Como podemos observar na figura 10, os neurônios da camada de entrada (X1,..,X10) são alimentados pelos tópicos de estudo e a saída Y1 libera o próximo tópico sinalizando ao sistema se o aprendiz pode prosseguir nos estudos. A seguir são descritos o algoritmo de treinamento, as etapas de treinamento e finalmente a configuração da rede neural.

5.5. Software de treinamento da RNA

Foi desenvolvido para esta pesquisa um software de treinamento escrito em linguagem Java que possui o algoritmo descrito a seguir:

1. Ler as entradas do formulário de parâmetros da rede:
   número de ciclos de treinamento,valor do parâmetro alfa, valor do momentum, capitulo a ser treinado e erro total desejado;
2. Ler do banco de dados os valores dos padrões de treinamento para o capítulo estipulado;
3. Inicializar os pesos para o capítulo estipulado;
4. Ler os pesos das camadas de entrada;
5. Ler pesos das camadas intermediárias;
6. Iniciar o algoritmo de retro-propagação;
7. Atualizar os pesos no banco de dados;
8. Calcular o Erro Total;
9. Repetir os passos de 1 a 8 se o erro total for maior que o erro desejado;
10. Fim do ciclo

Este programa permite ao usuário definir todos os parâmetros necessários para treinamento da rede tais como: taxa de aprendizado, momentum, épocas e erro total desejado.

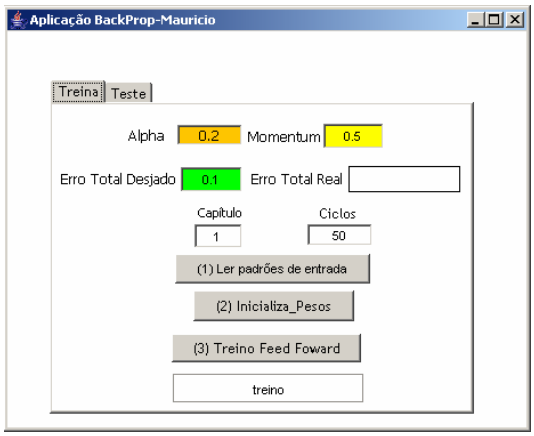

Figura 11 - Tela do software de treinamento da rede neural.

Durante todo o treinamento da rede neural os valores dos pesos são armazenados no banco de dados e mesmo que durante esta etapa ocorra algum problema no computador o treinamento pode ser reiniciado a partir do ponto onde foi encerrado. Outra característica que pode ser relevada neste programa é que o treinamento pode ser parado a qualquer momento pelo usuário e reiniciado do ponto onde foi interrompido.

Isto permite maior flexibilidade e segurança durante a fase de treinamento.

O banco de dados utilizado para armazenar os dados do treinamento é do tipo relacional e foi estruturado utilizando o Microsoft Access[6] contendo 3 tabelas: delta, "inout" e pesos.

	Nome do campo	Tipo de dados	Descrição
	padrao	Número	
	erro	Número	
	capitulo	Número	

Figura 12 – Estrutura da tabela delta.

A tabela chamada delta possui os seguintes campos: padrão, erro e capitulo que são utilizados para gravar os valores do erro parcial para cada padrão de entrada a cada época de treinamento da rede neural. Isto permite ao usuário que está executando o treinamento, consultar o erro parcial para cada padrão e verificar qual padrão esta causando maior erro no treinamento da rede.

Durante o treinamento da rede neural utilizada nesta pesquisa, a consulta do erro parcial, permitiu identificar que um padrão de entrada havia sido incluído erroneamente na lista de padrões de entrada. Este padrão que continha o erro foi removido e isto permitiu alcançar o erro total desejado.

	Nome do campo	Tipo de dados	Descrição
	padrao	Número	
	ytreina	Número	
	x0	Número	
	x1	Número	
	x2	Número	
	x3	Número	
	x4	Número	
	x5	Número	
	x6	Número	
	x7	Número	
	x8	Número	
	x9	Número	
	x10	Número	
	y	Número	
	capitulo	Número	

Figura 13 – Estrutura da tabela inout.

[6] Microsoft Access ® é um aplicativo para criação e edição de banco de dados criado pela Microsoft.

A tabela chamada “inout” possui os seguintes campos: padrão, “ytreina”, x0, x1, x2, x3, x4, x5, x6, x7, x8, x9, x10, y, capitulo.

Nesta tabela são guardados todos os padrões separados por capítulo, utilizados durante a etapa de treinamento. Isto permite ao instrutor treinar a rede neural com um padrão específico de navegação para cada capítulo. Esta é uma importante característica na implementação do sistema tutor porque fornece total flexibilidade ao instrutor na proposta do caminho de estudo a ser monitorado pelo STI.

O valor x0 é utilizado como “bias” para a camada de entrada da rede neural, os valores de x1 a x10 representam as entradas da rede neural e são empregados para representar cada tópico que deve ser navegado. O valor y é o valor desejado na saída da rede neural para cada padrão de entrada. A variável “ytreina” mostra ao instrutor qual o valor de saída após o treinamento da rede neural. O valor nomeado como padrão na tabela, representa o índice do padrão para aquele capítulo.

Os valores referentes aos padrões de entrada devem ser adicionados diretamente no banco de dados, logo o instrutor deve carregar manualmente todos os valores necessários para inicio do treinamento da rede. No STI criado para esta pesquisa o usuário é obrigado a abrir o banco de dados para realizar qualquer alteração referente aos padrões de entrada. Para realização de consultas foi criado o sistema de administração para o STI que está descrito no capítulo 7.

A tabela chamada de pesos armazena os valores dos pesos durante e depois do treinamento da RNA sendo que sua estrutura foi planejada para guardar o peso do “bias”, os pesos dos neurônios da camada de entrada e o os pesos dos neurônios da camada intermediária.

5.6. Software para o teste de resultados do treinamento da RNA

O software de treinamento da rede neural também é capaz de validar os padrões de entrada utilizados durante o treinamento assim como validar um novo conjunto de entradas que não foram previstas inicialmente. O teste pode ser executado ao final da etapa de treinamento da RNA uma vez que o valor do Erro Total desejado tenha sido atingido.

Figura 14 – Tela do programa para teste da RNA.

Esta característica disponibiliza ao usuário do programa o teste de eficiência da sua rede neural, imediatamente após a etapa de treinamento. No caso desta pesquisa foi muito importante porque permitiu a verificação do desempenho da RNA antes do início do desenvolvimento do STI.

Outro ponto importante a ser realçado é a reutilização de código através das classes criadas em linguagem Java. Todas as classes utilizadas para criar o aplicativo de treinamento e teste da RNA foram reaproveitadas no software do sistema tutor.

5.7. Etapas de treinamento da RNA:

Neste tópico estão descritas as etapas empíricas de treinamento da RNA.

O primeiro projeto da RNA possuía 10 neurônios na camada de entrada, 3 neurônios na camada intermediária e um neurônio na saída. Os parâmetros de treinamento iniciais foram 1.000 épocas, alfa igual 0,01, momentum igual 0,1 e o erro total desejado igual a 0,03. Após 1500 épocas de treinamento a rede neural não foi capaz de reduzir o erro total abaixo de 5,38758 mostrando assim sua ineficiência.

O gráfico abaixo mostra as épocas para a configuração da rede com 3 neurônios na camada intermediária.

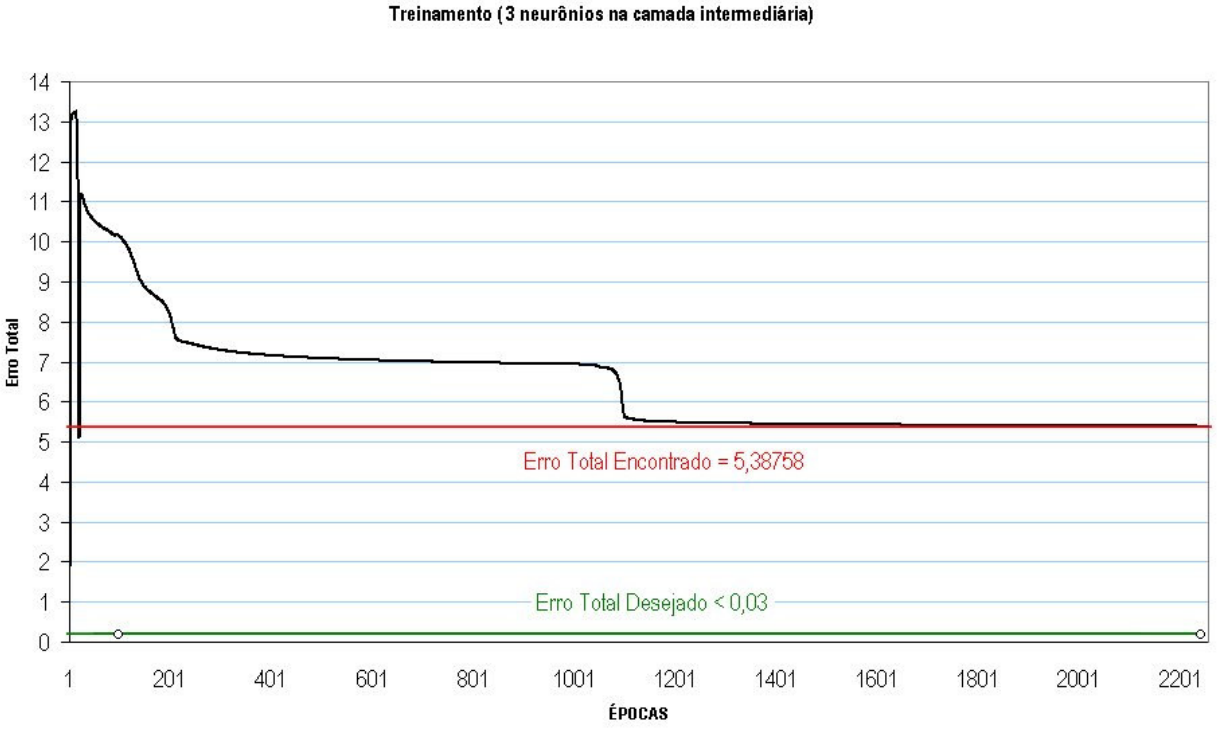

Figura 15 – Gráfico do Erro Total para 3 neurônios na camada intermediária.

