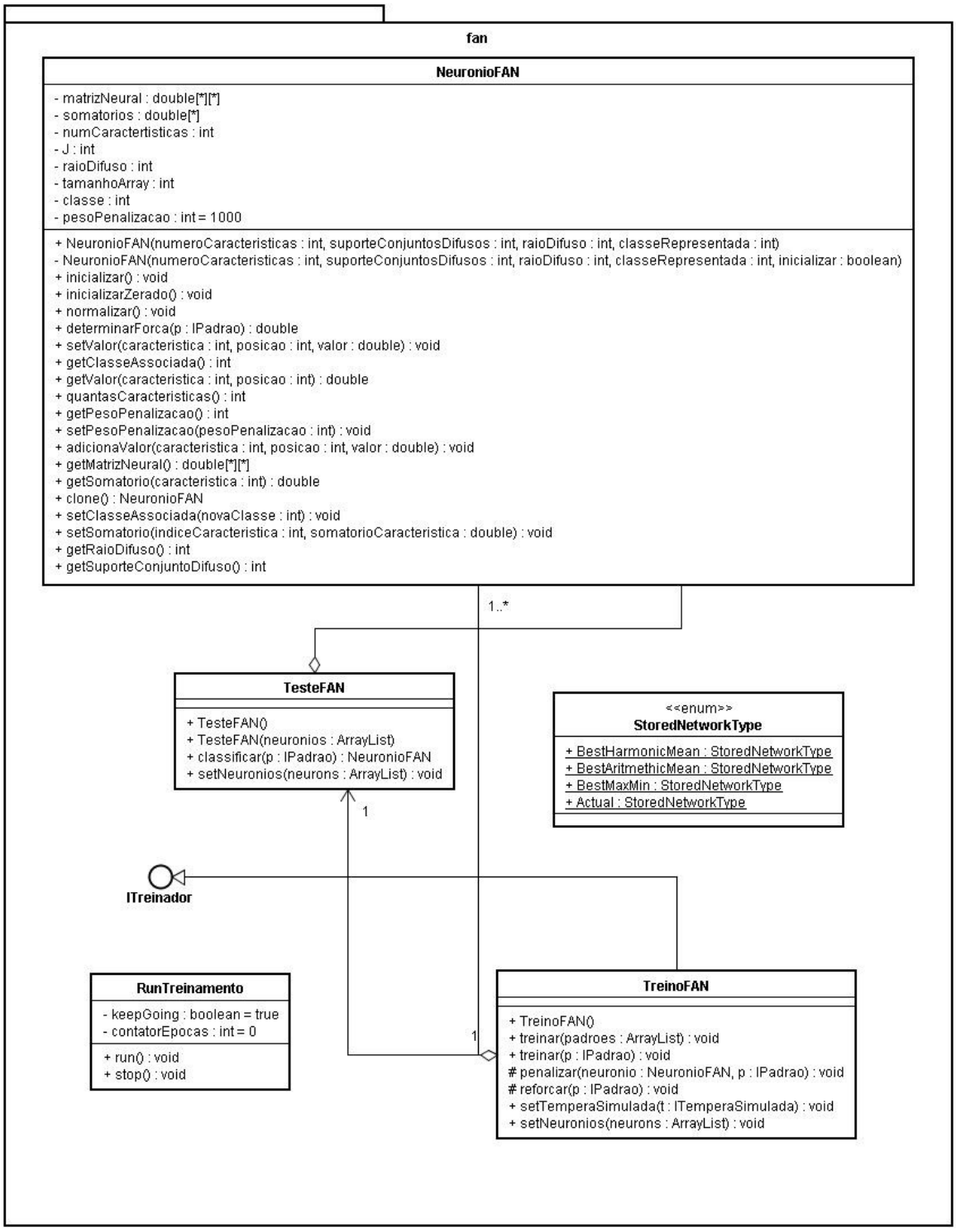

Figura 27 – Diagrama de Classes JFAN – Pacote fan 3

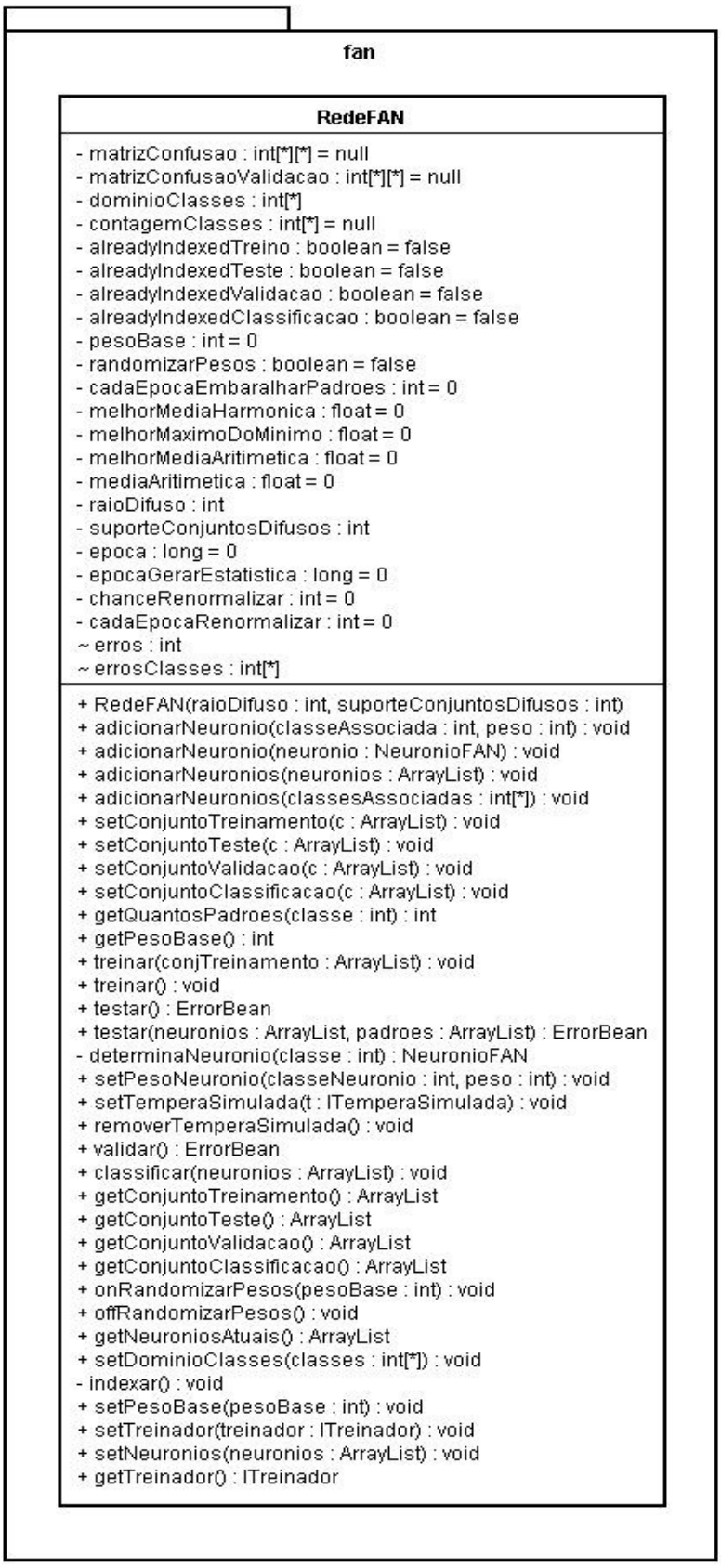

Figura 28 – Diagrama de Classes JFAN – Pacote fan 4

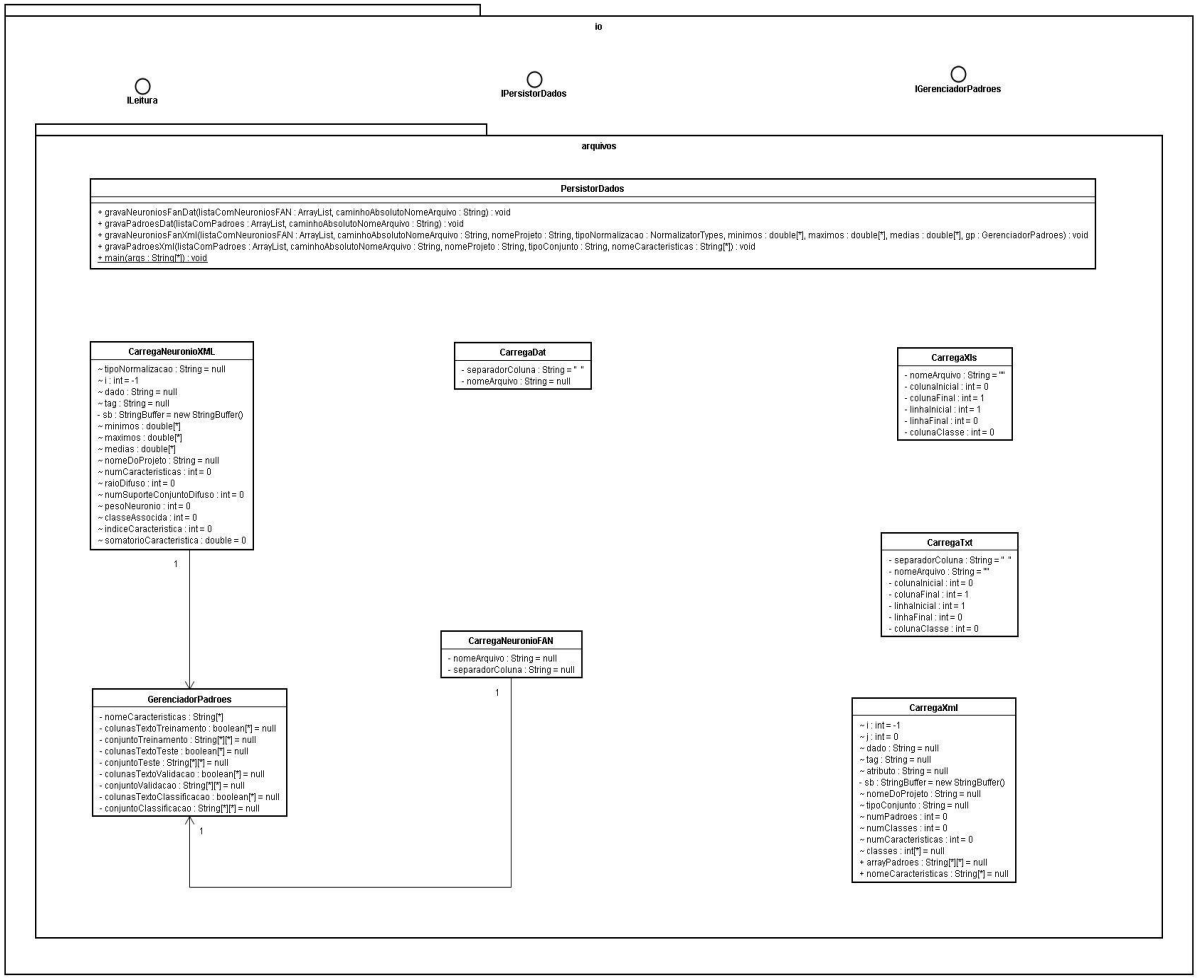

Figura 29 – Diagrama de Classes JFAN – Pacote IO 1

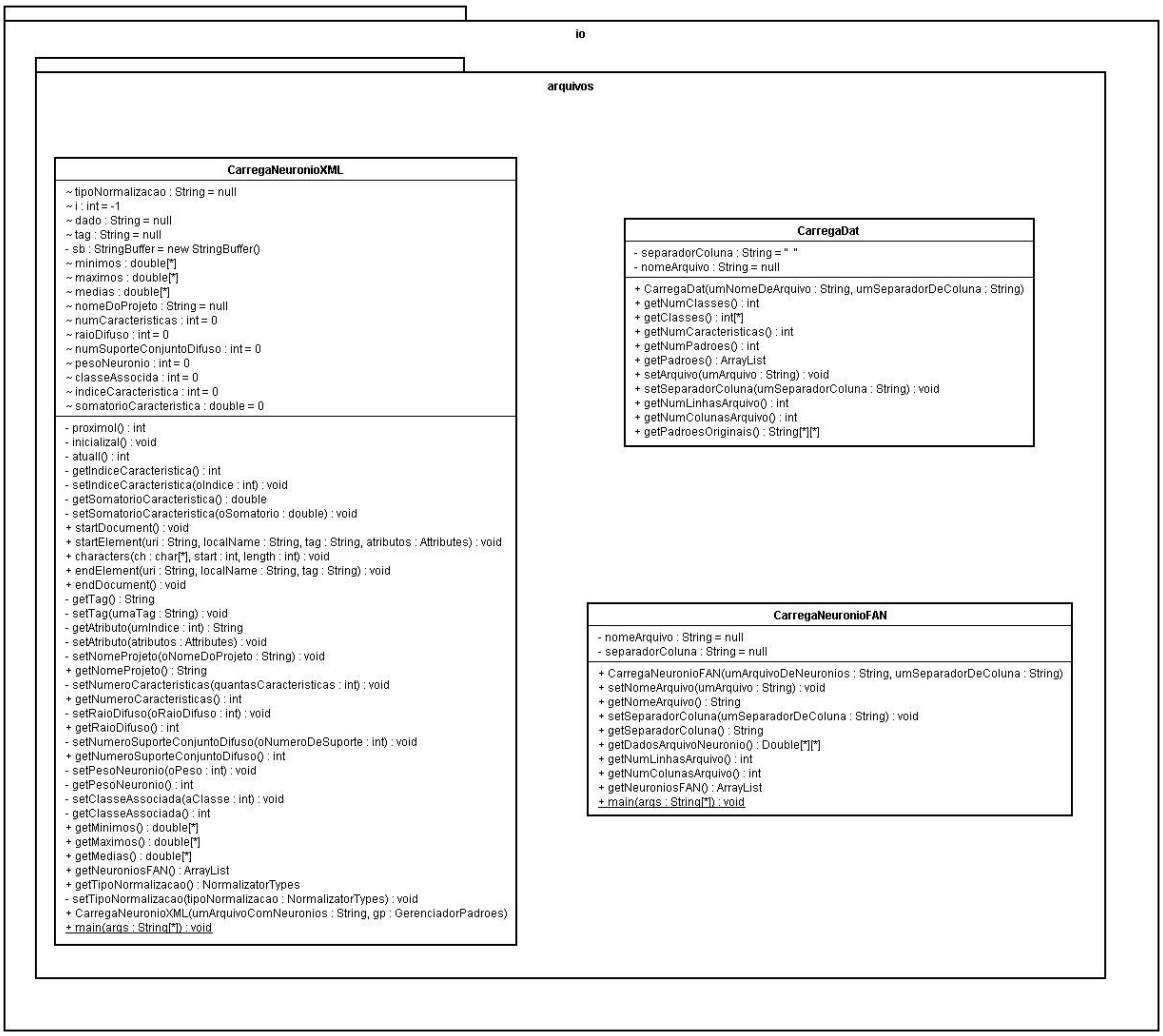

Figura 30 – Diagrama de Classes JFAN – Pacote IO 2

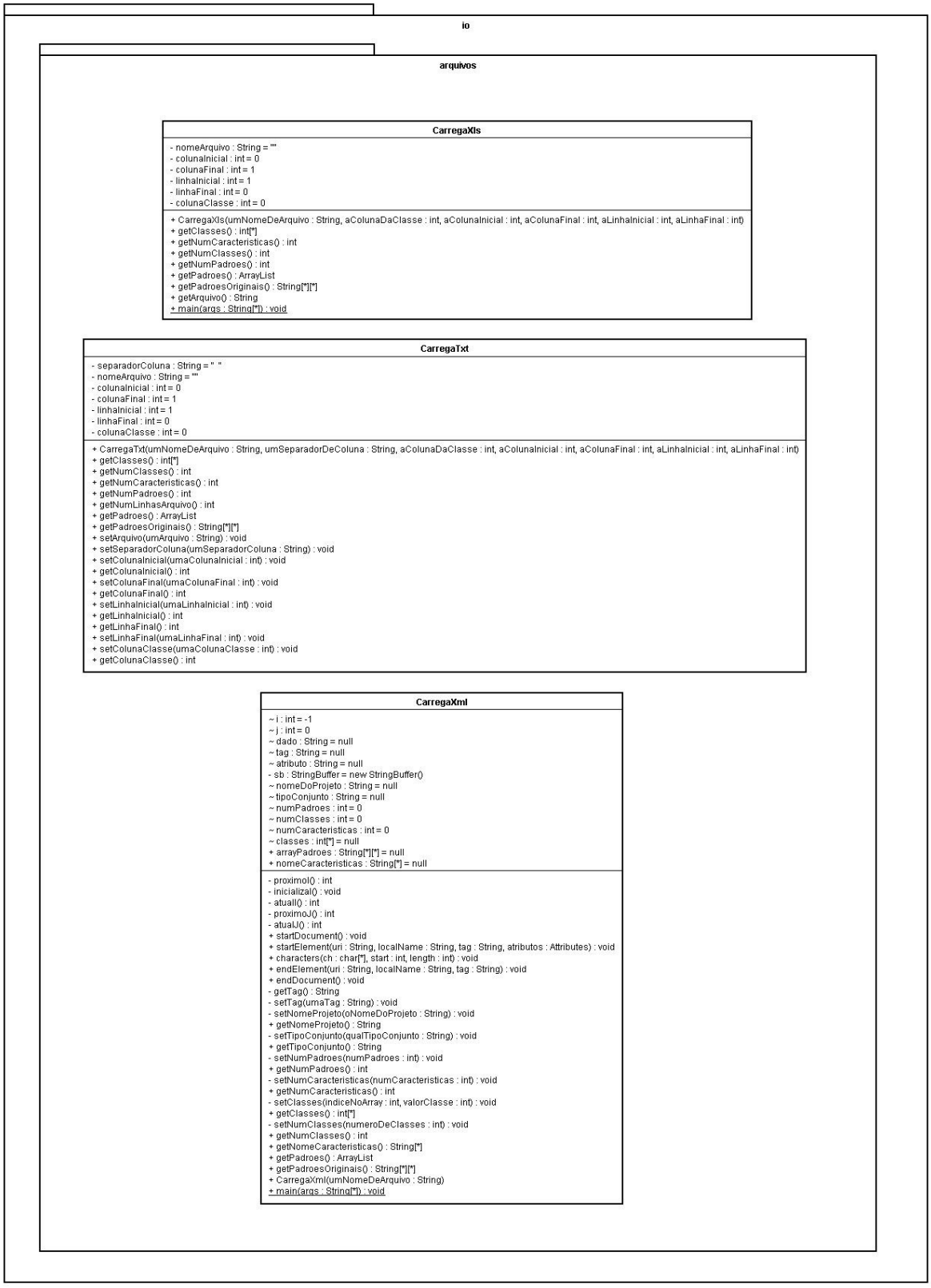

Figura 31 – Diagrama de Classes JFAN – Pacote IO 1

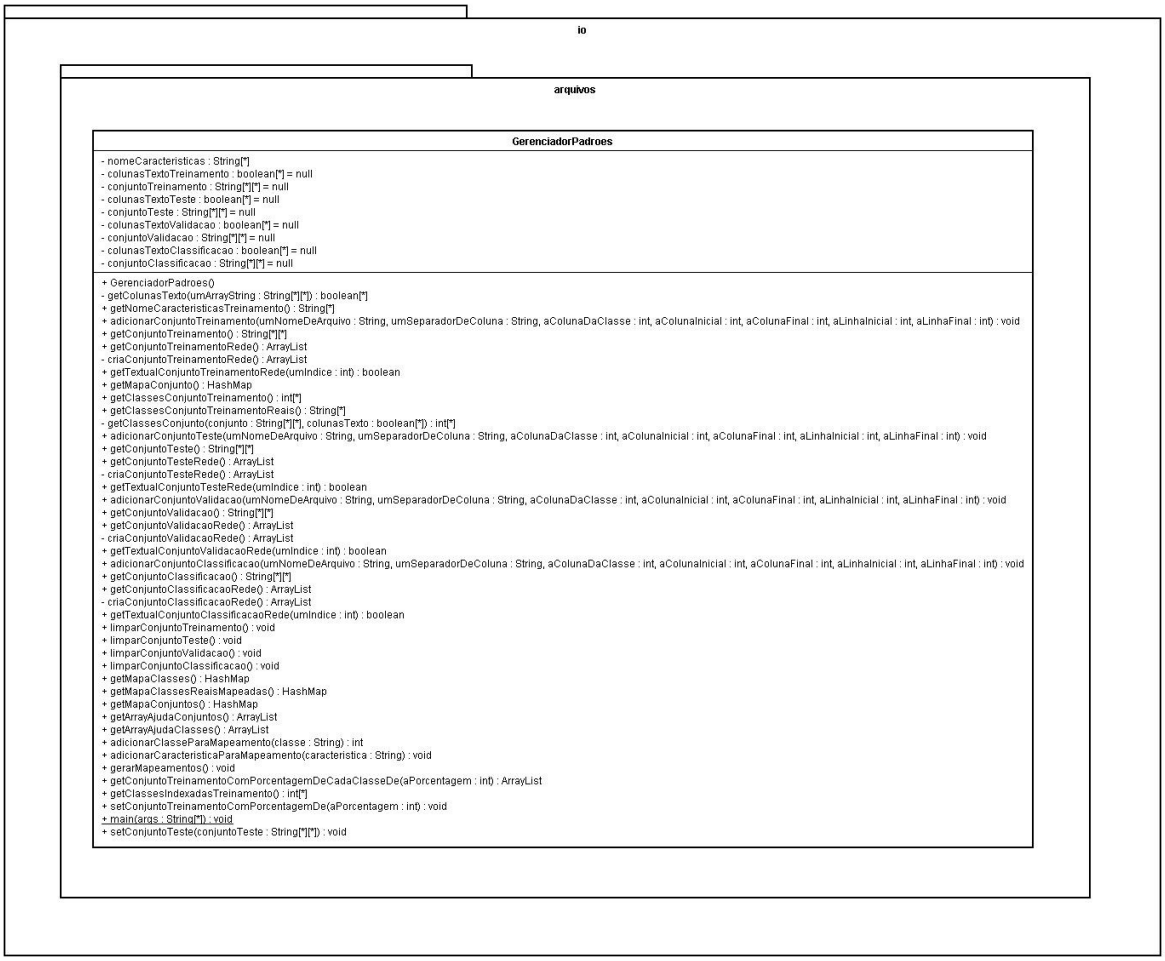

Figura 32 – Diagrama de Classes JFAN – Pacote IO 4

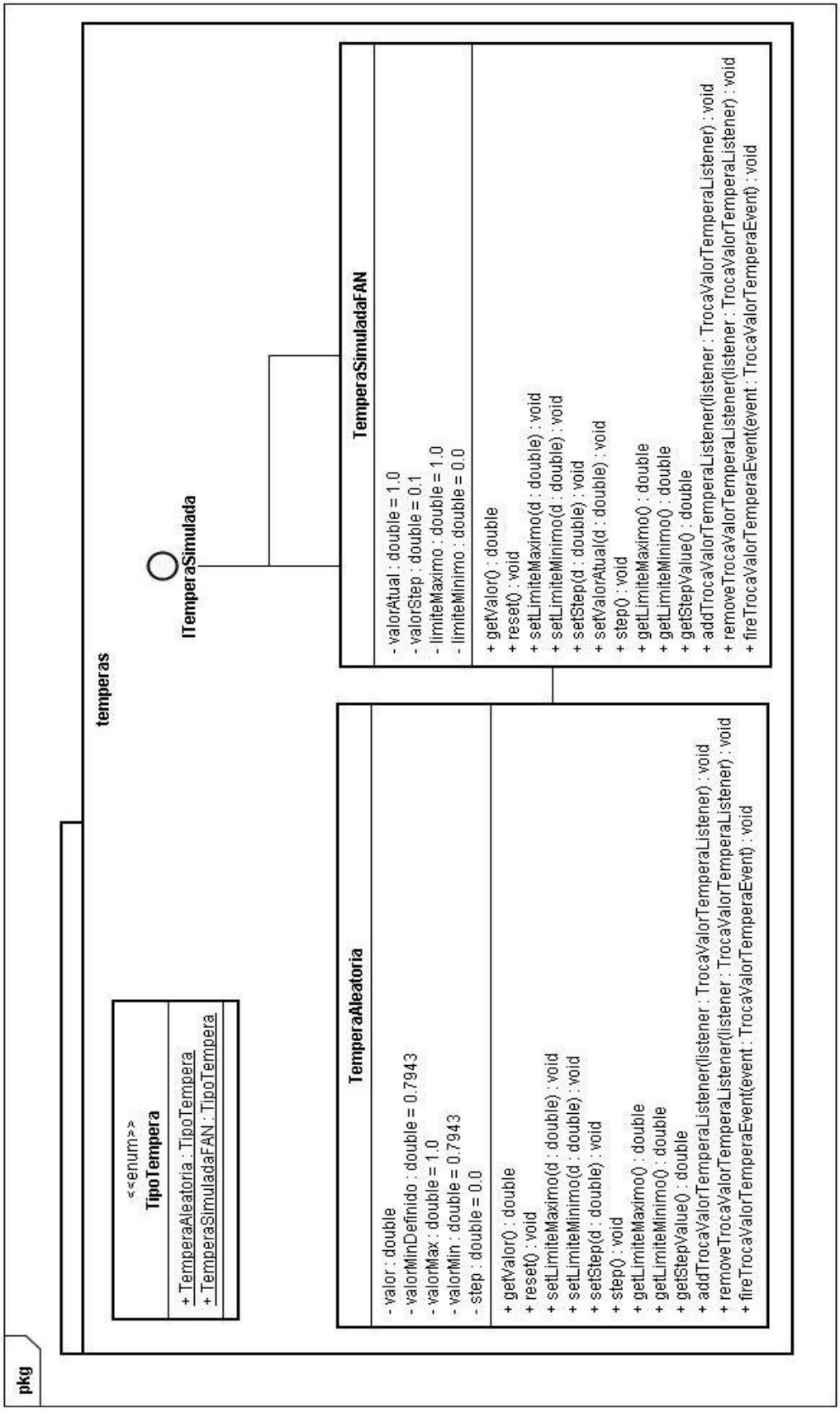

Figura 33 – Diagrama de Classes JFAN – Pacote temperas

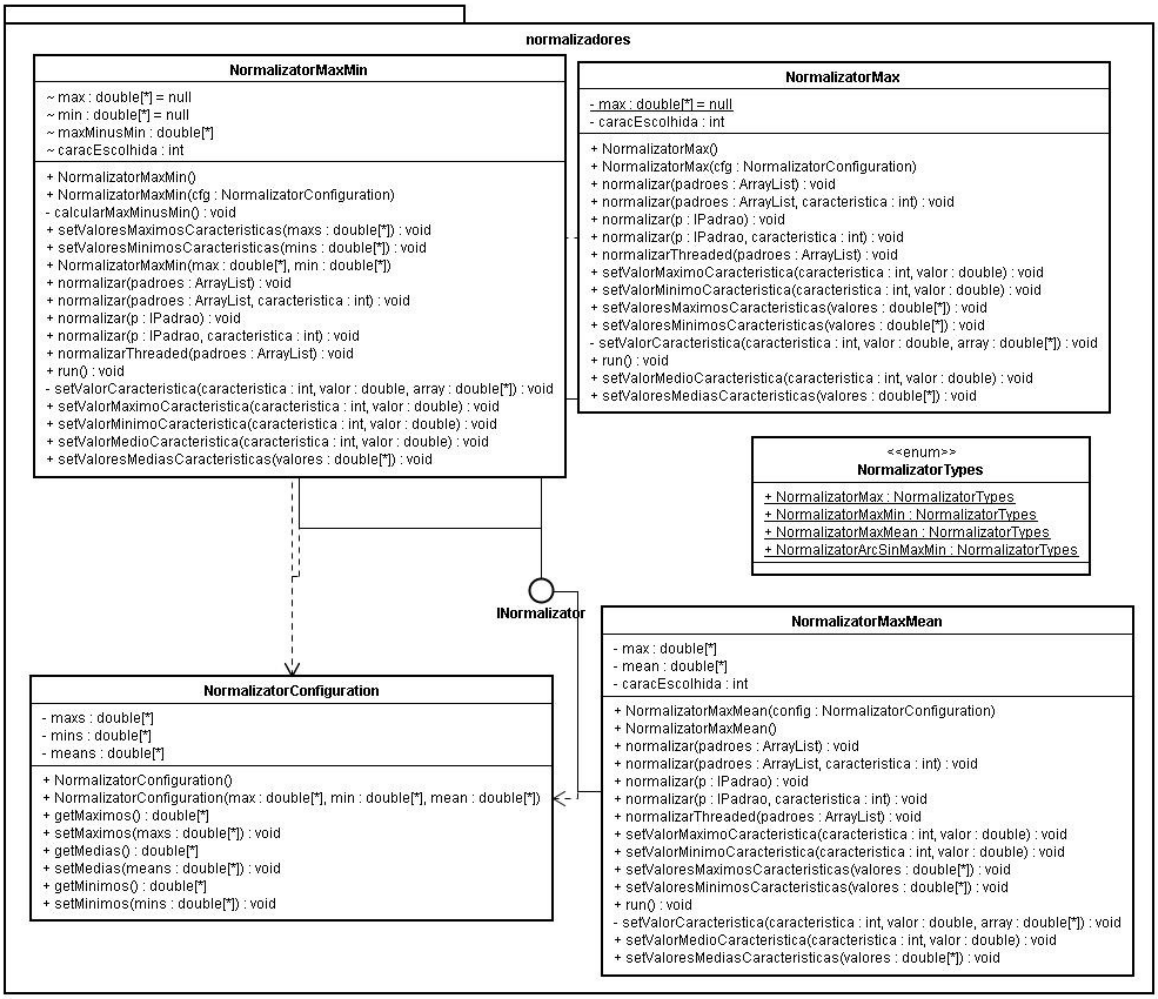

Figura 34 – Diagrama de Classes JFAN – Pacote normalizadores

4.3. DIAGRAMA DE SEQÜÊNCIA

Adicionar Conjunto de Treinamento

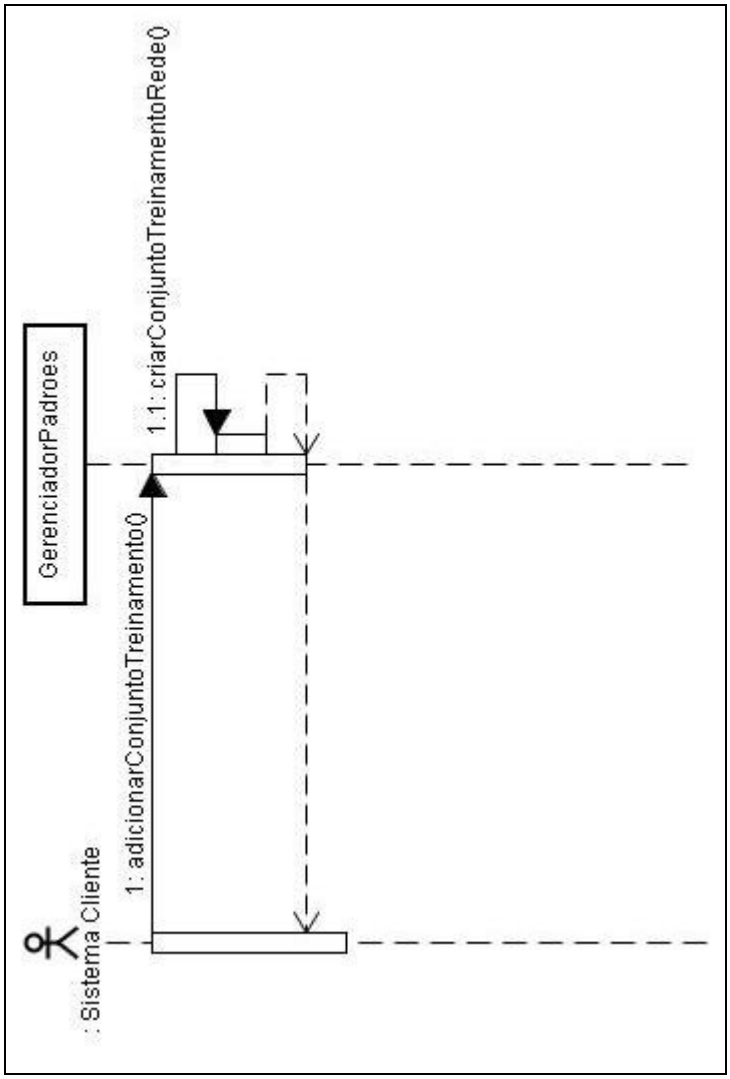

Figura 35 – Diagrama de Seqüência JFAN – Adicionar Conjunto de Treinamento

Adicionar Conjunto de Teste

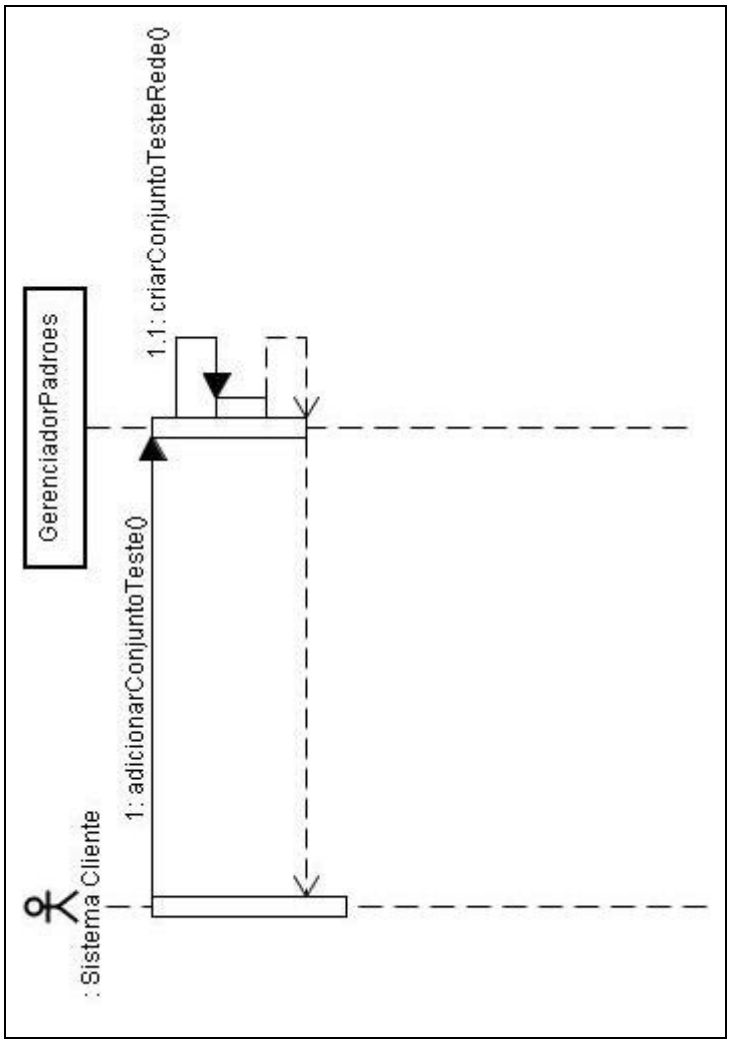

Figura 36 – Diagrama de Seqüência JFAN – Adicionar Conjunto de Teste

Adicionar Conjunto de Validação

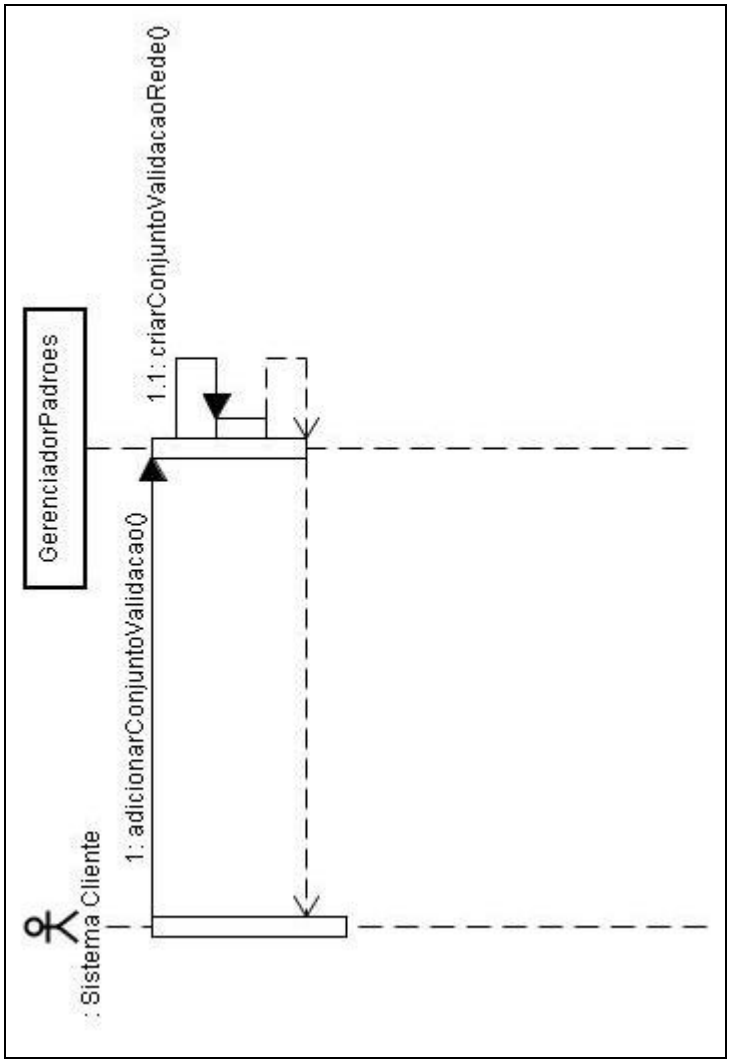

Figura 37 – Diagrama de Seqüência JFAN – Adicionar Conjunto de Validação

Adicionar Conjunto de Classificação

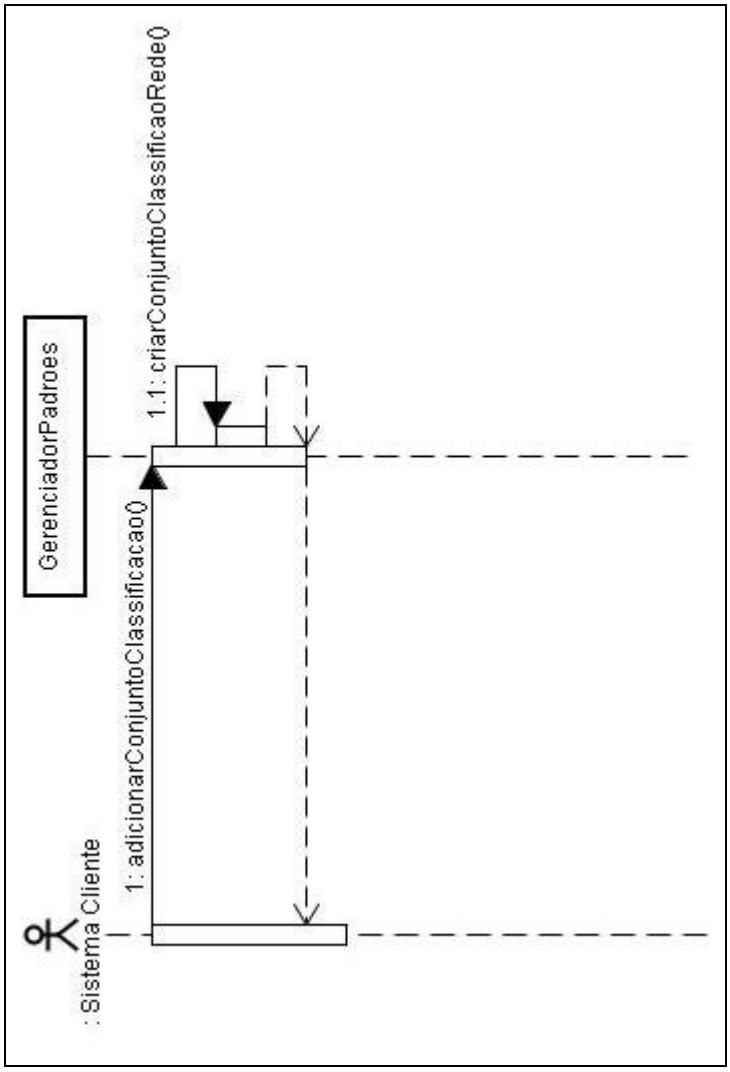

Figura 38 – Diagrama de Seqüência JFAN – Adicionar Conjunto de Classificação

Carregar Rede

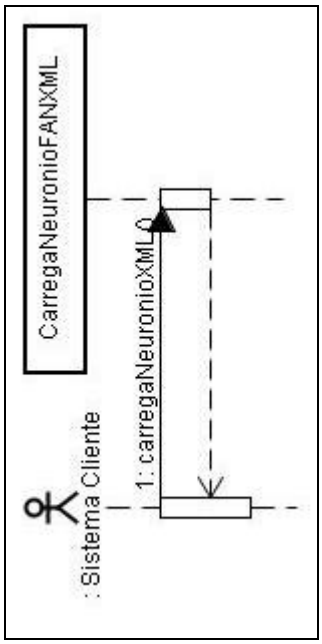

Figura 39 – Diagrama de Seqüência JFAN – Carregar Rede

Salvar Rede

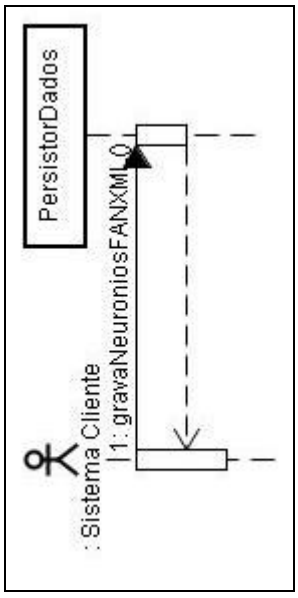

Figura 40 – Diagrama de Seqüência JFAN – Salvar Rede

Iniciar Treinamento

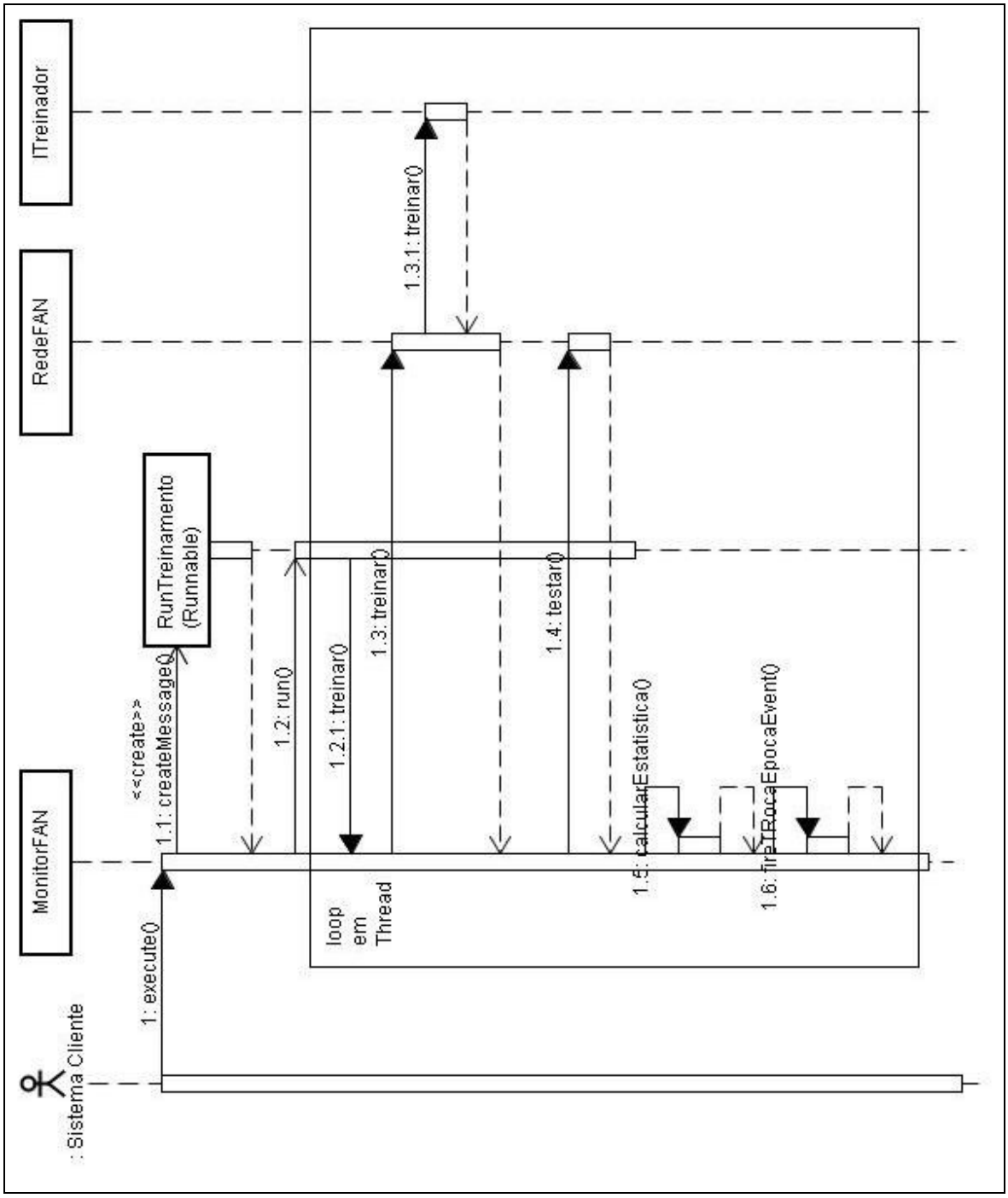

Figura 41 – Diagrama de Seqüência JFAN – Iniciar Treinamento

Testar

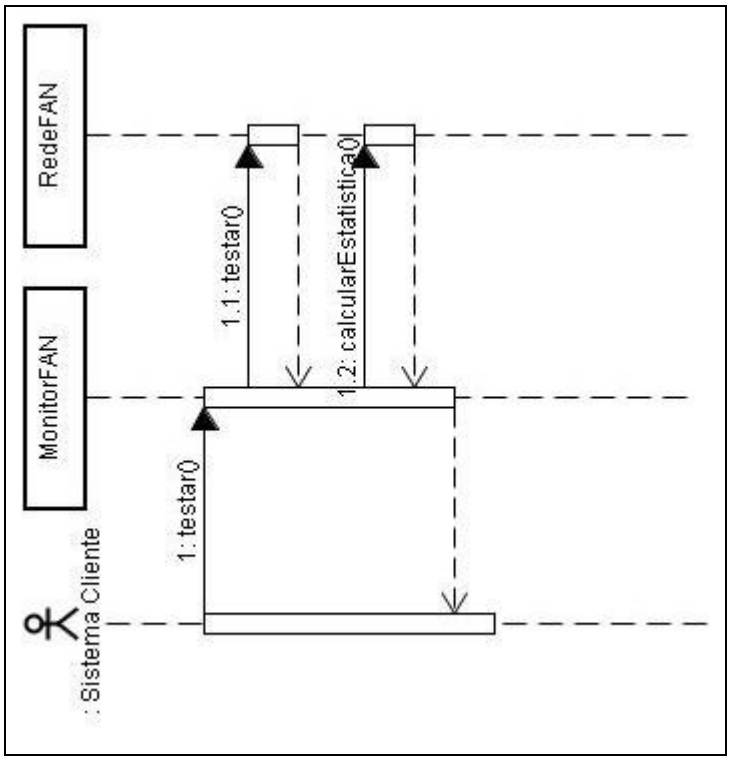

Figura 42 – Diagrama de Seqüência JFAN – Testar

Validar

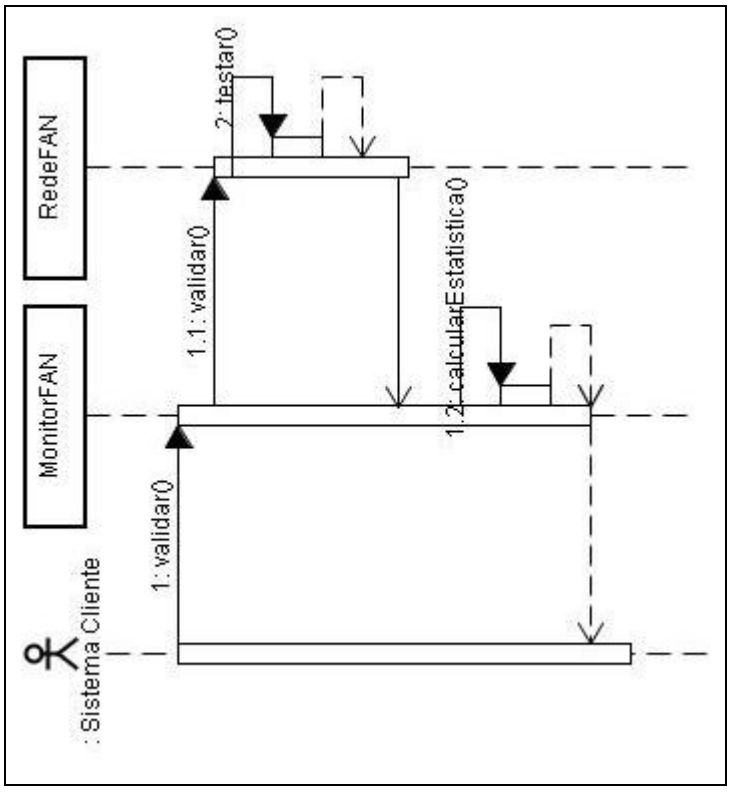

Figura 43 – Diagrama de Seqüência JFAN – Validar

Classificar

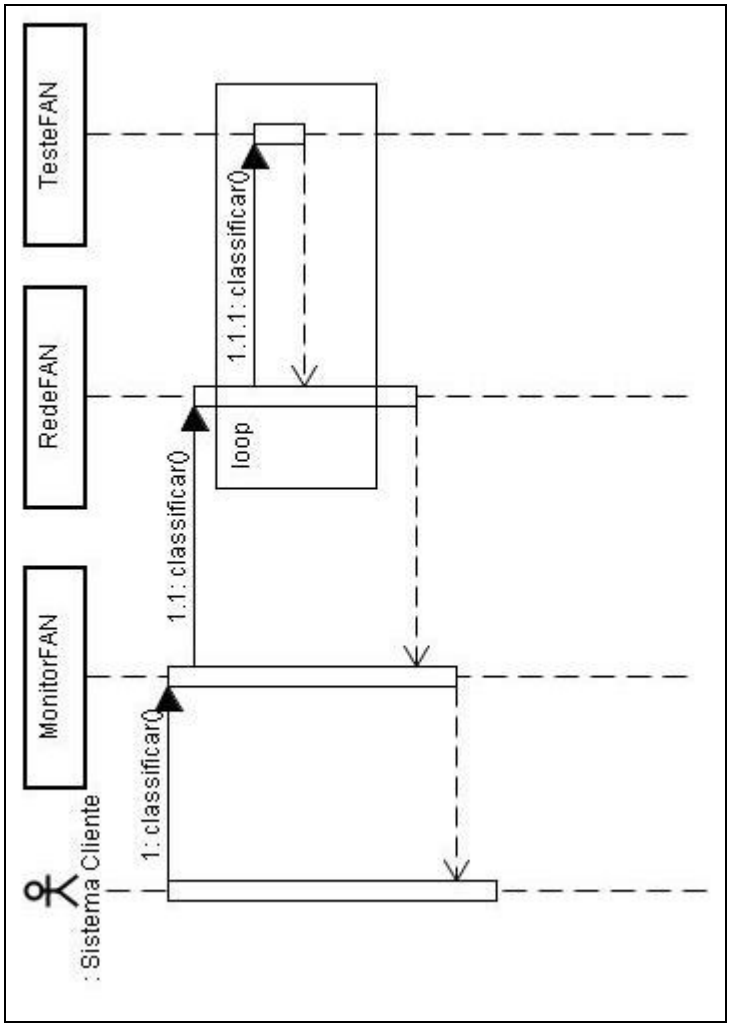

Figura 44 – Diagrama de Seqüência JFAN – Classificar

Parar Treinamento

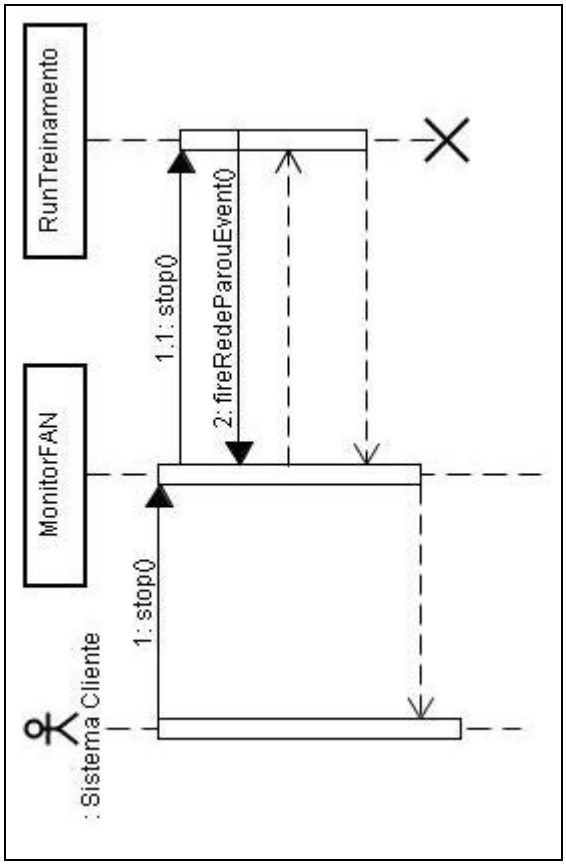

Figura 45 – Diagrama de Seqüência JFAN – Parar Treinamento

5. ESPECIFICAÇÃO TÉCNICA - EASYFAN

5.1. CASOS DE USO

5.1.1. Geral

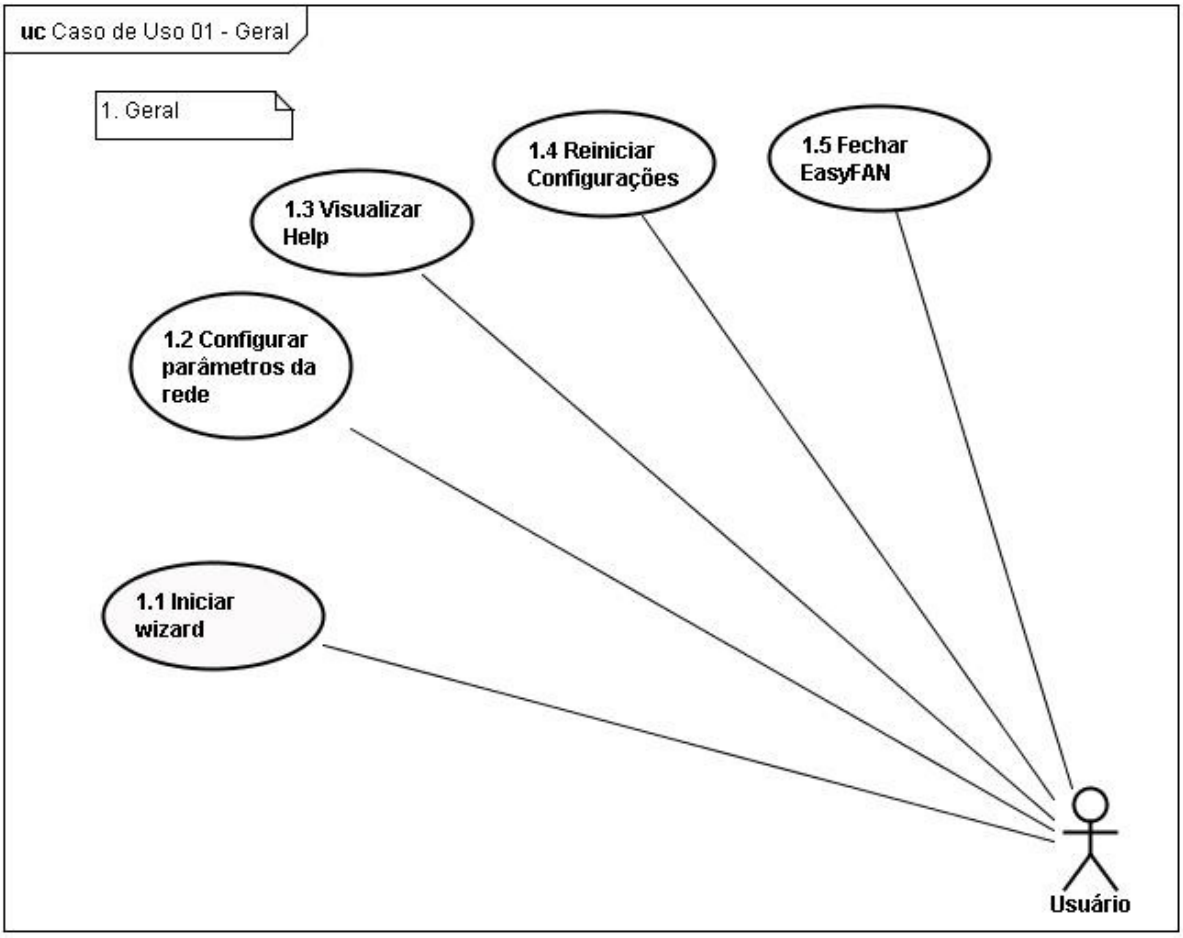

Figura 46 - Caso de uso 01 - geral

1.1 Iniciar Wizard

Prioridade: 3 (Média)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Realizar a abertura do programa no modo Wizard (O modo Wizard foi feito para auxiliar que usuários leigos saibam os conceitos básicos de reconhecimento de padrões e como podem treinar uma rede)

Tipo: Opcional

Fluxo de Eventos:

3. Usuário solicita modo Wizard
4. Sistema inicia modo Wizard

1.2 Configurar Parâmetros da Rede

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Usuário configura parâmetros da rede escolhendo o tamanho do Raio difuso (Conjunto difuso) e o tamanho do range (Suporte do conjunto difuso).
Tipo: Opcional

Fluxo de Eventos:
1. Usuário seleciona opção “Configurar Parâmetros da Rede”
2. Sistema mostra opções de configuração
3. Usuário escolhe parâmetros de configuração

1.3 Visualizar Help

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Abrir o Help do programa para auxiliar os usuários em eventuais dúvidas sobre o programa e suas funcionalidades

Tipo: Opcional

Fluxo de Eventos:
1. Usuário seleciona opção “Help” no menu Ajuda
2. Sistema Inicia o Help do EasyFAN

1.4 Reiniciar Configurações

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Sistema retorna as configurações padrões do programa como os parâmetros da rede e limpa os conjuntos carregados.

Tipo: Opcional

Fluxo de Eventos:
1. Usuário seleciona opção “Reiniciar Configurações”

2. Sistema pergunta se o usuário deseja realmente reiniciar as configurações
3. Sistema reinicia configurações

Fluxo alternativo para o passo 2, o caso usuário não deseje reiniciar configurações

2.1 Encerra caso de uso

1.5 Fechar EasyFAN

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Sistema finaliza a utilização do EasyFAN.

Tipo: Opcional

Fluxo de Eventos:
1. Usuário seleciona opção “Sair”
2. Sistema pergunta se o usuário deseja realmente fechar o EasyFAN
3. Sistema é encerrado

Fluxo alternativo para o passo 2, o caso usuário não deseje fechar o EasyFAN

2.1 Encerra caso de uso

5.1.2. Treinamento

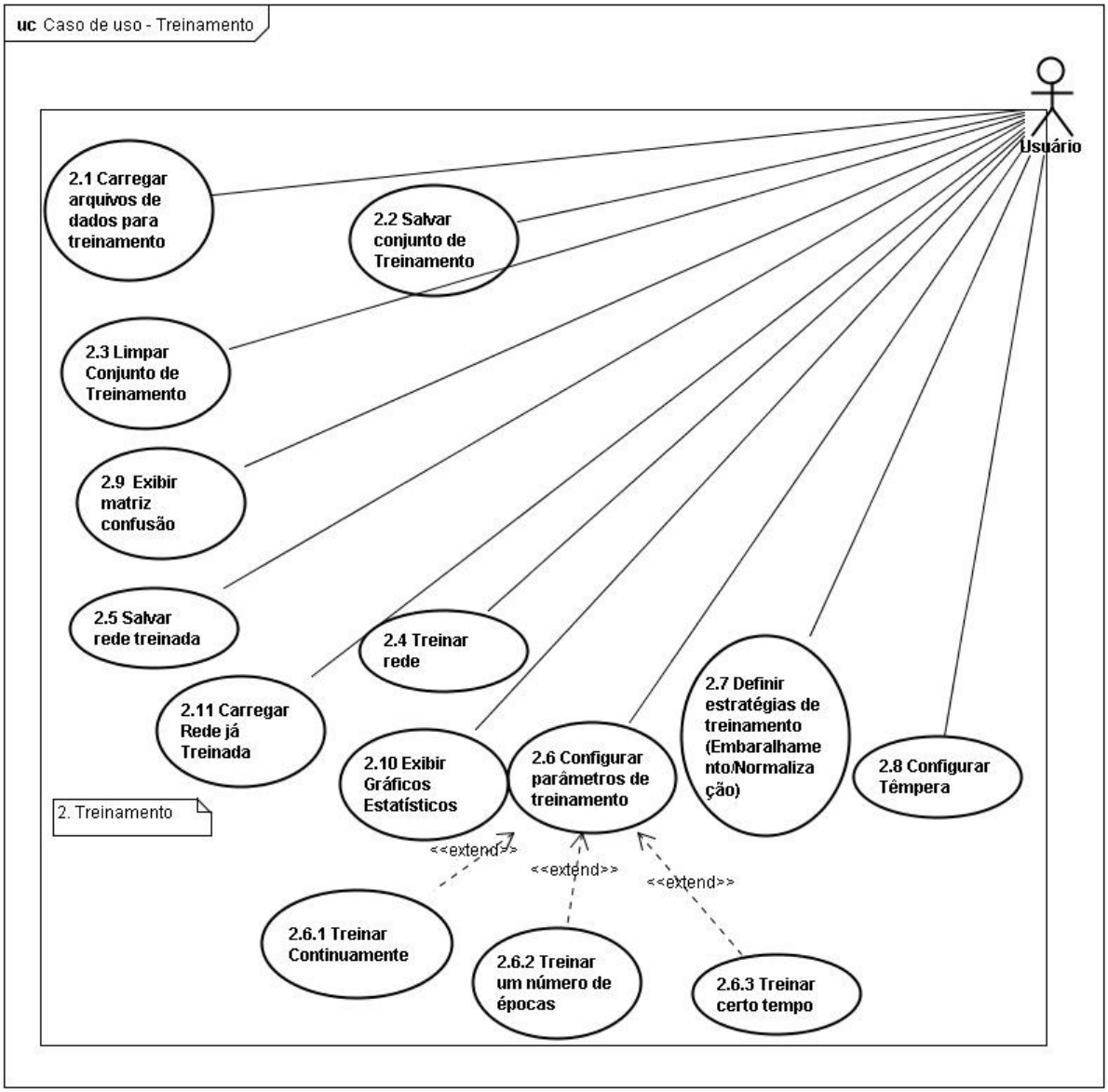

Figura 47 - Diagrama Caso de Uso - Treinamento

2.1 Carregar Arquivos de Dado para Treinamento

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Usuário realiza carregamento de arquivo(s) para o conjunto de treinamento que devem possuir uma das seguintes extensões: *.DAT, *.TXT, *.XLS e *.EPD.

Tipo: Secundário

Fluxo de Eventos:
1 Usuário escolhe opção “Carregar arquivos de treinamento”
2 Sistema mostra arquivos disponíveis e suas respectivas extensões para serem carregados
3 Usuário informa o arquivo de treinamento e extensão a ser carregado
4 Sistema carrega arquivo de treinamento

Fluxo alternativo para o passo 2, caso não exista nenhum arquivo de dado de treinamento dentro das extensões especificadas
1.1 Mensagem “Não há arquivos de dados para serem carregados”
1.2 Encerra Caso de Uso

Fluxo alternativo para o passo 3, caso o arquivo escolhido pelo usuário seja *.TXT
3.1 Usuário escolhe o separador de colunas, e os números que delimitarão a linha inicial, linha final, coluna inicial, coluna final e coluna da classe do conjunto de arquivos.
3.2 Sistema carrega arquivo de treinamento

Fluxo alternativo para o passo 3, caso o arquivo escolhido pelo usuário seja *.XLS
3.1 Usuário escolhe os números que delimitarão a linha inicial, linha final, coluna inicial, coluna final e coluna da classe do conjunto de arquivos.

3.2 Sistema carrega arquivo de treinamento

2.2 Salvar conjunto de treinamento

Prioridade: 5 (Alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Armazenar conjunto de treinamento carregado podendo ser nas seguintes extensões: *.DAT, *.EPD.

Tipo: Primário

Fluxo de Eventos:
1 Usuário escolhe opção “Salvar Conjunto de Treinamento”
2 Sistema solicita local de armazenamento, nome do arquivo, e extensão a ser salva
3 Usuário escolhe local de armazenamento, nome do arquivo e extensão a ser salva
4 Sistema salva arquivo de treinamento

Fluxo alternativo para o passo 1, caso não exista conjunto de treinamento carregado
1.1 Mensagem “Não há nenhum conjunto carregado para ser salvo”
1.2 Usuário escolhe opção “Carregar arquivo de dados para treinamento”

Fluxo alternativo para o passo 1.2, caso o usuário não deseje carregar o arquivo de treinamento.

1.1.1. Encerra caso de uso

Fluxo alternativo para o passo 3, caso o arquivo escolhido pelo usuário seja *.TXT
3.3. Usuário escolhe o separador de colunas que o arquivo terá.
3.4. Sistema salva arquivo de treinamento

Fluxo alternativo para o passo 3, caso já exista um arquivo de treinamento com o mesmo nome , o mesmo local de armazenamento e a mesma extensão
3.1. Mensagem “Esse arquivo já existe, deseja substituir o arquivo existente?”
3.2. Usuário opta por substituir arquivo existente
3.3. Sistema salva arquivo de treinamento

Fluxo alternativo para o passo 3.2, caso o usuário não deseje substituir o arquivo existente e opte por salvar um novo conjunto de dados de treinamento
3.2.1 Usuário escolhe novo local de armazenamento, novo nome ou nova extensão para o arquivo de treinamento
3.2.2 Sistema salva arquivo de treinamento

2.3 Limpar Conjunto de Treinamento

Prioridade: 3 (Média)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Limpar conjunto de treinamento carregado

Tipo: Secundário

Fluxo de Eventos:
1. Usuário seleciona opção “Limpar Conjunto de Treinamento”
2. Sistema limpa conjunto de treinamento

2.4 Treinar Rede

Prioridade: 5 (Alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Sistema inicia o treinamento da rede.

Tipo: Primário

Fluxo de Eventos:
1. Usuário seleciona opção “Treinar rede”
2. Sistema inicia treinamento

Fluxo alternativo para o passo 1, caso não exista arquivo de treinamento carregado
1.1. Sistema desabilita Opção “Treinar Rede” até que se tenha um arquivo de treino carreado

Fluxo alternativo para o passo 1, caso não exista arquivo de teste carregado
1.2. Sistema desabilita Opção “Treinar Rede” até que se tenha um arquivo de teste carregado

2.5 Salvar Rede

Prioridade: 5 (Alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Após treinar uma rede, o usuário escolhe qual tipo de rede que deseja arquivar (Atual, Melhor Máximo do Mínimo, Melhor Média harmônica e Melhor Média Aritmética) e qual extensão que a rede vai ter (*.ENN, *.FAN).

Tipo: Primário

Fluxo de Eventos:
1. Usuário escolhe opção “Salvar Rede”
2. Sistema solicita local de armazenamento, nome e extensão da rede a salvar
3. Usuário escolhe local de armazenamento, nome e extensão da rede
4. Sistema salva rede

Fluxo alternativo para o passo 3, caso já exista uma rede com o mesmo nome, mesmo local de armazenamento e mesma extensão
3.1 Mensagem “Essa rede já existe, deseja substituir a rede existente?”.
3.2 Usuário opta por substituir a rede existente
3.3 Sistema salva projeto

Fluxo alternativo para o passo 3.2, caso o usuário não deseje substituir a rede e opte por salvar uma nova rede
3.2.1 Usuário escolhe novo local de armazenamento ou novo nome ou nova extensão para a rede
3.2.2 Sistema salva rede

Fluxo alternativo para o passo 3.2, caso o usuário opte por não substituir a rede existente e nem salvar uma rede.
3.2.1 Encerra Caso de Uso

2.6 Configurar Parâmetros de Treinamento

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Usuário configura parâmetros de treinamento especificando como deseja eu seja o andamento do treinamento escolhendo uma das seguintes opções: Treinar continuamente, treinar a cada X épocas (X é um número inteiro especificado pelo usuário) e treinar por tempo especificando em quantos dias, horas, minutos e segundos é que o treinamento deve durar.

Tipo: Opcional

Fluxo de Eventos:
1. Usuário seleciona opção “Configurar Parâmetros de Treinamento”
2. Sistema mostra opções de configuração do treinamento
3. Usuário escolhe a configuração de treinamento

Fluxo alternativo para o passo 1, caso não exista arquivo de treinamento carregado
1.1 Sistema desabilita opção “Configurar Parâmetros de Treinamento”.

2.7 Definir Estratégias de Treinamento

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Usuário escolhe estratégias para treinar a rede escolhendo a cada quantas épocas deseja embaralhar o conjunto de treinamento, a porcentagem de padrões para treino, a cada quantas épocas deve se realizar a normalização, qual a chance de normalizar especificando a porcentagem, qual o peso de randomização. O usuário pode setar ainda a normalização por característica escolhendo a característica do conjunto de arquivo e qual o valor mínimo, máximo e médio que ela deve ter.

Tipo: Opcional

Fluxo de Eventos:
1 Usuário define estratégias de treinamento e de normalização

Fluxo alternativo para o passo 1, caso não exista arquivo de treinamento carregado
1.1. Mensagem “Não há nenhum arquivo de treinamento carregado, para escolher as estratégias de treinamento é necessário que se carregue um arquivo”.
1.2. Chama Caso de Uso “Carregar Arquivo de Dados para Treinamento”

2.8 Configurar Têmpera

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: O usuário pode configurar a têmpera especificando em quantas épocas deve ocorrer o step da tempera, em quantas épocas deve ser reiniciada a têmpera e escolher qual o limite mínimo, máximo e step da témpera.

Tipo: Opcional

Fluxo de Eventos:
1 Usuário Seleciona opção “Configurar Têmpera”
2 Usuário escolhe configurações da têmpera

2.9 Exibir Matriz de Confusão (Treino)

Prioridade: 4 (Média-Alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Sistema exibe matriz de confusão da época capturada do treino que mostra em quantidade de ocorrência ou porcentagem a taxa de acerto por classe, além de mostrar qual a época capturada e qual a média harmônica, média aritmética e máximo do mínimo da época.