A solução para atingir o Erro Total desejado, foi aumentar o número de neurônios na camada intermediária, inicialmente de 3 para 4, 5, 6, 7 até 12. Com 12 neurônios na camada intermediária rede foi capaz de responder com 100% de precisão a todos os padrões de treinamento e também foi hábil para generalizar a resposta para valores fora da coleção inicial de padrões. Os parâmetros de treinamento foram alfa igual 0,01, momentum igual 0,1, e o erro total desejado era menor que 0,01 que foi atingido após 1345 épocas.

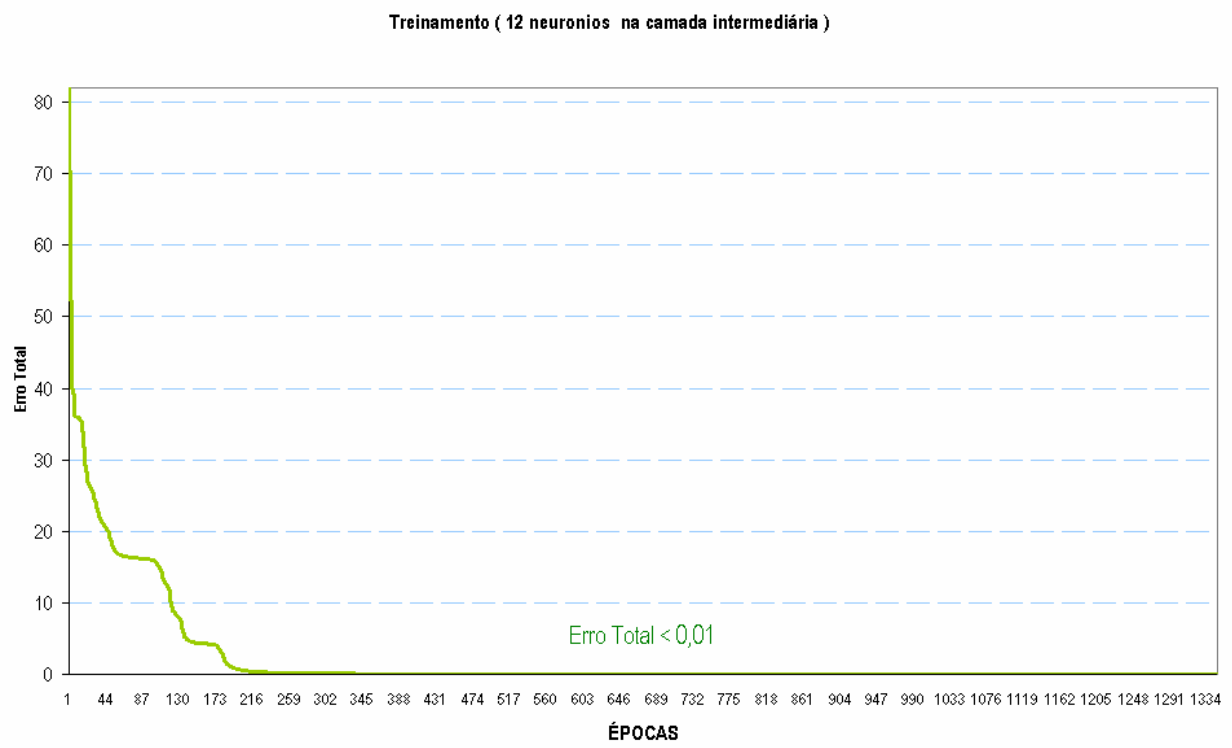

Figura 16 – Gráfico do Erro Total para 12 neurônios na camada intermediária.

Capítulo 6. Software do Sistema Tutor

Neste tópico está descrito como foi desenvolvido o programa do STI aplicado a esta pesquisa.

6.1. Recursos utilizados no desenvolvimento do STI

De acordo com Todd; Szolkowski (2003, p3-4) a linguagem Java surgiu no mundo da informação e tecnologia em meados de 1990, cerca de um ano após a Internet se estender ao grande público. Os "servlets" ou "applets" foram os primeiros programas que rodavam do lado do computador do cliente e não requeriam muito de um servidor sendo capazes de trazer algum tipo de animação para as páginas da internet. As aplicações típicas incluíam "applets" para manipulação de imagens e rolamento de texto na tela [8].

Quando era essencial o uso de aplicações do lado do servidor era necessário desenvolver aplicações CGI[7] ("Commom Gateway Interface") que traziam problemas sérios de desempenho para a aplicação, porque cada solicitação que entrava no servidor gerava uma aplicação CGI, como um processo separado, no próprio espaço de endereçamento e somente após o atendimento da solicitação é que o mesmo se desligava. Portanto um grande número de usuários resultava em muitos programas CGI executando concorrentemente o que podia afetar adversamente o desempenho do servidor.

Por fim foi desenvolvido um mecanismo onde era possível escrever programas que executavam do lado do servidor sem utilizar CGI, então nascia o "servlet" [8].

Embora os servlets fossem muito eficientes, não eram muito habilidosos para serem utilizados para desenvolvimento, pois era necessário muito conhecimento de linguagem de programação Java e HTML. Em sítios da internet que continham um grande número de servlets toda vez que uma das classes fosse alterada geralmente era necessário reiniciar o servidor para que a classe fosse recompilada e refletisse a modificação.

Isto para um "site" de grande porte era realmente uma dificuldade a ser superada.

[7] O CGI definiu um padrão para executar programas a partir de um servidor de Internet.

Resumidamente depois que o servlet API foi desenvolvido a Microsoft lançou as Active Sever Pages (ASP) que eram muito mais simples de trabalhar. As páginas ASP são uma mistura de script[8] e HTML[9], combinados para produzir conteúdo dinâmico. O desenvolvedor não precisa fazer nenhuma compilação.

Questões de portabilidade, porém, impediam que muitos sítios fossem desenvolvidos e instalados nesta plataforma. A única alternativa em Java eram esses servlets bastante desajeitados. Então na comunidade Java, foi inventada a Java Server Page, o que tornou o desenvolvimento de aplicações para Internet utilizando Java muito mais acessível, simples e direto. Com cada versão incremental de um novo padrão de JSP, as Java Server Pages tornaram-se progressivamente mais fáceis de desenvolver, manter e trabalhar.

Uma Java Server Page é o inverso da servlet, pois nela tínhamos código Java com HTML incorporado, agora temos código HTML com código Java incorporado. Uma página desta é primeira transformada em um servlet e então em um arquivo class. Esta classe então é instanciada para se tornar um objeto de servlet Java. Na primeira vez que a página for acessada todo este processo ocorrerá, em compensação, o segundo acesso será muito mais rápido praticamente instantâneo porque o objeto servlet já está criado.

Pelo fato dos recursos da linguagem Java estar disponível através da tecnologia Java Server Pages, esta serviu como plataforma base para o desenvolvimento do STI nesta pesquisa.

Utilizando uma página JSP também é possível acessar um banco de dados, recurso este considerado fundamental no caso do STI, pois permite resgatar as informações referentes ao aprendiz e os elementos chaves necessários para o funcionamento da RNA.

O acesso ao banco de dados feito pelo STI durante sua execução é através dos "drivers"[10] de ODBC[11] (Open Database Connectivity) providos pela linguagem Java. A tecnologia ODBC é um método padronizado de acesso a banco de dados desenvolvido em 1992 pelo grupo SQL Access com objetivo de disponibilizar o acesso a qualquer dado de qualquer aplicação independentemente do sistema de gerenciamento que criou, manipulou ou controlou originalmente o banco de dados [9].

---

[8] Script é um termo utilizado geralmente para linguagens de programação que possuem baixa complexidade em sua escrita e geralmente são programas interpretados.
[9] HTML é uma linguagem de marcação utilizada para estruturas texto ou documentos de multimídia onde é possível criar hiperlinks.
[10] Drive é um programa que interage em particular com um dispositivo ou software.
[11] ODBC é um padrão para acesso a diferentes tipos de banco de dados.
< http://dictionary.reference.com/search?q=Open%20Database%20Connectivity>

O ODBC consegue executar esta operação porque insere uma camada intermediária chamada “drive” do banco de dados, entre a aplicação e o sistema gerenciador do banco de dados (DBMS)[12]. O propósito desta camada é traduzir as consultas de dados da aplicação em comandos que o gerenciador do banco de dados entenda. Para que o ODBC funcione é necessário que ambos trabalhem com a mesma tecnologia. A aplicação deve ser capaz de lançar comandos e o gerenciador do banco de dados deve ser capaz de responder [9].

Para que uma aplicação possa acessar um banco de dados através dos drives ODBC é necessário configurar o sistema operacional do computador, onde está instalado o banco de dados, para que o mesmo reconheça que uma aplicação irá acessar os dados através destes drives. O banco de dados do STI foi estruturado utilizando o programa Microsoft Access.

Para que o STI possa ser acessado através de uma rede de computadores é necessário instalar em servidor um programa capaz de servir as páginas JSP e para tal propósito foi utilizado o Jakarta Tomcat[13] em sua versão 4.5. Este programa é um container do tipo internet que tem licença de uso livre, é facilmente configurável e implementa os padrões até o JDK 1.4 (Java Development Kit)[14] . Após desenvolver a aplicação basta uma cópia para a pasta de módulos e o início do serviço no servidor da rede de computadores. Para atualizações basta substituir os arquivos modificados ou então substituir toda a pasta e a aplicação estará atualizada.

O acesso a aplicação é feito através de qualquer navegador da internet apenas digitando o endereço no navegador. O acesso ao STI é feito especificamente através do endereço : http://localhost:8080/WebModule1/index.jsp.

Para esta pesquisa o Tomcat foi instalado em um computador Pentium IV com 512 de memória RAM e rodando o sistema operacional Windows 2003. A aplicação mostrou muita estabilidade durante toda a utilização com os aprendizes.

[12] DBMS é um programa gerenciador que facilita a criação, manutenção e utilização de um banco de dados eletrônico.
[13] O Jakarta Tomcat é um software de código aberto com a licença de uso de Apache software foundation. < http://www.apache.org/>
[14] JDK é um ambiente de desenvolvimento criado pela Sun Microsystem que inclui máquina virtual Java, compilador, depurador e outras ferramentas para desenvolvimento de aplicações em Java

Tabela 3 – Quadro resumo do critério de decisão para escolha do ambiente de execução do STI.

Tipo de aplicativo	Características					
	Servidor grátis	Tipo do Banco de dados	Dificuldade Configuração do banco	Ambiente de execução	Dificuldade no desenvolvimento do código Para acesso ao BD.	Dimensão do Acesso
Executável	N/A	Não testado	Não testado	local	Alta (código pronto mas sem tutorial)	Local
Paginas JSP(1) Internet	Sim	HSQLDB	ALTA Pouca documentação falta de experiência	Internet	Media Possível encontrar documentação	Internet
Paginas JSP(2) Internet	Não R$29,90/mês	MySQL	MEDIA Muita documentação	Internet	Media Muita doc. Pouca Experiência	Internet
Paginas JSP em rede	N/A	Access	Baixa	Rede atende teste alunos.	Baixa	Rede

Outro recurso utilizado, fundamental para o desenvolvimento do STI, foi o ambiente integrado de desenvolvimento JBuilder da Borland[15] . No caso do STI foi possível gerar a interface gráfica, criar e depurar o programa executável e as páginas JSP tudo dentro do mesmo ambiente de desenvolvimento.

Para desenvolvimento do STI o JBuilder foi importante porque além do ambiente integrado de desenvolvimento possui o servidor "Tomcat" integrado o que permite toda depuração do código Java, HTML simulando o ambiente real de funcionamento da aplicação.

[15] O JBuilder é um ambiente integrado de desenvolvimento criado pela Borland. <http://www.borland.com/us/products/jbuilder/index.html>

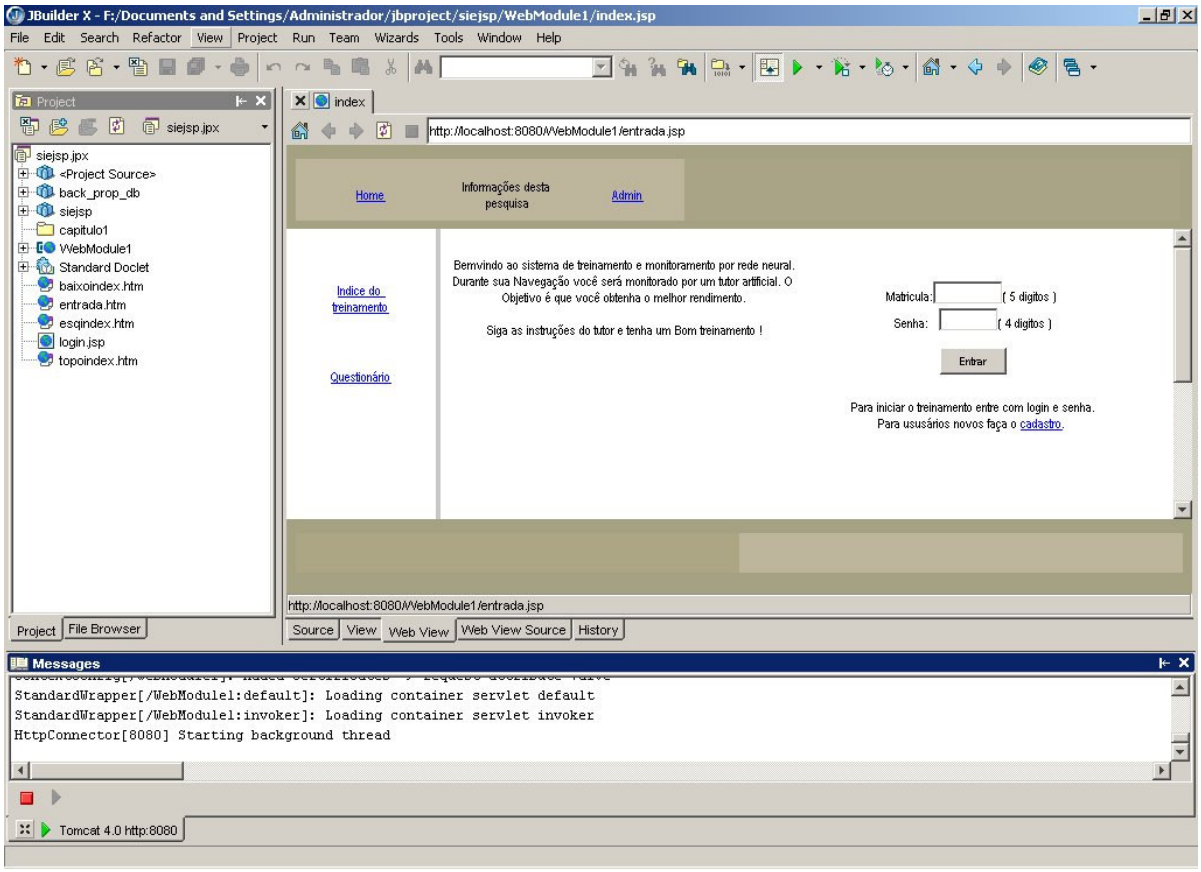

Figura 17 – Ambiente integrado de desenvolvimento JBuilder.

No ambiente integrado é possível visualizar os recursos utilizados na aplicação dentro de uma mesma janela. Este modo de trabalho permite uma rápida localização das páginas JSP, HTML e das classes utilizadas na aplicação.

O material de estudo foi feito utilizando o software Microsoft Office Power Point, esta ferramenta permite inicialmente gerar os eslaides e depois aplicar uma conversão para o formato HTML para que possa ser visualizado dentro do STI através do navegador.

A grande vantagem na escolha desta ferramenta é o ganho de produtividade, porque o material produzido anteriormente pelo instrutor pode ser reaproveitado, o que significa um grande ganho de tempo na produção de material didático para o STI.

No tópico 7.6 desta pesquisa, onde está descrito o sistema de gerenciamento do STI, será mostrado com mais detalhes que qualquer outro material produzido em formato HTML pode ser facilmente incorporado ao STI.

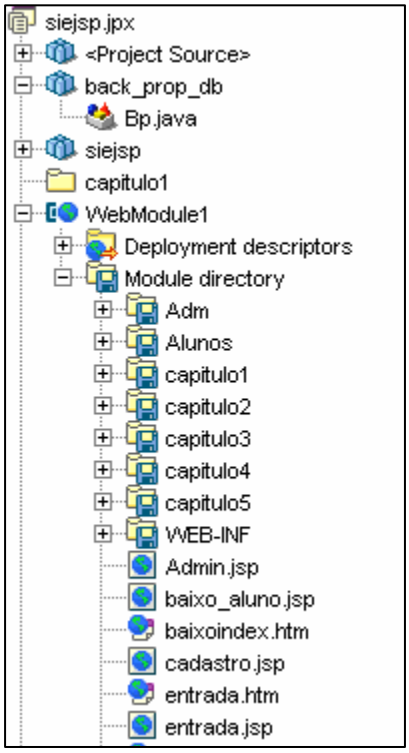

Figura 18 – Visualização dos recursos no ambiente integrado.