Tipo: Opcional

Fluxo de Eventos:
1 Usuário seleciona opção “Matriz de Confusão”
2 Sistema mostra dados da matriz de confusão

Fluxo alternativo para o passo 1, caso não exista uma rede treinada
1.1 Sistema desabilita opção “Exibir Matriz Confusão”

2.10 Exibir Gráficos Estatísticos

Prioridade: 4 (Média-Alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Usuário escolhe qual gráfico estatístico deseja visualizar (Evolução de erros, gráfico de neurônios, comparação de características e visualização de características).

Tipo: Secundário

Fluxo de Eventos:
1. Usuário seleciona opção “Exibir gráfico estatístico”
2. Sistema mostra os tipos de gráficos visualizáveis (Evolução de erros, gráfico de neurônios, comparação de características e visualização de características).

3. Usuário escolhe o gráfico que deseja visualizar
4. Sistema exibe gráfico estatístico

2.11 Carregar Rede já Treinada

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Usuário realiza carregamento do arquivo de rede manualmente.
Tipo: Secundário

Fluxo de Eventos:
1. Usuário escolhe opção “Carregar Rede”
2. Sistema mostra redes que podem ser carregados e extensões de redes permitidas (*.ENN, *.FAN)
3. Usuário informa a rede a ser carregada
4. Sistema carrega rede

Fluxo alternativo para o passo 2, caso não exista nenhuma rede armazenada nas extensões permitidas
1.1 Mensagem “Não há redes a serem carregadas”
1.2 Encerra Caso de Uso

5.1.3. Teste

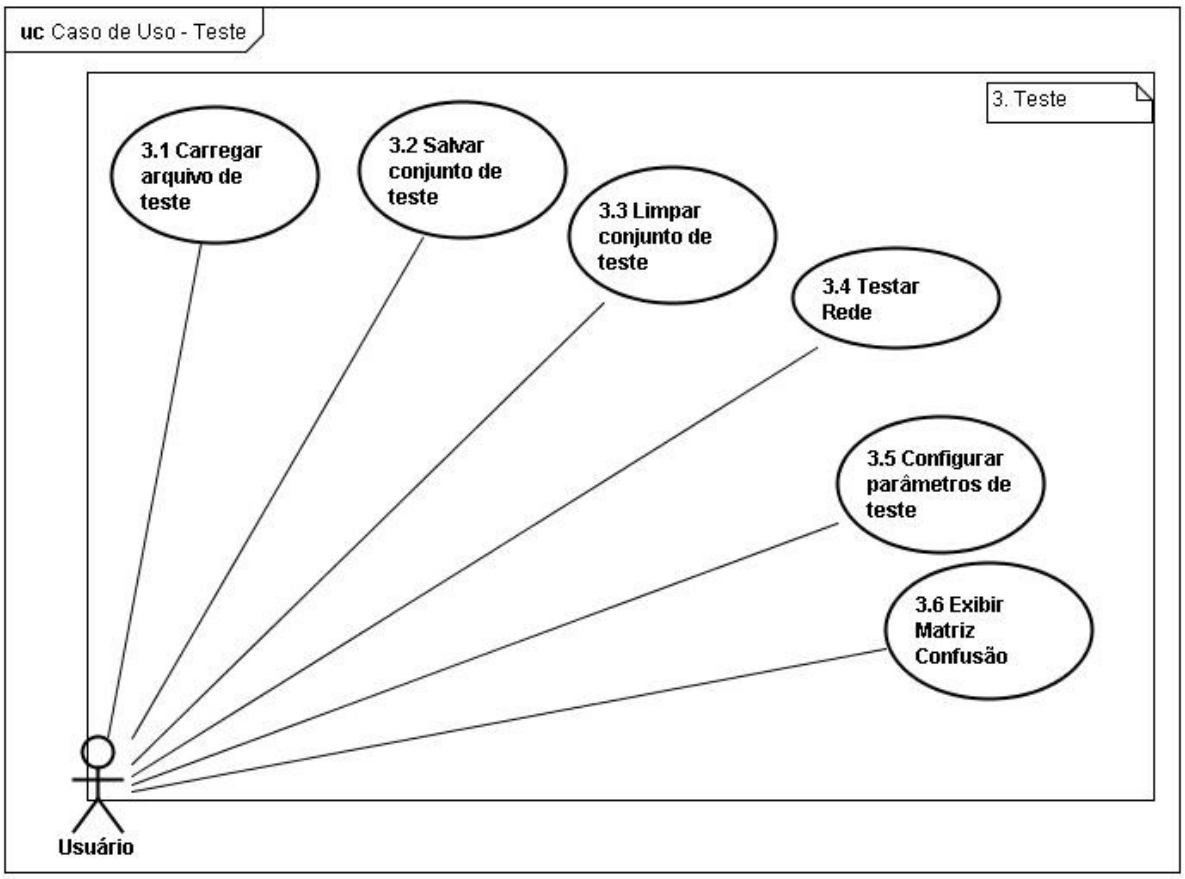

Figura 48 - Diagrama Caso de Uso - Teste

3.1 Carregar Arquivos de Dado para Teste

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Usuário realiza carregamento de arquivo(s) para o conjunto de teste que devem possuir uma das seguintes extensões: *.DAT, *.TXT, *.XLS, *.EPD.

Tipo: Secundário

Fluxo de Eventos:
1. Usuário escolhe opção “Carregar arquivos de teste”
2. Sistema mostra arquivos disponíveis e suas respectivas extensões para serem carregados
3. Usuário informa o arquivo de teste e extensão a ser carregado
4. Sistema carrega arquivo de teste

Fluxo alternativo para o passo 2, caso não exista nenhum arquivo de dado de teste dentro das extensões especificadas
2.1. Mensagem “Não há arquivos de dados para serem carregados”
2.2. Encerra Caso de Uso

Fluxo alternativo para o passo 3, caso o arquivo escolhido pelo usuário tenha a extensão *.TXT
3.1. Usuário escolhe o separador de colunas, e os números que delimitarão a linha inicial, linha final, coluna inicial, coluna final e coluna da classe do conjunto de arquivos.
3.2. Sistema carrega arquivo de teste

Fluxo alternativo para o passo 3, caso o arquivo escolhido pelo usuário tenha a extenão *.XLS
3.1. Usuário escolhe os números que delimitarão a linha inicial, linha final, coluna inicial, coluna final e coluna da classe do conjunto de arquivos.
3.2. Sistema carrega arquivo de teste

3.2 Salvar conjunto de teste

Prioridade: 5 (Alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Armazenar conjunto de teste carregado podendo ser nas seguintes extensões: *.DAT, *.EPD.

Tipo: Primário

Fluxo de Eventos:
1. Usuário escolhe opção “Salvar Conjunto de Teste”
2. Sistema solicita local de armazenamento, nome do arquivo, e extensão a ser salva
3. Usuário escolhe local de armazenamento, nome do arquivo e extensão a ser salva
4. Sistema salva arquivo de teste

Fluxo alternativo para o passo 1, caso não exista conjunto de teste carregado
1.1. Mensagem “Não há nenhum conjunto carregado para ser salvo”
1.2. Usuário escolhe opção “Carregar arquivo de dados para teste”

Fluxo alternativo para o passo 3, caso já exista um arquivo de teste com o mesmo nome , o mesmo local de armazenamento e a mesma extensão
3.1. Mensagem “Esse arquivo já existe, deseja substituir o arquivo existente?”.
3.2 Usuário opta por substituir arquivo existente
3.3 Sistema salva arquivo de teste

Fluxo alternativo para o passo 3.2, caso o usuário não deseje substituir o arquivo existente e opte por salvar um novo conjunto de dados de teste.

3.2.1 Usuário escolhe novo local de armazenamento, novo nome ou nova extensão para o arquivo de teste.
3.2.2 Sistema salva arquivo de teste

Fluxo alternativo para o passo 3.2, caso o usuário opte por não substituir o arquivo existente e nem e nem salvar um novo arquivo de teste.
3.2.1 Encerra Caso de Uso

3.3 Limpar Conjunto de Teste

Prioridade: 3 (Média)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Limpar conjunto de teste carregado

Tipo: Secundário

Fluxo de Eventos:
1. Usuário seleciona opção “Limpar Conjunto de Teste”
2. Sistema limpa conjunto de teste

3.4 Testar Rede

Prioridade: 5 (Alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Usuário escolhe qual tipo de rede que deseja testar (Carregada, Atual, Melhor Média Harmônica, Melhor Máximo do Mínimo e Melhor Média Aritmética) .

Tipo: Primário

Fluxo de Eventos:
1. Usuário seleciona opção “Testar rede”
2. Sistema inicia teste

Fluxo alternativo para o passo 1, caso não tenha nenhuma rede treinada ou carregada
1.1. Mensagem “Não há nenhuma rede de treinamento carregada, para se testar uma rede é necessário que se carregue uma rede ou se treine uma nova rede”.
1.2. Usuário decide por carregar uma rede
1.3. Chama Caso de Uso “Carregar Rede”

Fluxo alternativo para o passo 1.2, caso o usuário decida por treinar uma nova rede
1.2.1 Chama Caso de Uso “Treinar Rede”

Fluxo alternativo para o passo 1, caso não exista arquivo de teste carregado
1.1. Sistema desabilita opção “Testar Rede”

3.5 Configurar Parâmetros de Teste

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Usuário configura parâmetros do teste da rede decidindo se deve se testar sempre ou apenas a cada X épocas (X é um número inteiro positivo que deve ser especificado pelo usuário).

Tipo: Opcional

Fluxo de Eventos:
1 Usuário seleciona opção “Configurar Parâmetros de Teste”
2 Sistema mostra opções de configuração do teste
3 Usuário escolhe parâmetros de configuração

Fluxo alternativo para o passo 1, caso não exista arquivo de teste carregado
1.1. Sistema desabilita opção “Configurar Parâmetros de Teste”

3.6 Exibir Matriz de Confusão (Teste)

Prioridade: 4 (Média-Alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Sistema exibe matriz de confusão do teste realizado mostrando em quantidade de ocorrência ou porcentagem a taxa de acerto por classe e qual a média harmônica, média aritmética e máximo do mínimo.

Tipo: Opcional

Fluxo de Eventos:
1 Usuário seleciona opção “Matriz de Confusão”
2 Sistema mostra dados da matriz de confusão

Fluxo alternativo para o passo 1, caso não exista uma rede testada
1.1. Sistema desabilita opção “Exibir Matriz Confusão”

5.1.4. Validação

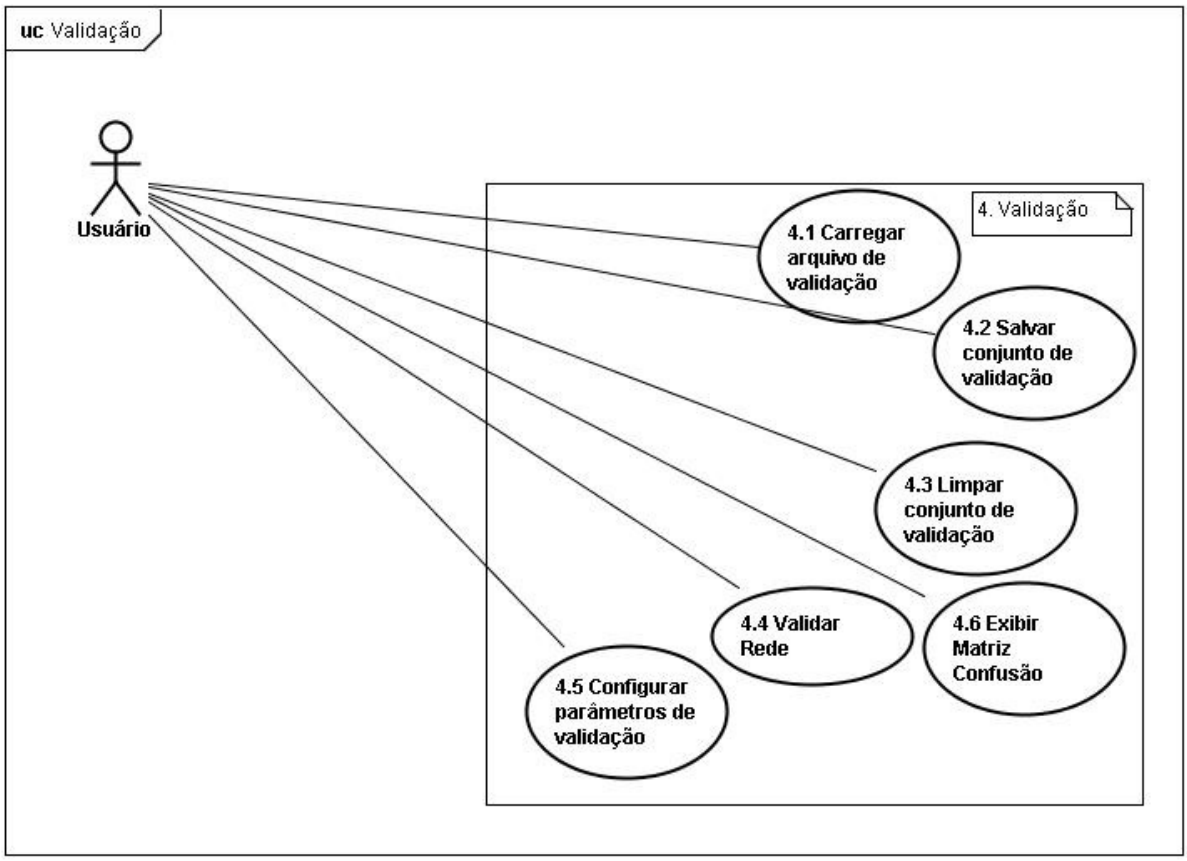

Figura 49 - Casos de Uso - Validação

4.1 Carregar Arquivos de Validação

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Usuário realiza carregamento do arquivo de dados para validação que pode ter as seguintes extensões: *.DAT, *.TXT, *.XLS, *.EPD.

Tipo: Secundário

Fluxo de Eventos:
1. Usuário escolhe opção “Carregar arquivos de Validação”
2. Sistema mostra arquivos disponíveis e suas respectivas extensões para serem carregados
3. Usuário informa o arquivo de validação e extensão a ser carregado
4. Sistema carrega arquivo de validação

Fluxo alternativo para o passo 2, caso não exista nenhum arquivo de dado de validação dentro das extensões especificadas

1.1. Mensagem “Não há arquivos para serem carregados”
1.2. Encerra Caso de Uso

Fluxo alternativo para o passo 3, caso o arquivo escolhido pelo usuário tenha extensão *.TXT
3.1. Usuário escolhe o separador de colunas, e os números que delimitarão a linha inicial, linha final, coluna inicial, coluna final e coluna da classe do conjunto de arquivos.
3.2. Sistema carrega arquivo de validação

Fluxo alternativo para o passo 3, caso o arquivo escolhido pelo usuário tenha extensão *.XLS
3.1. Usuário escolhe os números que delimitarão a linha inicial, linha final, coluna inicial, coluna final e coluna da classe do conjunto de arquivos.
3.2. Sistema carrega arquivo de validação

4.2 Salvar Conjunto de Validação

Prioridade: 5 (Alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Armazenar conjunto de validação carregado podendo ser nas seguintes extensões: *.DAT e *.EPD.

Tipo: Primário

Fluxo de Eventos:
1. Usuário escolhe opção “Salvar Conjunto de Validação”
2. Sistema solicita local de armazenamento, nome do arquivo, e extensão a ser salva
3. Usuário escolhe local de armazenamento, nome do arquivo e extensão a ser salva
4. Sistema salva arquivo de validação

Fluxo alternativo para o passo 1, caso não exista conjunto de validação carregado
1.1 Mensagem “Não há nenhum conjunto carregado para ser salvo”
1.2 Usuário escolhe opção “Carregar arquivo de dados para validação”

Fluxo alternativo para o passo 1.2, caso o usuário não deseje carregar o arquivo de validação.
1.2.1. Encerra caso de uso

Fluxo alternativo para o passo 3, caso já exista um arquivo de validação com o mesmo nome , o mesmo local de armazenamento e a mesma extensão
3.1 Mensagem “Esse arquivo já existe, deseja substituir o arquivo existente?”.
3.2 Usuário opta por substituir arquivo existente
3.3 Sistema salva arquivo de validação

Fluxo alternativo para o passo 3.2, caso o usuário não deseje substituir o arquivo existente e opte por salvar um novo conjunto de dados de validação.
3.2.1 Usuário escolhe novo local de armazenamento, novo nome ou nova extensão para o arquivo de validação.
3.2.2 Sistema salva arquivo de validação

Fluxo alternativo para o passo 3.2, caso o usuário opte por não substituir o arquivo existente e nem e nem salvar um novo arquivo de validação.
3.2.1 Encerra Caso de Uso

4.3 Limpar Conjunto de Validação

Prioridade: 3 (Média)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Limpar conjunto de validação carregado

Tipo: Secundário

Fluxo de Eventos:
1. Usuário seleciona opção “Limpar Conjunto de Validação”
2. Sistema limpa a tabela de Validação.
3. Sistema limpa conjunto de Validação

4.4 Validar Rede

Prioridade: 5 (Alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Usuário escolhe qual tipo de rede que deseja validar (Carregada, Atual, Melhor Média Harmônica, Melhor Máximo do Mínimo e Melhor Média Aritmética) .

Tipo: Primário

Fluxo de Eventos:
1 Usuário seleciona opção “Validar rede”
2 Usuário escolhe o tipo de rede a ser validada.
3 Sistema inicia Validação.

Fluxo alternativo para o passo 1, caso não tenha nenhuma rede treinada ou carregada
1.1 Mensagem “Não há nenhuma rede de treinamento carregada, para se validar uma rede é necessário que se carregue uma rede ou se treine uma nova rede”.
1.2 Usuário decide por carregar uma rede
1.3 Chama Caso de Uso “Carregar Rede”

Fluxo alternativo para o passo 1.2, caso o usuário decida por treinar uma nova rede
1.2.1 Chama Caso de Uso “Treinar Rede”

Fluxo alternativo para o passo 1, caso não exista arquivo de validação carregado
1.1. Sistema desabilita opção “Validar Rede”

4.5 Configurar Parâmetros de Validação

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Usuário configura parâmetros de validação da rede decidindo se nunca se deve validar a rede ou validar apenas a cada X épocas (X é um número inteiro positivo que deve ser especificado pelo usuário). O usuário também define o tipo de rede a ser validada.

Tipo: Opcional

Fluxo de Eventos:
1 Usuário seleciona opção “Configurar Parâmetros de Validação”
2 Sistema mostra opções de configuração de validação
3 Usuário escolhe parâmetros de configuração

Fluxo alternativo para o passo 1, caso não exista arquivo de validação carregado
1.1. Sistema desabilita opção “Configurar Parâmetros de Validação”

4.6 Exibir Matriz de Confusão (Validação)

Prioridade: 4 (Média-Alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Sistema exibe matriz de confusão da validação realizada mostrando em quantidade de ocorrência ou porcentagem a taxa de acerto por classe e qual a média harmônica, média aritmética e máximo do mínimo.