6.2. Tecnologia aplicada no desenvolvimento do STI

Segundo Todd; Szolkowski (2003, p157-159) existem dois modelos de arquitetura utilizadas em aplicações JSP denominadas Modelo 1 e Modelo 2.

No Modelo 1 a arquitetura é mais intuitiva, envolve o cliente que solicita a página JSP através de uma URL[16] e o servidor que devolve a resposta através da entrega de outra página JSP. A arquitetura do modelo 1 é descentralizada porque a próxima página a ser exibida é sempre determinada pela anterior. Não há necessidade de servlets nessa arquitetura.

O Modelo 2 trate-se de uma aplicação para Internet centralizada com aquilo que é denominado de servlet controlador que direciona as solicitações para JSP específicas. A determinação da JSP apropriada é feita através de parâmetros dentro de uma "string" de consulta. Esta arquitetura não é tão intuitiva como a do Modelo 1, pois de um modo geral segue o padrão chamado de controle do modelo de visualização (MVC).

A visualização é a apresentação da aplicação como um programa que centraliza as entradas. Todas as solicitações passam por um servlet controlador. O servlet controlador cria o que é necessário, para então passar para a JSP, só então é devolvida a resposta ao cliente. O padrão MVC é composto por modelo, visualização e controlador.

O modelo é o centro da funcionalidade da aplicação que implementa classes também chamadas beans[17] com métodos "get" e "set".

A visualização é a apresentação ou a interface da aplicação que em Java poderia, por exemplo, ser implementada por um tipo de interface utilizando componentes Swing [18], isto é, uma interface dentro de uma página do tipo HTML ou até mesmo uma linha de comando. Na interface de visualização é feita a entrada de dados para o modelo, embora não modifique o modelo. Esta interface recebe também as informações de eventuais modificações que ocorram no modelo e quando isto ocorre, se caracteriza então por uma página do tipo JSP.

O controlador está situado entre a visualização e modelo e dependendo daquilo que o usuário faz, então, pode criar ou modificar o modelo.

A aplicação deste modelo inicia quando uma solicitação passa pelo servlet controlador ou o controlador no caso do modelo MVC, então este servlet cria um objeto da classe requerida e passa para a aplicação de modo que os dados gerados por este objeto possam ser

[16] URL é o endereço global de um recurso ou documento na rede local ou Internet.
< http://www.webopedia.com/TERM/U/URL.html>
[17] Java Bean é um componente reutilizável de software escrito em linguagem Java.
<http://java.sun.com/developer/onlineTraining/Beans/Beans1/simple-definition.html>
[18] Swing é um conjunto de componentes da linguagem Java para o desenvolvimento de interface gráfica.

acessados a partir da página JSP. No caso do MVC as classes tornam-se o modelo que são gerados pelos “beans”.

Depois de criar ou modificar os objetos, a solicitação é encaminhada à JSP apropriada, que envia a resposta ao solicitante ou usuário. Normalmente as páginas JSP são o padrão de saída para visualização no MVC.

Não existe uma arquitetura considerada a ideal, portanto a decisão de qual modelo utilizar deve levar em consideração alguns aspectos para cada modelo.

No Modelo 1 as aplicações menores serão beneficiadas com esta arquitetura, não havendo a necessidade de recorrer a um servlet controlador se houver apenas algumas páginas JSP. Se não houver muitos detalhes na maneira como deve ocorrer o processamento enquanto o usuário navega entre as páginas, é provável que não haja benefícios em ter um controlador.

Um controlador pode tornar a aplicação desnecessariamente complexa.

As aplicações que requerem um menor grau no controle da segurança também podem ser construídas utilizando a arquitetura no Modelo 1. Um controlador pode ajudar a gerenciar a segurança, mas se não é necessário gerenciar a segurança, este controlador se torna desnecessário. A principal desvantagem do Modelo 1 é que na medida em que as aplicações construídas com esta arquitetura aumentam de tamanho e em número de páginas, torna-se mais difícil o seu gerenciamento, pelo fato da aplicação ter várias páginas e cada uma ser responsável por um trecho de execução da aplicação. Por esta razão, isto pode também, dificultar a depuração da aplicação.

O Modelo 2 pode realmente ser útil em grandes aplicações e seu principal benefício residem no fato da existência da centralização de todo o sistema em um controlador que gerencia toda a navegação.Sendo assim, a manutenção da aplicação torna-se mais simples. A segurança pode ser mais facilmente gerenciada, pois cada solicitação que passa pelo controlador central é verificada e credenciada para aquele determinado usuário.

A apresentação em diferentes idiomas, embora simples e direta, pode ser implementada mais facilmente quando o Modelo 2 é utilizado.

Com este modelo é possível também apresentar visualizações diferentes para diversos tipos de dispositivos de navegação, sendo assim, um tipo de visualização pode ser apresentado para um dispositivo de navegação móvel e outro tipo para navegação em computador de mesa. O Modelo 2 permite também uma construção mais modular do sistema e a realização de testes no sitio por meio de blocos funcionais discretos. Isto porque as classes são criadas e configuradas através de um único servlet controlador, opostamente ao que ocorre em uma aplicação do Modelo 1, onde diferentes páginas JSP criam suas classes.

Para esta pesquisa foi adotado o Modelo 1 com servlets controladores localizados em nas páginas JSP de navegação do sistema. A razão principal para escolha desta alternativa foi o fato de que um MVC dificultaria a passagem de variáveis através de sessão para regravação de URL como será mostrado no próximo tópico desta pesquisa.

Uma segunda razão é que todo sistema que gerencia a navegação e monitoração do aprendiz dentro do STI é composto por 10 páginas JSP, o que permite um fácil gerenciamento.

A terceira razão é que apesar do Modelo 2 não ter sido adotado em sua totalidade, parte de seu conceito foi aplicado no que se refere a redirecionamento de navegação, pois a página nomeada como Tópico.jsp concentra todo o redirecionamento e gerenciamento para os tópicos e capítulos de estudo dentro do STI. Por isso, em caso de modificação no sistema referente a redirecionamento, exclusão ou inclusão de capítulos, basta a modificação de uma página o que diminuiu a complexidade no caso de manutenção do sistema. Isto é muito semelhante ao servlet concentrador adotado no Modelo 2.

Quanto ao aspecto de segurança adotado nesta pesquisa, toda a validação de credenciais do aprendiz é feita no início do sistema na página nomeada como “login.jsp” e o aprendiz não entrará no sistema se não houver seu credenciamento desde seu início.

E finalmente, um MVC não pode ser adotado pelo fato de restringir os aspectos de navegação do aprendiz, pois neste sistema, caso o aprendiz deseje, pode navegar livremente pelos tópicos e estar ou não sendo monitorado pela rede neural durante sua navegação, deixando assim livre a escolha de uma URL, o que seria mais difícil se o MVC fosse aplicado pois este é mais centralizador.

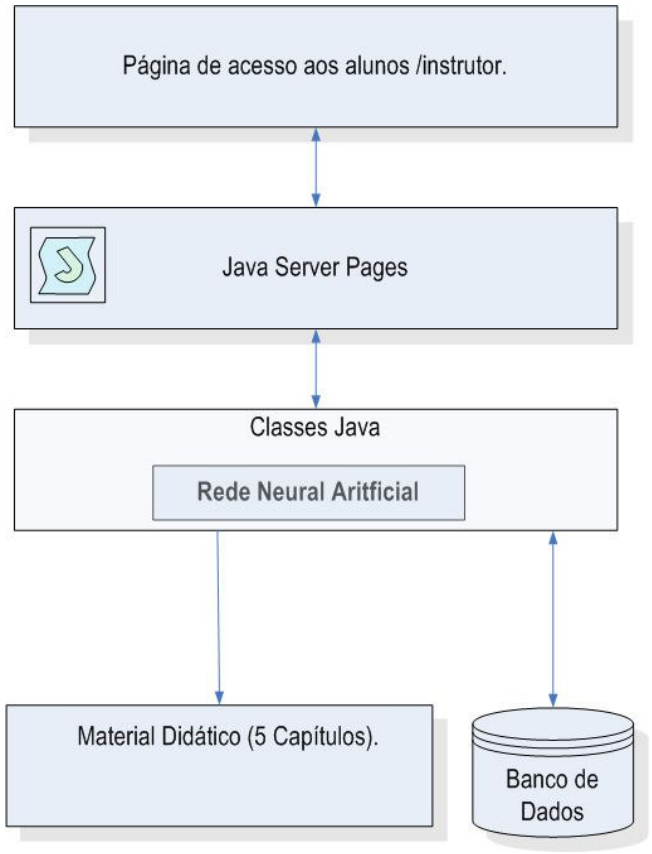

Figura 19 – Diagrama modelo adotado na aplicação STI

O acesso aos valores armazenados no banco de dados tem importância crucial nesta aplicação.

Para isto é necessário que haja uma maneira de acessar ou modificar os valores armazenados no banco através das páginas JSP. Como já foi mostrado no tópico 7.1 é necessário que haja uma interface entre o meio de acesso e os dados.

Uma maneira de tornar possível o acesso a estes dados é um programa de interface para a aplicação chamado de API [19] , no caso da linguagem Java, foi criado a JDBC que executa a conectividade com banco de dados.