Tipo: Opcional

Fluxo de Eventos:
1 Usuário seleciona opção “Matriz de Confusão”
2 Sistema mostra dados da matriz de confusão

Fluxo alternativo para o passo 1, caso não exista uma rede validada
1.1 Sistema desabilita opção “Exibir Matriz Confusão”

5.1.5. Classificação

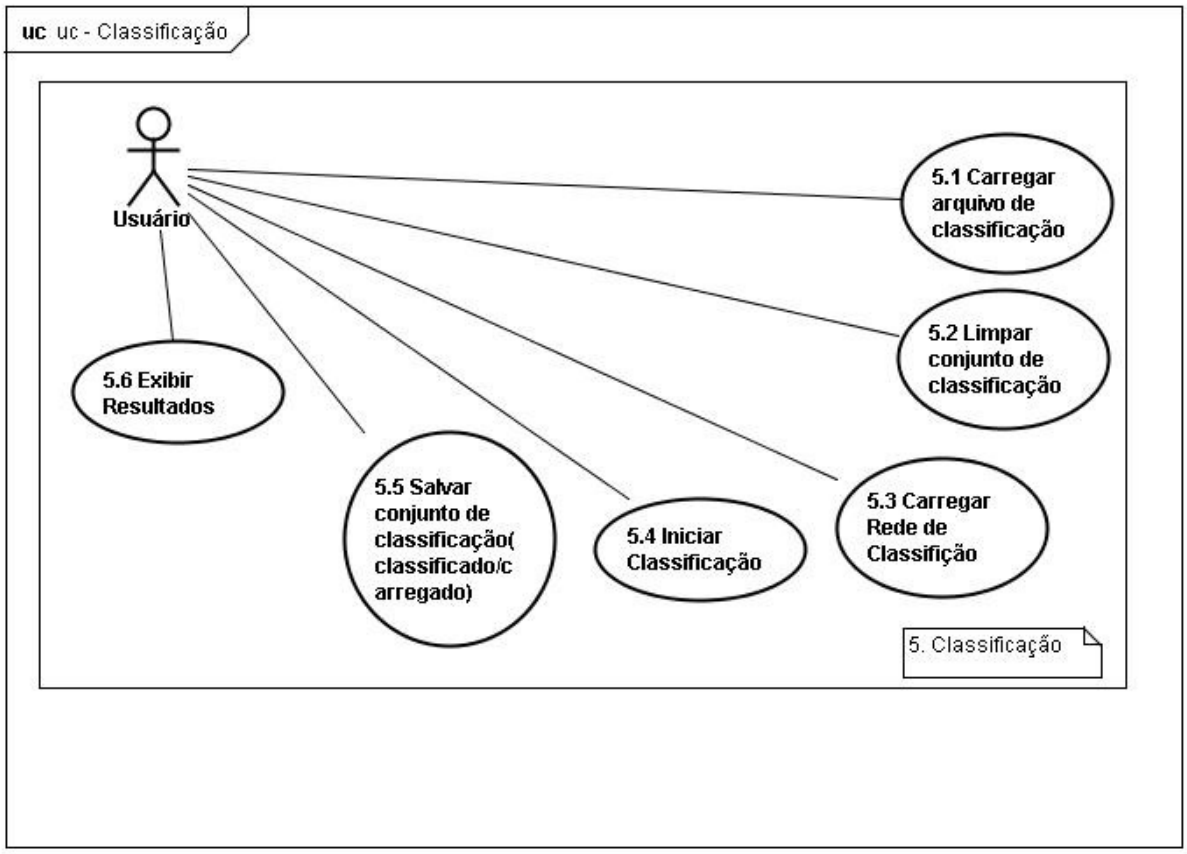

Figura 50 - Casos de Usos - Classificação

5.1 Carregar Arquivos de Classificação

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Usuário realiza carregamento do arquivo de dados para classificação que pode ter as seguintes extensões: *.DAT, *.TXT, *.XLS, *.EPD.

Tipo: Secundário

Fluxo de Eventos:
1 Usuário escolhe opção “Carregar arquivos de Classificação”
2. Sistema mostra arquivos disponíveis e suas respectivas extensões para serem carregados
3. Usuário informa o arquivo de classificação e extensão a ser carregado
4. Sistema carrega arquivo de classifcação

Fluxo alternativo para o passo 2, caso não exista nenhum arquivo de dado de classificação dentro das extensões especificadas
2.1. Mensagem “Não há arquivos para serem carregados”
2.2. Encerra Caso de Uso

Fluxo alternativo para o passo 3, caso o arquivo escolhido pelo usuário tenha extensão *.TXT
3.1. Usuário escolhe o separador de colunas, e os números que delimitarão a linha inicial, linha final, coluna inicial, coluna final e coluna da classe do conjunto de arquivos.
3.2. Sistema carrega arquivo de classificação

Fluxo alternativo para o passo 3, caso o arquivo escolhido pelo usuário tenha extensão *.XLS
3.1. Usuário escolhe os números que delimitarão a linha inicial, linha final, coluna inicial, coluna final e coluna da classe do conjunto de arquivos.
3.2. Sistema carrega arquivo de classificação

5.2 Limpar Conjunto de Classificação

Prioridade: 3 (Média)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Limpar conjunto de classificação carregado

Tipo: Secundário

Fluxo de Eventos:
1. Usuário seleciona opção “Limpar Conjunto de Classificação”
2. Sistema limpa conjunto de Classificação

5.3 Carregar Rede de Classificação

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Usuário realiza carregamento da rede manualmente para realizar a classificação do conjunto de arquivos.

Tipo: Secundário

Fluxo de Eventos:
1. Usuário escolhe opção “Carregar Rede”
2. Sistema mostra redes que podem ser carregados e extensões de redes permitidas (*.ENN, *.FAN)

3. Usuário informa a rede a ser carregada
4. Sistema carrega rede

Fluxo alternativo para o passo 2, caso não exista nenhuma rede armazenada nas extensões permitidas
1.1 Mensagem “Não há redes a serem carregadas”
1.2 Encerra Caso de Uso

5.4 Iniciar Classificação

Prioridade: 5 (Alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Sistema realiza classificação do conjunto de arquivos de dados de classificação carregado

Tipo: Primário

Fluxo de Eventos:
1 Usuário seleciona opção “Classificar Conjunto”
2 Sistema inicia classificação

Fluxo alternativo para o passo 1, caso não se tenha nenhuma rede de classificação carregada
1.1. Mensagem “Não há nenhuma rede de classificação carregada, para se classificar um conjunto é necessário que se carregue uma rede uma rede de classificação”.
1.2. Chama Caso de Uso “Carregar Rede de Classificação”

Fluxo alternativo para o passo 1, caso não exista arquivo de dados classificação carregado
1.1. Sistema desabilita opção “Classificar Conjunto”

5.5 Salvar Conjunto de Classificação

Prioridade: 5 (Alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Usuário escolhe qual tipo de conjunto de classificação que deseja arquivar, carregado (Apenas os conjuntos carregados) e classificado (Após a classificação especificando as classes dos padrões). Após a escolha do tipo de conjunto o usuário deve escolher a extensão que o conjunto deve ter (*.DAT, *.EPD).
Tipo: Primário

Fluxo de Eventos:

1. Usuário escolhe opção “Salvar Conjunto de Classificação”
2. Sistema solicita local de armazenamento, nome do arquivo, e extensão a ser salva
3. Usuário escolhe local de armazenamento, nome do arquivo e extensão a ser salva
4. Sistema salva arquivo de classificação

Fluxo alternativo para o passo 1, caso não exista conjunto de classificação carregado
1.1. Mensagem “Não há nenhum conjunto carregado para ser salvo”
1.2. Usuário escolhe opção “Carregar arquivo de dados para Classificação”

Fluxo alternativo para o passo 1.2, caso o usuário não deseje carregar o arquivo de classificação.
1.2.1. Encerra caso de uso

Fluxo alternativo para o passo 3, caso já exista um arquivo de classificação com o mesmo nome , o mesmo local de armazenamento e a mesma extensão
3.1. Mensagem “Esse arquivo já existe, deseja substituir o arquivo existente?”.
3.2. Usuário opta por substituir arquivo existente
3.3. Sistema salva arquivo de classificação

Fluxo alternativo para o passo 3.2, caso o usuário não deseje substituir o arquivo existente e opte por salvar um novo conjunto de dados de classificação.
3.2.1. Usuário escolhe novo local de armazenamento, novo nome ou nova extensão para o arquivo de classificação.
3.2.2. Sistema salva arquivo de classificação

Fluxo alternativo para o passo 3.2, caso o usuário opte por não substituir o arquivo existente e nem e nem salvar um novo arquivo de classificação.
3.2.1. Encerra Caso de Uso

5.6 Exibir resultados de Classificação

Prioridade: 4 (Média-Alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Sistema exibe resultados da classificação realizada no conjunto de arquivos, mostrando um gráfico de barras de características por classes classificadas.

Tipo: Secundário

Fluxo de Eventos:
1 Usuário escolhe opção “Exibir Resultados de Classificação”
2 Sistema Exibe Resultados de Classificação

Fluxo alternativo para o passo 1, caso não exista arquivo de classificação carregado

1.1. Sistema não mostra resultados da classificação e desabilita opções “Classificar Conjunto” e “Exibir Resultados de Classificação”

Fluxo alternativo para o passo 1, caso não tenha sido realizada a classificação
1.1 Sistema desabilita opção “Exibir Resultados de Classificação”

5.1.6. Wizard

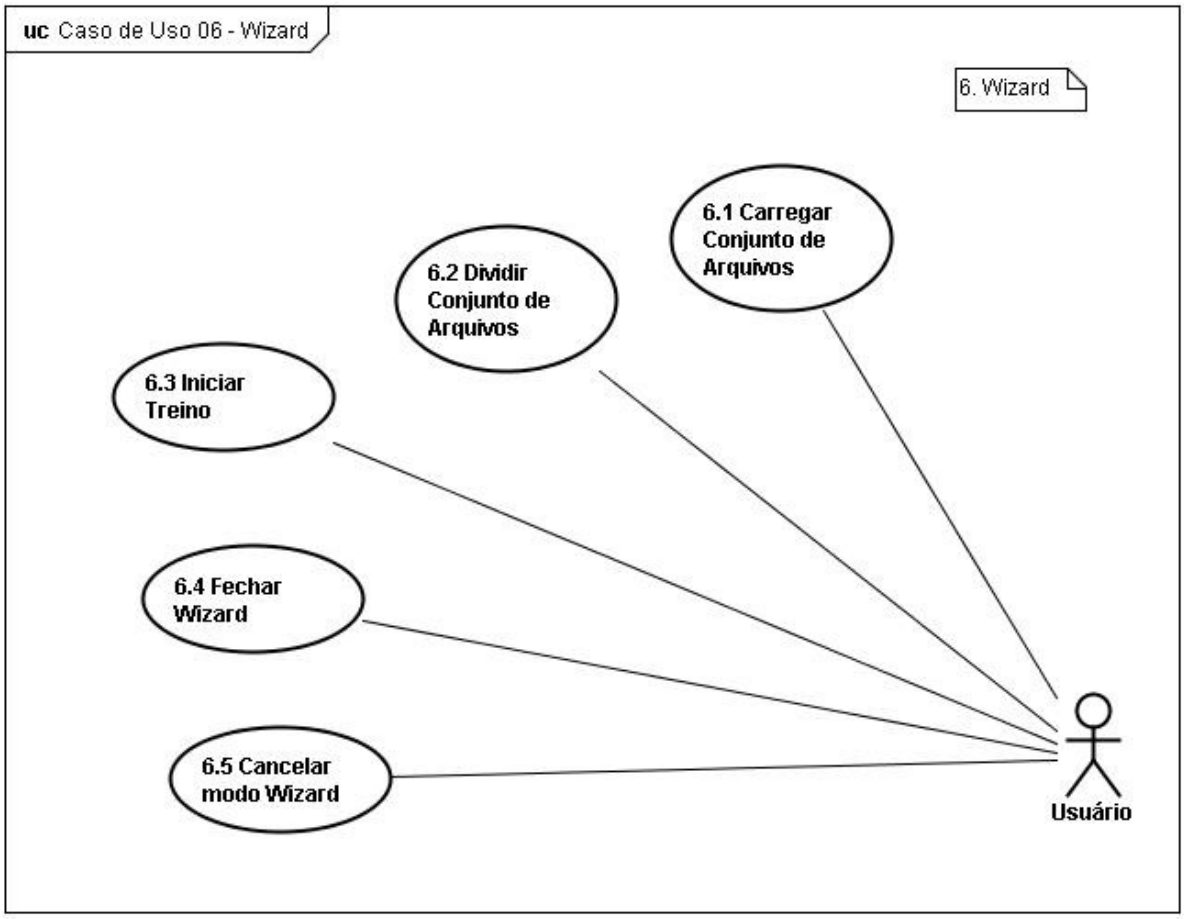

Figura 51 - Casos de Usos - Wizard

6.1Carregar Conjunto de Arquivos

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Usuário realiza carregamento de um conjunto de arquivos que podem ser nas seguintes extensões: *.DAT, *.TXT, *.XLS, *.EPD.

Tipo: Secundário

Fluxo de Eventos:
1 Usuário escolhe opção “Carregar Conjunto de Arquivos”
2. Sistema mostra arquivos disponíveis e suas respectivas extensões para serem carregados
3. Usuário informa o arquivo e extensão a ser carregado
4. Sistema carrega arquivo

Fluxo alternativo para o passo 2, caso não exista nenhum arquivo de dentro das extensões especificadas
2.1. Mensagem “Não há arquivos para serem carregados”
2.2. Encerra Caso de Uso

Fluxo alternativo para o passo 3, caso o arquivo escolhido pelo usuário tenha extensão *.TXT
3.1. Usuário escolhe o separador de colunas, e os números que delimitarão a linha inicial, linha final, coluna inicial, coluna final e coluna da classe do conjunto de arquivos.
3.2. Sistema carrega arquivo

Fluxo alternativo para o passo 3, caso o arquivo escolhido pelo usuário tenha extensão *.XLS
3.1. Usuário escolhe os números que delimitarão a linha inicial, linha final, coluna inicial, coluna final e coluna da classe do conjunto de arquivos.
3.2 Sistema carrega arquivo

6.2 Dividir Conjunto de Arquivos

Prioridade: 4 (Média-alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Usuário realiza a divisão do conjunto de arquivos carregados anteriormente em Treino e Teste. A divisão pode se dar por: Mesmo Conjunto de Treino e Teste, por porcentagem indicando a porcentagem do conjunto que irá para treino e para teste, e adicionar novo conjunto de teste sendo que o conjunto carregado irá para a carga de treino e o usuário terá de escolher um novo conjunto para a carga de teste.

Tipo: Secundário

Fluxo de Eventos:
1 Sistema mostra opções de divisão de conjunto
2 Usuário escolhe uma opção de divisão do conjunto
3 Sistema divide conjunto em treino e teste

6.3 Iniciar Treino

Prioridade: 4 (Alta)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Usuário seleciona opção “Iniciar Treinamento” que cancelará a tela do Wizard e retornará ao EasyFAN na aba de treinamento com as configurações básicas carregas e o treino em execução.

Tipo: Primário

Fluxo de Eventos:
1 Usuário seleciona opção “Iniciar Treinamento”
2 Sistema fecha modo Wizard
3 Sistema volta ao EasyFAN
4 Chama caso de uso “Iniciar Treinamento(EasyFAN)”

6.4 Fechar Wizard

Prioridade: 3(Média)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Em qualquer parte das telas de navegação do wizard o usuário pode fechar o modo Wizard e retornar ao EasyFAN sem ter nenhum comprometimento no que ele realizou no Wizard, podendo voltar a qualquer momento sem que suas configurações sejam reiniciadas.

Tipo: Secundário

Fluxo de Eventos:
1 Usuário seleciona opção “Fechar Wizard”
2 Modo Wizard é fechado
3 Sistema retorna ao EasyFAN

6.5 Cancelar Wizard

Prioridade: 3(Média)

Autor: Filipe Lenfers, Claiton Küster, Luiz Fabiano, Fábio Ignácio, Sérgio Zotto

Ator: Usuário

Propósito: Em qualquer parte das telas de navegação do wizard o usuário pode cancelar o modo Wizard e retornar ao EasyFAN, mas difrerente do modo fechar, a opção cancelar reseta todas as configurações do wizard e se o usuário quiser retornar ao modo wizard terá de setar todas as configurações novamente.

Tipo: Secundário

Fluxo de Eventos:
1. Usuário seleciona opção “Fechar Wizard”
2. Sistema limpa conjunto de arquivos e divisões de arquivos setados
3. Modo Wizard é fechado
4. Sistema retorna ao EasyFAN