Esta interface é parte da linguagem a partir da versão 1.0.2 onde os pacotes principais que contém as classes e objetos disponíveis para utilização são: java.sql e javax.sql [8].

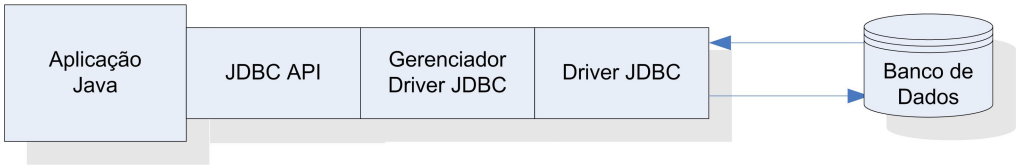

Figura 20 – Arquitetura básica JDBC.

O “drive” JDBC é responsável por interagir com o próprio banco de dados por isso fica claro que o acesso ao banco não é feito diretamente pela aplicação e sim pelo gerenciador.

Nesta aplicação o “drive” utilizado é definido como do Tipo 1 ou uma ponte JDBC-ODBC que mapeia a conectividade ao banco de dados (JDBC) para conexão aberta ao banco (ODBC) permitindo acesso aos dados. ODBC é o mecanismo utilizado para permitir o acesso ao banco de dados para as aplicações que executam no sistema operacional Windows.

O kit de desenvolvimento para Java é distribuído com um “drive” de ponte que permite que a aplicação possa interagir com ODBC, que por sua vez interage com o banco de dados. O “drive” JDBC precisa ser registrado e isto é feito durante a sua carga na aplicação.

O STI utiliza este “drive” e dentro de uma página JSP a sua carga é feita utilizando a declaração Class.forName("sun.jdbc.odbc.JdbcOdbcDriver").

Isto carrega o “drive” na máquina virtual, fazendo com que o inicalizador estático dentro da classe “drive” seja executado. O inicializador contém a linha de código para

[19] API é um conjunto de ferramentas, protocolos e rotinas para construção de um programa. <http://www.webopedia.com/TERM/A/API.html>

registrar o “drive” com o gerenciador. A partir de então é necessário abrir uma conexão com o banco de dados e então realizar as operações possíveis através de consultas no padrão SQL.

6.3. Sistema de navegação do STI.

Toda esta aplicação foi desenvolvida utilizando o poder do objeto sessão agregado a JSP, o que permite grande capacidade na recuperação das informações dos “hiperlinks” navegados pelo usuário. O protocolo http é conhecido como um protocolo sem estado, porque para cada solicitação do cliente, é aberta em uma conexão separada e os servidores não mantém quaisquer tipos de informações contextuais que poderiam formar a base para um diálogo entre o cliente e o servidor. Existem várias soluções para contornar este problema, uma delas a gravação de “cookies” no computador do cliente. Os “cookies” são pequenos arquivos enviados pelo servidor e armazenados no computador do cliente. Esta solução apresenta um grave problema porque depende totalmente do cliente habilitar a permissão para gravação destes arquivos em sua máquina.

A solução considerada nesta pesquisa é chamada de regravação de URL. Neste caso cada “hiperlink” de texto navegado pelo usuário acrescenta no final da URL informações extras requeridas pelo servidor. Essas informações extras identificam o cliente para o servidor e carregam muito valor agregado [8]. Um exemplo de regravação de URL é: http://servidor/pagina.jsp?aluno=1234. Este exemplo mostra que no momento que esta URL for ativada a variável aluno é carregada com o valor igual a 1234.

Do lado do servidor esta sessão é lida de duas maneiras. Um delas é a leitura através do comando request.getParameter(“nome da variável da sessão”) dentro da página JSP. Após a leitura da sessão, a variável pode ser empregada dentro do programa através de conversões de tipo e assim ser utilizada para pesquisa no banco de dados, cálculos ou qualquer outro tipo de lógica desejada dentro do programa.

Uma outra maneira que também disponibiliza uma variável de sessão durante a utilização do programa é a atribuição da variável através de um objeto de uma classe na aplicação.

Para tal é necessário criar funções membros dentro de uma classe que atribuam e retornem os valores das variáveis que se deseja manter na sessão. Uma vez criado um objeto desta classe podemos invocar um método nativo chamado de session.setAttribute(“nome do objeto”) para atribuir o valor à sessão e utilizá-lo em qualquer outra página. Para recuperar os valores das variáveis atribuídas à sessão desta maneira invocamos o método

session.getAttribute(“nome do objeto”) dentro de qualquer outra página JSP do contexto e a informação estará disponível.

A regravação de URL e os objetos de sessão foram extensamente utilizados nesta pesquisa. É através da regravação, que é feita a carga do tópico que está sendo estudado pelo aprendiz, para o STI e assim a rede neural artificial é alimentada adequadamente para executar seu julgamento. O fluxo de navegação permitido pelo STI é mostrado na figura 21.

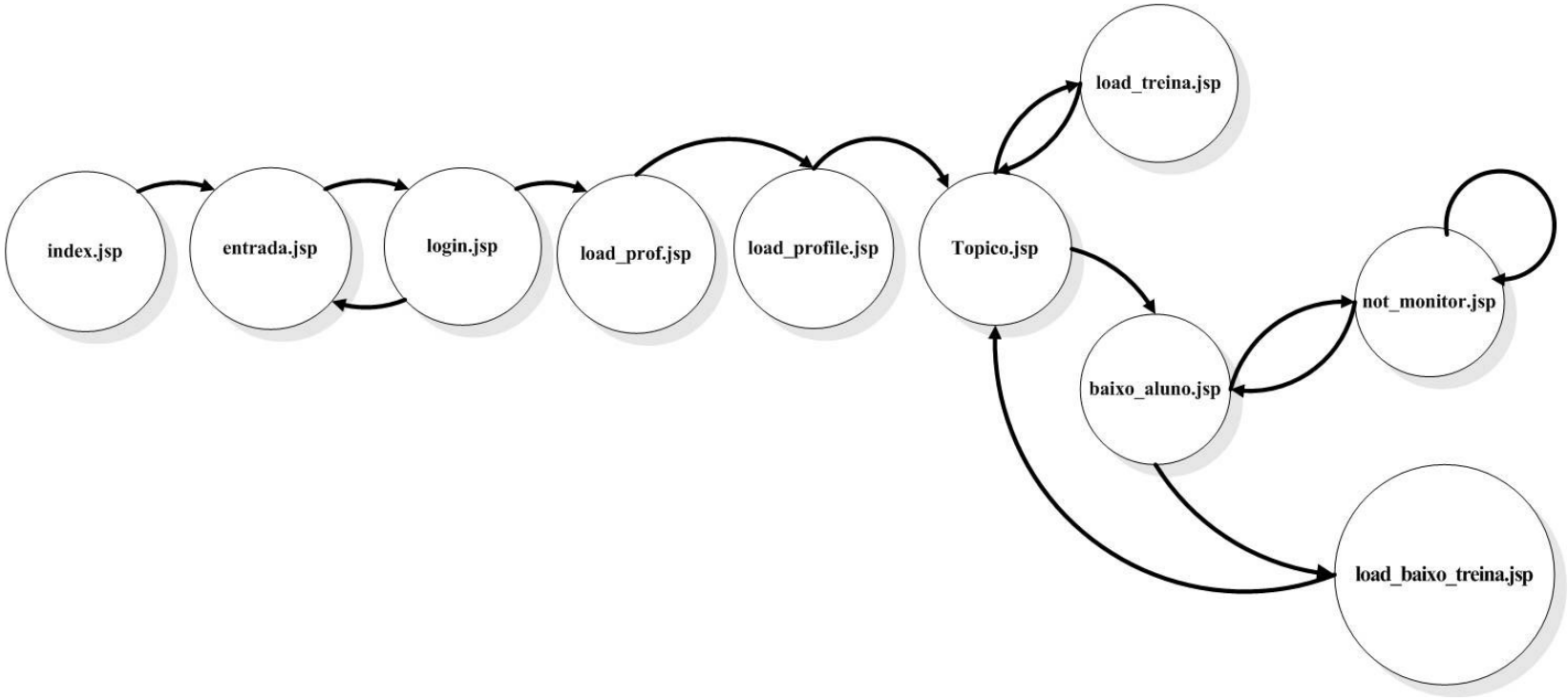

Figura 21 – Diagrama de estados do STI para navegação.

A entrada no STI é feita por “index.jsp” que carrega uma página com 3 quadros. No quadro central está localizado a página “entrada.jsp” com os campos de validação do aprendiz para validação no sistema.

Sua entrada é validada pelo número de matrícula e senha sendo que estes campos são numéricos e passados através do formulário para a página nomeada login.jsp onde efetivamente é feita a validação.

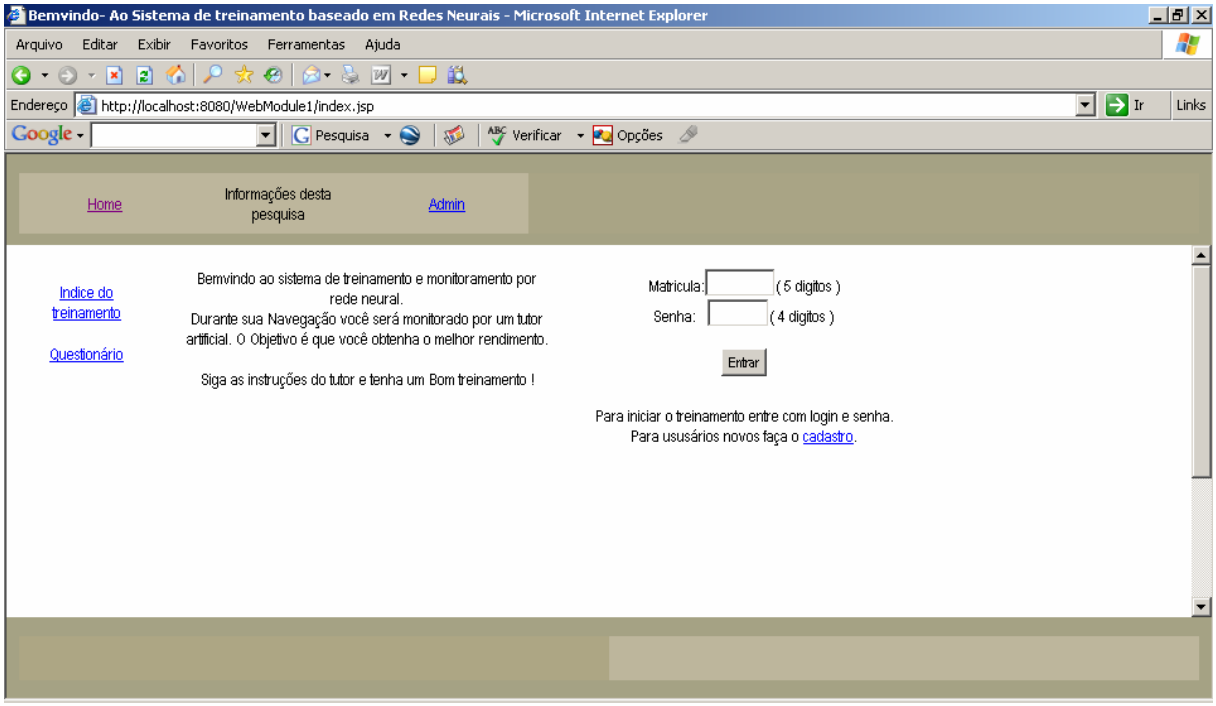

Figura 22 – Interface de entrada do STI.

Uma vez passados os dados para a página “login.jsp” é feita a validação do número de dígitos na matrícula e senha. Caso o aprendiz tente entrar no sistema e seu número de matrícula não foi encontrado no banco de dados o mesmo será redirecionado para uma página chamada “cadastro.jsp”.

Na página de cadastro o aprendiz novamente digita seu número de matrícula, escolhe uma senha para utilizar durante todo o aprendizado, digita seu nome e envia o formulário para uma página chamada “registra.jsp”. A página de registro faz toda a validação dos campos buscando no banco de dados se o número de matrícula ou senha não estão duplicados.

Uma vez passado pela validação, então o aprendiz é cadastrado no sistema e automaticamente são gravados sua matrícula, senha, nome, status e nível. As variáveis de registro do aprendiz denominadas status e nível significam que o aprendiz será inicialmente monitorado pela rede neural e o mesmo é considerado como tendo um nível de conhecimento básico para aquele capítulo a ser estudado, ou seja, é um iniciante. Estes dados são gravados em uma tabela chamada login. Este aprendiz também tem registrado na tabela "status_cap" e "status_top" o capítulo e tópico iniciais de estudo que recebem o valor de registro igual a um, pois o mesmo irá iniciar seus estudos. Uma vez que os dados foram gravados com sucesso, este aprendiz também é registrado em uma tabela no banco de dados chamada Diário, que tem o objetivo de registrar a data e hora toda vez que o aprendiz acessar o STI para estudo. Estes dados podem ser utilizados mais tarde pelo instrutor para acompanhamento da freqüência de estudo do aprendiz.

Se o aprendiz já foi cadastrado no sistema o processo de registro não é necessário, então é feita a validação de suas entradas e o mesmo é direcionado para a página chamada "load_prof.jsp" para iniciar os estudos.

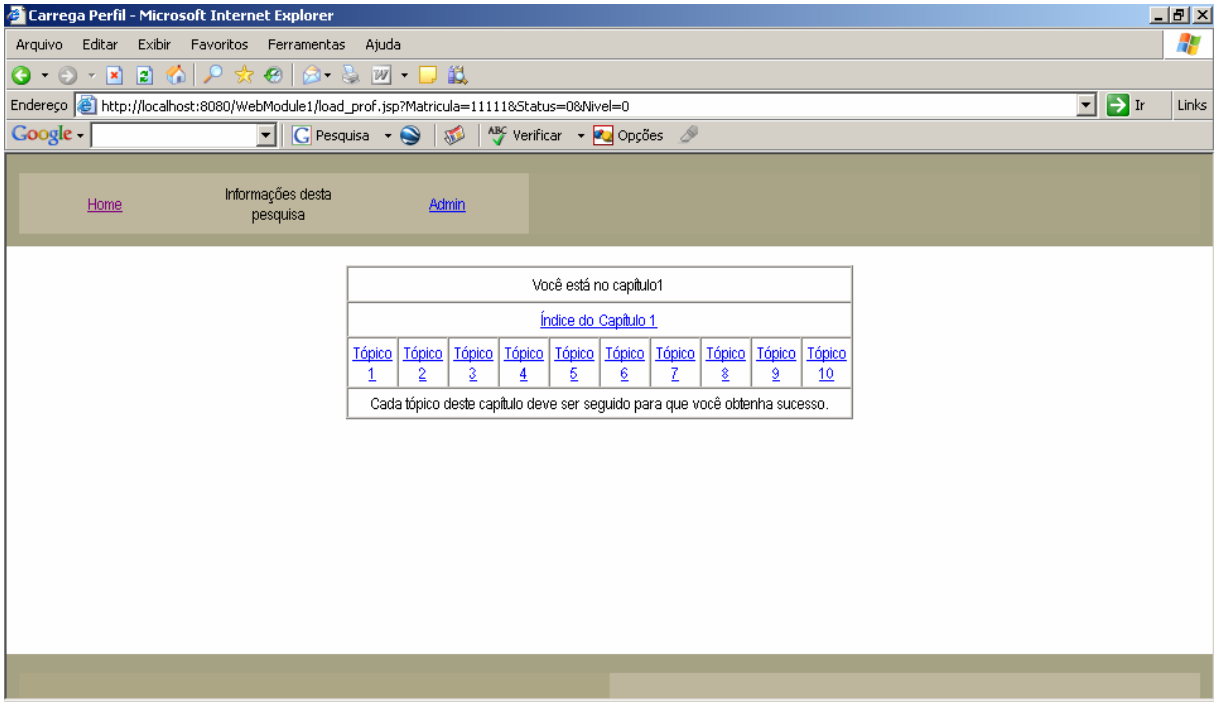

Figura 23 – Interface de carga de tópicos.

Esta página carrega no quadro central a página “load_profile.jsp”, que se encarrega de montar a primeira interface de visualização dos tópicos a serem estudados. A partir desde momento assim que os tópicos forem sendo ativados é feita a chamada da página “Tópico.jsp”.

Os valores tais como nível do aprendiz, status de monitoração, matrícula, capítulo e tópico de estudo são sempre carregados, pois o STI sempre considera a carga de dados que representa exatamente a situação de estudo do aprendiz naquele momento em que está no sistema.

Uma vez construída a interface de estudo, para cada ativação de tópico é chamada a página “load_treina.jsp”, que é o centro do STI, através dela a rede neural artificial entra em ação para rastreamento do aprendiz.

A página “load_treina.jsp” grava o “hiperlink” ativado pelo aprendiz, verifica os “hiperlinks” já estudados, monta as entradas para que a rede neural julgue o “hiperlink” ativado pelo aprendiz naquele momento, carrega os pesos de treinamento para aquele capítulo em específico e executa o algoritmo de retro-propagação da rede neural artificial. Como resultado temos o julgamento da rede neural. Caso a rede neural julgue que aquele tópico possa ser estudado libera então o conteúdo de estudo para visualização.