5.2. DIAGRAMA DE CLASSES

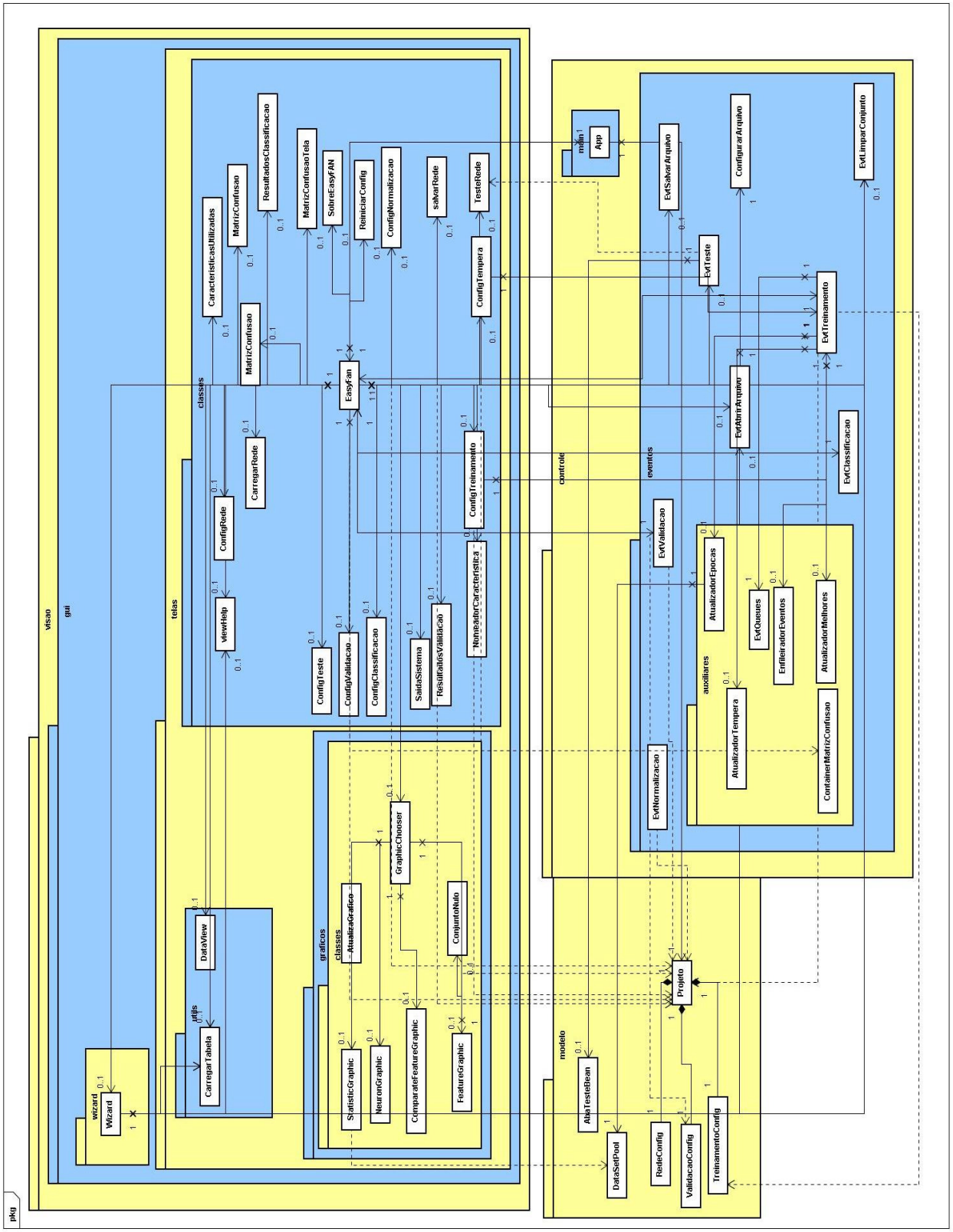

Figura 52 – Diagrama de Classes EasyFAN 1

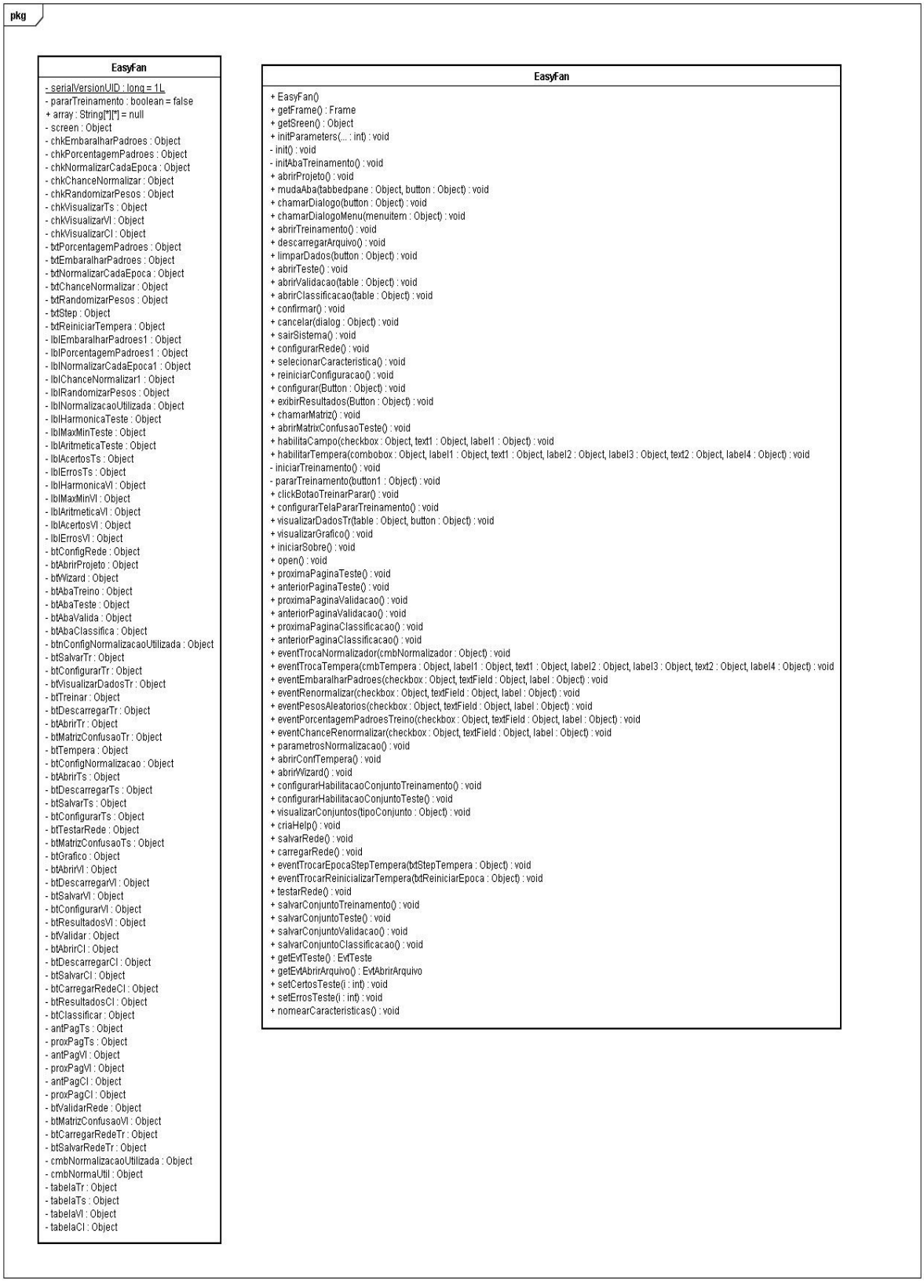

Figura 53 – Diagrama de Classes EasyFAN – Classe EasyFAN

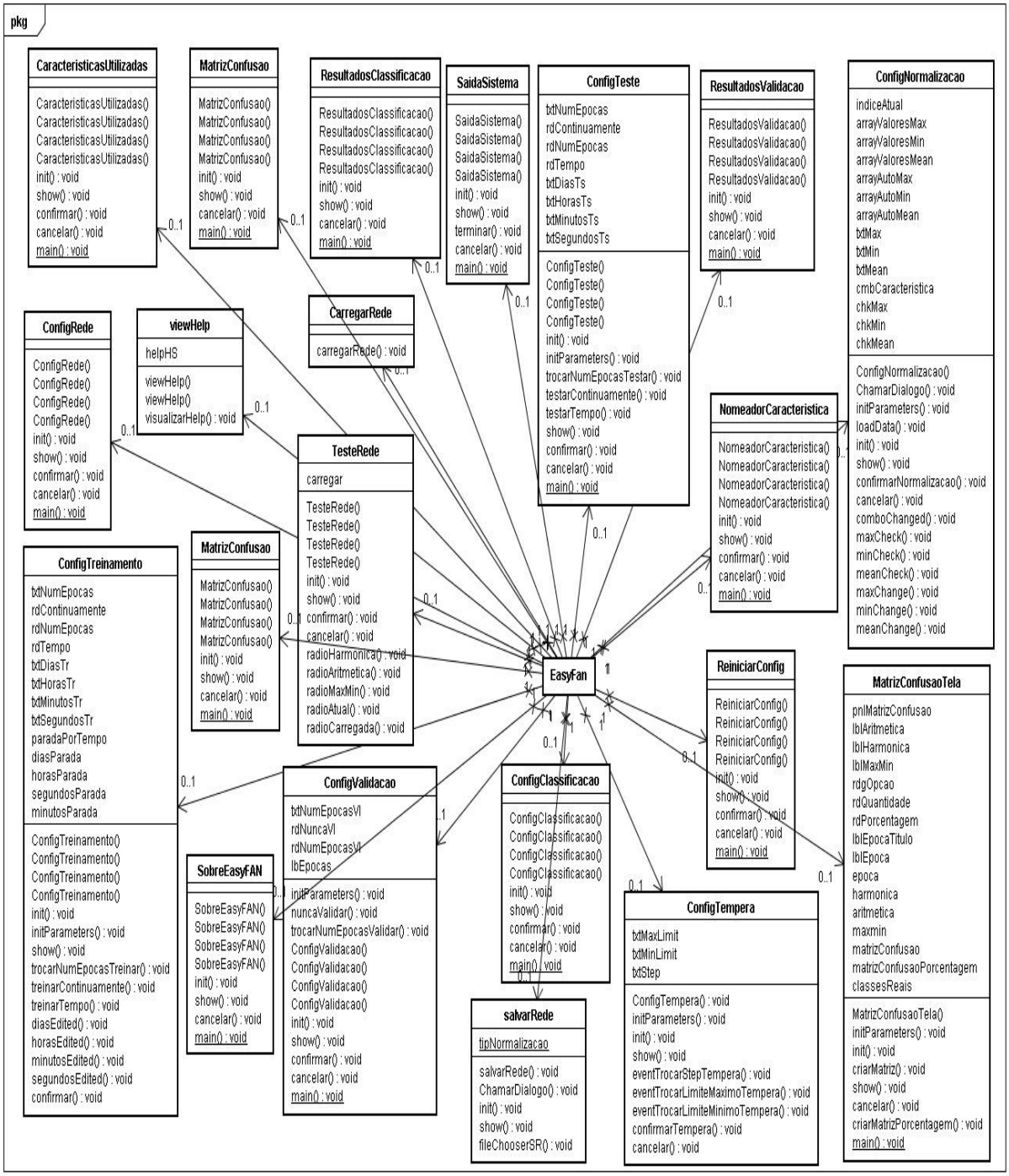

Figura 54 – Diagrama de Classes EasyFAN 2

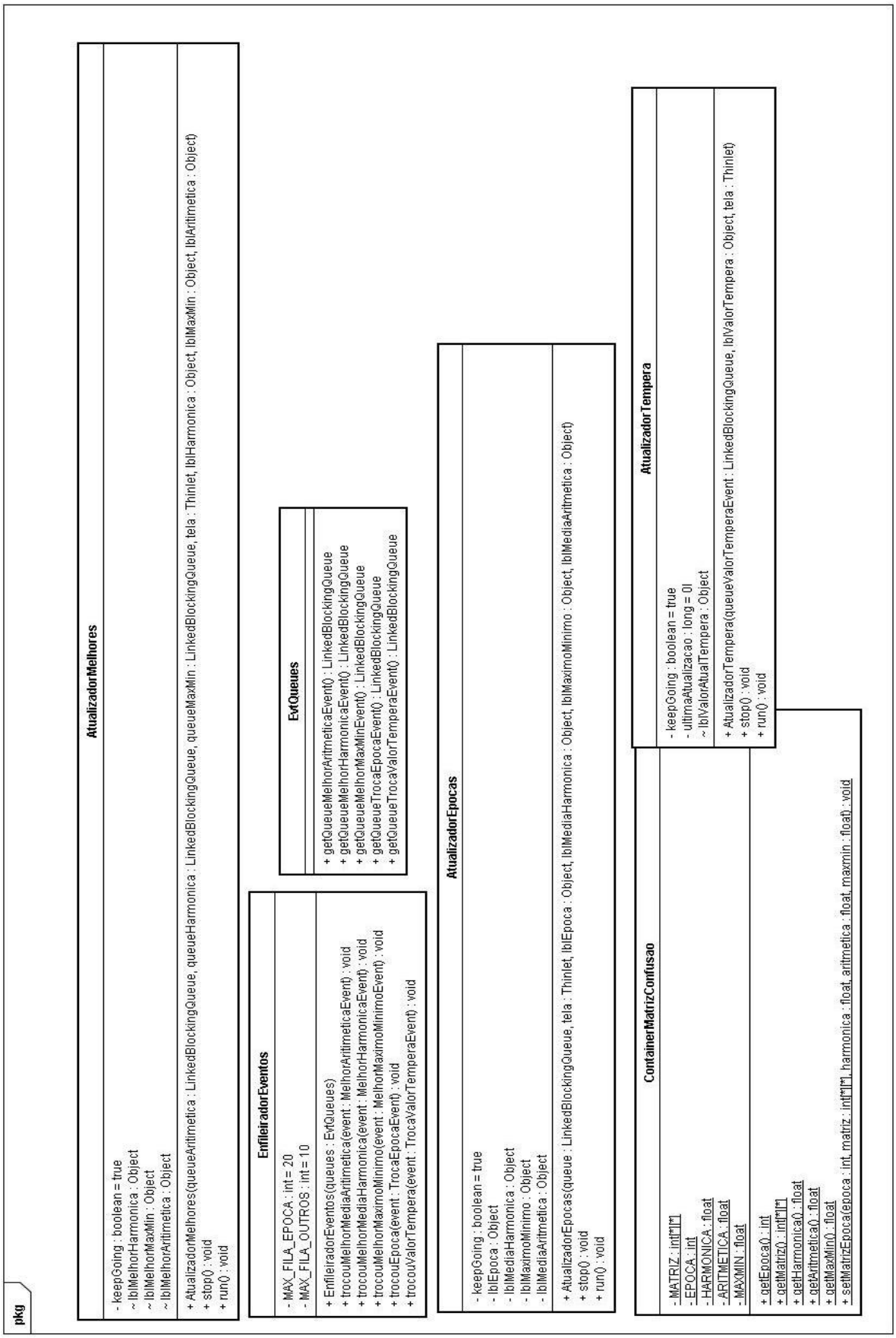

Figura 55 – Diagrama de Classes EasyFAN – Pacote auxiliares

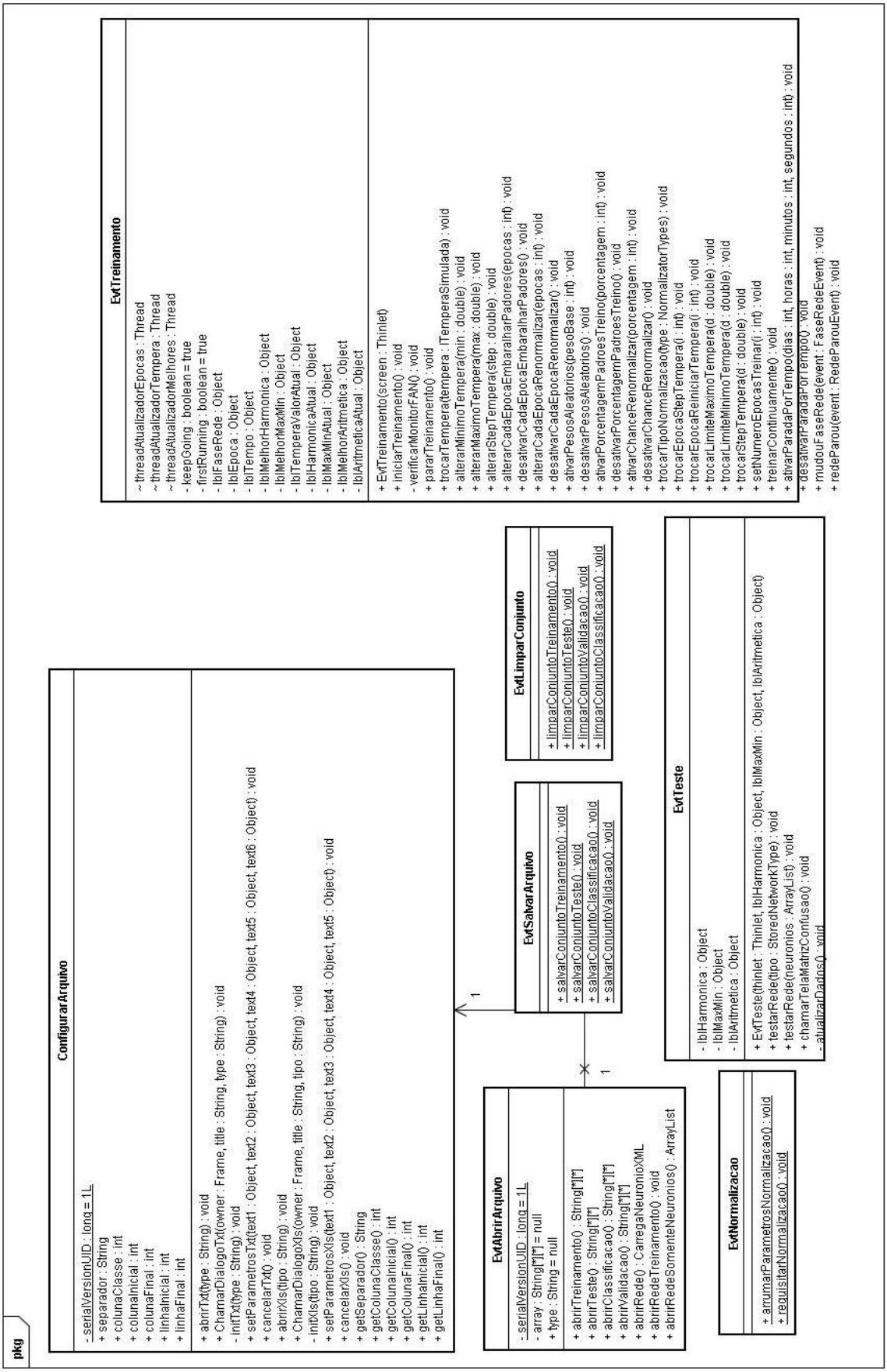

Figura 56 – Diagrama de Classes EasyFAN – Pacote eventos

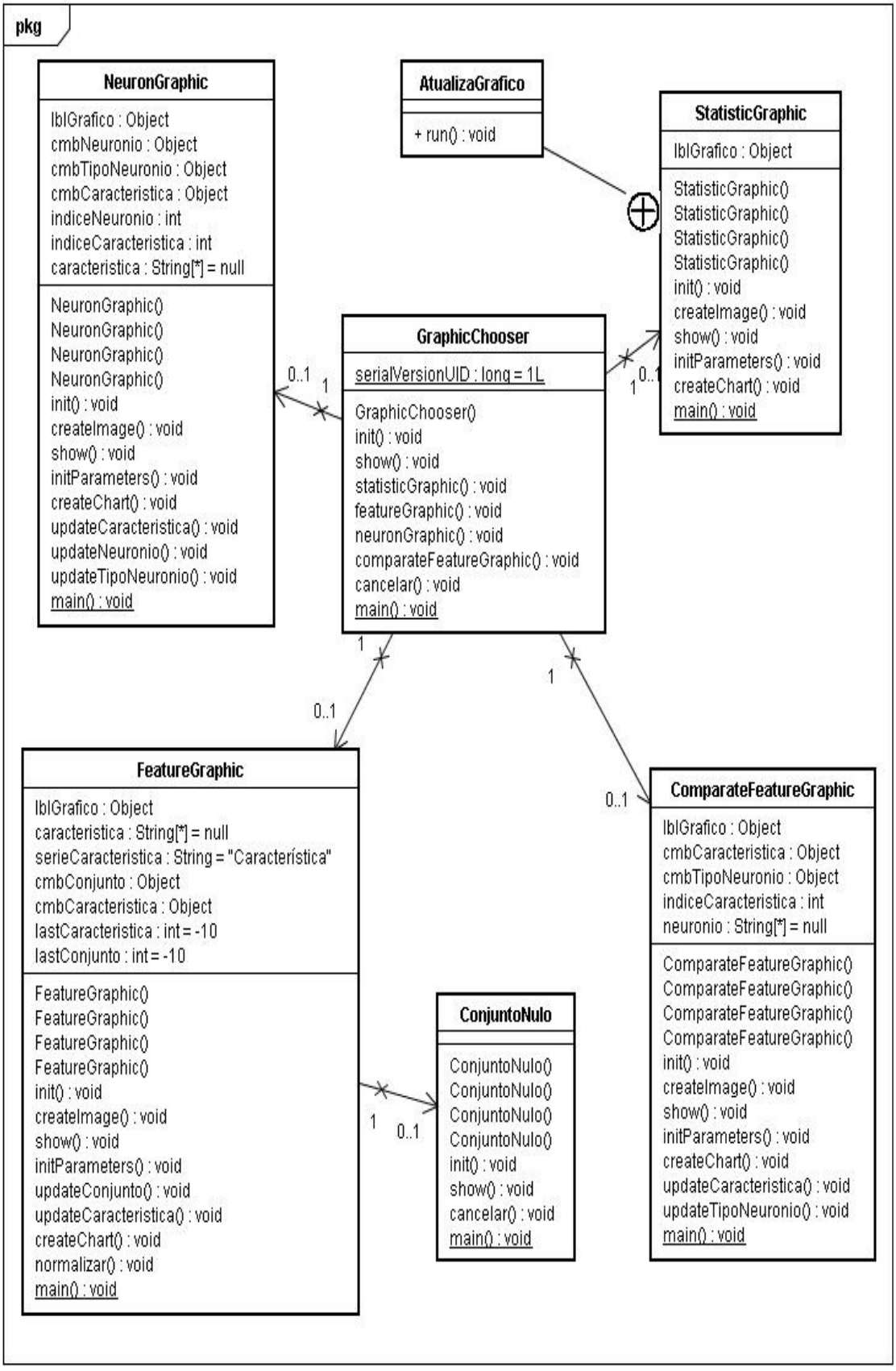

Figura 57 – Diagrama de Classes EasyFAN – Pacote gráficos

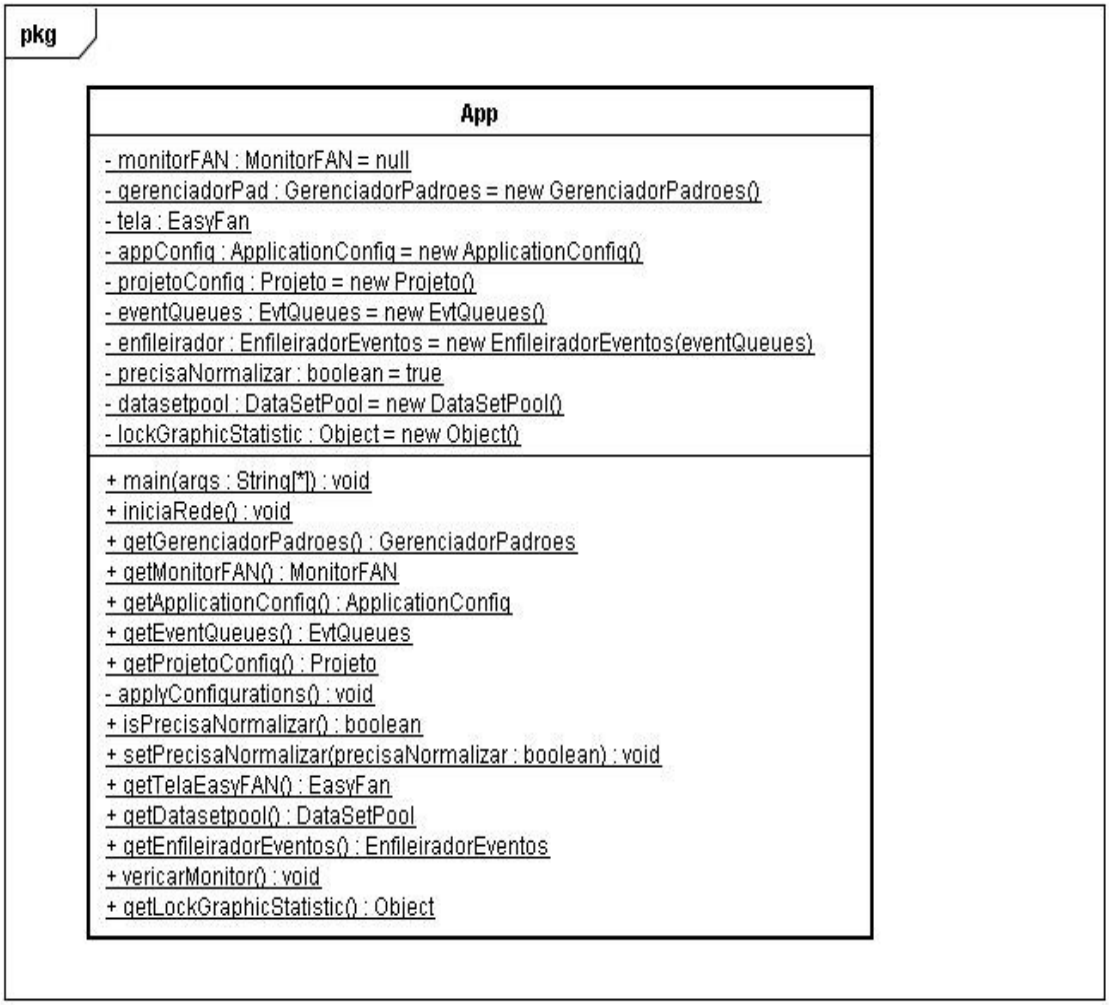

Figura 58 – Diagrama de Classes EasyFAN – Pacote main

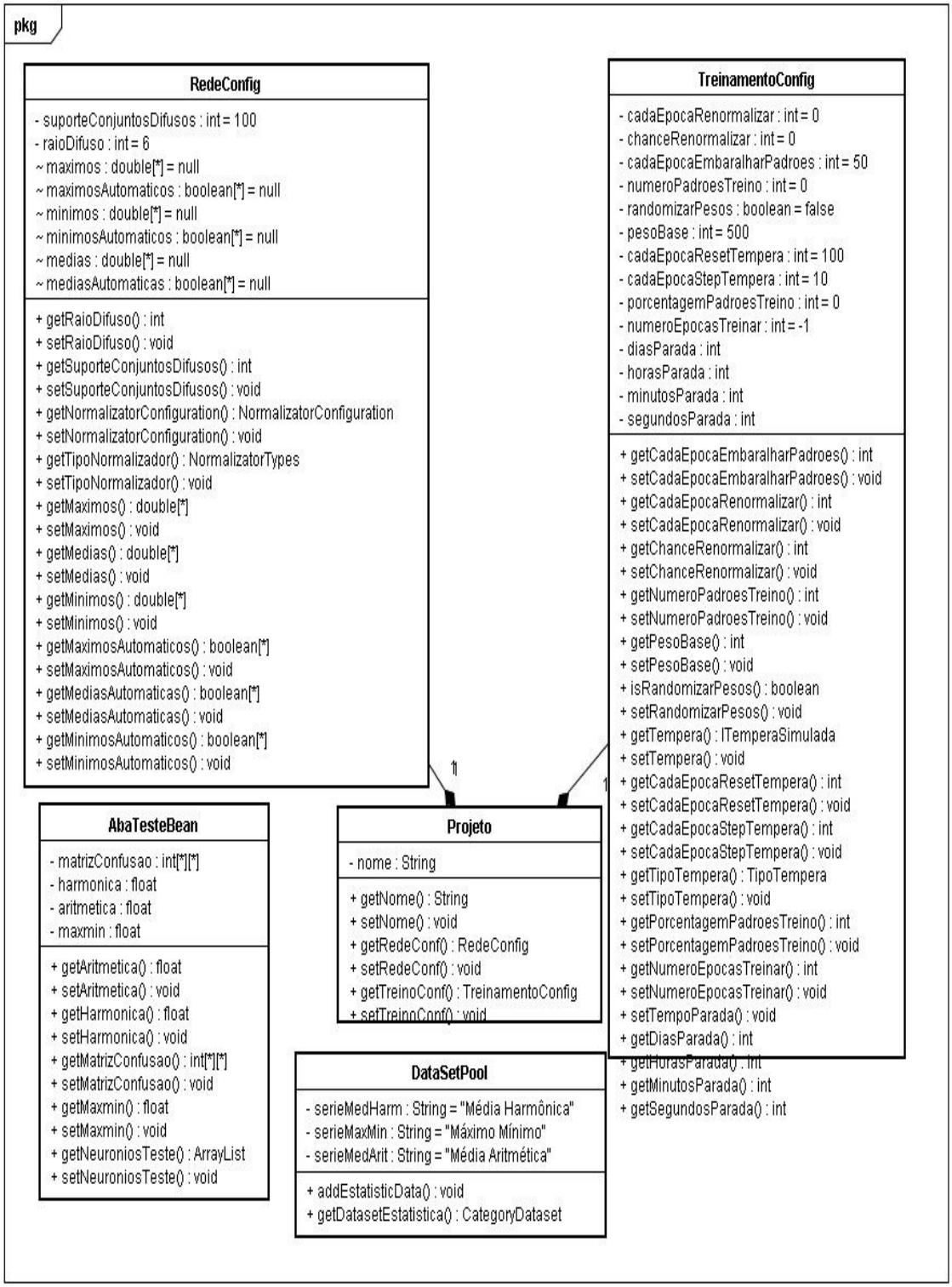

Figura 59 – Diagrama de Classes EasyFAN – Pacote modelo

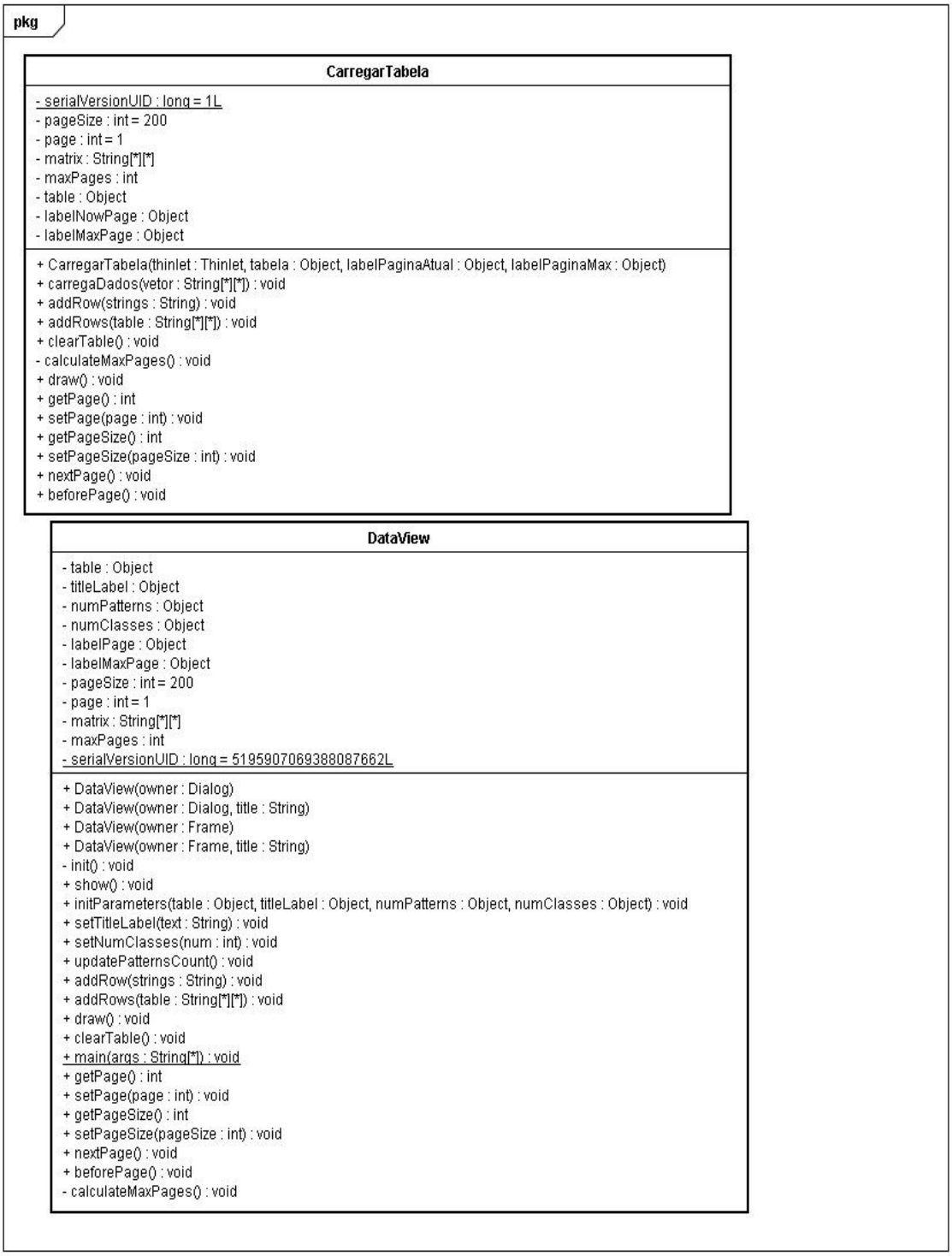

Figura 60 – Diagrama de Classes EasyFAN – Pacote utils

Figura 61 – Diagrama de Classes EasyFAN – Classe Wizard

5.3. DIAGRAMA DE SEQÜÊNCIA

5.3.1. Geral

Configurar Parâmetros

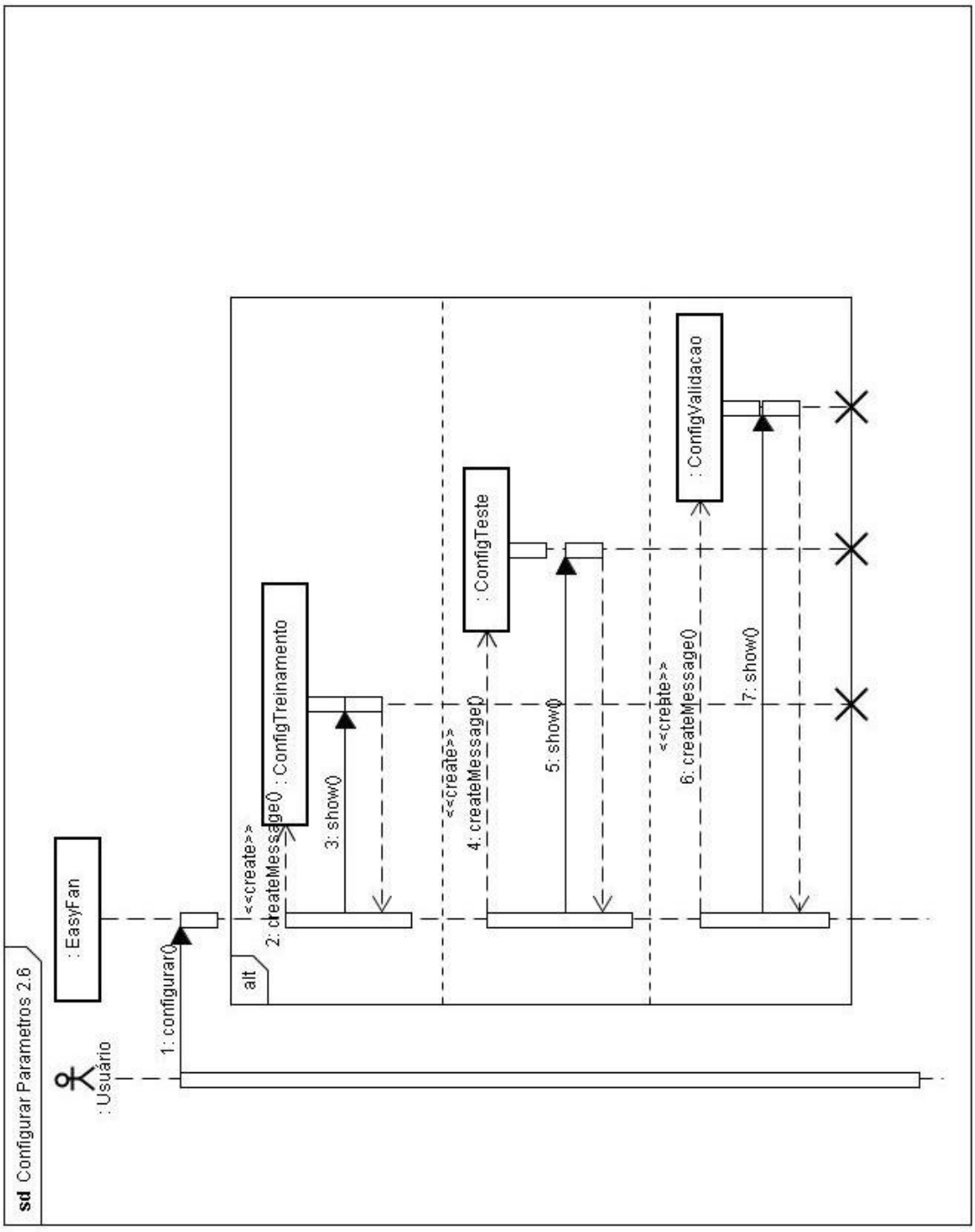

Figura 62 – Diagrama de Seqüência EasyFAN – Configurar Parâmetros

Limpar Conjuntos

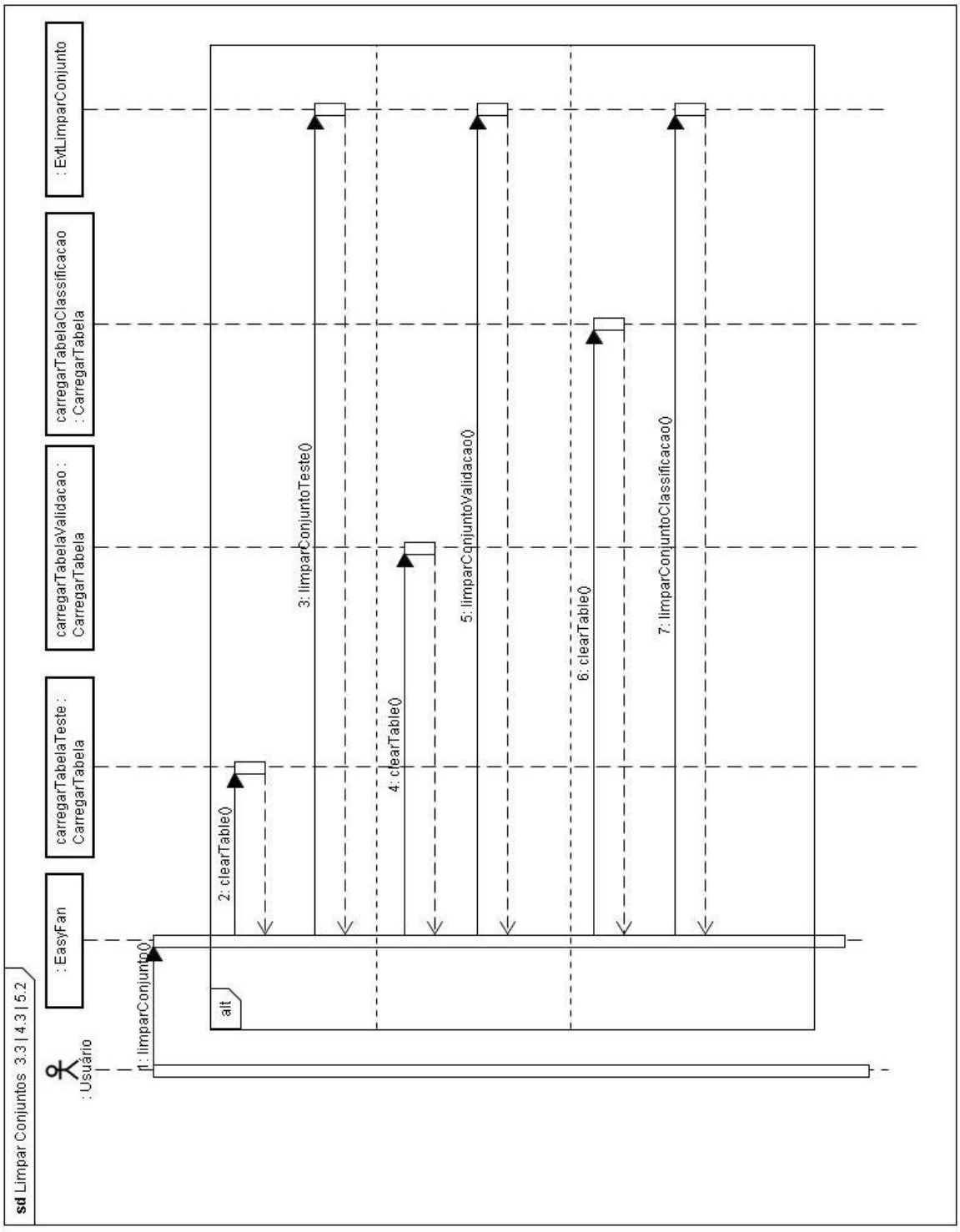

Figura 63– Diagrama de Seqüência EasyFAN – Limpar Conjuntos

5.3.2. Treinamento

Treinamento

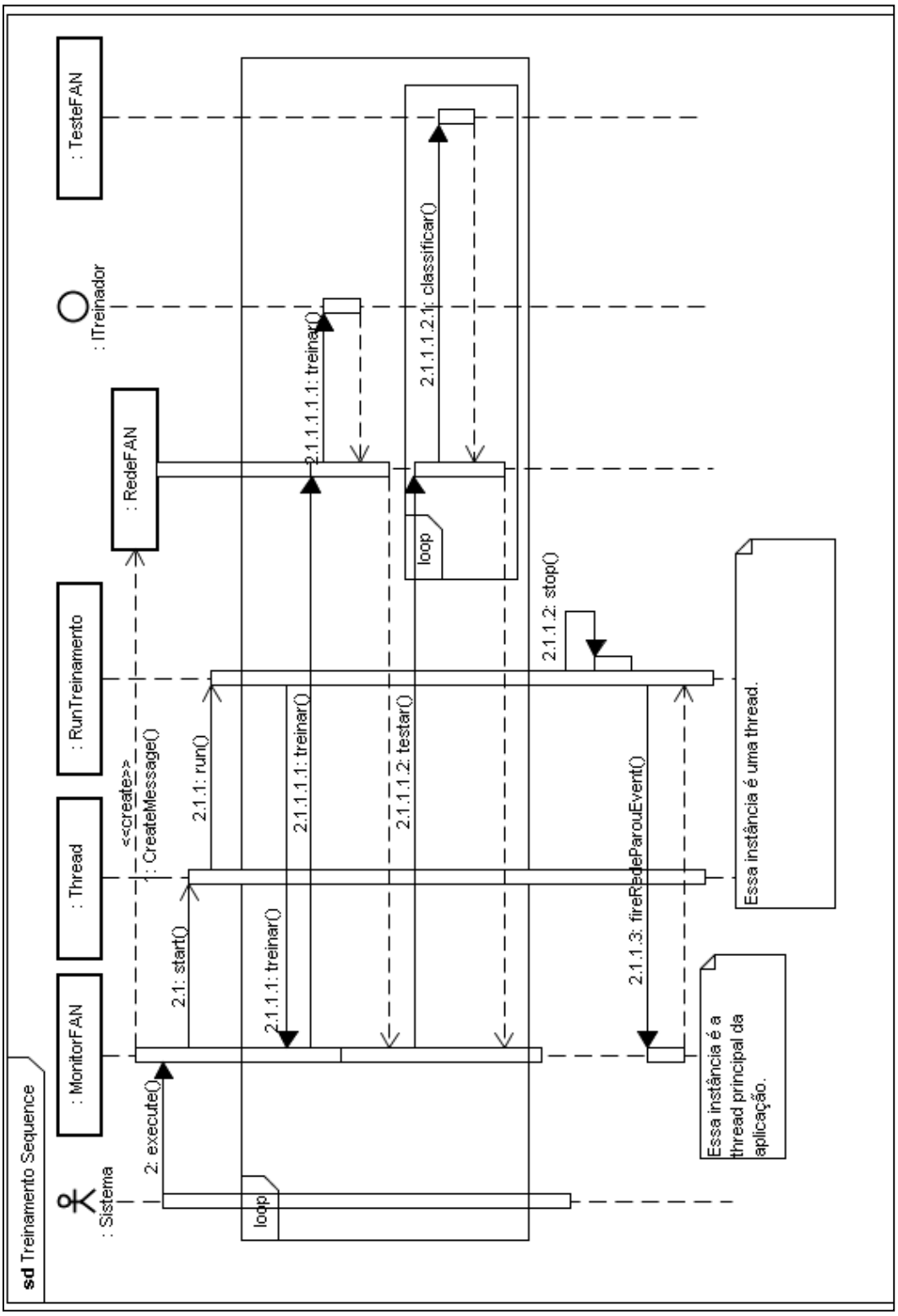

Figura 64 – Diagrama de Seqüência EasyFAN – Treinamento

Treinamento – Carregar Arquivo

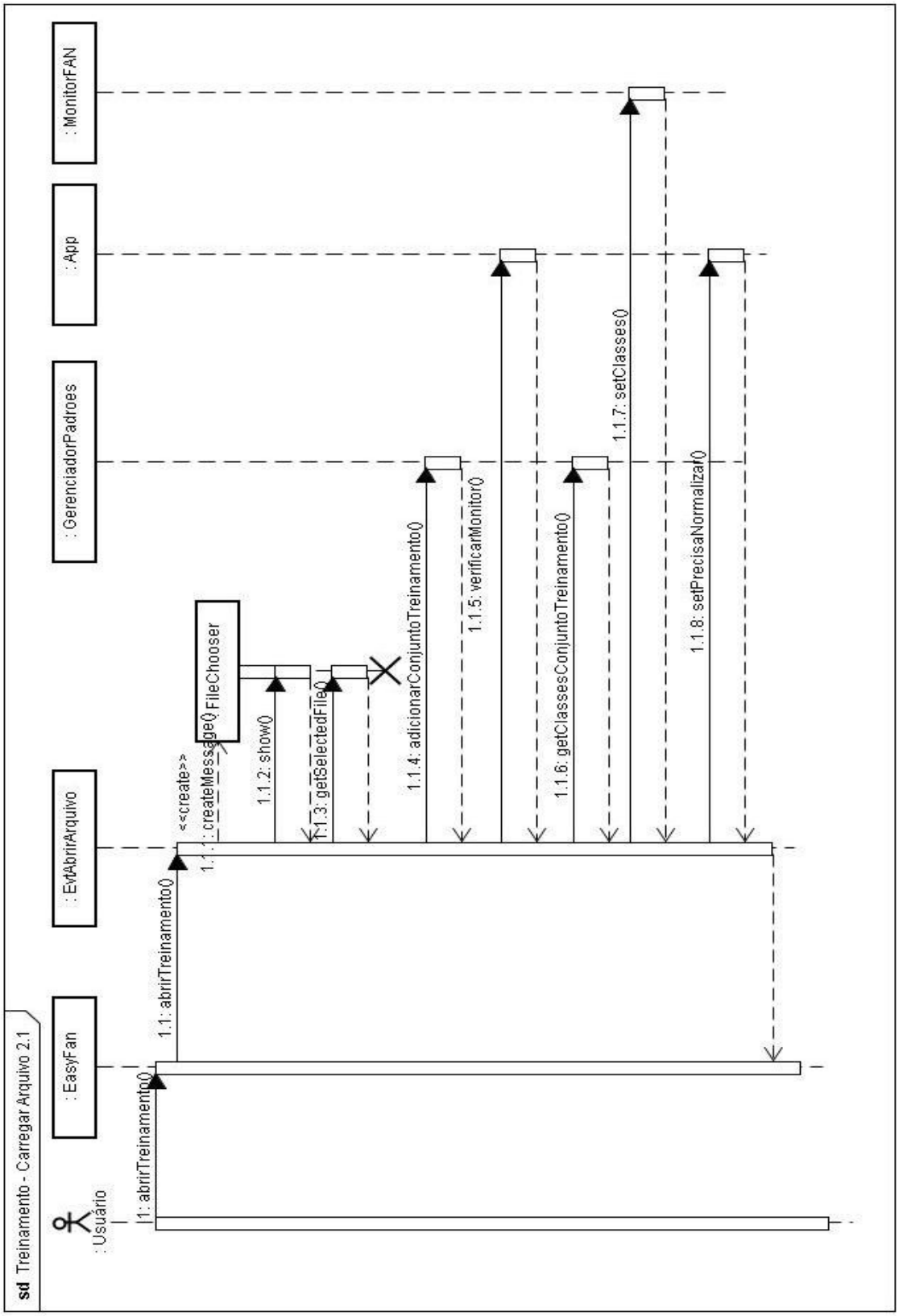

Figura 65 – Diagrama de Seqüência EasyFAN – Carregar Arquivo de Treinamento

Iniciar Treinamento

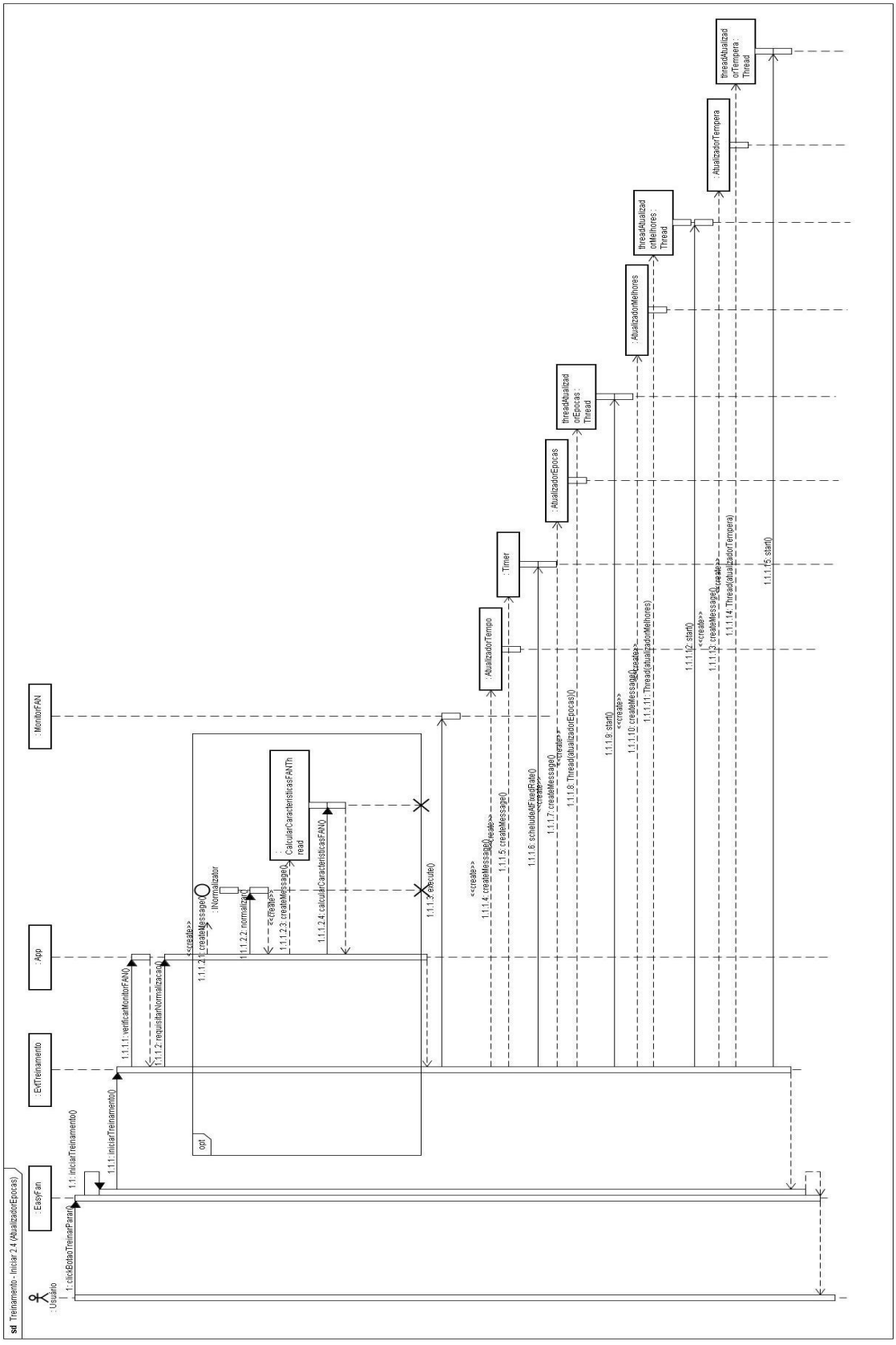

Figura 66 – Diagrama de Seqüência EasyFAN – Iniciar Treinamento

Salvar Conjunto de Treinamento

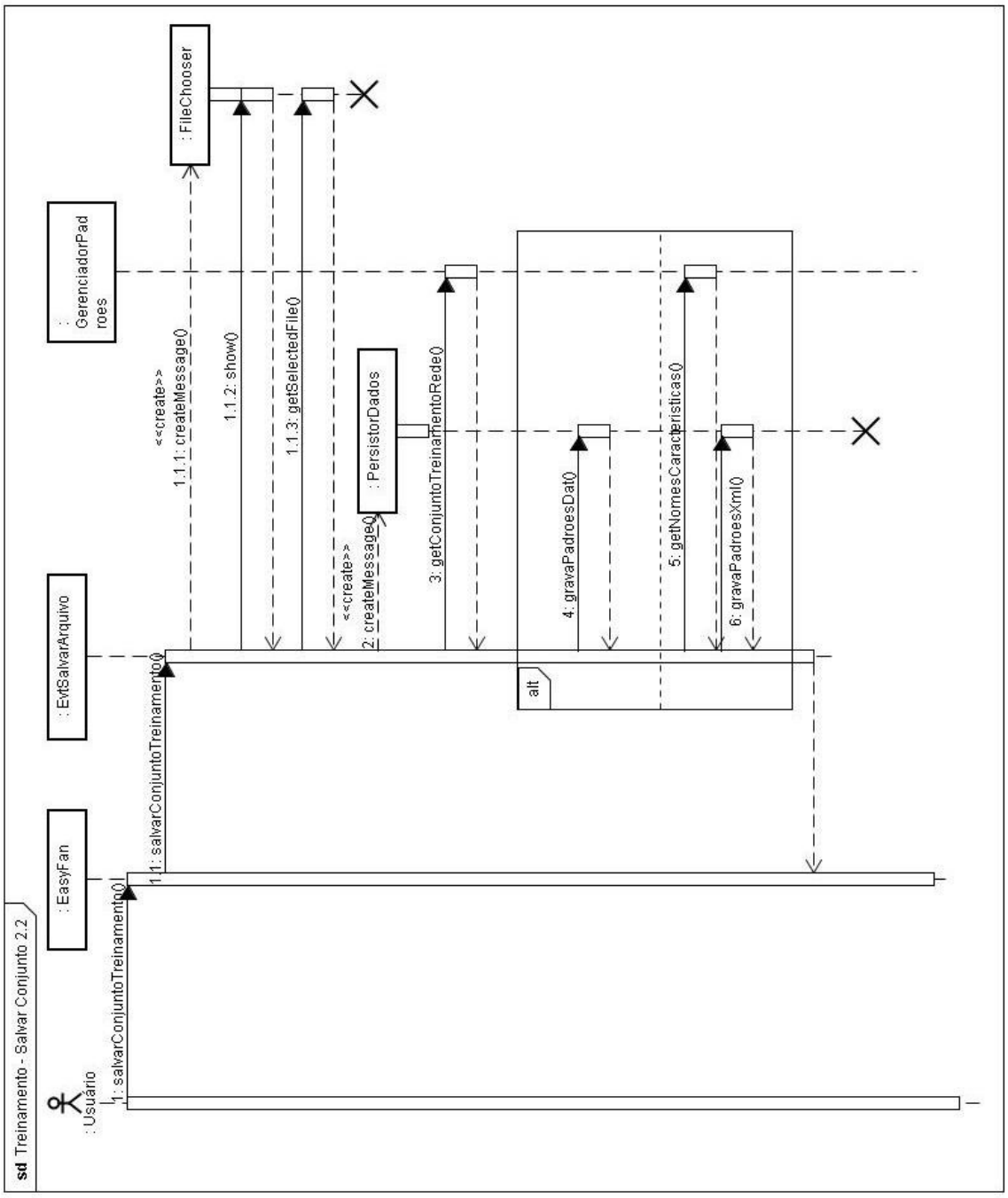

Figura 67 – Diagrama de Seqüência EasyFAN – Salvar Conjunto de Treinamento

Treinamento – Exibir Matriz de Confusão

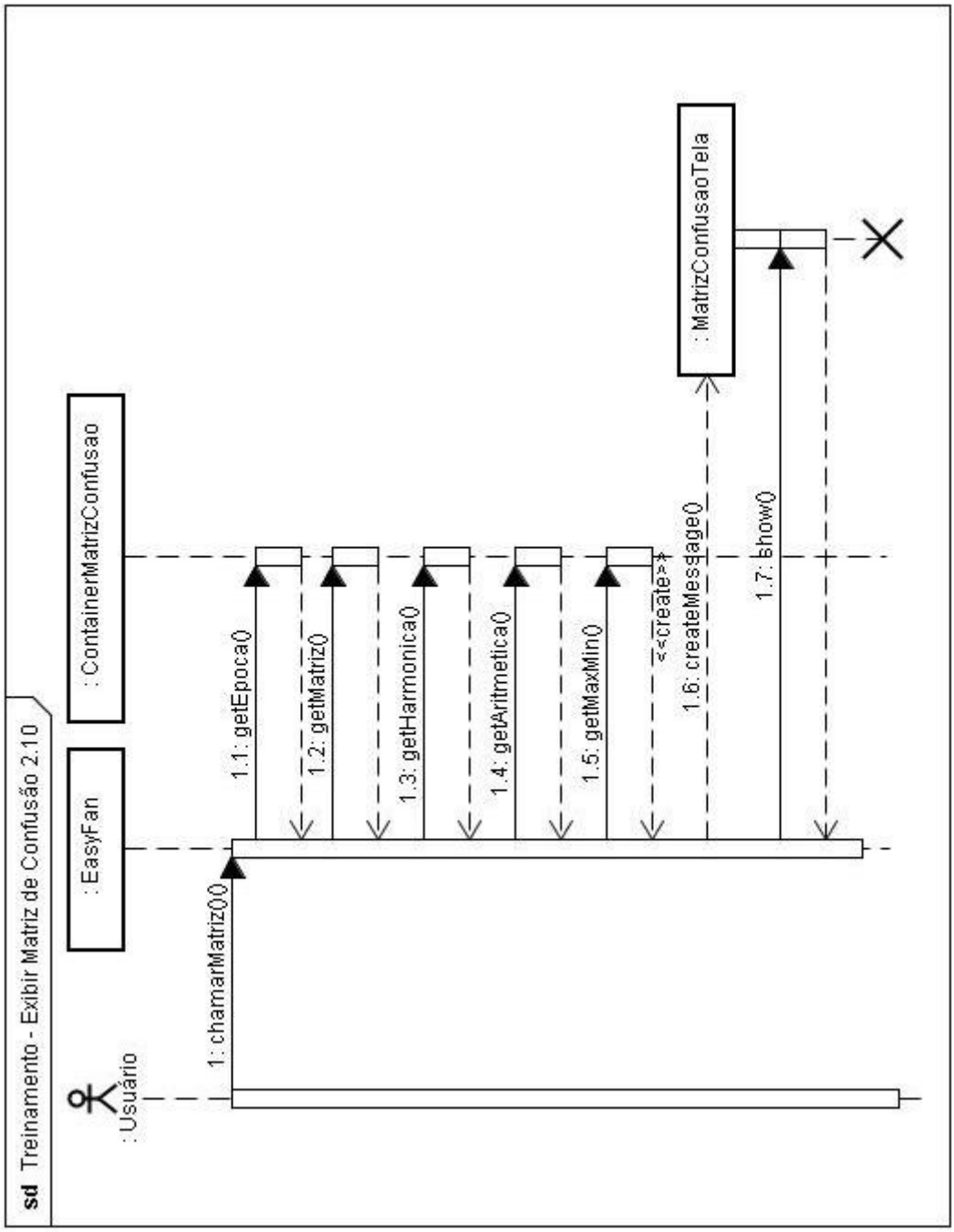

Figura 68 – Diagrama de Seqüência EasyFAN – Exibir Matriz de Confusão do Treinamento