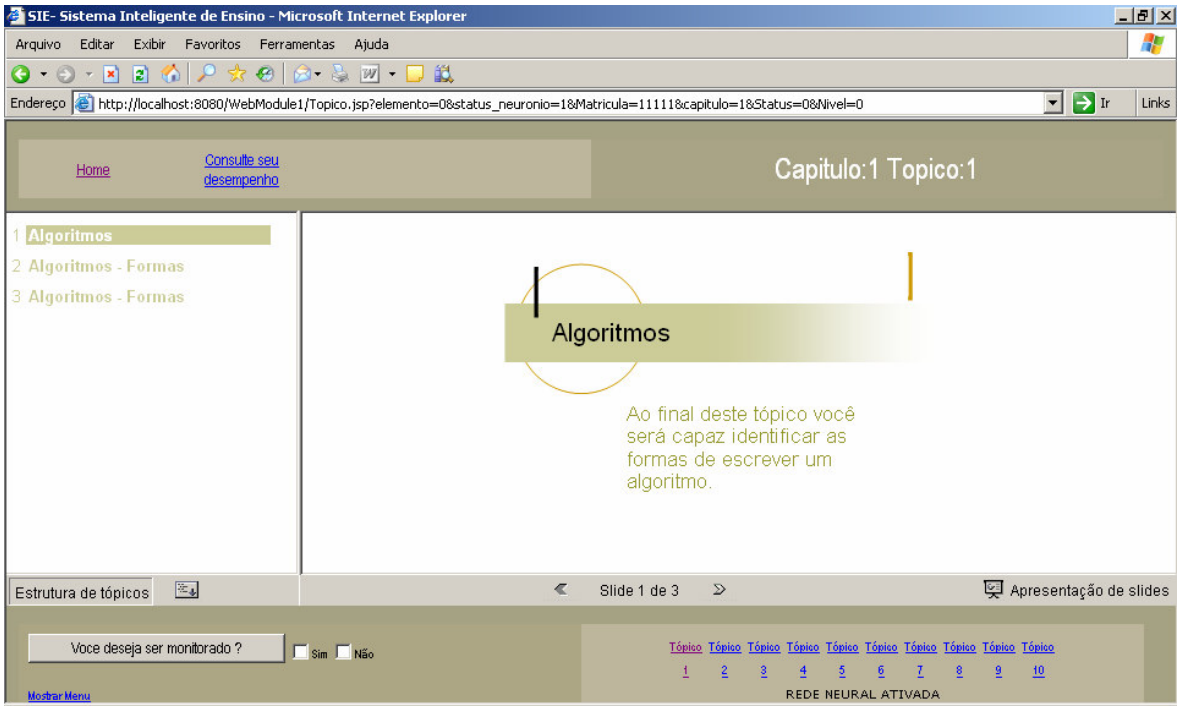

Figura 24 – Interface para os tópicos de estudo.

Uma vez liberada a visualização do tópico os itens a serem estudados aparecem do lado esquerdo do quadro principal e o aprendiz pode navegar livremente até que tenha concluído os sub-tópicos. Para ativar um novo tópico basta navegar pelo “hiperlink” de texto apresentado no quadro inferior e no canto direito da interface. A apresentação dos tópicos na parte inferior da interface é feito pelas páginas “baixo_aluno.jsp” e “load_baixo_treina.jsp”, que executa as mesmas funções descritas para a página “load_treina.jsp”.

Este processo é continuo e se repete toda vez que um tópico é ativado. Caso a RNA tenha liberado o tópico para estudo, o mesmo é gravado no banco de dados para que o processo de julgamento do sistema considere sempre a situação mais atualizada do aprendiz.

Durante a carga de cada capítulo o nível de conhecimento do aluno também é considerado. O valor zero carregado na gravação de URL significa que o aprendiz tem acesso ao material didático em nível básico e valor um significa material didático em nível avançado.

Este valor varia em função do resultado da avaliação feita no final de cada capítulo.

Isto permite ao aprendiz estudar com diferentes níveis de material didático e ao instrutor proporciona a flexibilidade na estratégia de ensino a ser adotada.

Para o STI aplicado nesta pesquisa foi desenvolvido somente o material didático para o nível básico, mas o sistema está preparado para receber o material para nível avançado.

A página “baixo_aluno.jsp” tem outra função muito especial, a de carregar no quadro inferior direito da interface, os campos de seleção que permitem ao aprendiz optar por ser ou não monitorado pelo STI. Caso o aprendiz escolha a opção de não ser monitorado, o sistema muda o status do aprendiz alterando o valor desta variável de “zero” para “um” na tabela “login”. Os menus de navegação devem então ser re-exibidos na parte inferior da página de quadros para isto o aprendiz deve ativar o “hiperlink” Mostrar Menu.

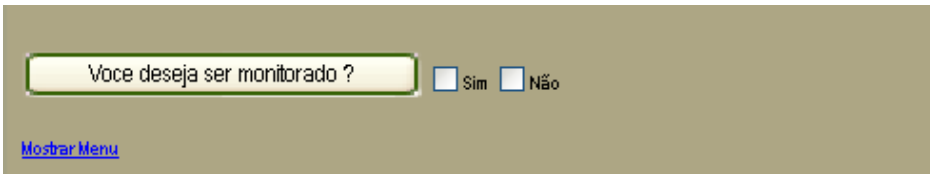

Figura 25 – Menu de seleção para alterar o status da monitoração do STI.

A qualquer momento o sistema permite que o aprendiz passe sua monitoração de ativa para inativa. Quando a monitoração da rede neural estiver inativa o tópico e capítulo continuam sendo gravados no banco de dados pelas páginas load_treina.jsp ou load_baixo_treina.jsp apenas para efeito de consulta futura pelo instrutor dos caminhos alternativos percorridos pelo aprendiz.

Quando o aprendiz está sendo monitorado e o tópico ativado não faz parte das regras de estudos determinadas pelo treinamento da RNA, uma mensagem de alerta é exibida indicando que o aprendiz não pode seguir naquele caminho e sugere que outro tópico seja ativado para que a sua seqüência de estudos possa continuar. Nesta mesma página são recarregados os tópicos para que o aprendiz faça uma nova opção.

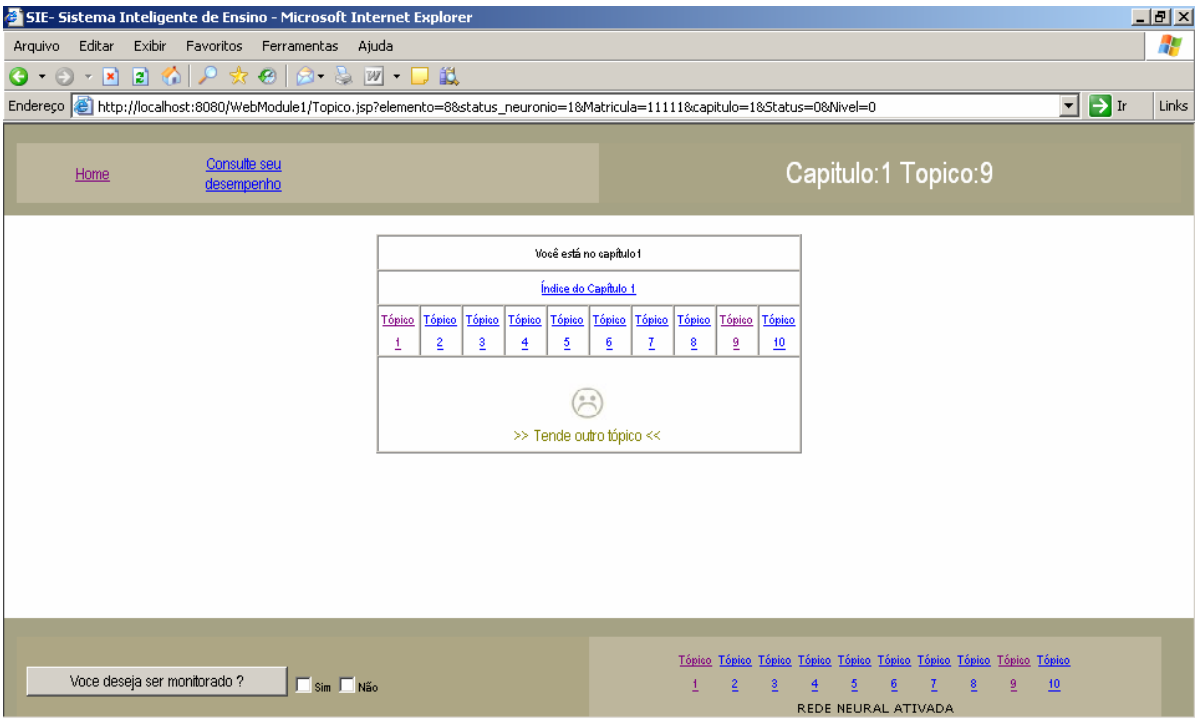

Figura 26 – Mensagem de alerta para aprendiz monitorado pelo STI.

No quadro superior da interface de navegação o aprendiz pode consultar seu desempenho a qualquer momento durante os estudos. A página ativada exibe suas notas até aquele momento.