Treinamento – Limpar Conjunto

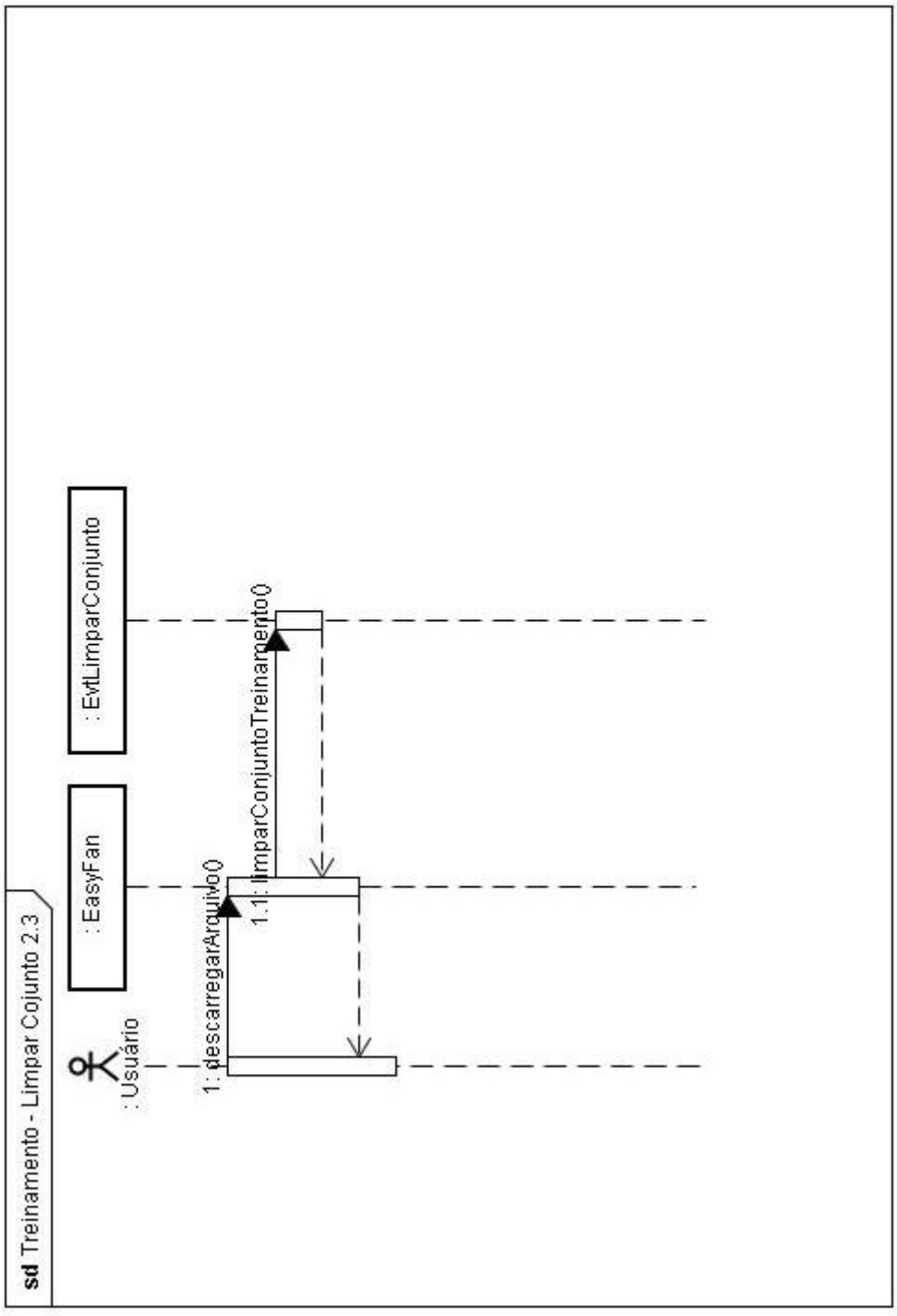

Figura 69 – Diagrama de Seqüência EasyFAN – Limpar Conjunto de Treinamento

Treinamento – Salvar Rede

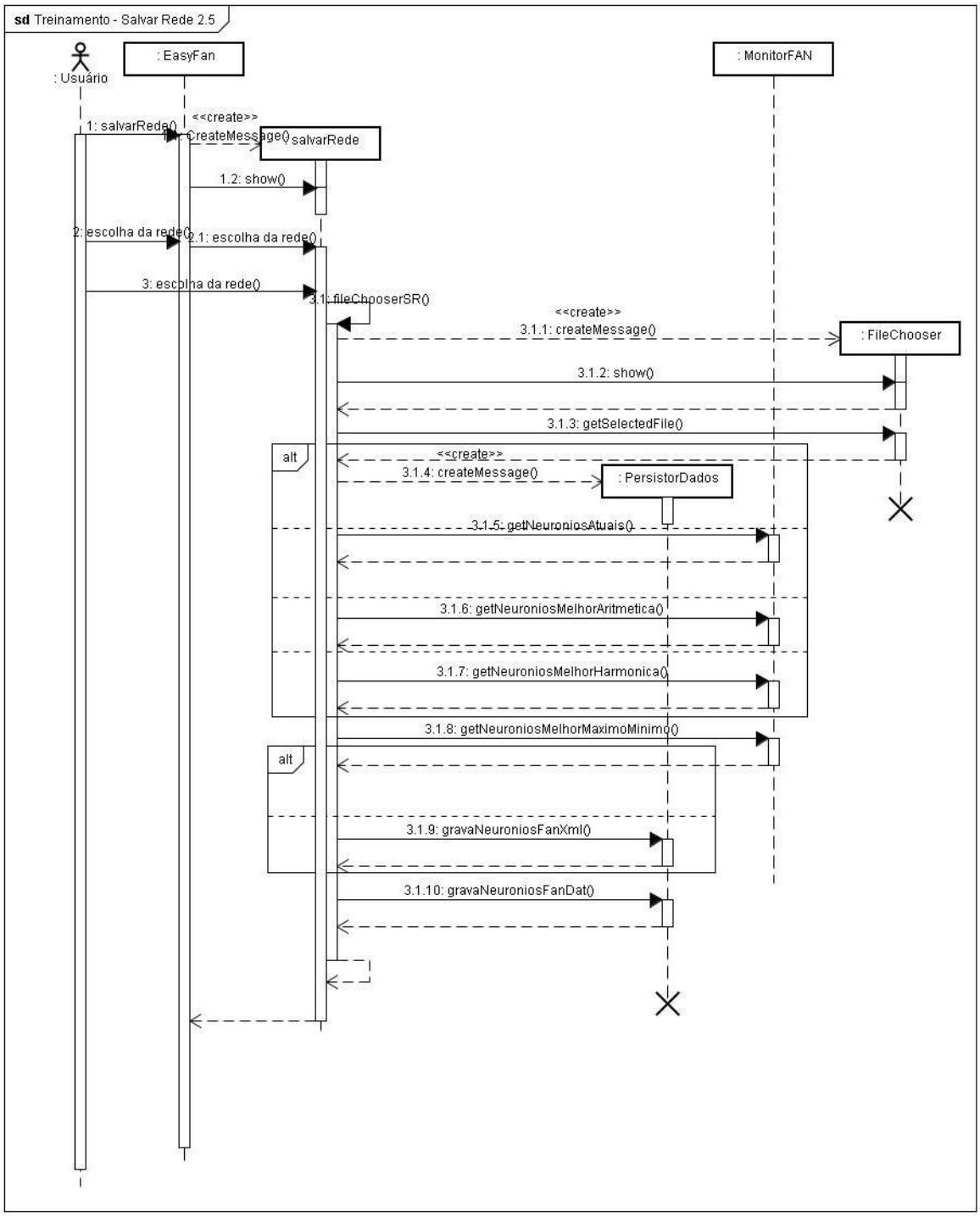

Figura 70 – Diagrama de Seqüência EasyFAN – Salvar Rede

Treinamento – Treinar Épocas

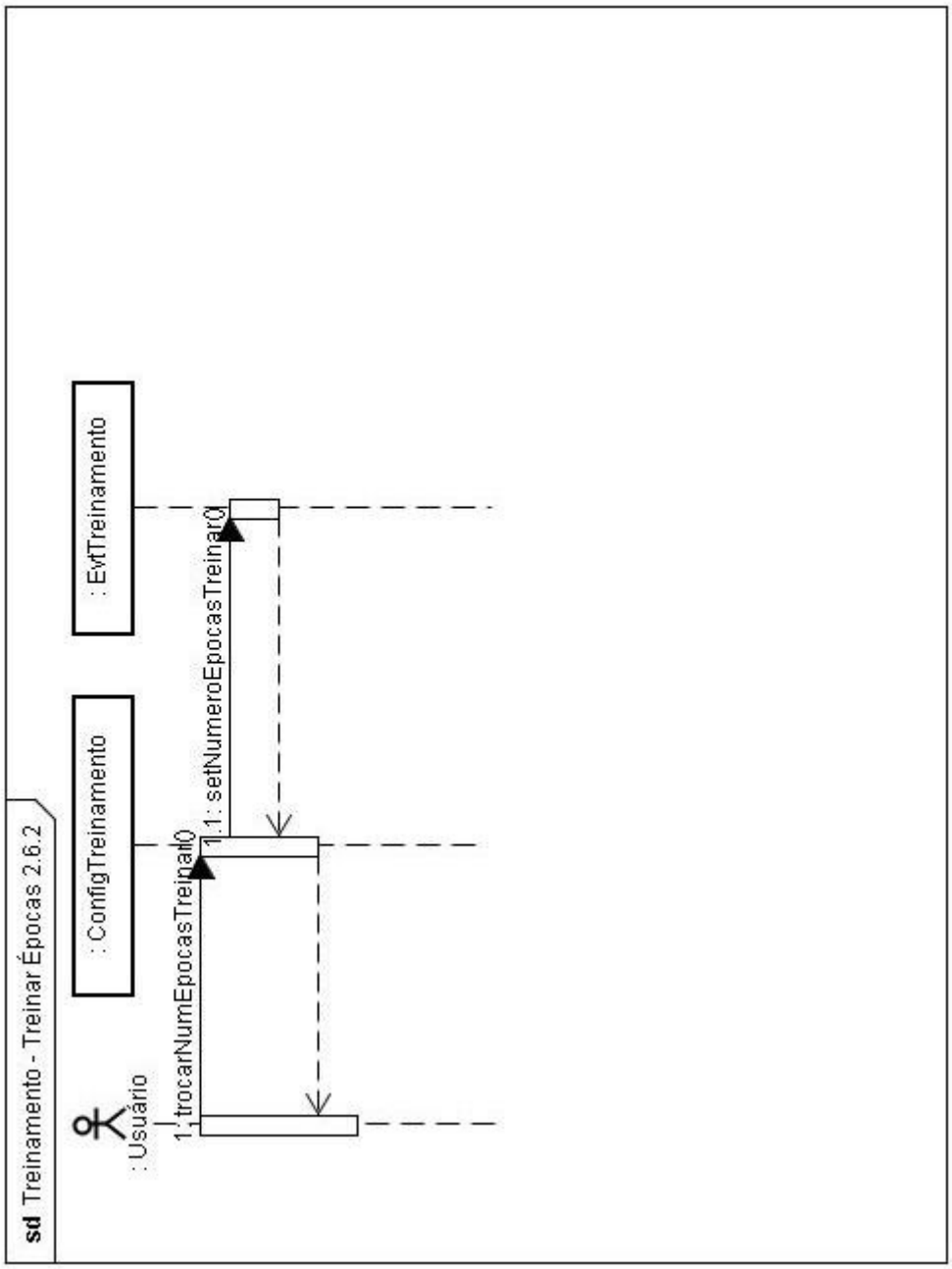

Figura 71 – Diagrama de Seqüência EasyFAN – Treinar Épocas

Treinamento – Treinar Sempre

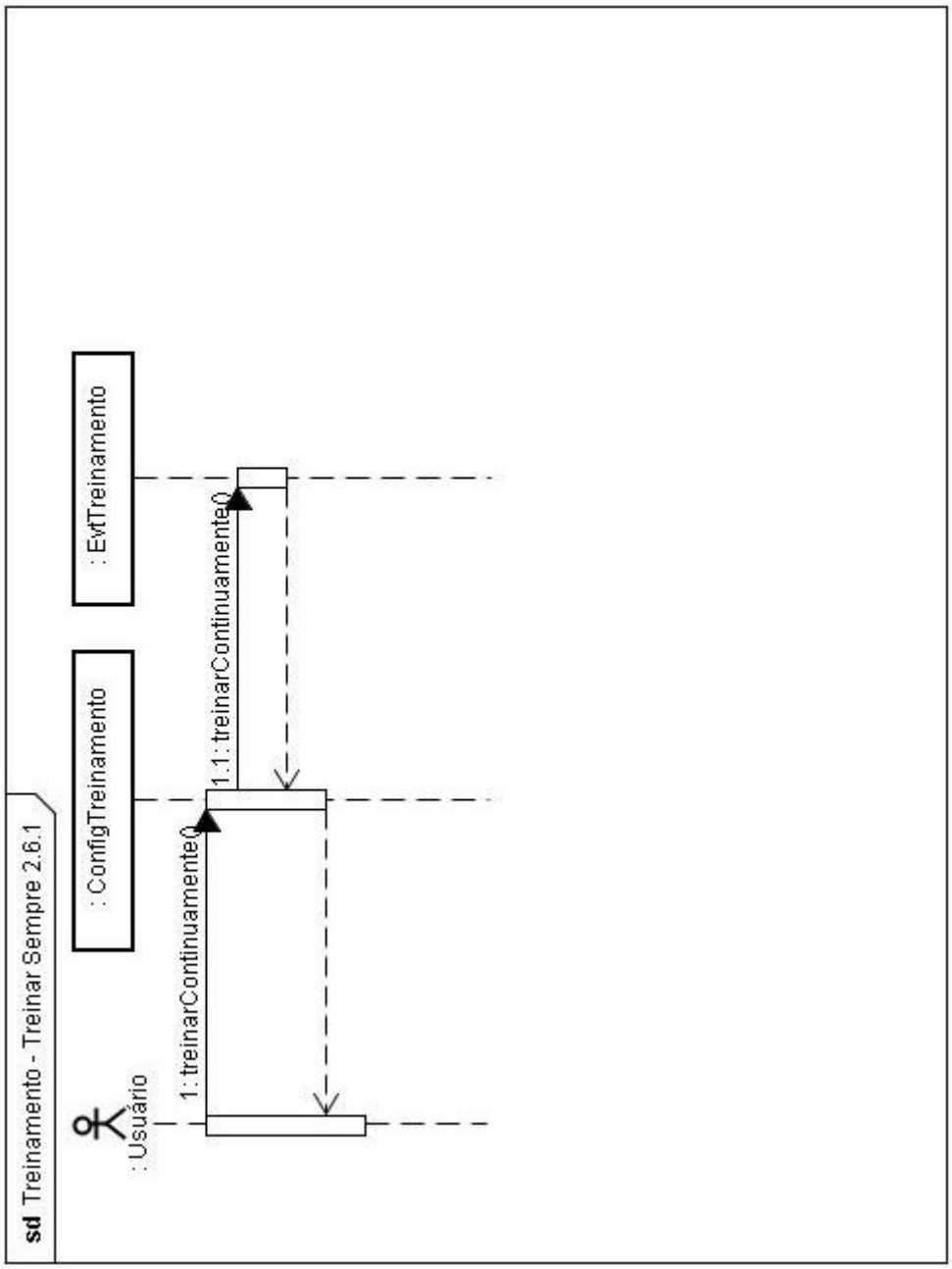

Figura 72 – Diagrama de Seqüência EasyFAN – Treinar Sempre

Treinamento – Treinar Tempo

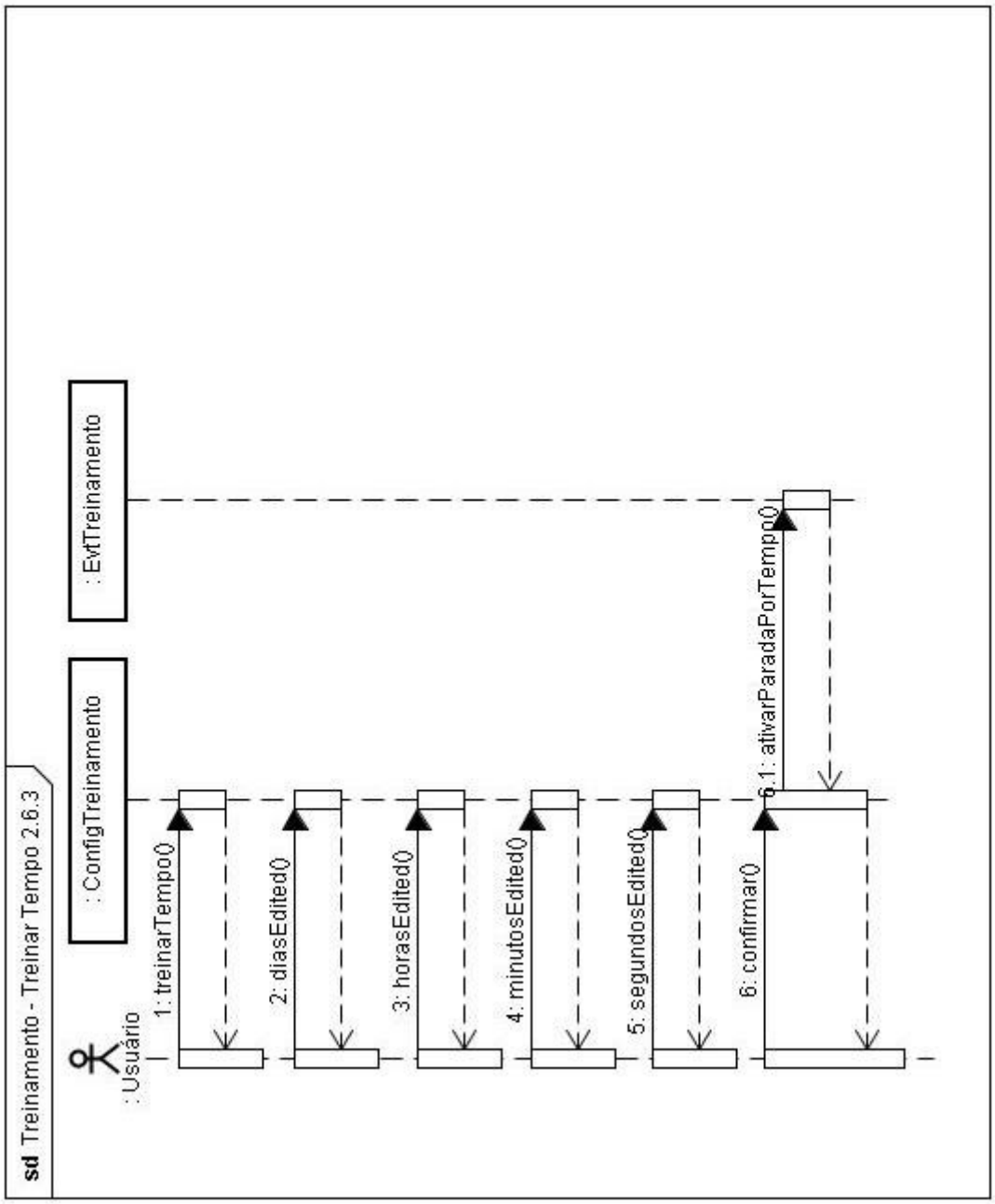

Figura 73 – Diagrama de Seqüência EasyFAN – Treinar Tempo

5.3.3. Teste

Teste

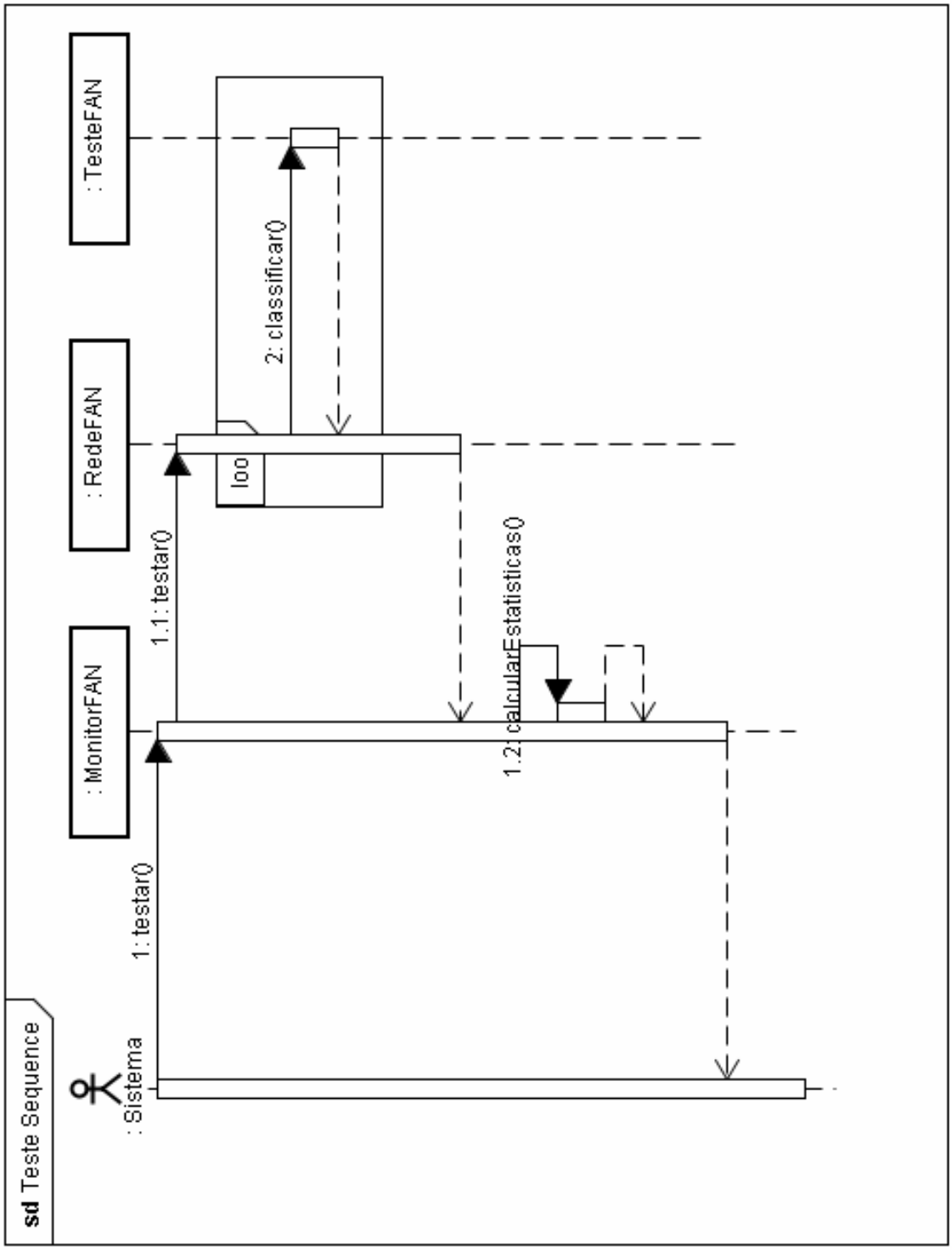

Figura 74 – Diagrama de Seqüência EasyFAN – Teste

Teste – Exibir Matriz de Confusão

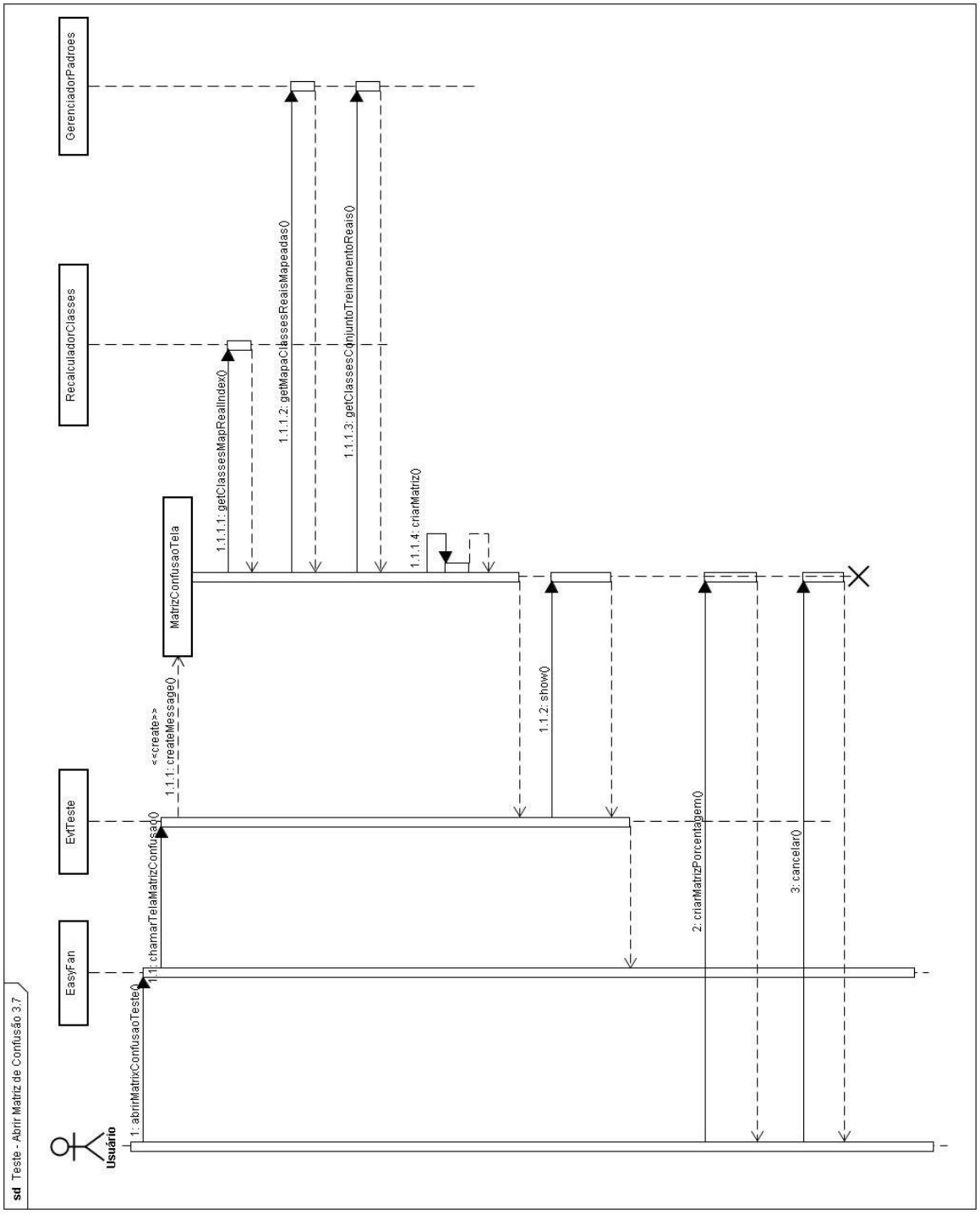

Figura 75 – Diagrama de Seqüência EasyFAN – Exibir Matriz Confusão do Teste

Teste – Carregar Conjunto de Teste

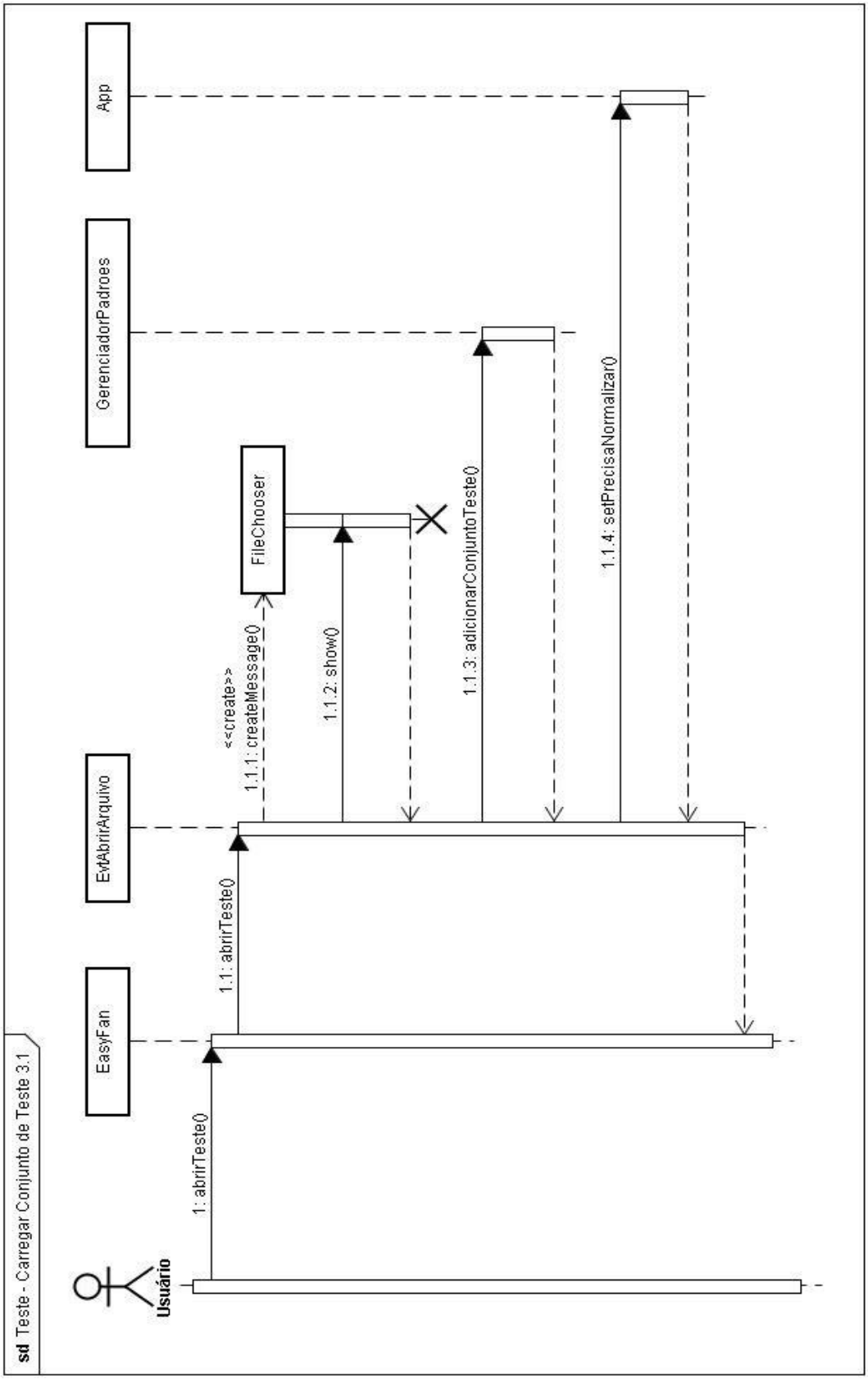

Figura 76 – Diagrama de Seqüência EasyFAN – Carregar Conjunto de Teste

Teste – Salvar Conjunto de Teste

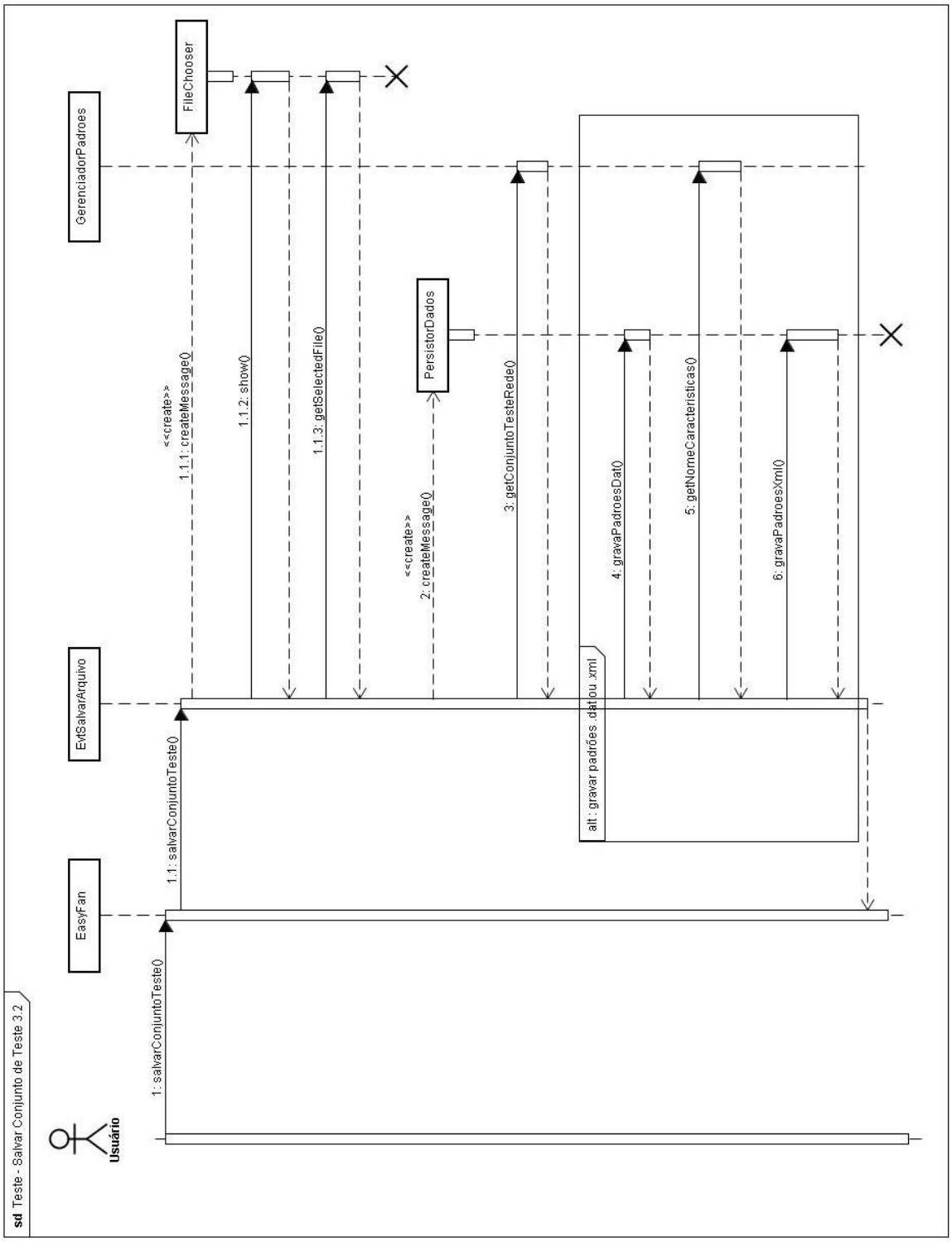

Figura 77 – Diagrama de Seqüência EasyFAN – Salvar Conjunto de Teste

Teste – Testar Conjunto de Teste

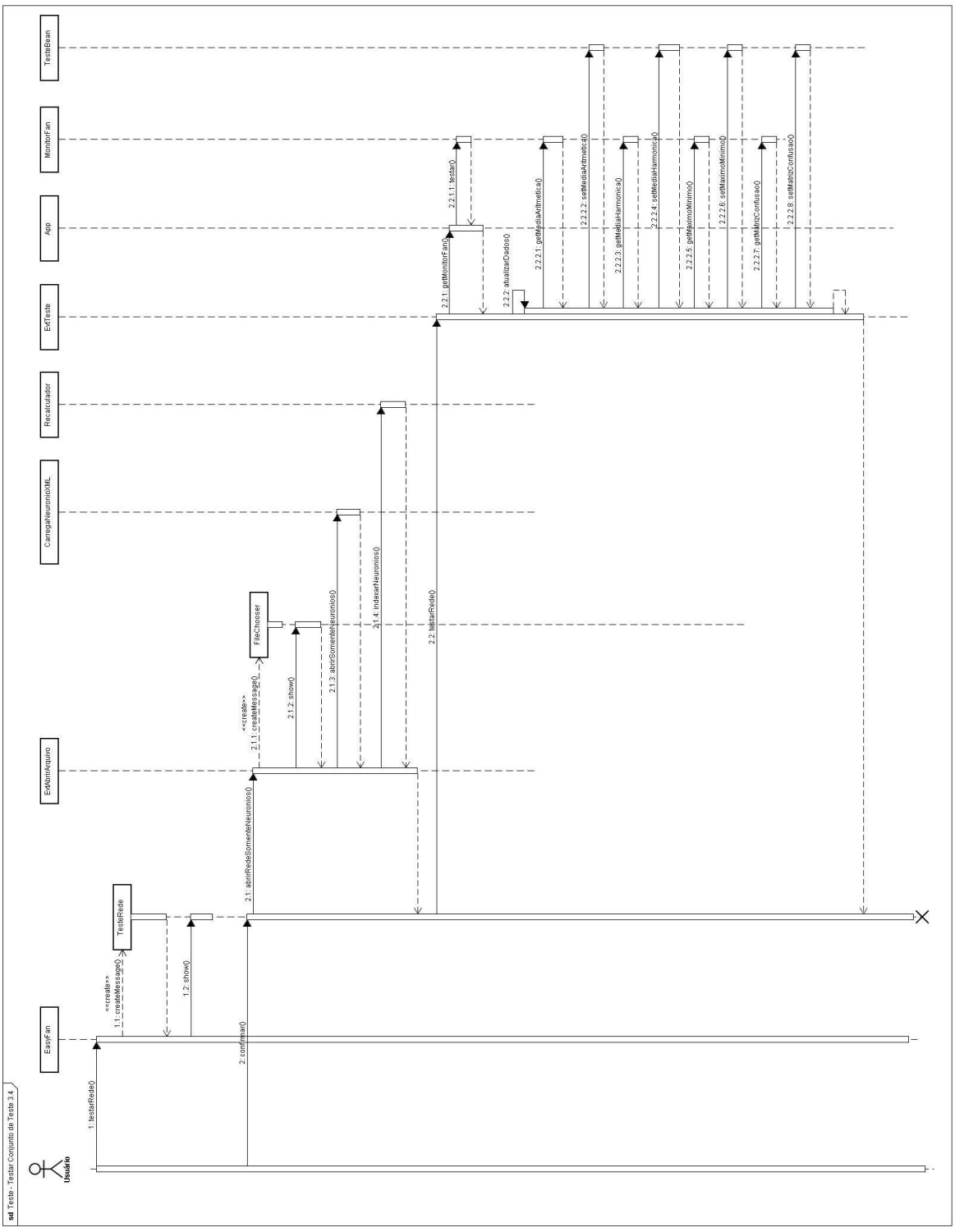

Figura 78 – Diagrama de Seqüência EasyFAN – Testar Conjunto de Teste

5.3.4. Validação

Validação

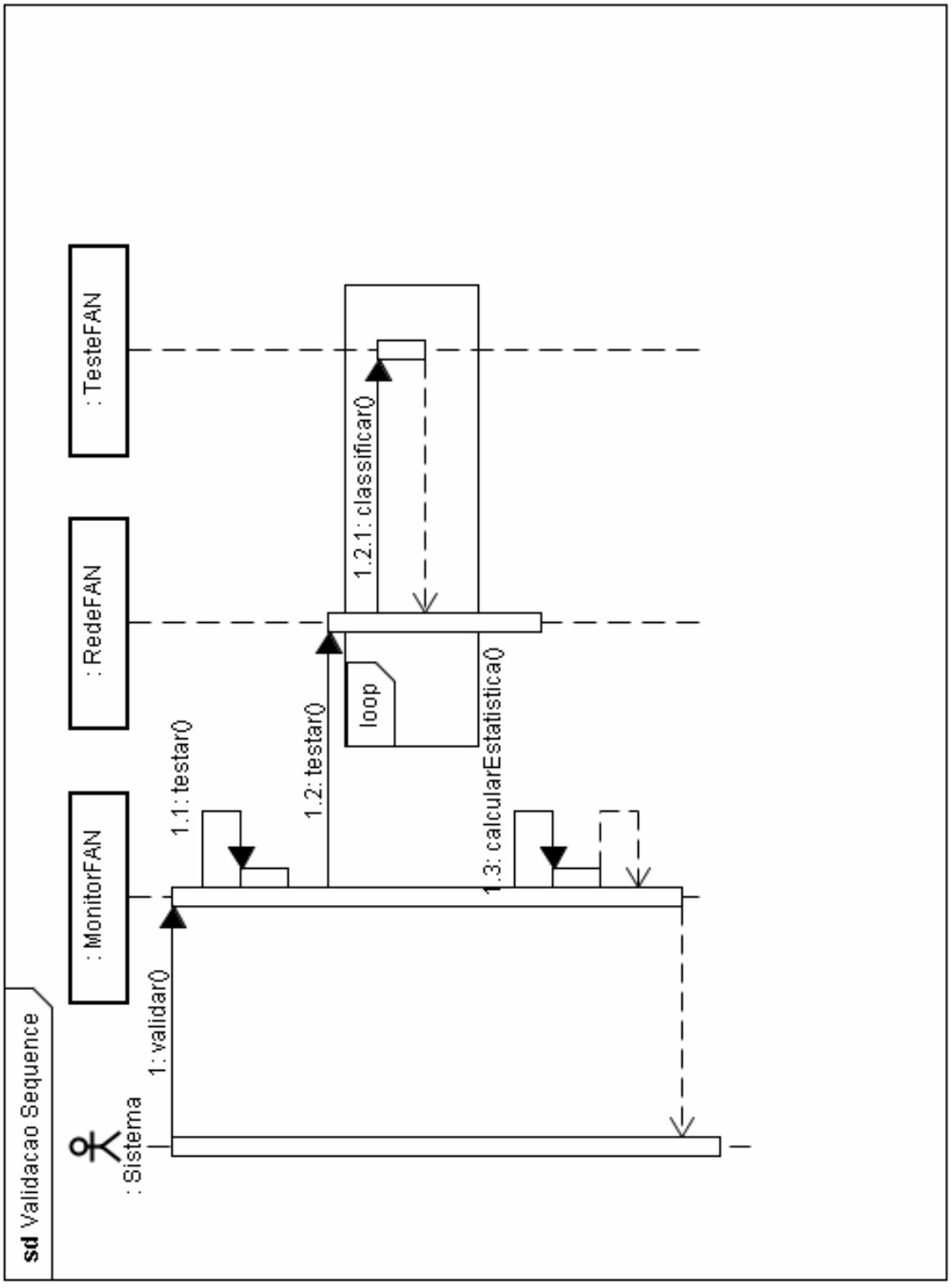

Figura 79 – Diagrama de Seqüência EasyFAN – Validação

Validação – Adicionar Conjunto

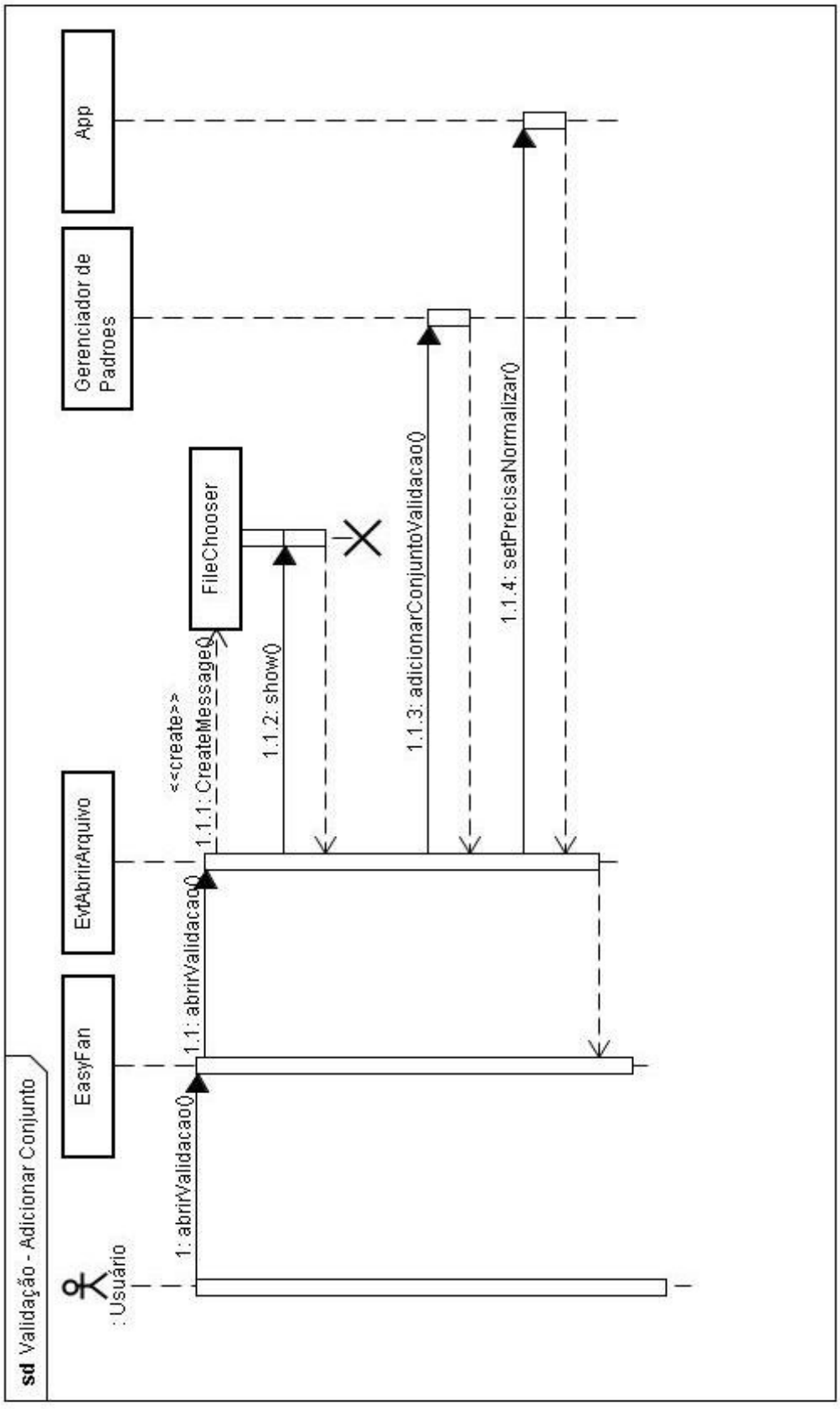

Figura 80 – Diagrama de Seqüência EasyFAN – Adicionar Conjunto de Validação

Validação – Configurar Validação

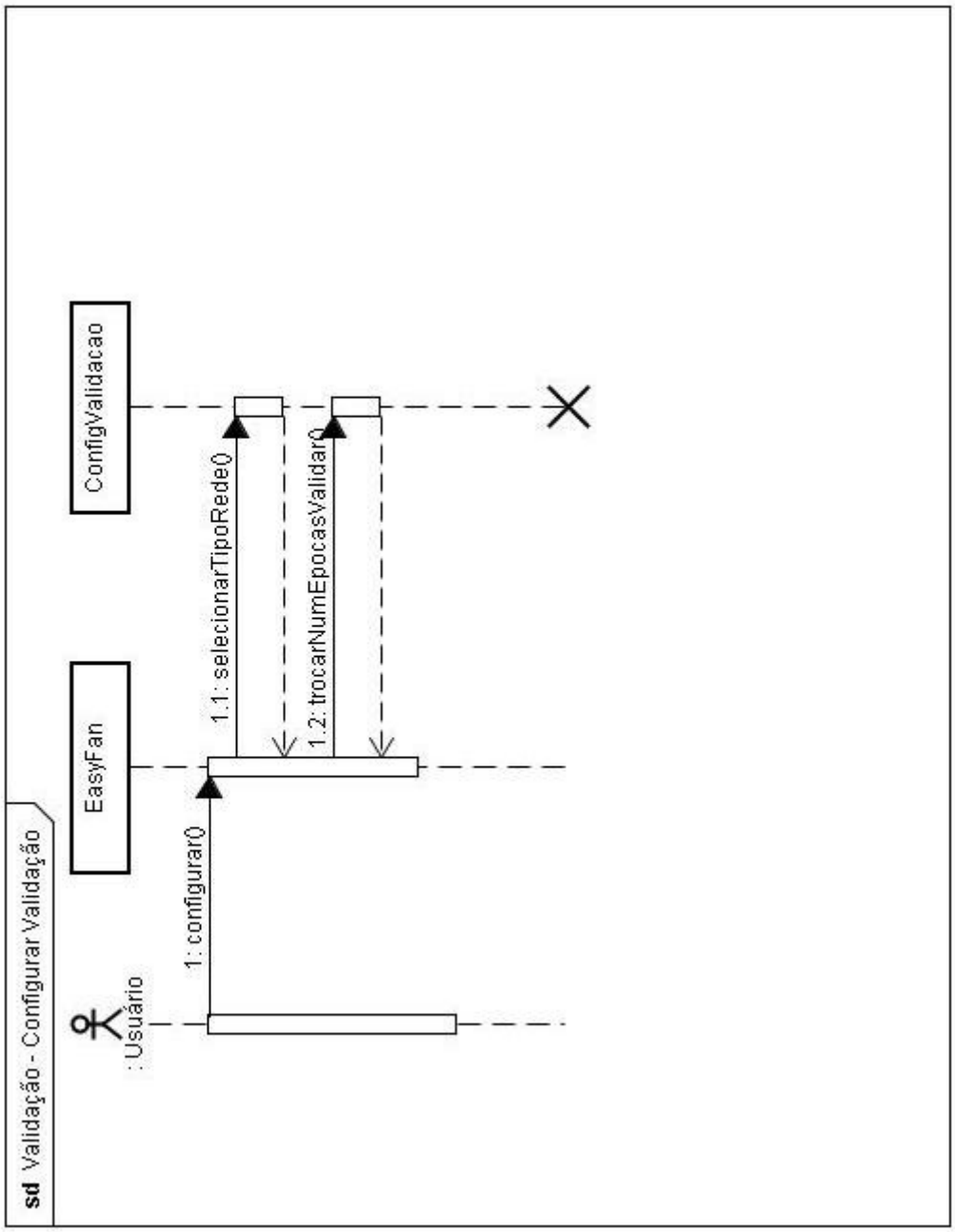

Figura 81 – Diagrama de Seqüência EasyFAN – Configurar Validação

Validação – Limpar Conjunto

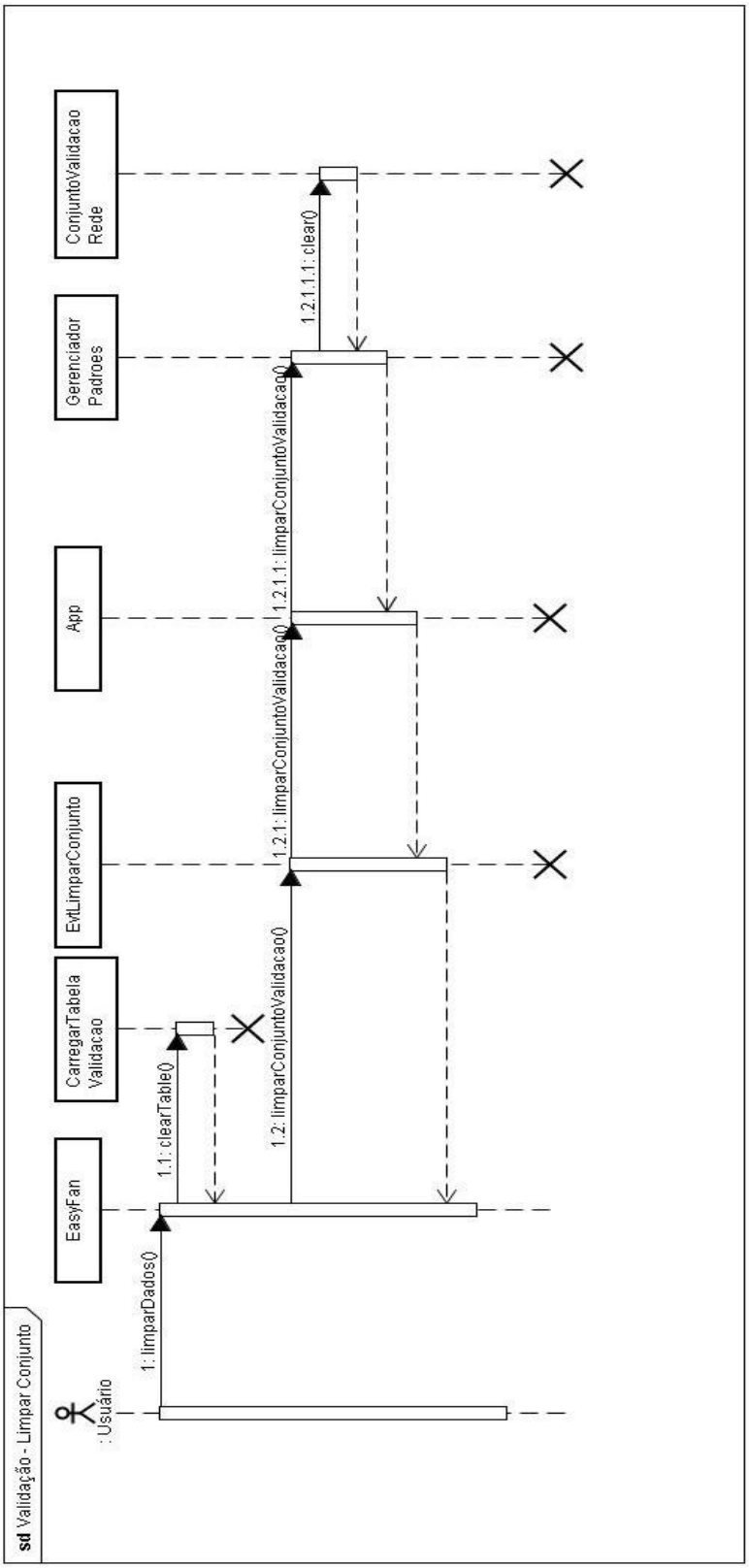

Figura 82 – Diagrama de Seqüência EasyFAN – Limpar Conjunto de Validação

Validação – Exibir Matriz de Confusão

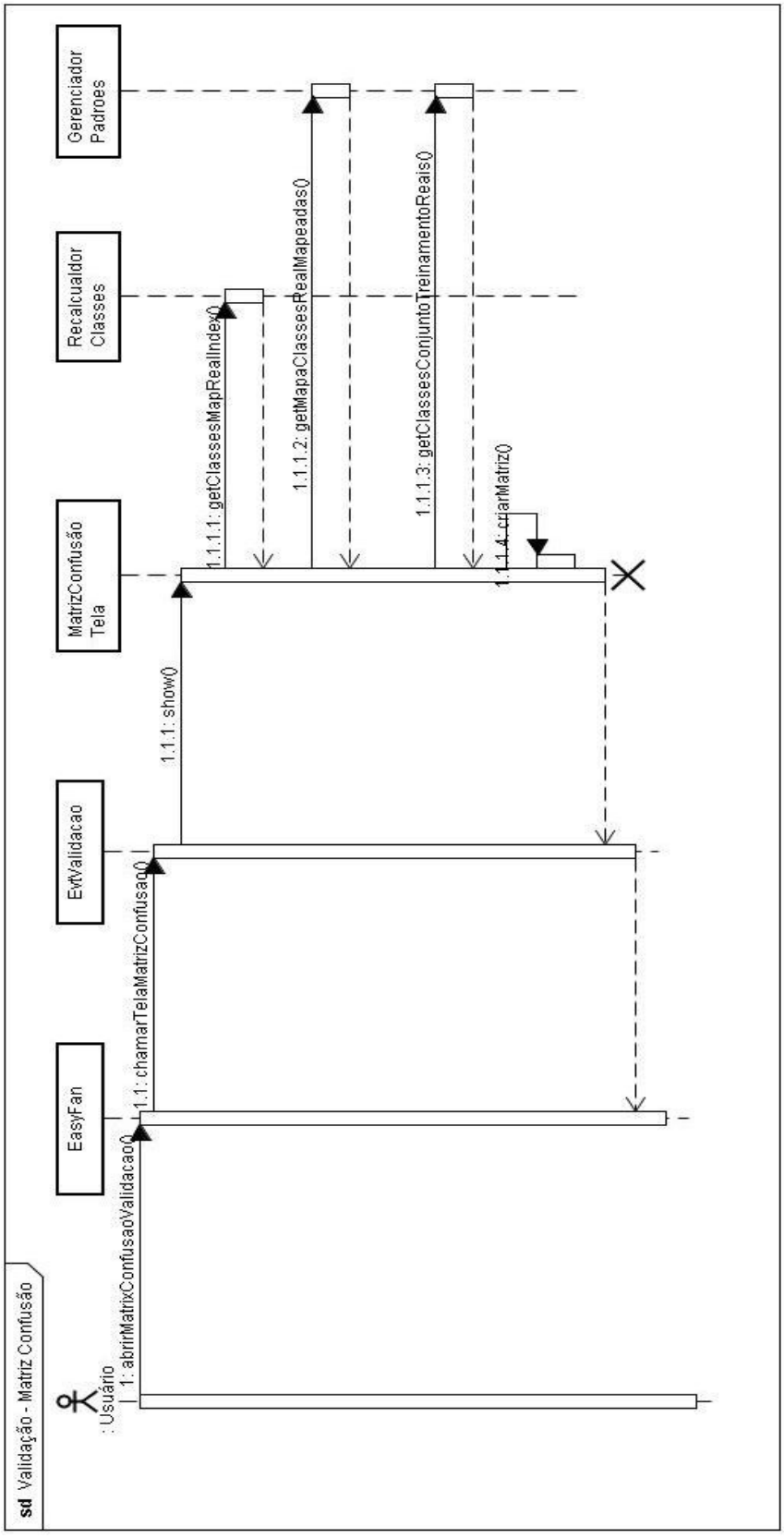

Figura 83 – Diagrama de Seqüência EasyFAN – Exibir Matriz de Confusão Validação

Validação – Salvar Conjunto

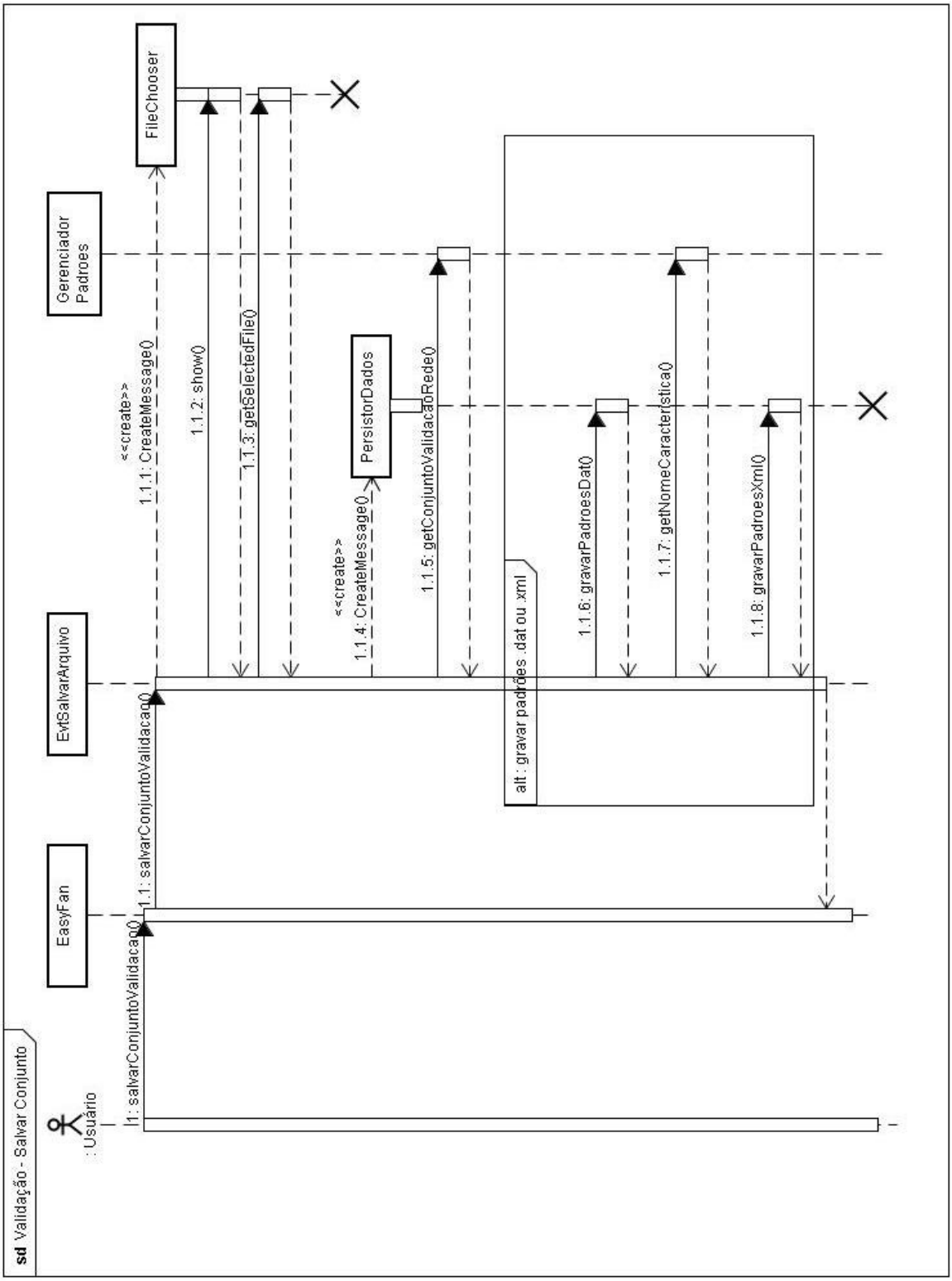

Figura 84 – Diagrama de Seqüência EasyFAN – Salvar Conjunto de Validação

Validação – Validar Rede

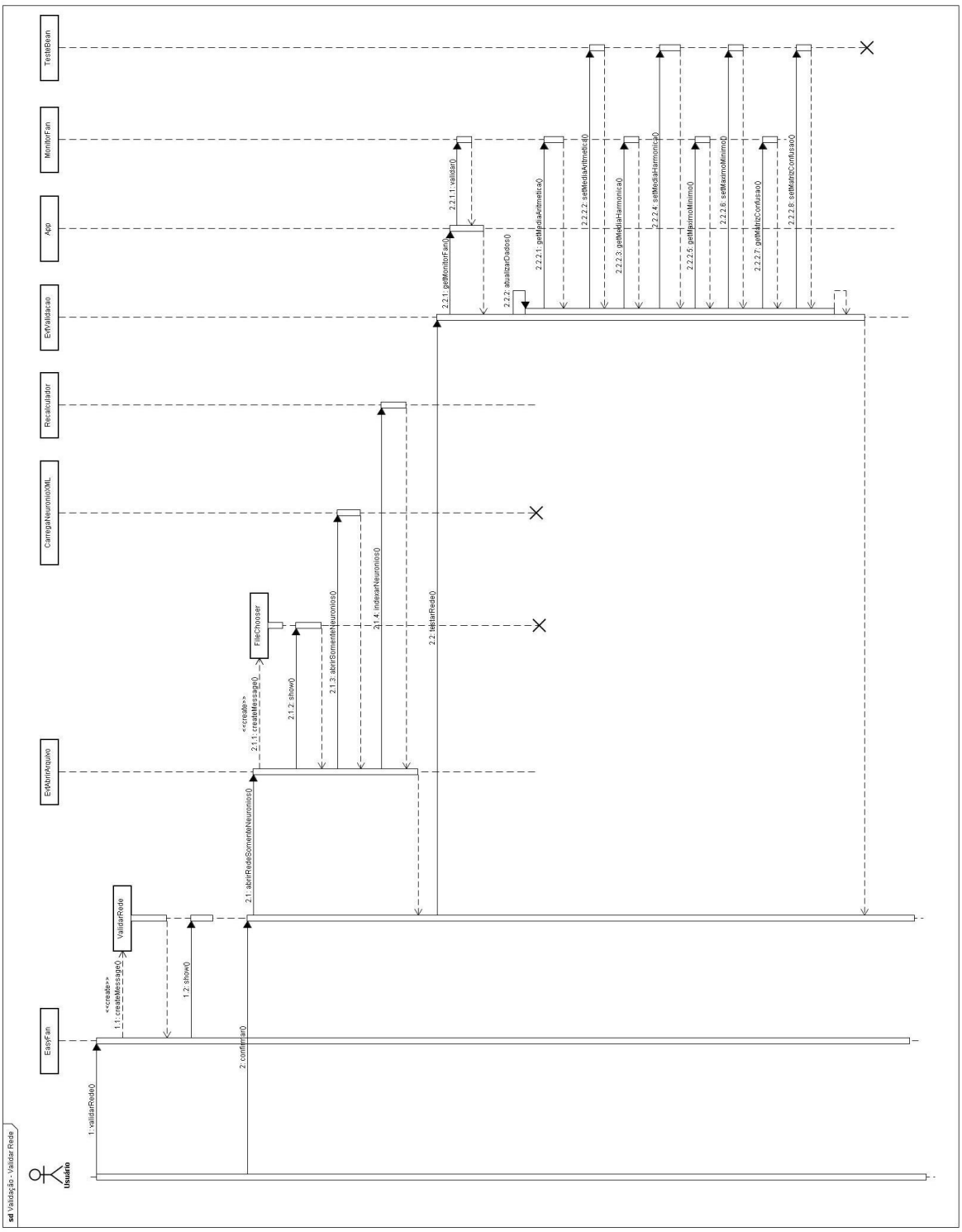

Figura 85 – Diagrama de Seqüência EasyFAN – Validar Rede

5.3.5. Classificação

Classificação

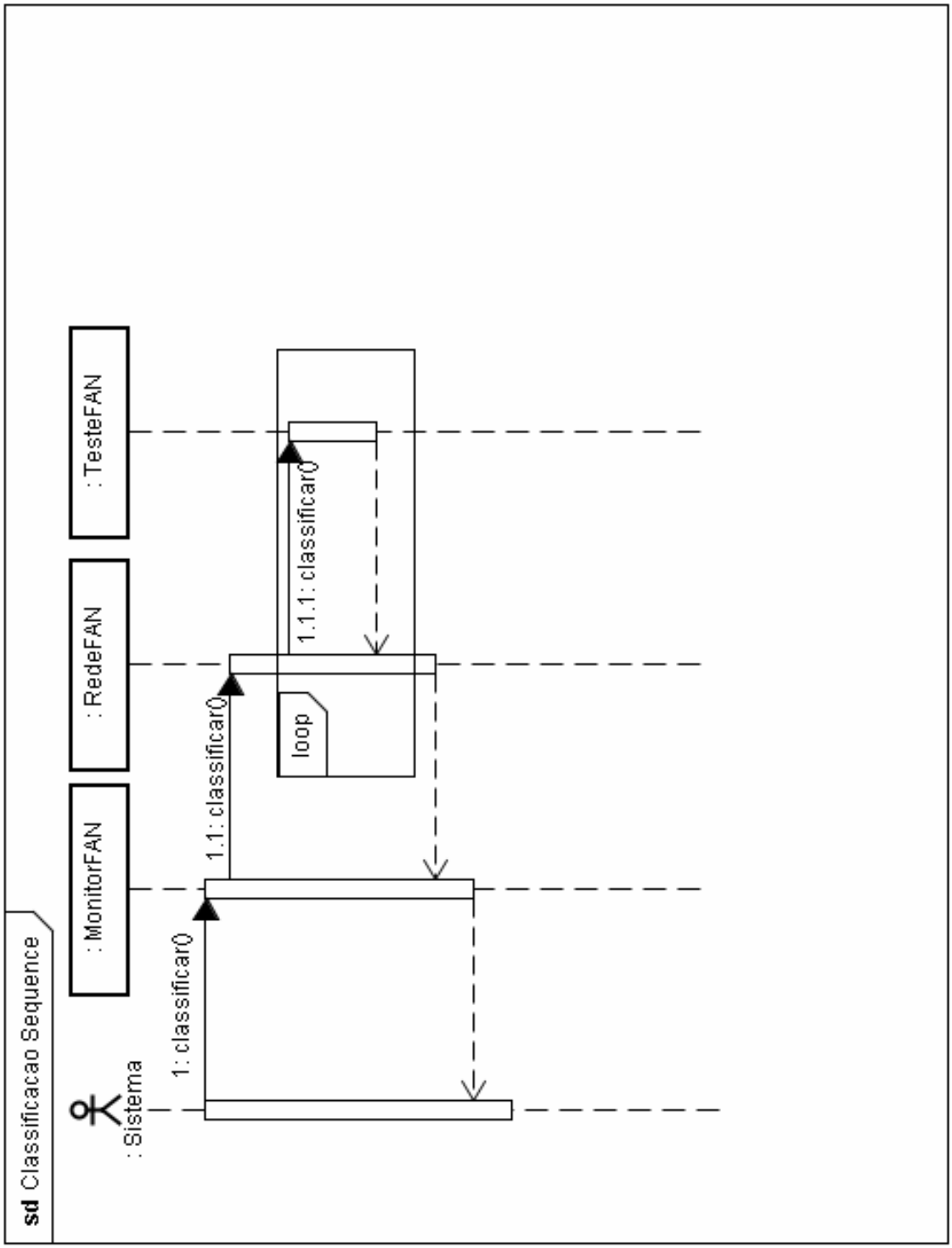

Figura 86 – Diagrama de Seqüência EasyFAN – Classificação

Classificação – Iniciar Classificação

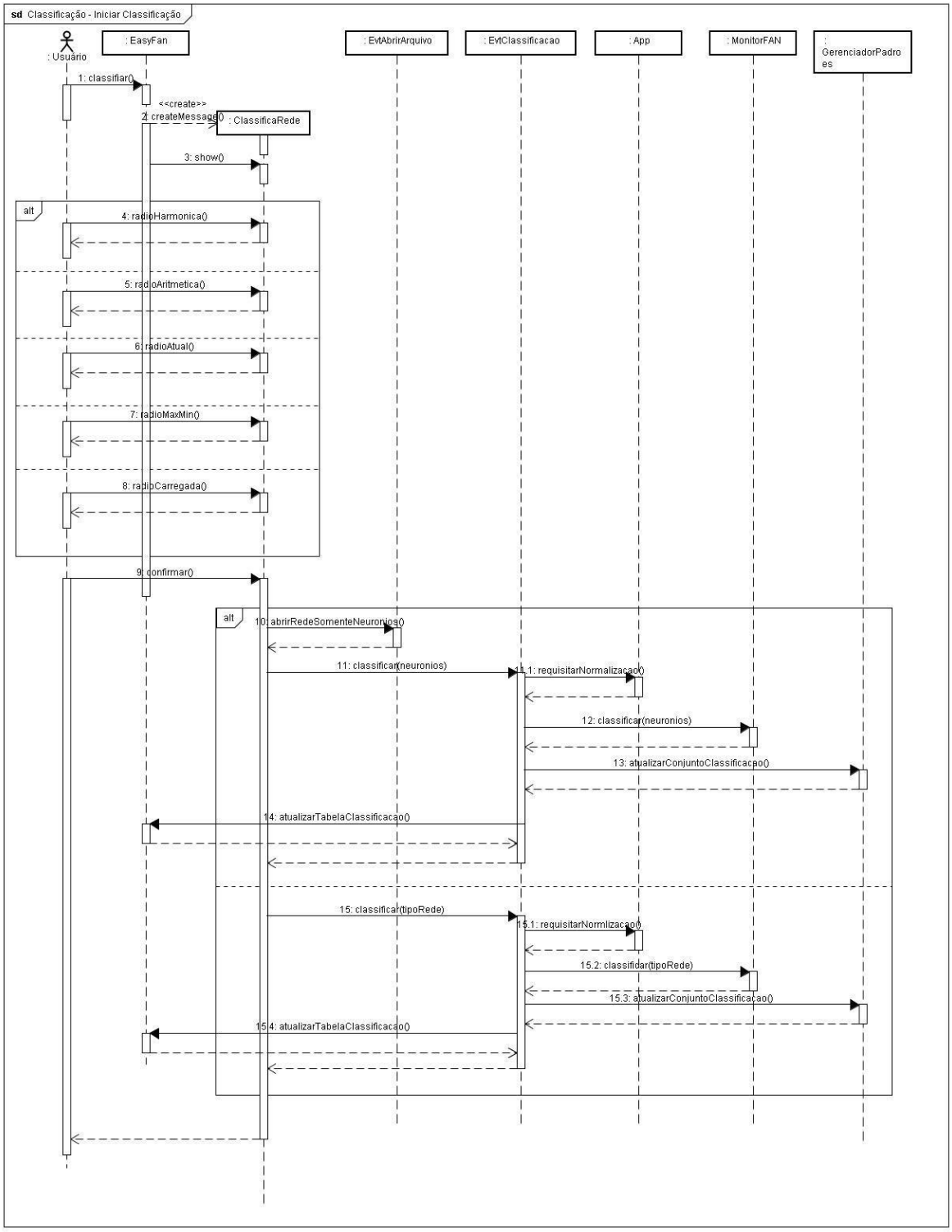

Figura 87 – Diagrama de Seqüência EasyFAN – Iniciar Classificação

Classificação – Abrir Arquivo

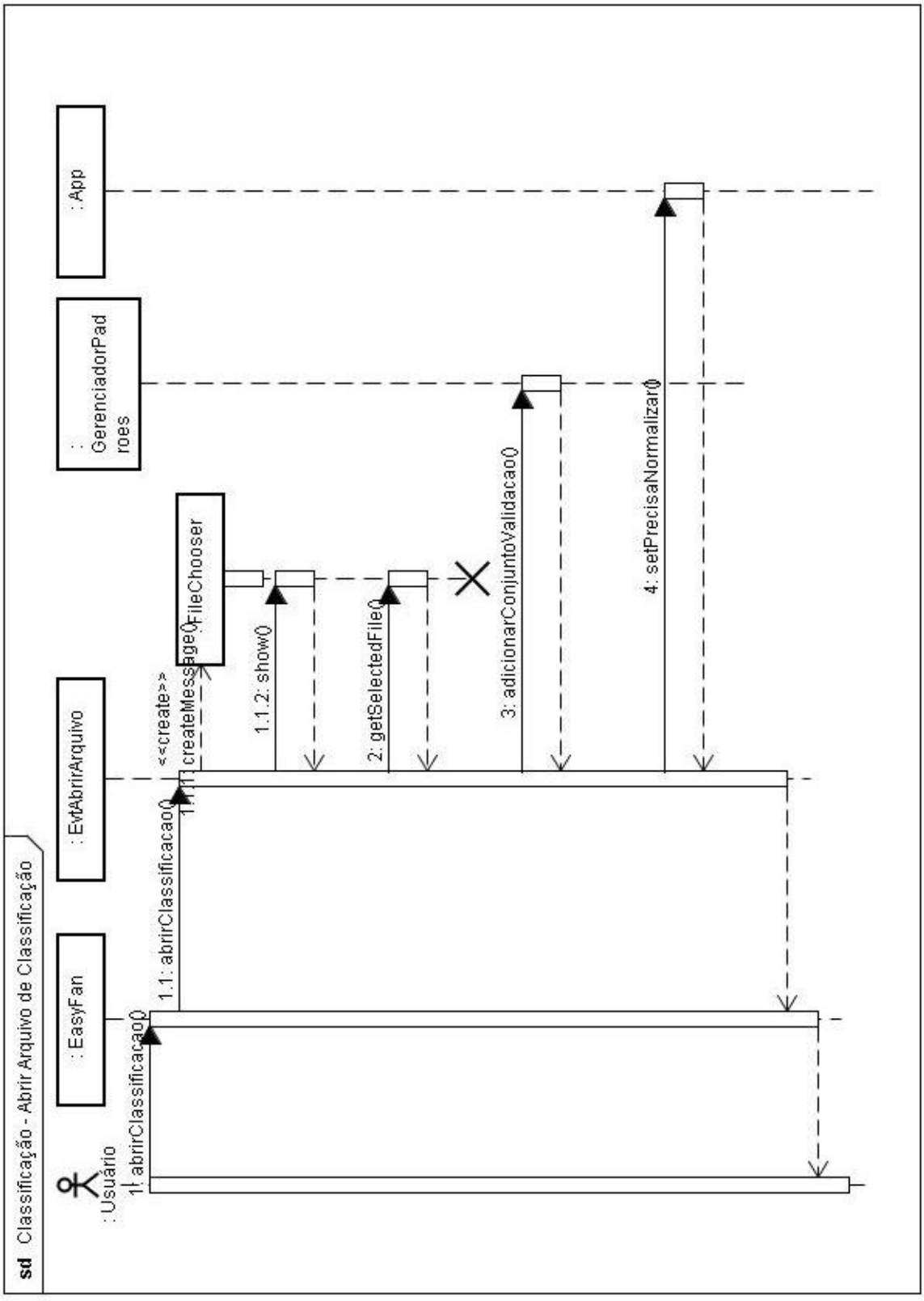

Figura 88 – Diagrama de Seqüência EasyFAN – Abrir Conjunto de Classificação

Classificação – Salvar Conjunto

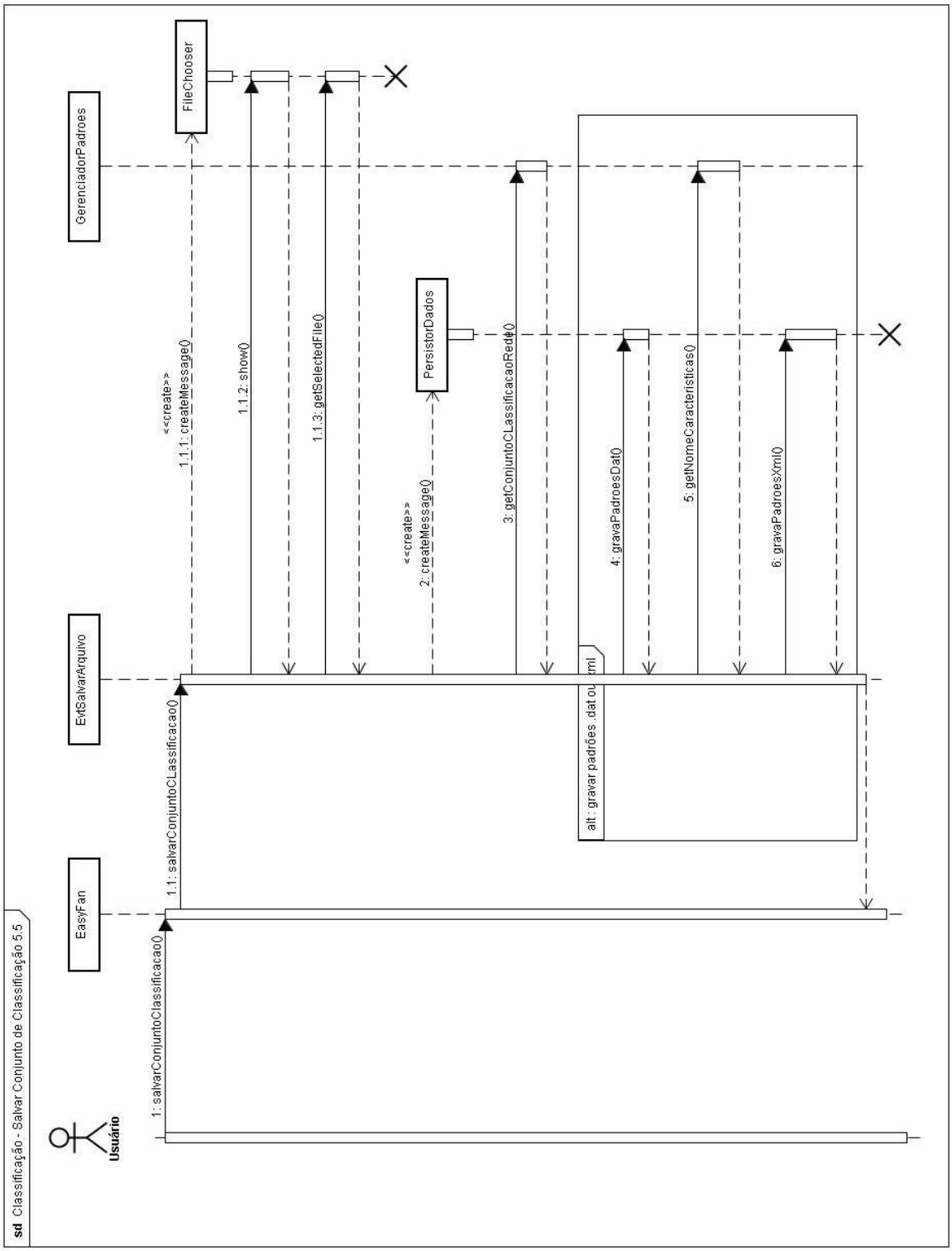

Figura 89 – Diagrama de Seqüência EasyFAN – Salvar Conjunto de Classificação

5.3.6. Wizard

Iniciar Wizard

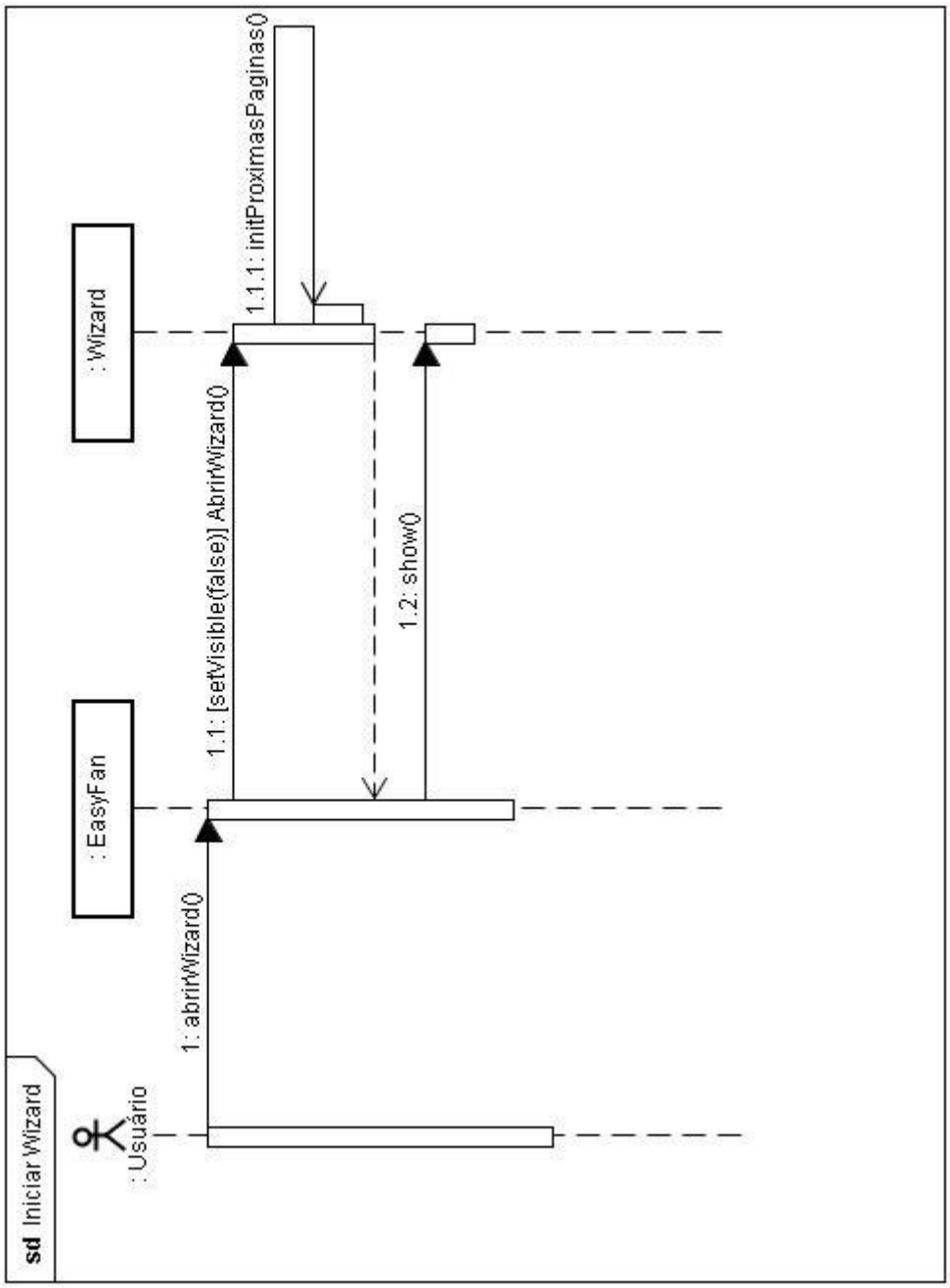

Figura 90 – Diagrama de Seqüência EasyFAN – Iniciar Wizard

Wizard – Carregar Conjunto

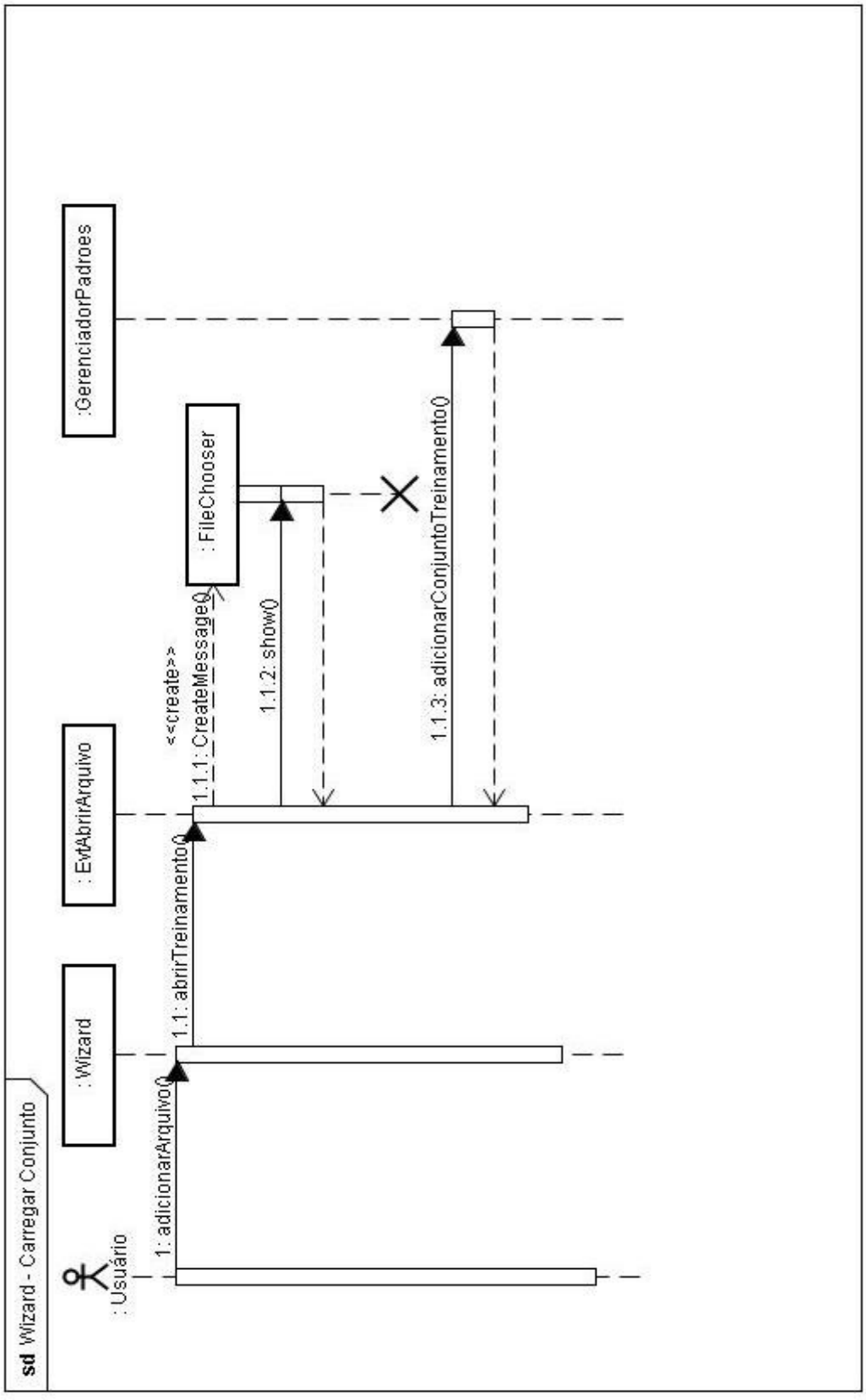

Figura 91 – Diagrama de Seqüência EasyFAN – Carregar Conjunto Wizard

Wizard – Dividir Conjunto de Arquivos

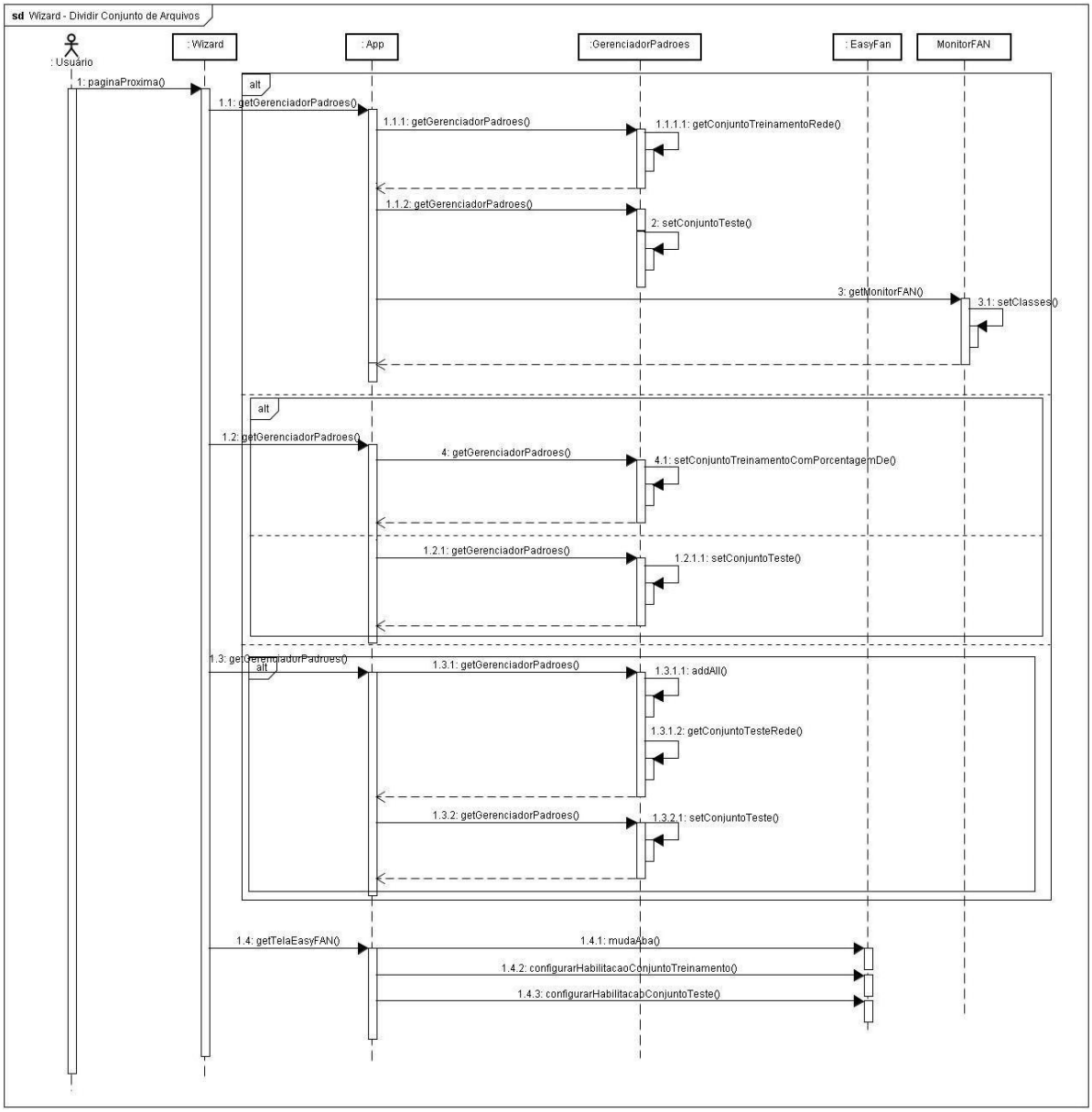

Figura 92 – Diagrama de Seqüência EasyFAN – Dividir Conjunto Wizard

Wizard – Iniciar Treinamento

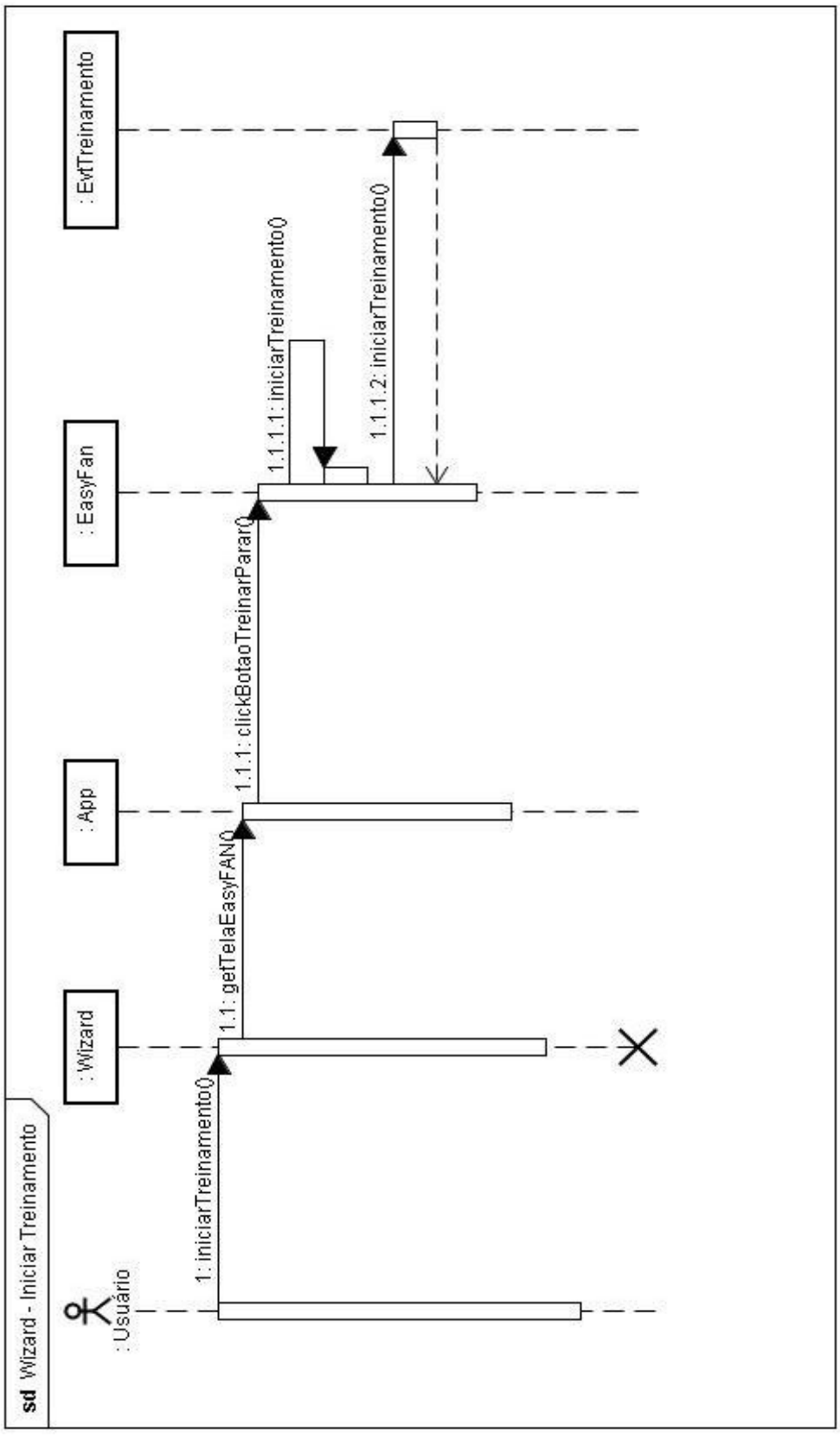

Figura 93 – Diagrama de Seqüência EasyFAN – Iniciar Treinamento Wizard

Wizard – Fechar Wizard

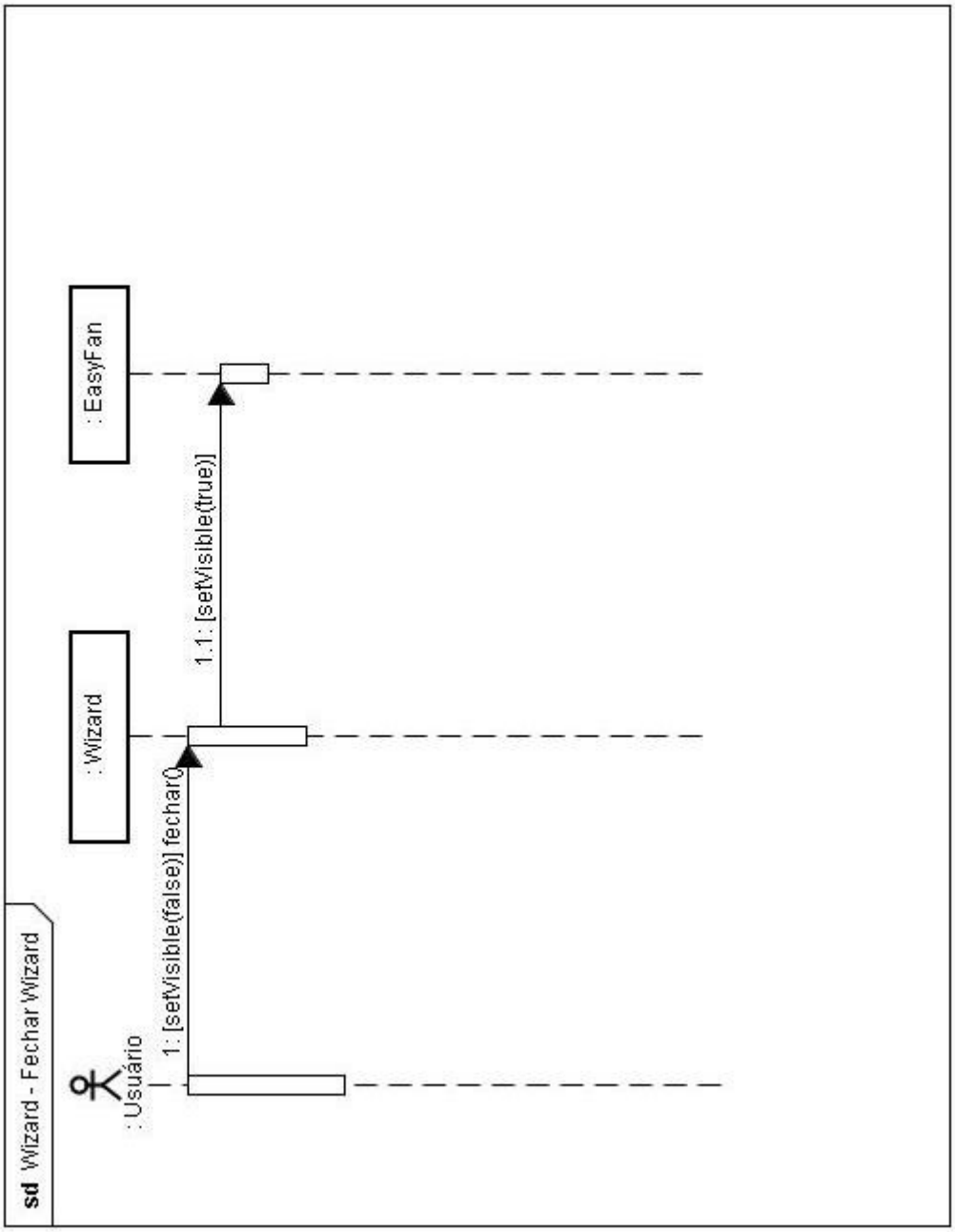

Figura 94 – Diagrama de Seqüência EasyFAN – Fechar Wizard

Wizard – Cancelar Wizard

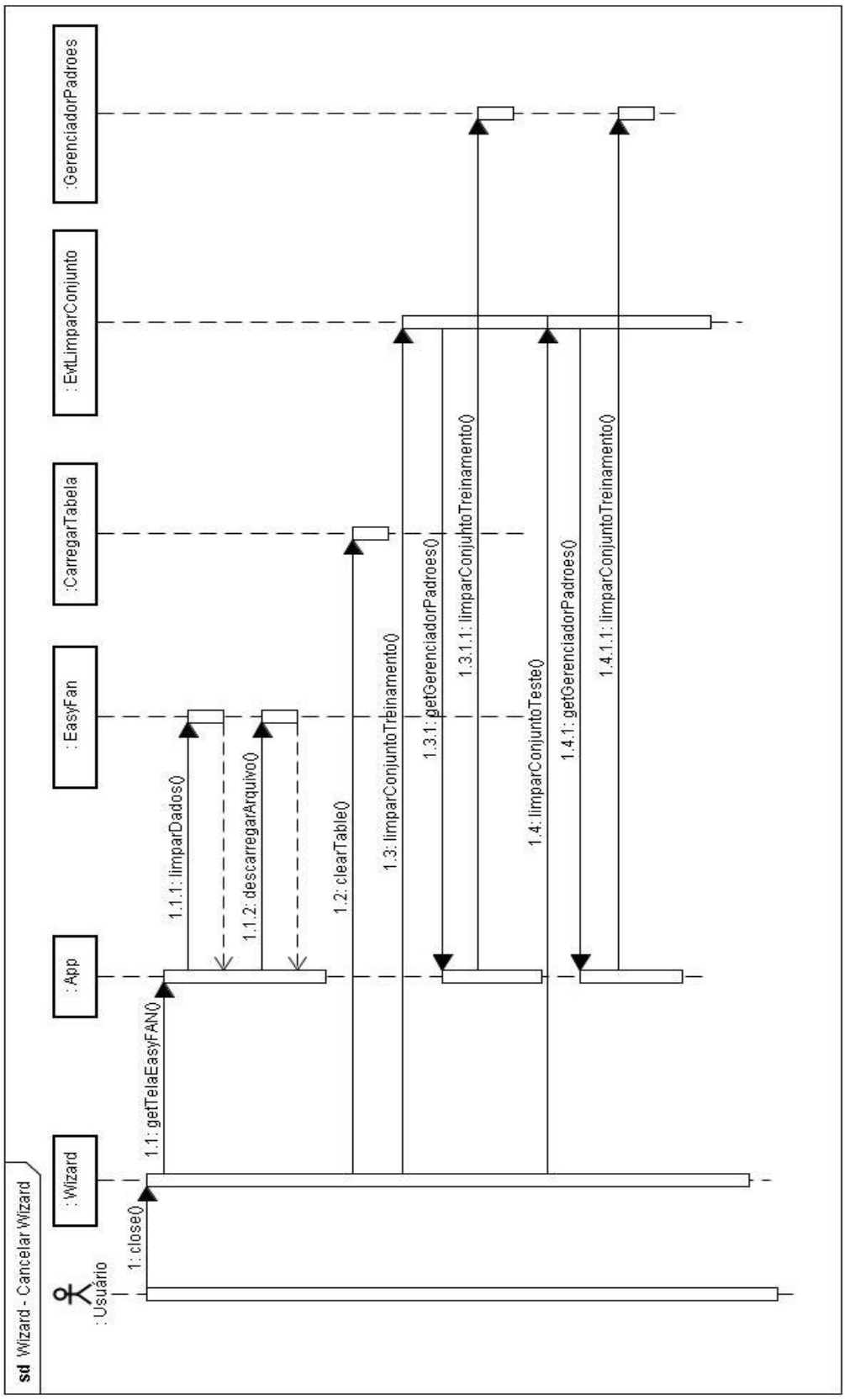

Figura 95 – Diagrama de Seqüência EasyFAN – Cancelar Wizard

5.4. DIAGRAMA DE SEQÜÊNCIA DE TELAS

DIAGRAMA DE SEQÜÊNCIA DE TELAS - GERAL (1/5)

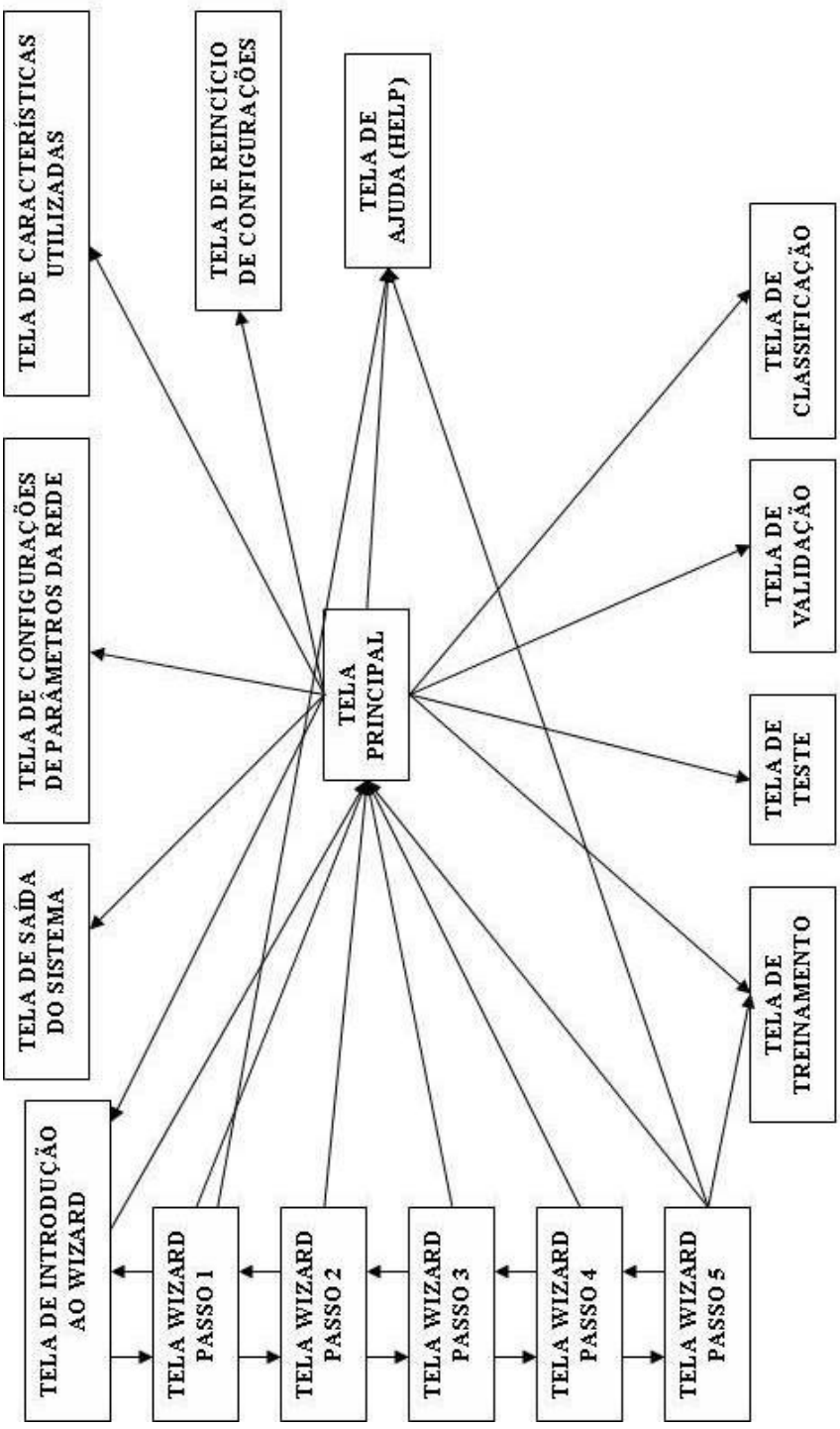

Figura 96– Diagrama de Seqüência de Telas EasyFAN – Geral

DIAGRAMA DE SEQÜÊNCIA DE TELAS - TREINO (2/5)

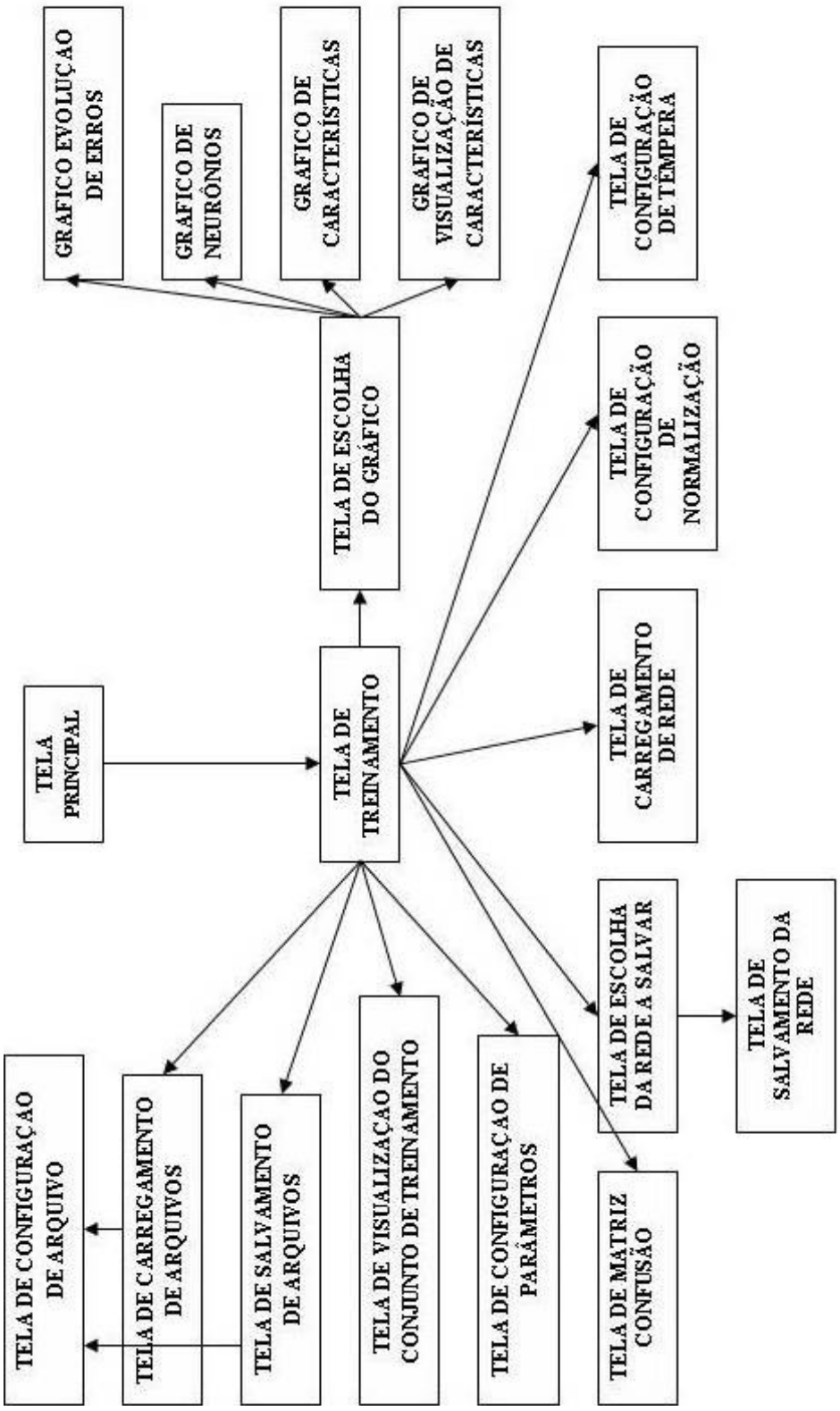

Figura 97– Diagrama de Seqüência de Telas EasyFAN – Treino

DIAGRAMA DE SEQÜÊNCIA DE TELAS - TESTE (3/5)

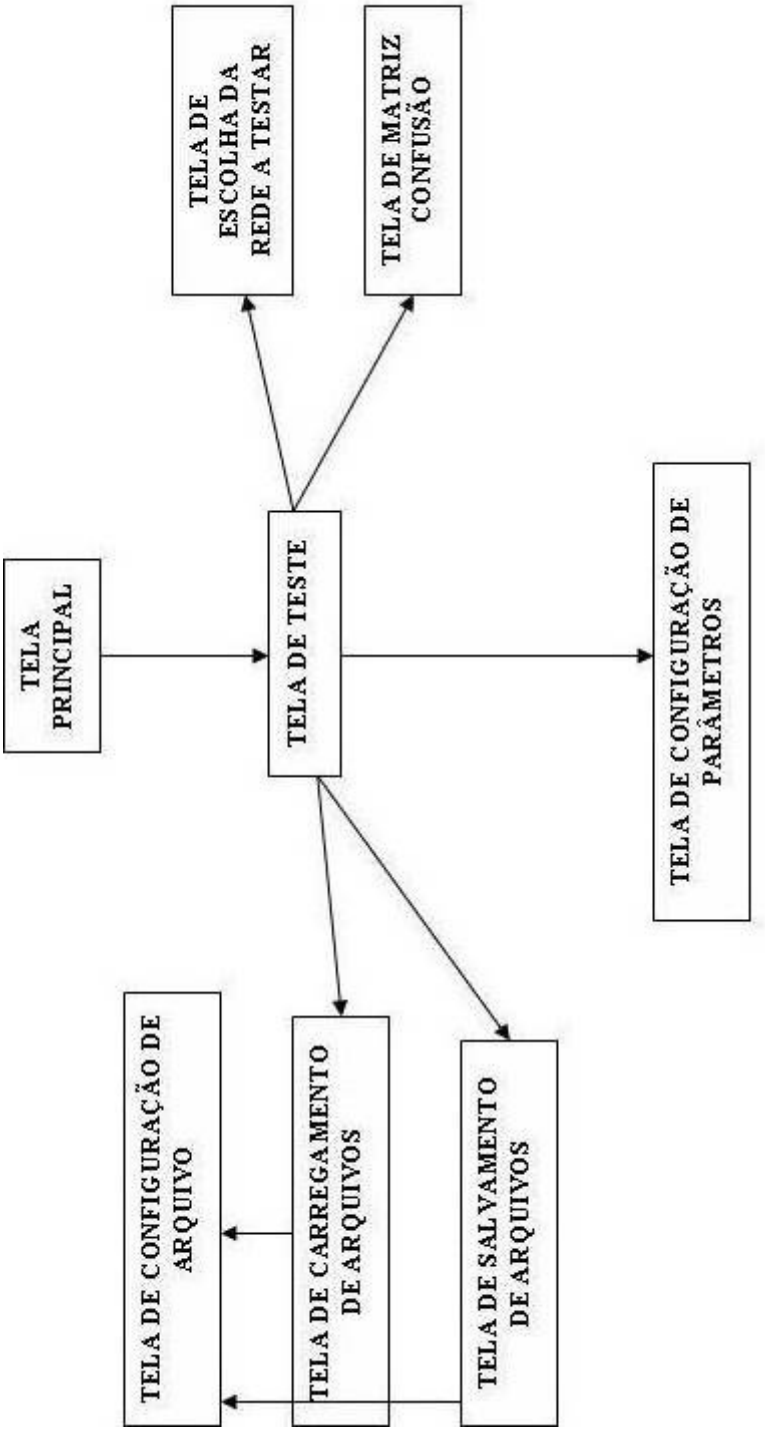

Figura 98– Diagrama de Seqüência de Telas EasyFAN – Teste

DIAGRAMA DE SEQÜÊNCIA DE TELAS - VALIDAÇÃO (4/5)

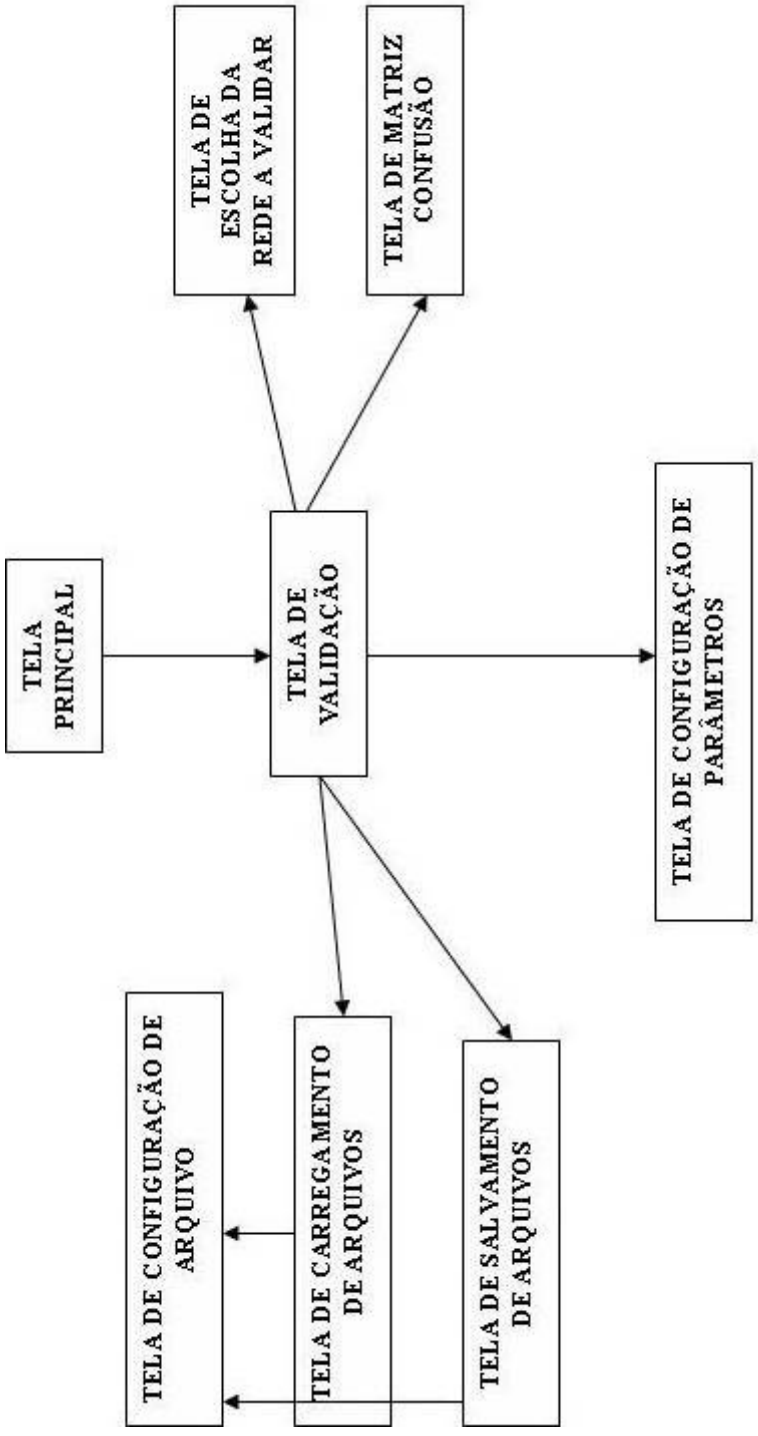

Figura 99 – Diagrama de Seqüência de Telas EasyFAN – Validação

DIAGRAMA DE SEQÜÊNCIA DE TELAS - CLASSIFICAÇÃO (5/5)

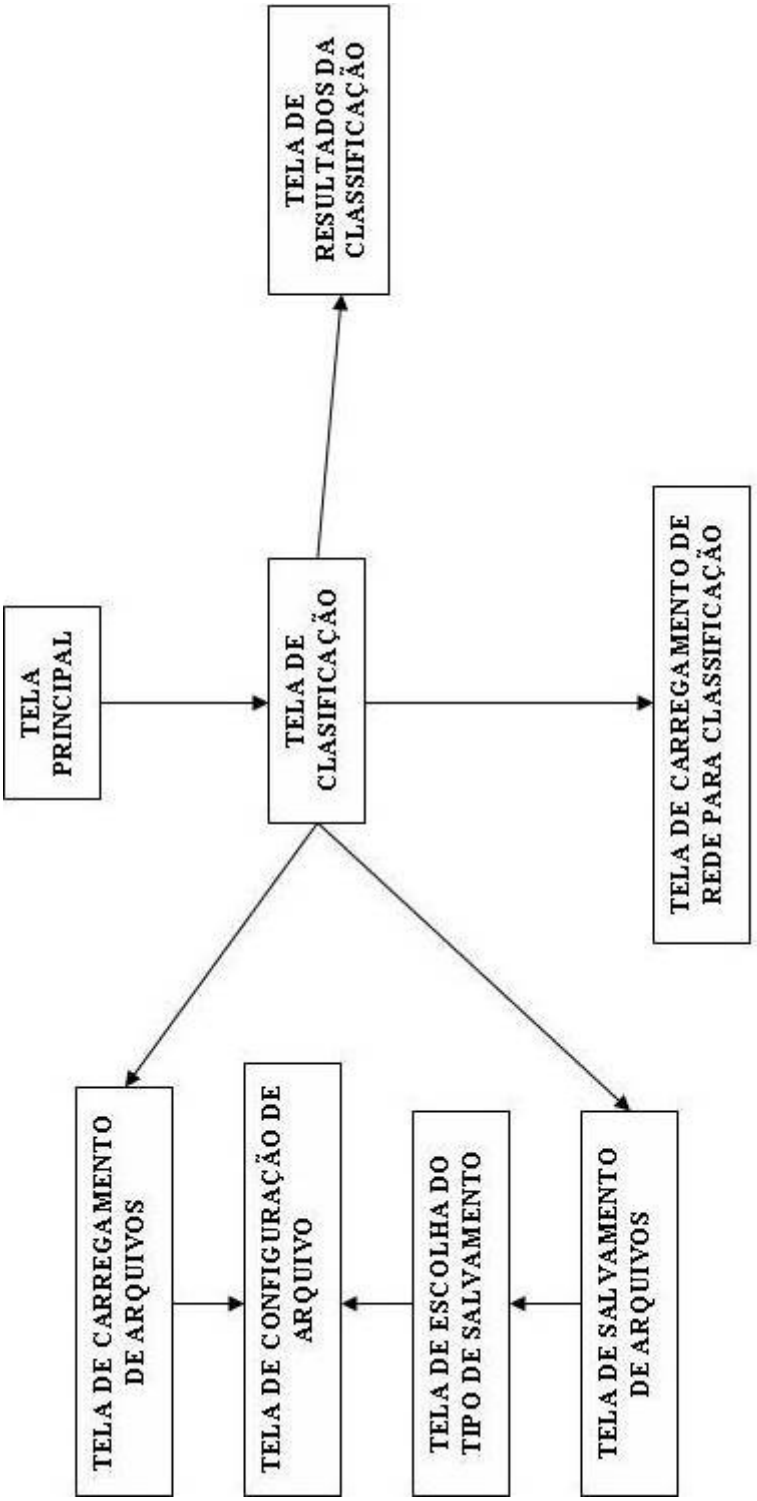

Figura 100 – Diagrama de Seqüência de Telas EasyFAN – Classificação

5.5. DIAGRAMA DE TELAS

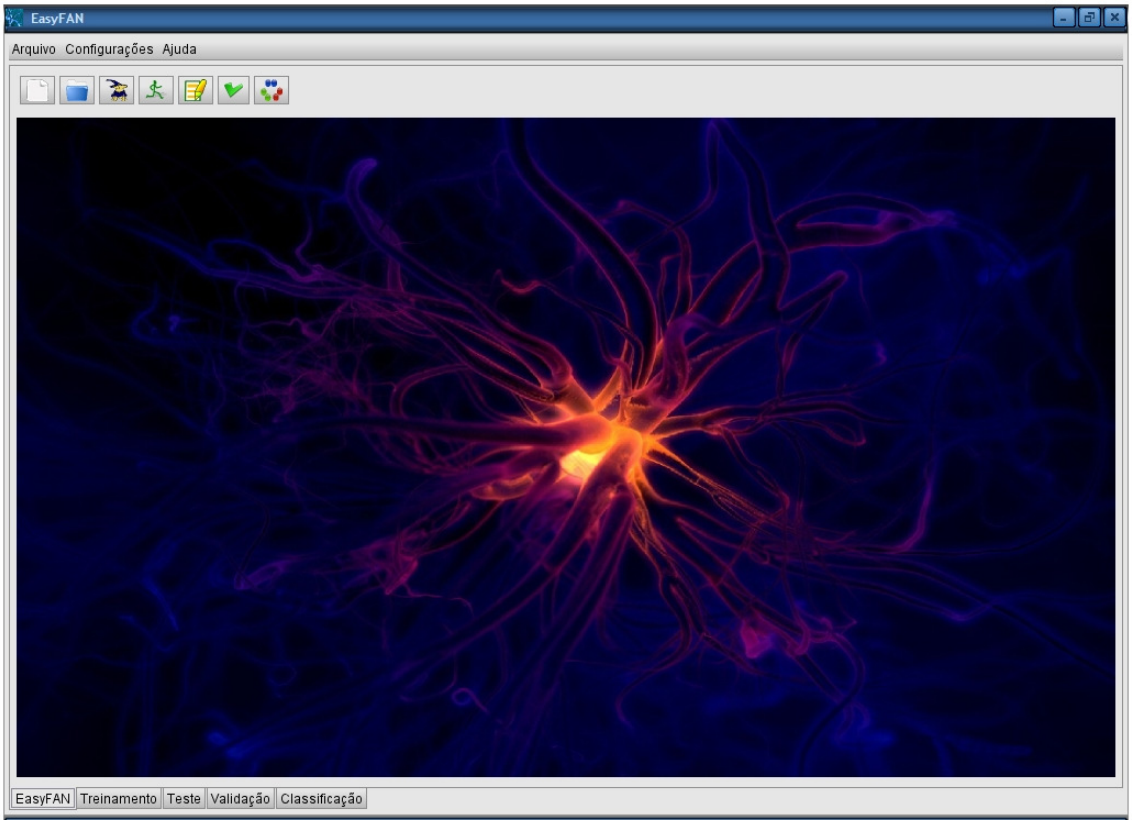

Figura 101 – Tela Inicial

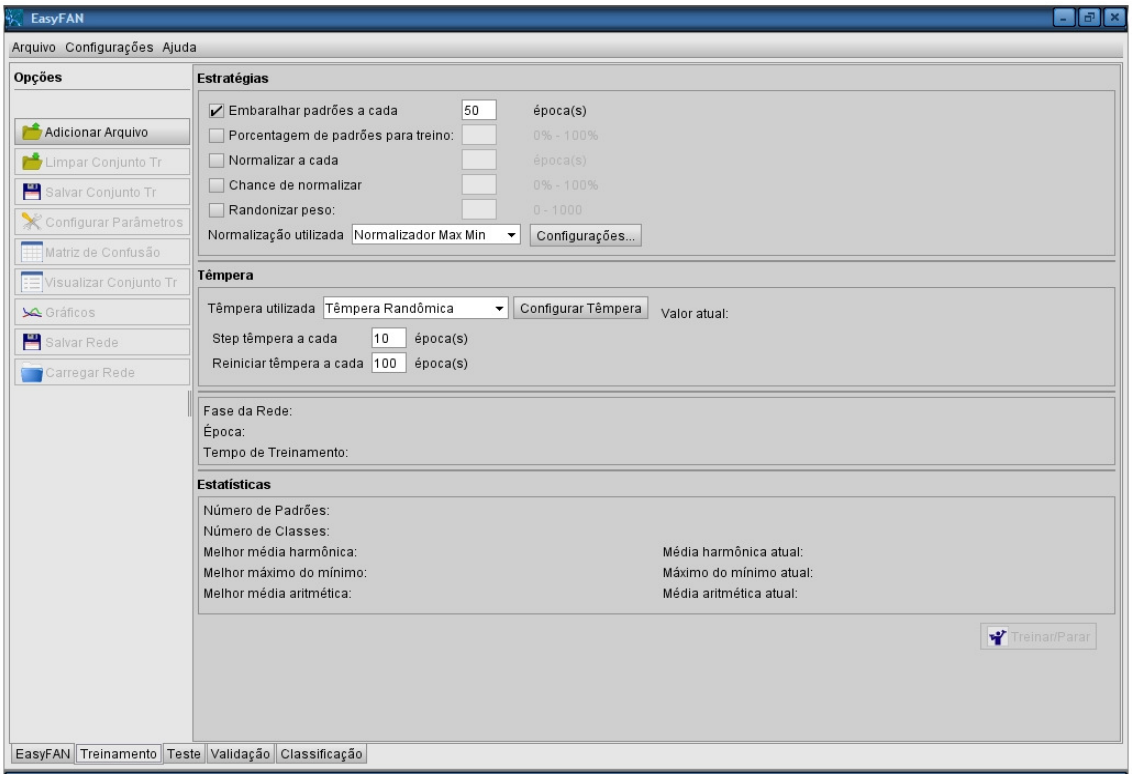

Figura 102 – Tela Aba Treinamento

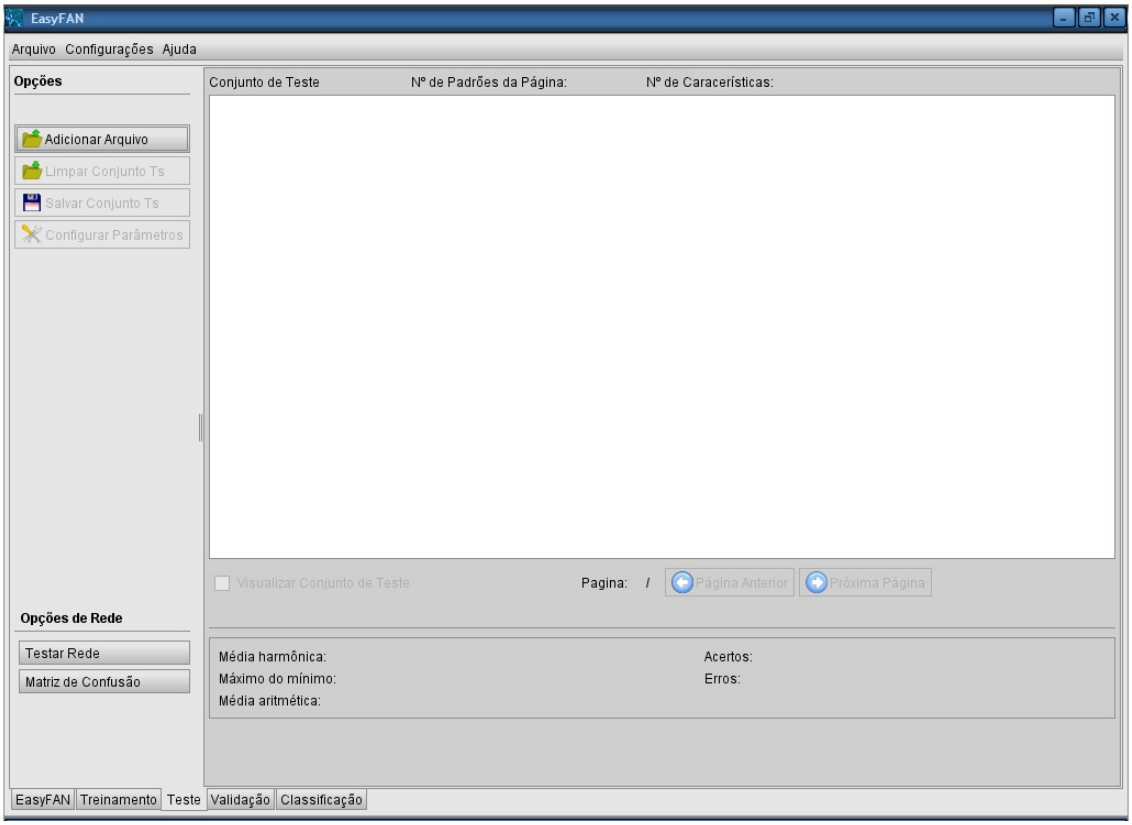

Figura 103 – Tela Aba Teste

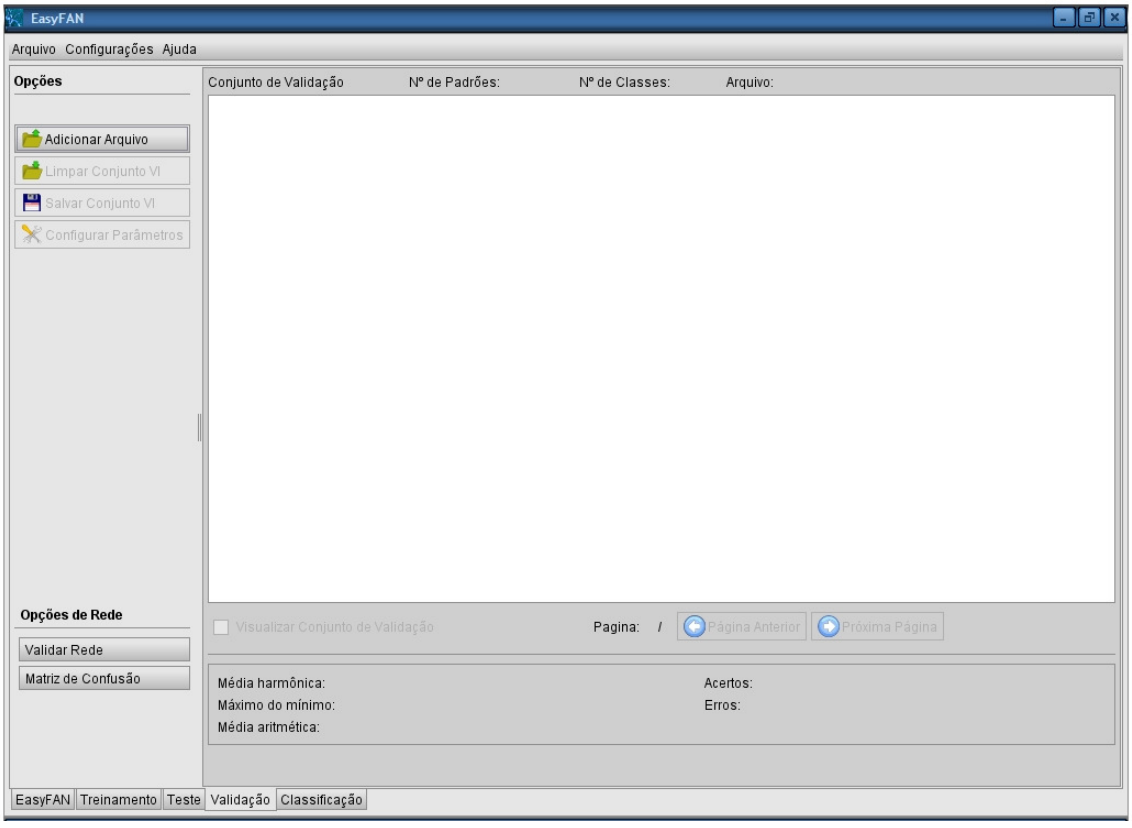

Figura 104 – Tela Aba Validação

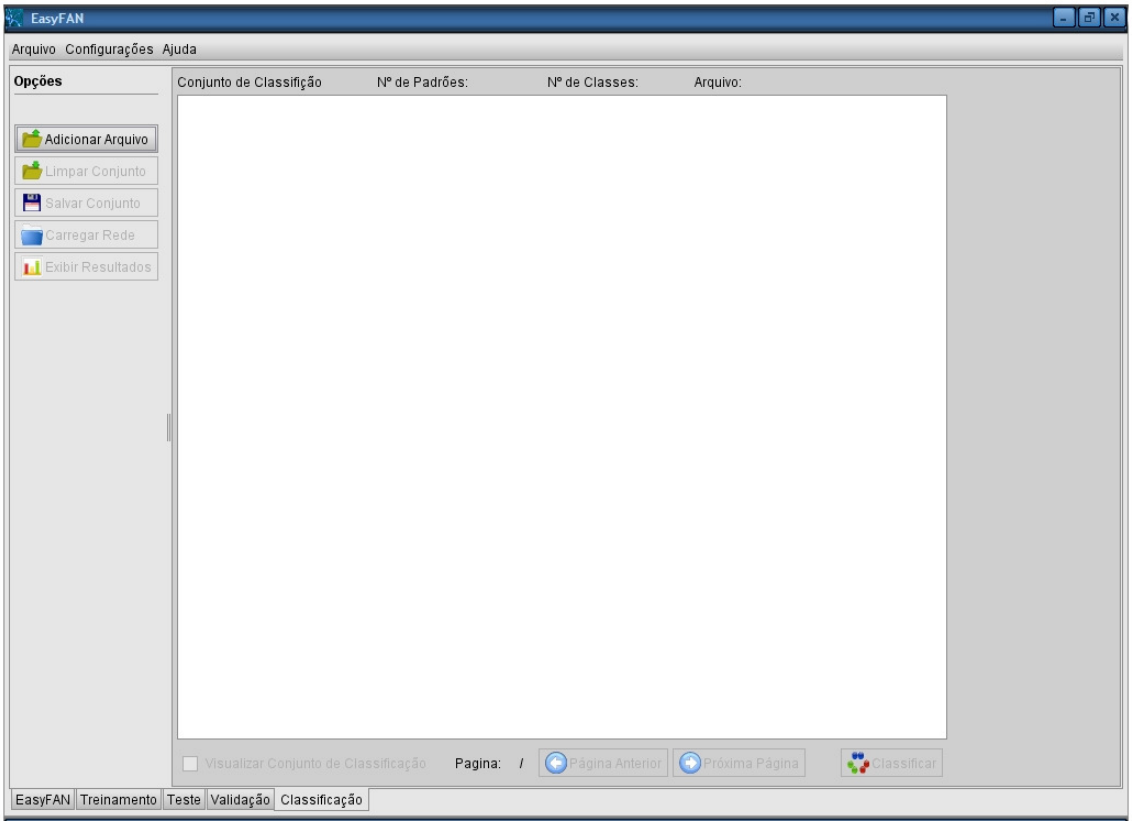

Figura 105 – Tela Aba Classificação

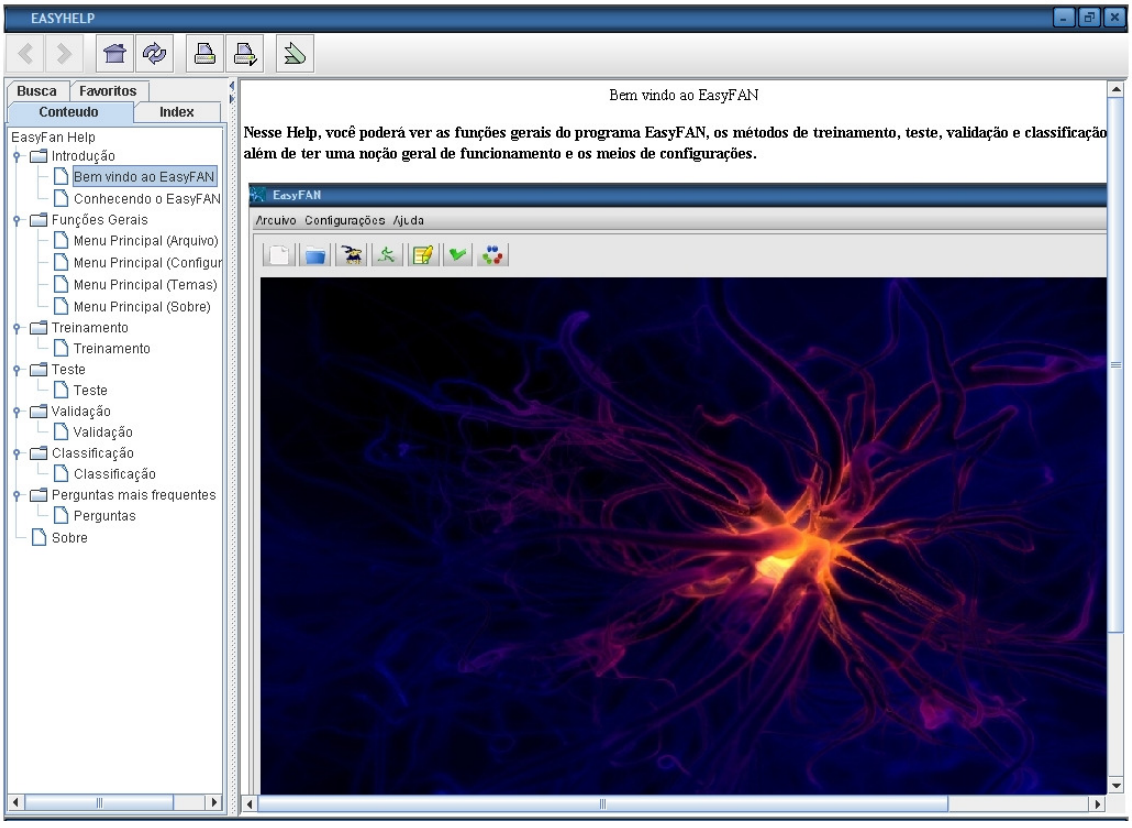

Figura 106 – Tela Ajuda (Help)

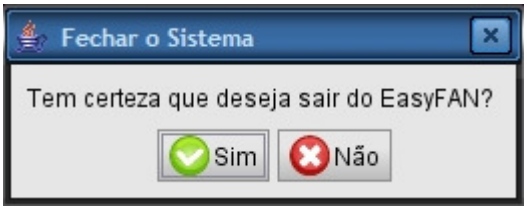

Figura 107 – Tela de Saída do Sistema

Figura 108 – Tela de Configurações de Parâmetros de Rede

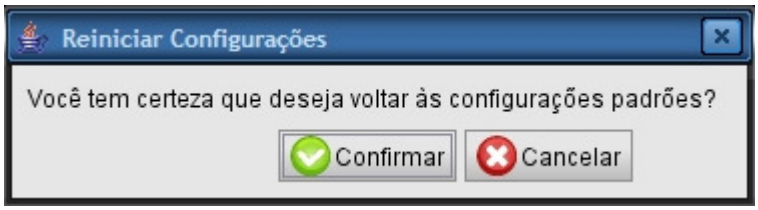

Figura 109 – Tela de Reinício de Configurações

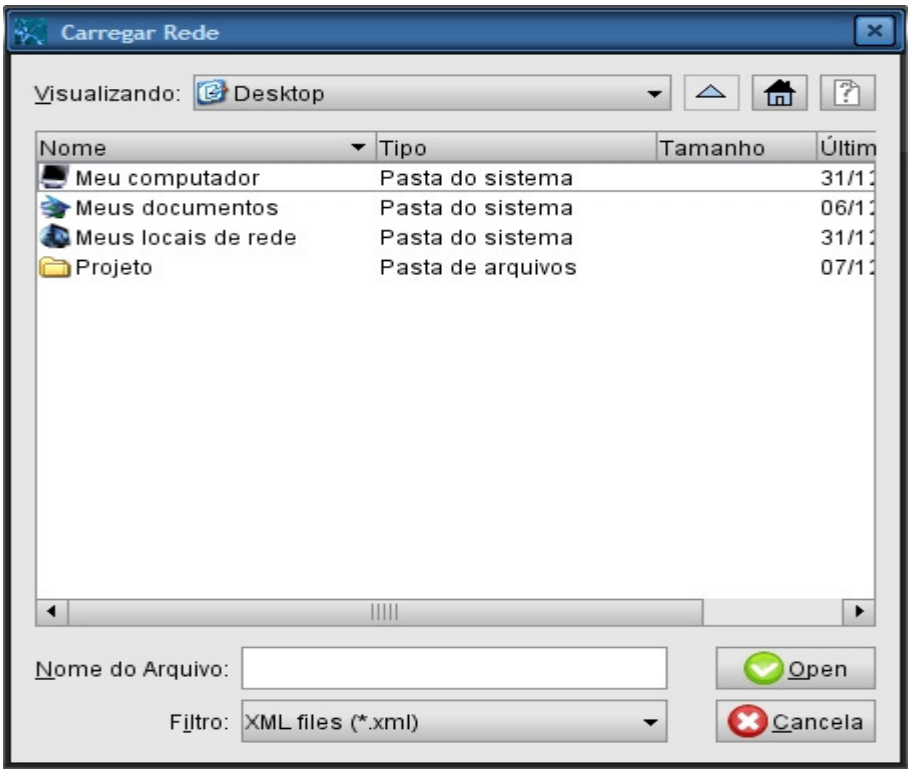

Figura 110 – Tela de Carregamento de Rede

Figura 111 – Tela de Introdução ao Wizard

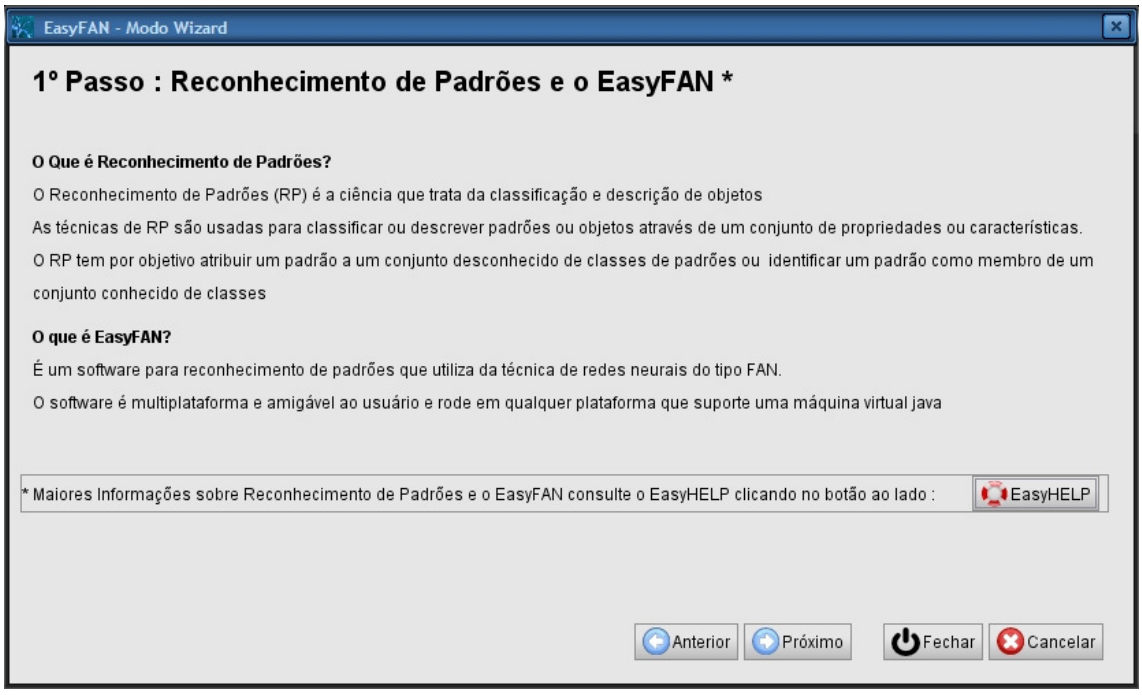

Figura 112 – Tela Wizard Passo 1

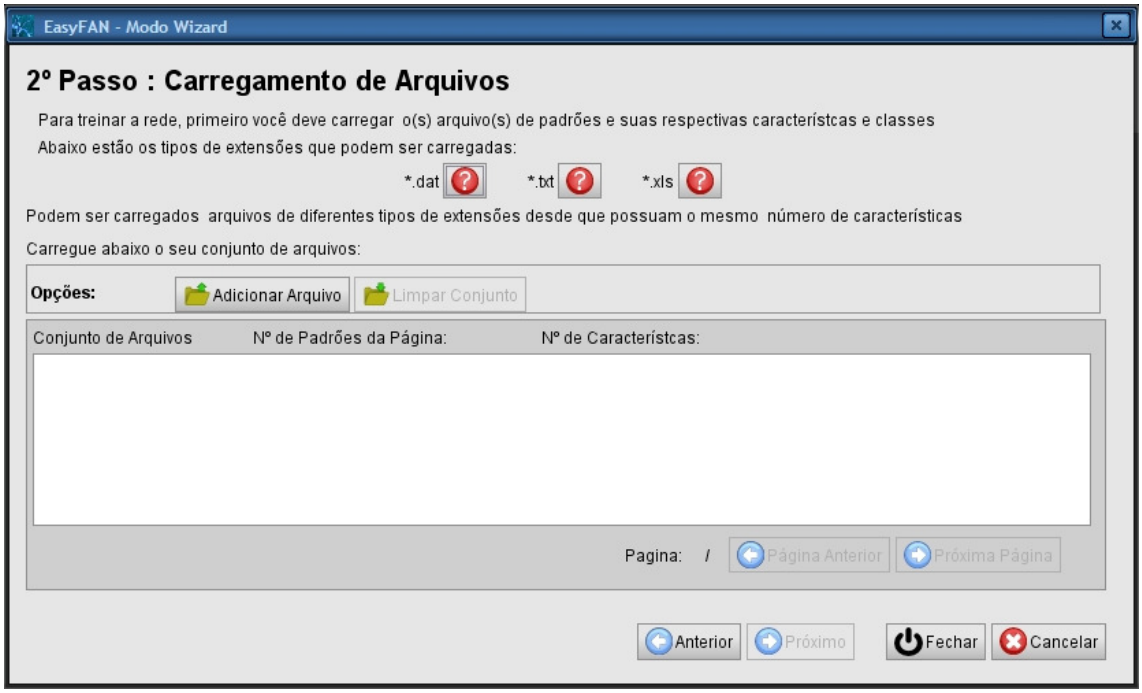

Figura 113 – Tela Wizard Passo 2

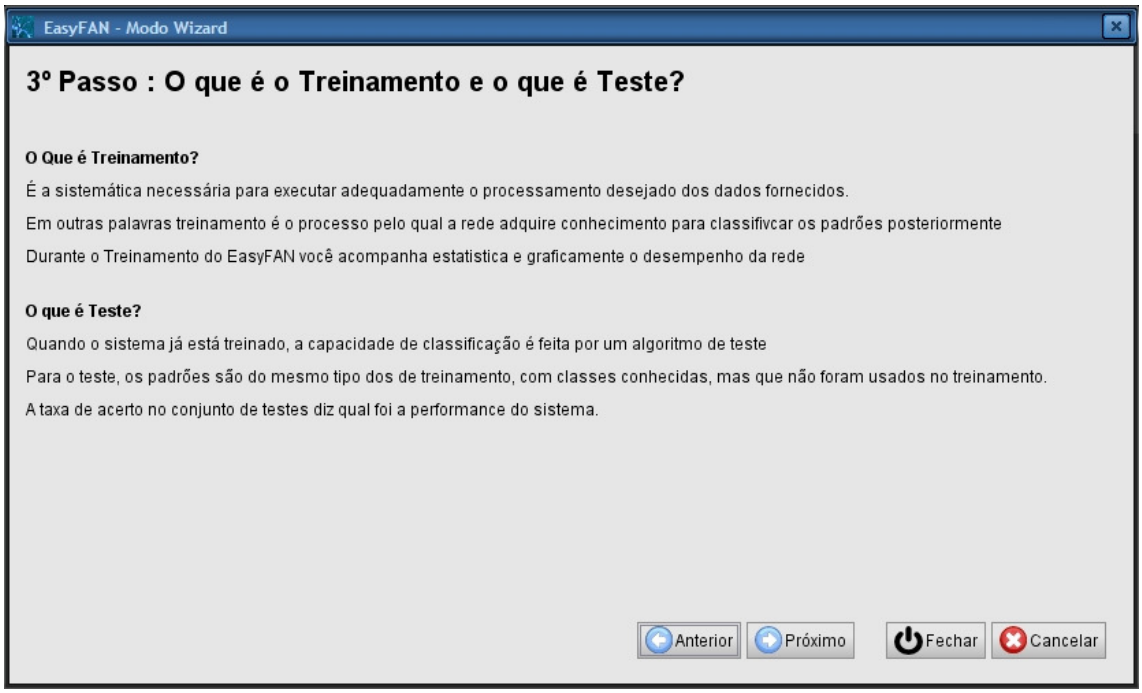

Figura 114 – Tela Wizard Passo 3

Figura 115 – Tela Wizard Passo 4

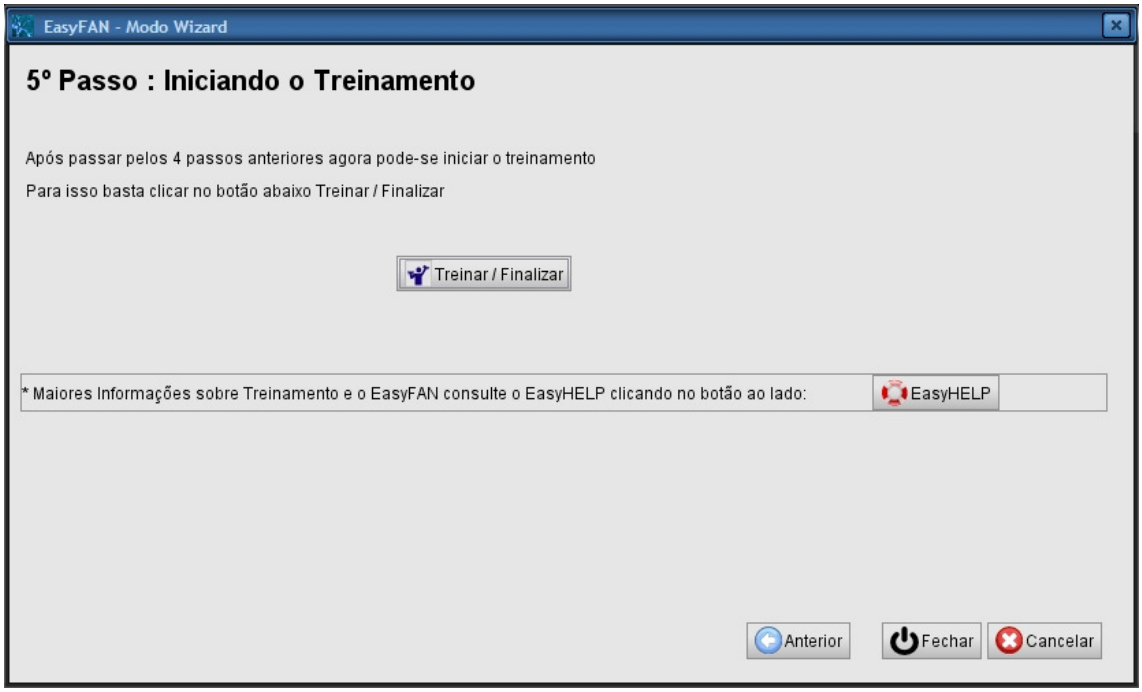

Figura 116 – Tela Wizard Passo 5

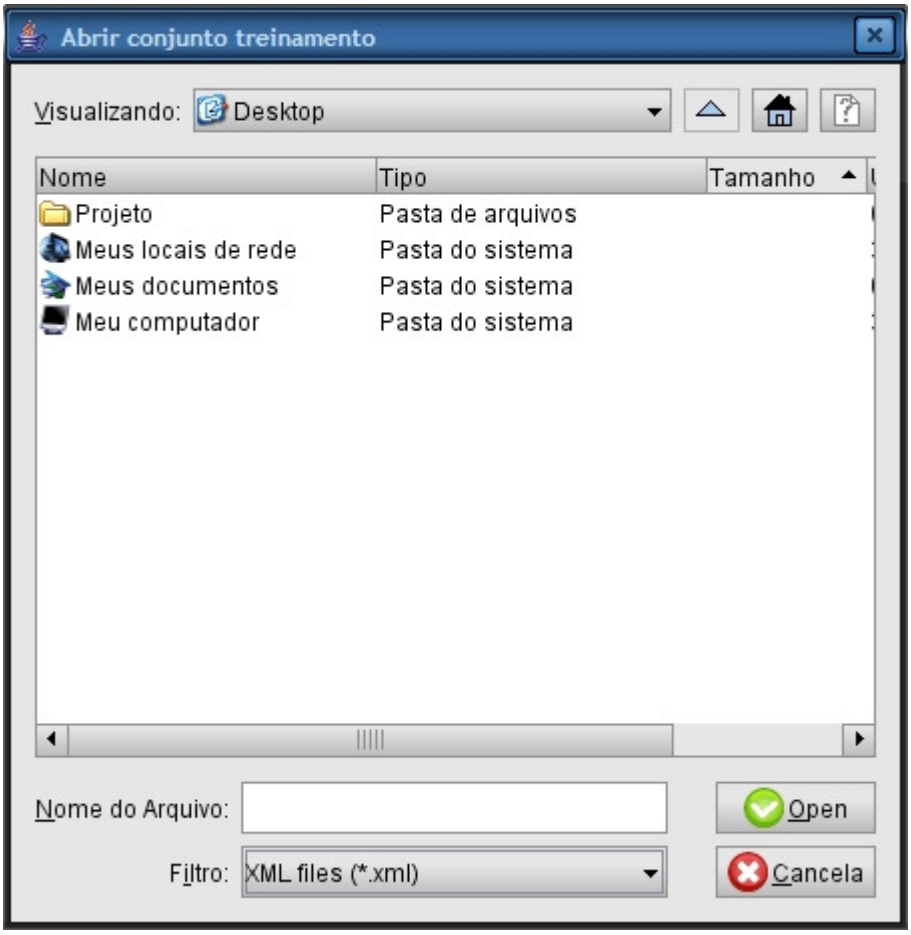

Figura 117 – Tela de Carregamento de Arquivo

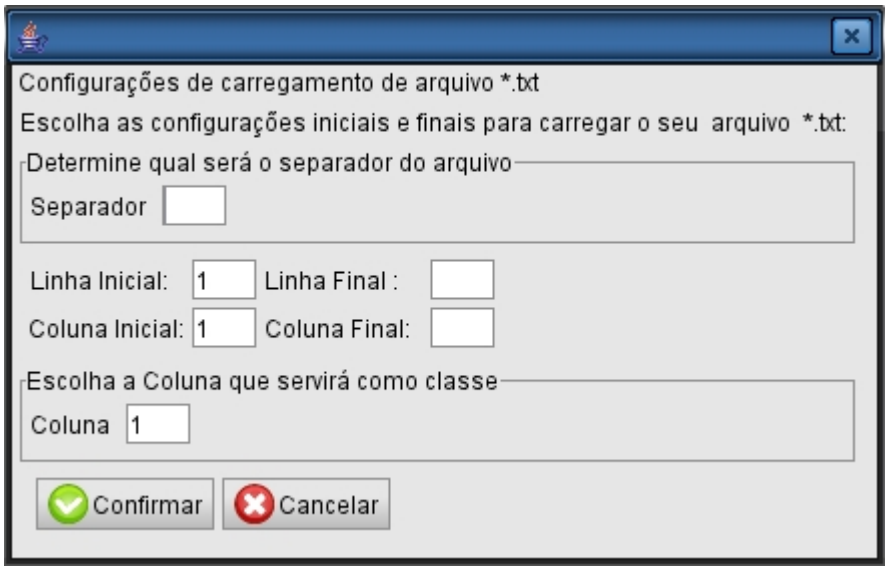

Figura 118 – Tela de Configuração de Arquivo

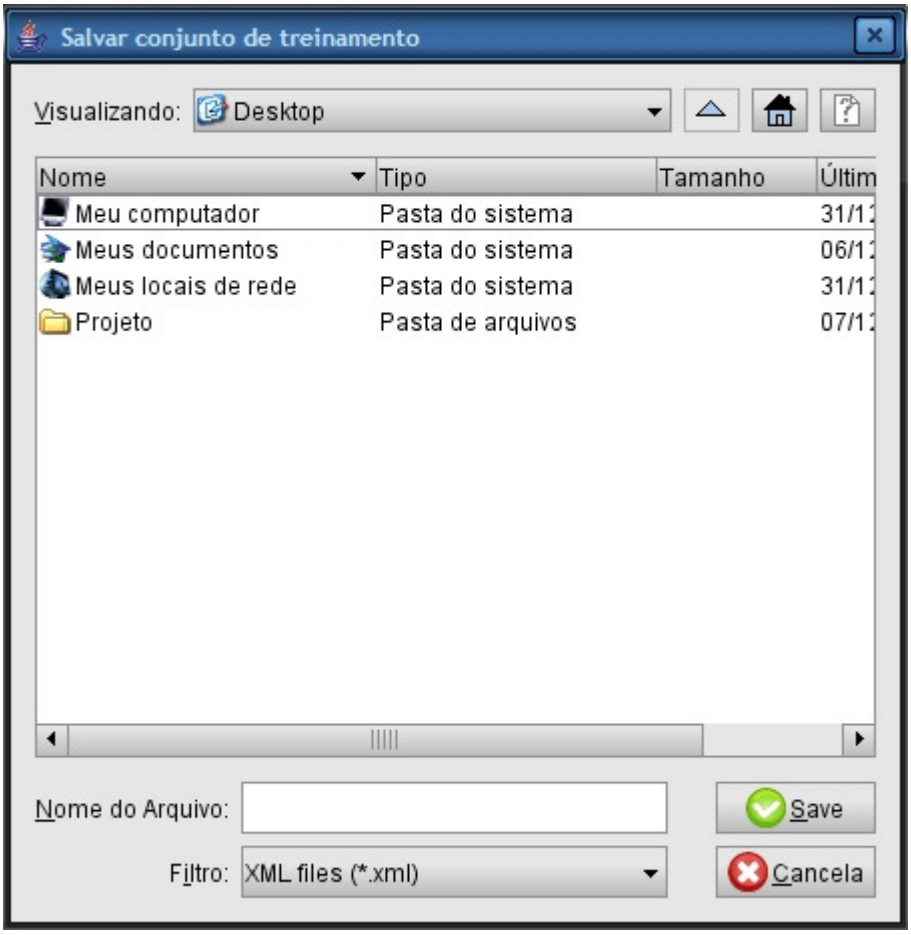

Figura 119 – Tela de Salvamento de Arquivos

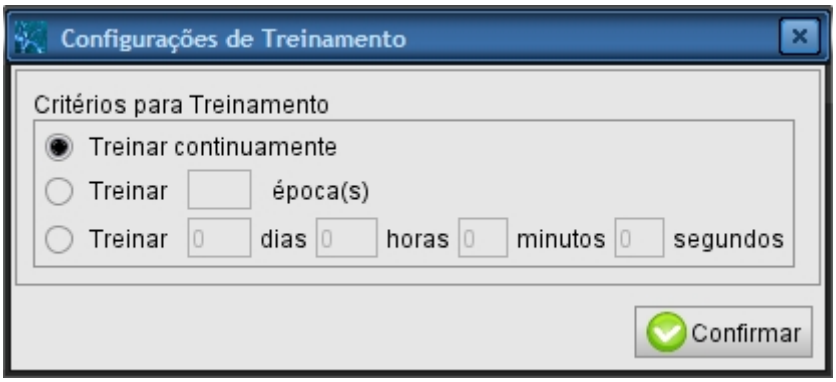

Figura 120 – Tela de Configuração de Parâmetros de Treinamento

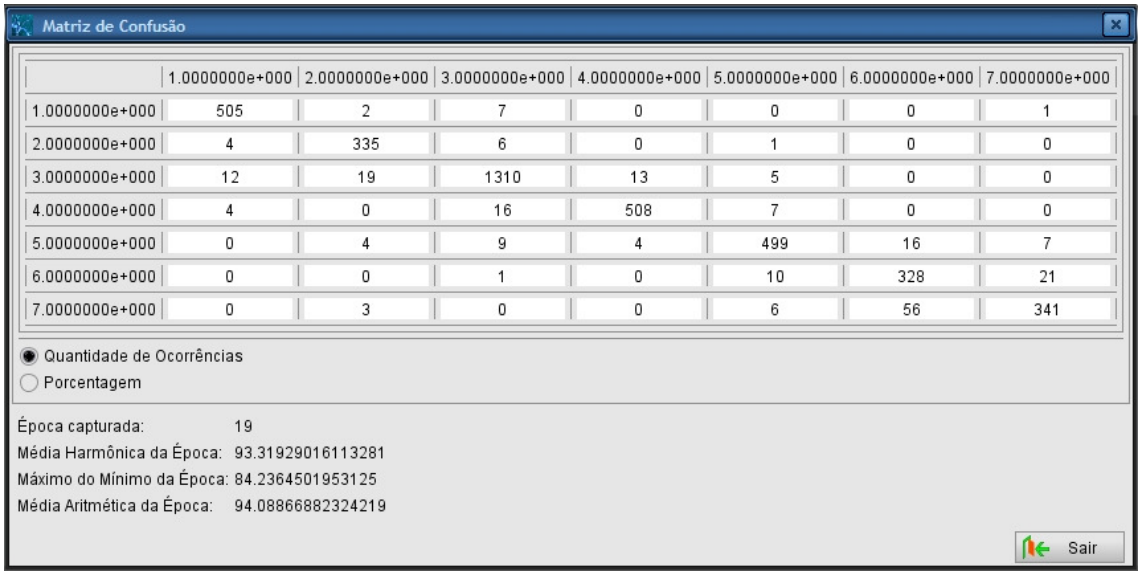

Figura 121 – Tela de Matriz de Confusão

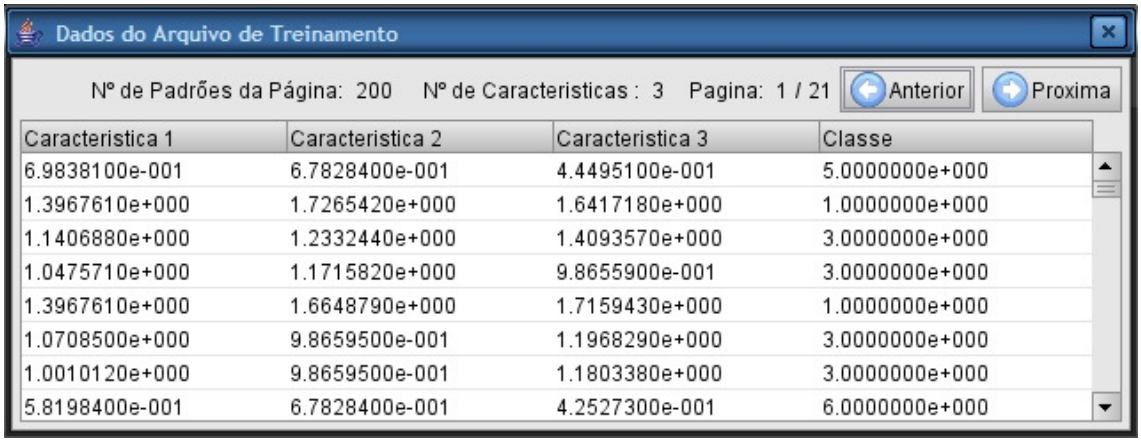

Figura 122 – Tela de Visualização do Conjunto de Treinamento

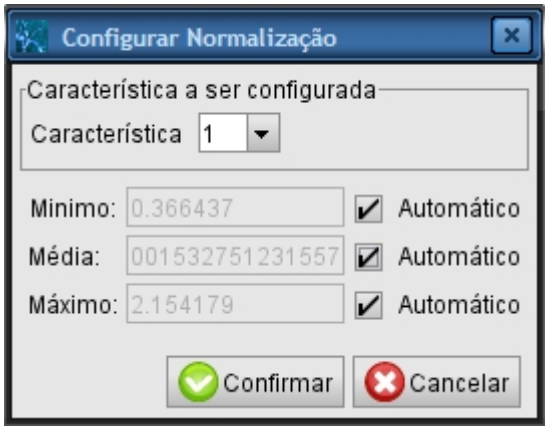

Figura 123 – Tela de Configuração de Normalização

Figura 124 – Tela de Configuração de Têmpera

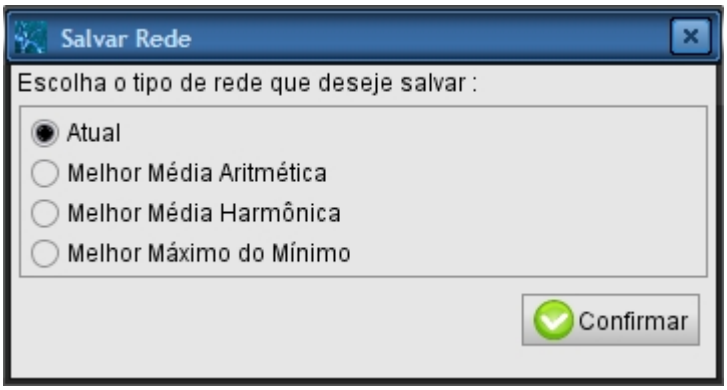

Figura 125 – Tela de Escolha da Rede a ser Salva

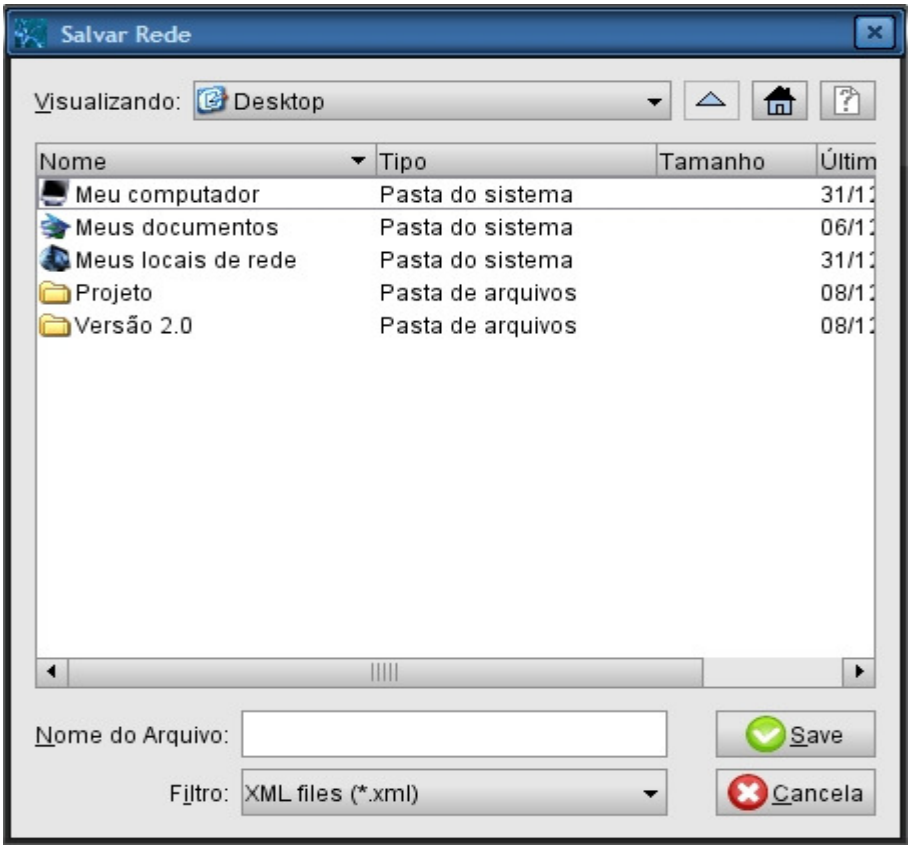

Figura 126 – Tela de Salvamento de Rede

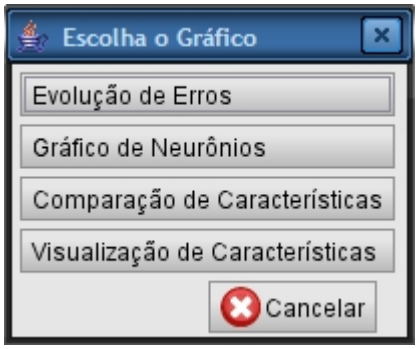

Figura 127 – Tela de Escolha de Gráfico

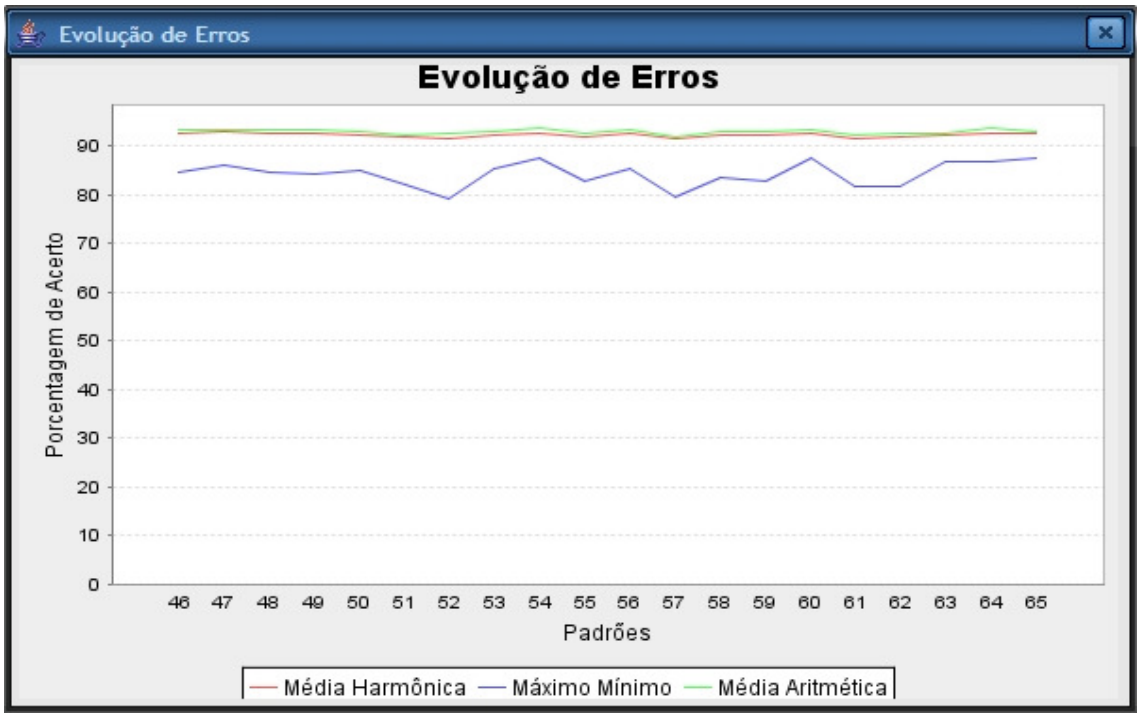

Figura 128 – Gráfico de Evolução de Erros

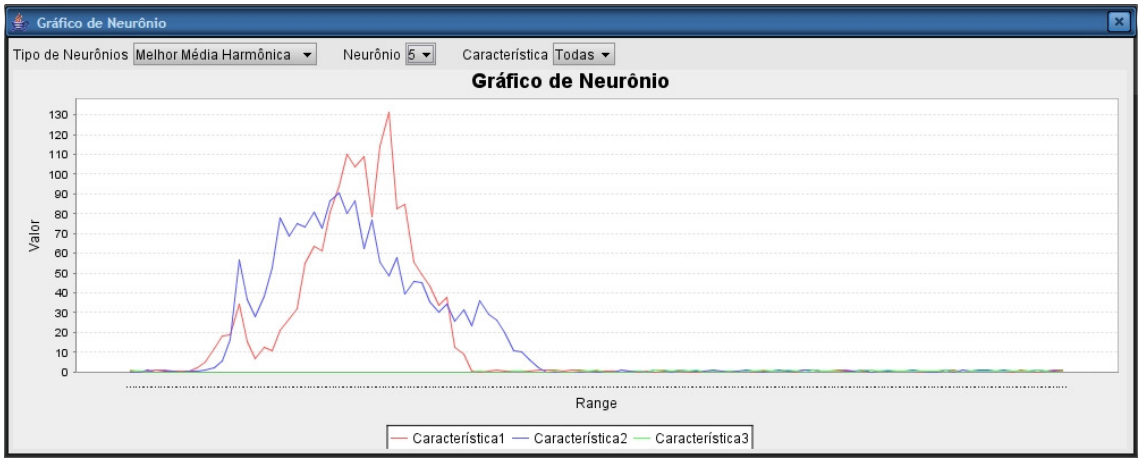

Figura 129 – Gráfico de Neurônios

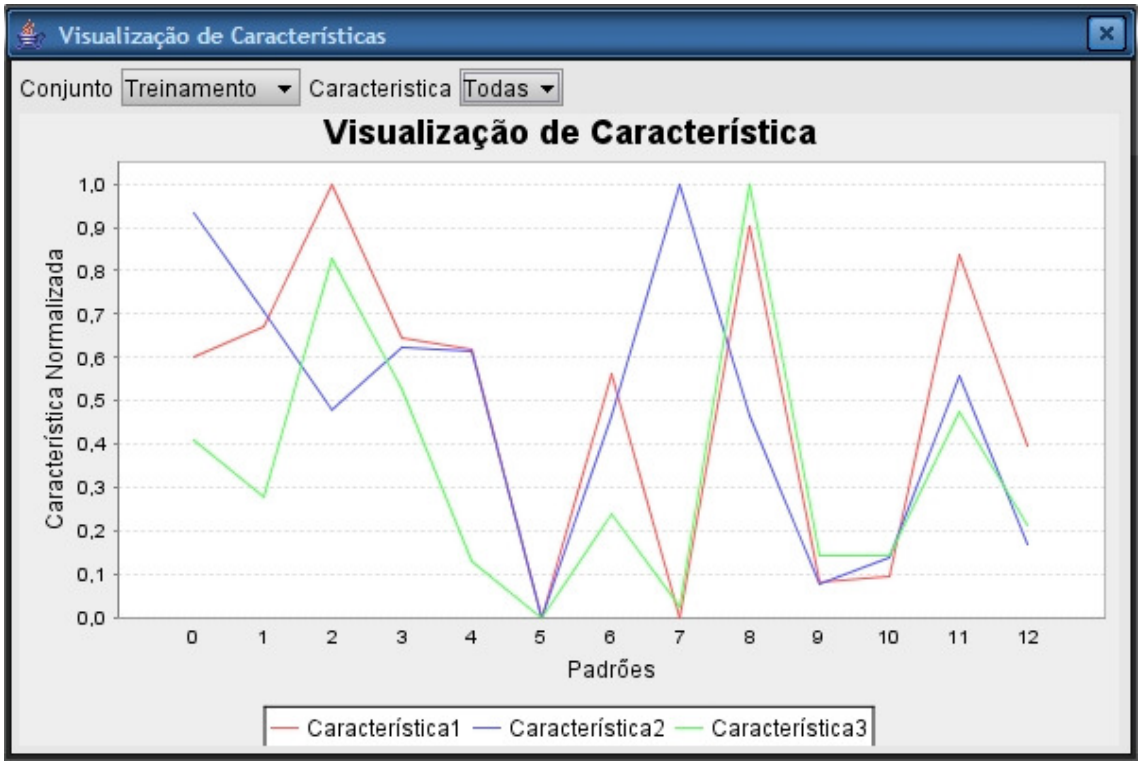

Figura 130 – Gráfico de Visualização de Características

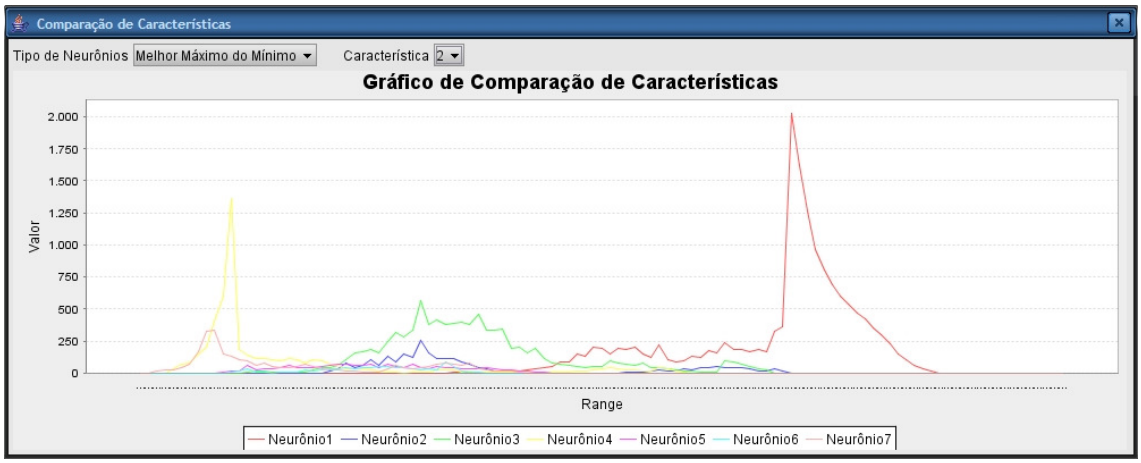

Figura 131 – Gráfico de Comparação de Características

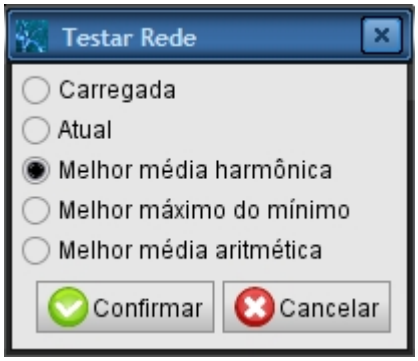

Figura 132 – Tela de Escolha de Rede a Testar

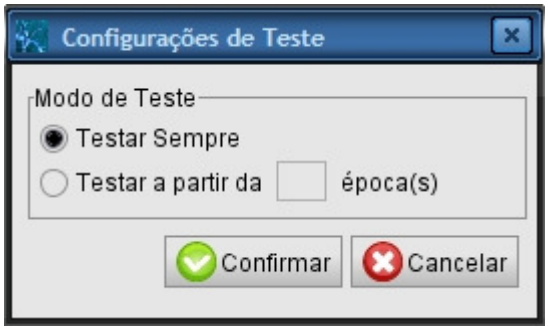

Figura 133 – Tela de Configurações de Parâmetros de Teste

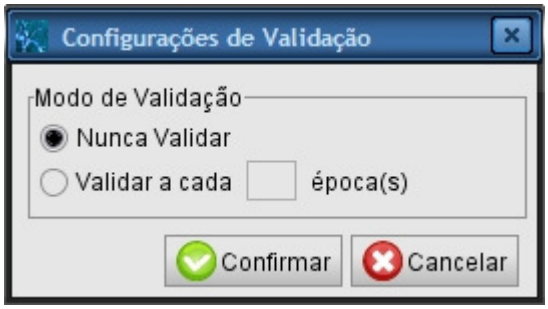

Figura 134 – Tela de Configurações de Parâmetros de Validação

5.6 DIAGRAMA DE ESTADO

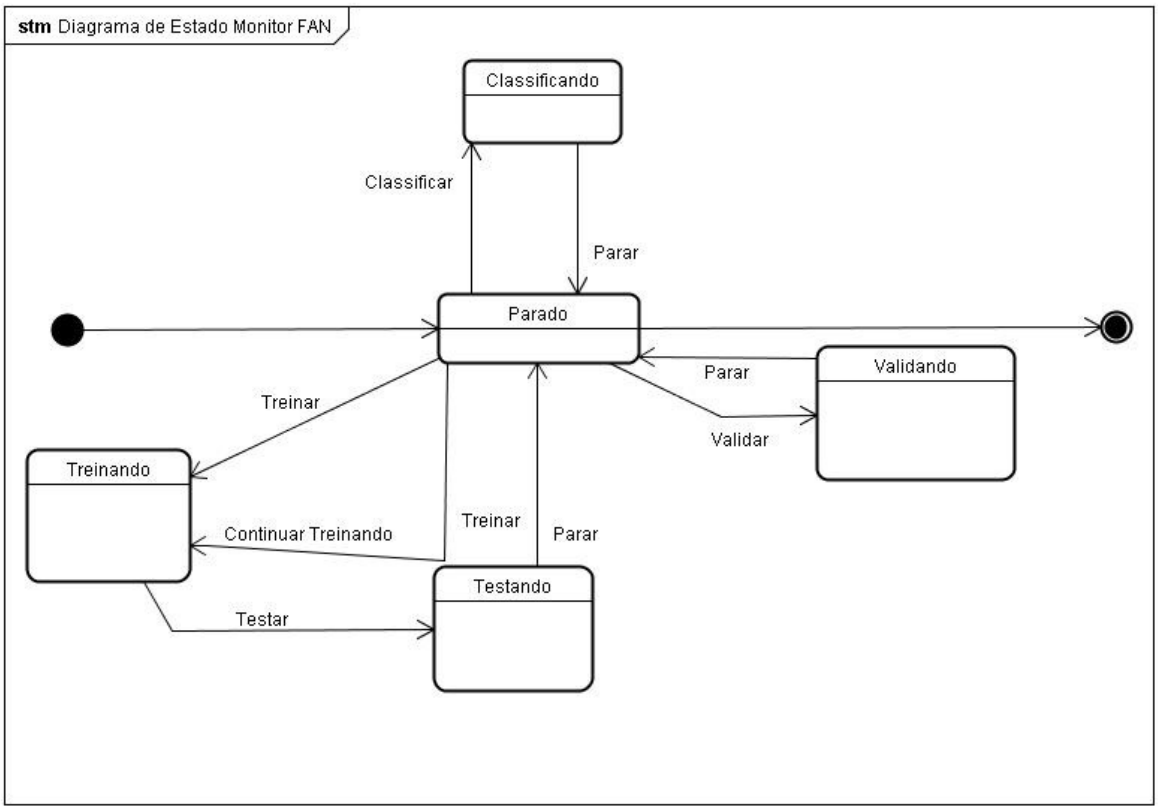

Figura 135 – Diagrama de Estado

5.7 DIAGRAMA DE COMPONENTES

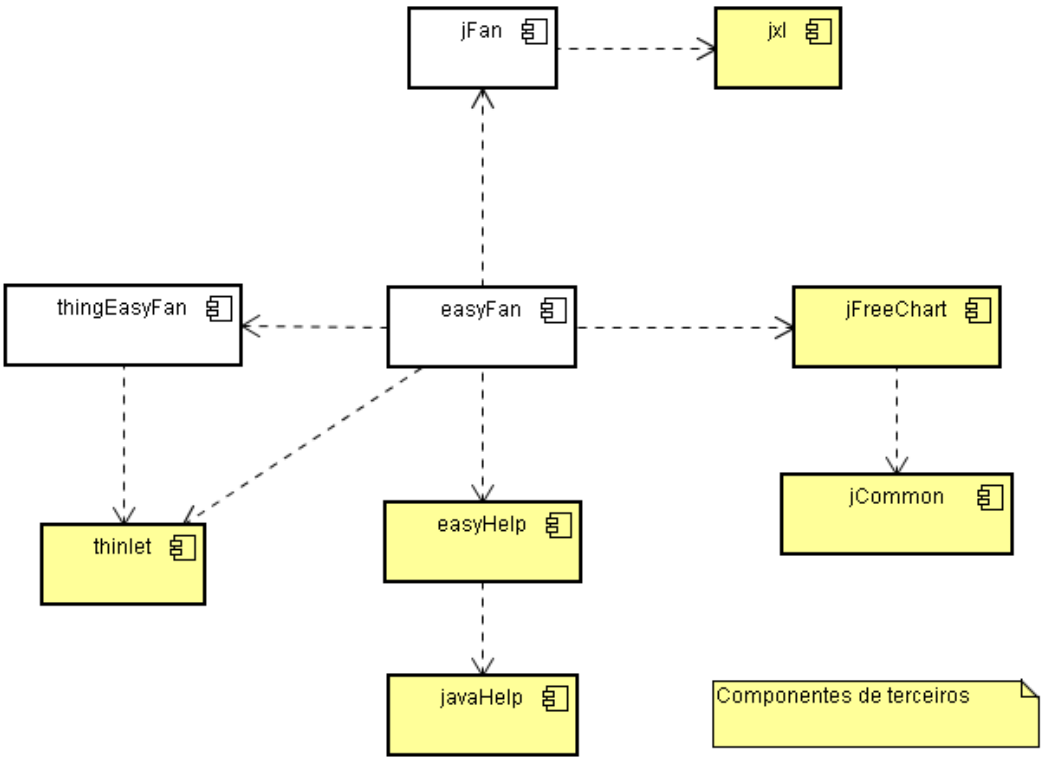

Figura 136 – Diagrama de Componentes

6. CONCLUSÃO

O EasyFan acabou tendo um desempenho melhor do que o LABFAN. Conseguiu fazer as tarefas em menos tempo e consumindo menos recursos. Testes apresentaram uma média de 320% de superação de desempenho, podendo ser verificados pela analise do relatório presente no Anexo 1.

Está apto a abrir arquivos com extensão .dat, .txt, .xls e .epd, gerando mais possibilidades de tipos de arquivos que o anterior LabFan.

Não está dependente da plataforma Windows, pois o fato de ter sido escrito na linguagem Java proporcionou esta vantagem.

Com relação ao aproveitamento do projeto por terceiros, o EasyFAN foi utilizado por Wladimir Bastos, estudante do curso de Sistemas de Informação, na Faculdade Hélio Rocha em Salvador-BA, para o reconhecimento automático de locutor em modo dependente de texto, projeto esse orientado pelo Professor Ernesto Massa.

Tanto o jFAN quanto o EasyFAN estão sendo utilizados em Projeto de conclusão do curso de Tecnologia em Informática da UFPR pela equipe composta por Amanda Campos Costa, Marcelo Preiss e Jonath Rodrigues, orientados pelo professor Mauro José Belli. O projeto tem por objetivo o desenvolvimento de um software que fará projeções futuras das cotações da bolsa de valores. Sua utilidade pode ser comprovada pela declaração que consta no Anexo 2.

Uma abordagem ainda a ser considerada, que não foi contemplada no projeto, é a possibilidade de internacionalização do software, permitindo a utilização do EasyFan por parte de outros usuários de outras línguas.

REFERÊNCIAS BIBLIOGRÁFICAS

BOOCH, G., RUMBAUGH, J., JACOBSON, I. *UML, guia do usuário,* 2ª ed. Rio de Janeiro: Elsevier, 2005.

Eclipse. Disponível em: <http://www.eclipse.org>. Acesso em: 3 abr. 2006.

GNU Lesser General Public License. Disponível em: <http://www.gnu.org/copyleft/lesser.html>. Acesso em: 1º dez. 2006.

GNU Lesser General Public License da Free Software Foundation. Disponível em: <http://creativecommons.org/licenses/LGPL/2.1/deed.pt>. Acesso em 1º dez. 2006.

Java. Disponível em: <http://java.sun.com>. Acesso em: 3 abr. 2006.

JavaHelp. Disponível em: <http://java.sun.com/products/javahelp>. Acesso em: 6 set. 2006.

JFreeChart. Disponível em: <http://www.jfree.org/jfreechart>. Acesso em: 6 set. 2006.

Licença Pública Geral GNU. <http://www.magnux.org/doc/GPL-pt_BR.txt>. Acesso em 1º dez. 2006.

Licença Pública Geral Menor do GNU. Disponível em: <http://creativecommons.org/licenses/LGPL/2.1/legalcode.pt>. Acesso em 1º dez. 2006.

NetBeans. Disponível em: <http://www.netbeans.org>. Acesso em: 3 abr. 2006.

O que é o Software Livre? Disponível em: <http://www.gnu.org/philosophy/free-sw.pt.html>. Acesso em 1º dez. 2006.

RAITTZ, R. T. *Free Associative Neurons – FAN: uma abordagem para reconhecimento de padrões.* Florianópolis, 1997. Dissertação (Mestrado em Engenharia de Produção) - Universidade Federal de Santa Catarina.

_____. *FAN 2002: Um modelo neuro-fuzzy para reconhecimento de padrões*. Florianópolis, 2002. Tese (Doutorado em Engenharia de Produção) Universidade Federal de Santa Catarina.

Reconhecimento de Padrões. Disponível em: <http://www.lx.it.pt/~afred/rp-ist/ACETATOS/RP_1_Introducao.PDF>. Acesso em 5 dez. 2006.

RIPLEY, B. D. *Pattern Recognition and Neural Networks,* 8ª ed. Cambridge: Cambridge University Press, 2005.

STRUNK, Jr William. *The Elements of Style.* Disponível em: <http://www.bartleby.com/141/strunk1.html>. Acesso em 10 dez. 2006.

Thing. Disponível em: <http://thing.sourceforge.net>. Acesso em: 3 abr. 2006.

Thinlet. Disponível em: <http://www.thinlet.com>. Acesso em: 3 abr. 2006.

ANEXO I - COMPARAÇÃO ENTRE OS SOFTWARES LABFAN E EASYFAN

Os dois aplicativos foram testados simultaneamente utilizando os mesmos conjuntos de padrões, tanto para treinamento quanto para teste; os conjuntos usados continham padrões sobre cromossomos. Foram usadas as configurações mínimas dos dois aplicativos para a realização do teste de comparação.

Dados dos conjuntos utilizados na comparação de desempenho:

Conjunto de Treinamento:

Arquivo: Cromo_Tr.dat

Número de Padrões: 4060 divididos em 7 Classes em 3 Classes

Conjunto de Teste:

Arquivo: Cromo_Ts.dat

Número de Padrões: 4060 divididos em 7 Classes em 3 Classes

Resultado do Teste de Desempenho

Os aplicativos foram testados num período de 10 minutos, e foi observado que o LabFAN foi executado por 370 épocas e o EasyFAN foi executado por 1185 épocas.

Neste teste simples, foi constatado que o EasyFAN supera o LabFAN em 3,2 vezes.

Cabe ressaltar que dependendo do hardware é possível atingir níveis melhores de desempenho, como em ambientes multiprocessado ou em

processadores de dois núcleos, o EasyFAN terá uma performance superior por ter sido construído no modelo de multithreads.

ANEXO II – DECLARAÇÃO DE USO DO JFAN

Relatório – Equipe SAIB – Sistema de Apoio a Investidores da Bovespa.

Conforme solicitação feita pela equipe EasyFan, na data de 13 de Dezembro de 2006, realizamos no laboratório 3 do Setor Escola técnica, na Universidade Federal do Paraná, alguns testes com a finalidade de comparar o desempenho entre o Laboratório LabFan, na linguagem C, e a compilação JFAN, em Java, sendo este desenvolvido pela equipe solicitante.

Os requisitos que apuramos são média harmônica e tempo de execução dos treinos / testes, sempre utilizando os mesmos arquivos, redes e épocas, que rodando em computadores distintos, pudemos obter conclusões, complementar o trabalho do solicitante e desta forma, corroborar com a apresentação da aplicação da equipe solicitante junto a uma banca avaliadora, lembrando que a mesma nos forneceu, sem ônus e para fins acadêmicos a biblioteca em Java, a qual estamos utilizando também para testes em nosso projeto, por se tratar de uma mesma plataforma de desenvolvimento.

Dos testes finais pudemos concluir que, com arquivos idênticos e rodando nos aplicativos propostos e em todas as ações realizadas, a biblioteca JFAN sempre apresentou como resultados finais médias hamônicas maiores que as do Laboratórios LabFan, e isto sempre em tempos iguais ou inferiores.

Não podemos expressar opinião técnica sobre a melhor performance relatada, pois essa não nos caberia, porém como usuários da biblioteca do solicitante, podemos nos afirmar satisfeitos e agradecidos pela incumbência dos testes e pela disponibilidade da referida biblioteca.

Curitiba, 13 de Novembro de 2006 – Equipe SAIB

Amanda Campos Costa

Jonath Rodrigues Ignácio

Marcelo Preiss