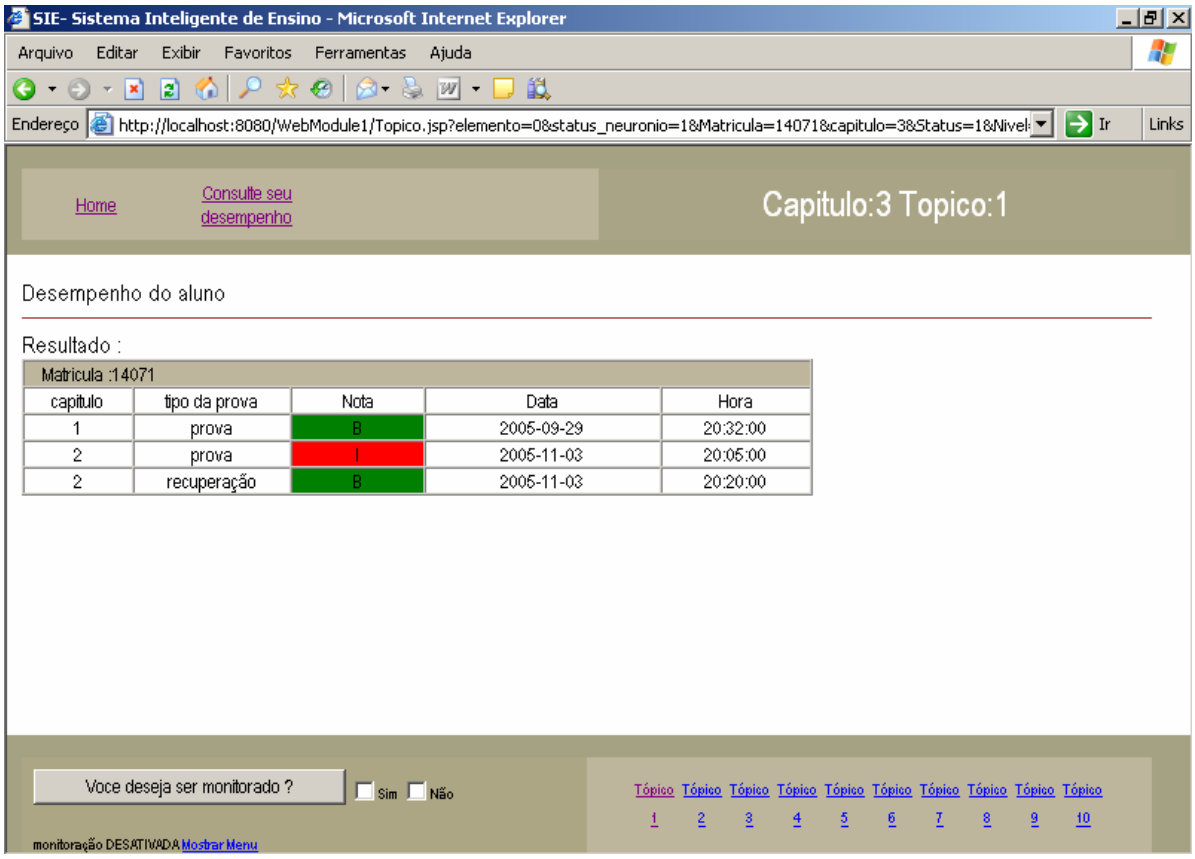

Figura 27 – Interface para consulta de desempenho.

6.4. Sistema de avaliação do STI

O sistema de avaliação do STI é feito de modo convencional, onde, dois cenários podem ser aplicados ao aprendiz. No primeiro cenário, ele percorre todo o treinamento com a monitoração ativada e atinge o último tópico de cada capítulo para realizar a avaliação. No segundo cenário o aprendiz opta por não ser monitorado e pode fazer a avaliação navegando diretamente para o tópico 10 do capítulo.

O tópico 10 da interface de navegação é sempre responsável pela chamada da avaliação, que é executado pela página nomeada como “topico10.jsp”.

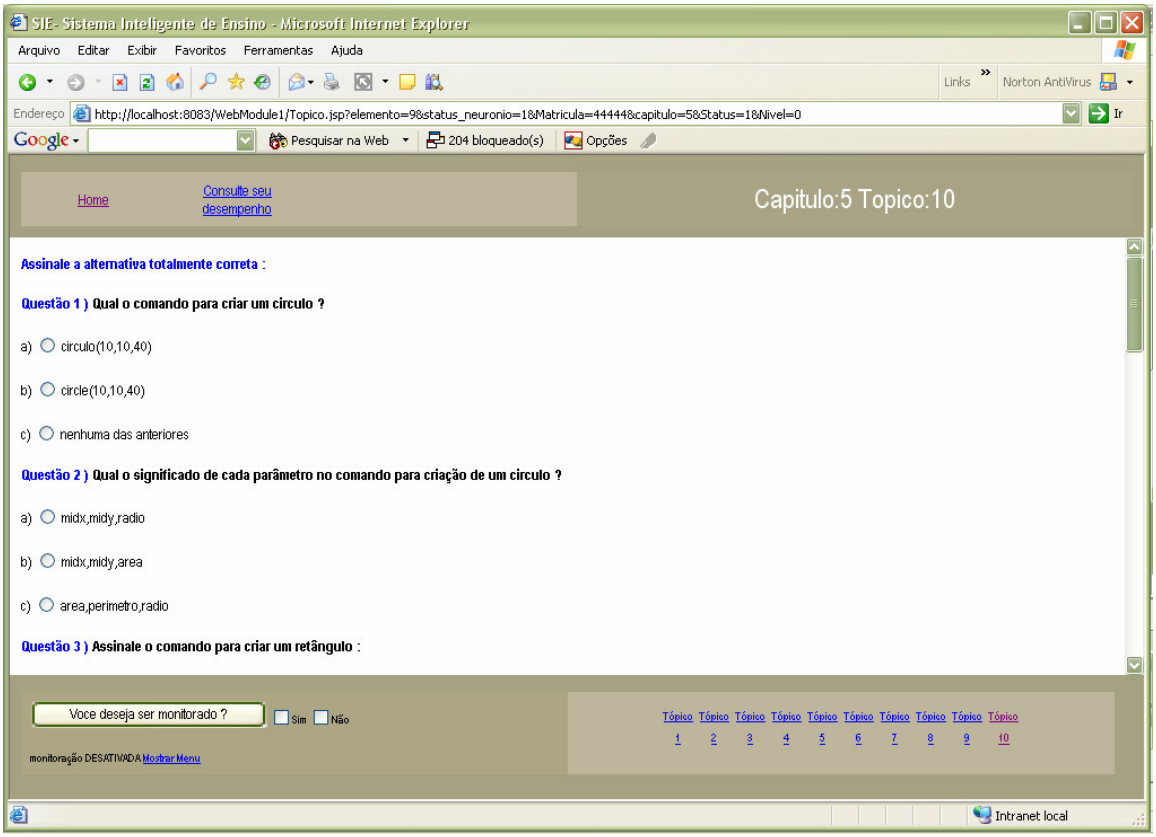

Figura 28 - Página de avaliação do STI.

A página de avaliação apresenta dez questões, cada uma contendo três alternativas de múltipla escolha, sendo que somente uma delas é a correta. Quando a prova é preenchida e enviada para correção, é invocada a página chamada “cap_prova.jsp”. Esta página, primeiramente, lê as repostas assinaladas no formulário para verificar se houve alguma questão que foi deixada em branco pelo aprendiz.

Em seguida monta uma matriz com todas as repostas e invoca a função membro da classe “Prova” que verifica se a avaliação esta sendo realizada pela primeira ou segunda vez.

Se a avaliação estiver sendo realizada pela segunda vez, o sistema registra no banco de dados que está é uma avaliação de recuperação. Indicar que a avaliação é de recuperação serve para o controle do instrutor, que pode verificar através do sistema de administração, se o aprendiz necessita de auxílio extra naquele capítulo de estudo.

Sempre que a prova for realizada pelo aprendiz mais de uma vez ficará registrado no banco de dados que aquela prova foi de recuperação. Este tipo de registro no banco de dados auxilia também o instrutor a verificar se foi satisfatório o desempenho do aprendiz que optou por não ser monitorado pelo sistema e se o caminho escolhido o levou ao sucesso.

O próximo passo da página “cap_prova.jsp” é invocar outra função membro da classe chamada “grav_Respaluno”, para gravar no banco de dados as respostas, tipo de prova, data, horário e capítulo.

O último passo é comparar as repostas assinaladas pelo aprendiz com as respostas do gabarito feito pelo instrutor, o que resultará na nota ou conceito do aprendiz para aquele capítulo. No final do processo é mostrado para o aprendiz as questões assinaladas corretamente e seu conceito final.

O aprendiz só poderá prosseguir os estudos se obteve acerto de pelo menos 50% das questões, então o STI libera o aprendiz para o próximo capítulo.

A liberação do próximo capítulo é feita pela outra função membro da classe Prova chamada “atualiza_Capitulo”, que atualiza no banco de dados a tabela “status_cap” para o correspondente aprendiz, incrementado o número capítulo a ser estudado.

Caso o aprendiz acerte um número menor que 50% do total de respostas então permanecerá no mesmo capítulo. A partir deste momento o aprendiz pode navegar em qualquer um dos tópicos do capítulo para sanar as suas dúvidas, pois já percorreu todos os tópicos de estudos anteriormente. Neste sistema ficou estabelecido que o próprio aprendiz deve verificar dentro do capítulo, a razão pelo qual seu desempenho não foi satisfatório, baseando-se nas repostas que não foram assinaladas corretamente e que o STI indicou ao final da avaliação.

6.5. Sistema de gerenciamento do STI

O sistema de gerenciamento do STI foi desenvolvido com o objetivo de facilitar o trabalho do instrutor na consulta do desempenho individual ou do grupo de aprendizes, na investigação de caminhos percorridos sem monitoração e para um novo treinamento da RNA.

No topo da interface principal de navegação o sistema de administração é acessado através do “hiperlink” “Admin”.

O primeiro passo ao entrar no sistema de gerenciamento é a validação do chamado administrador que é feito pela página “Admin.jsp”.

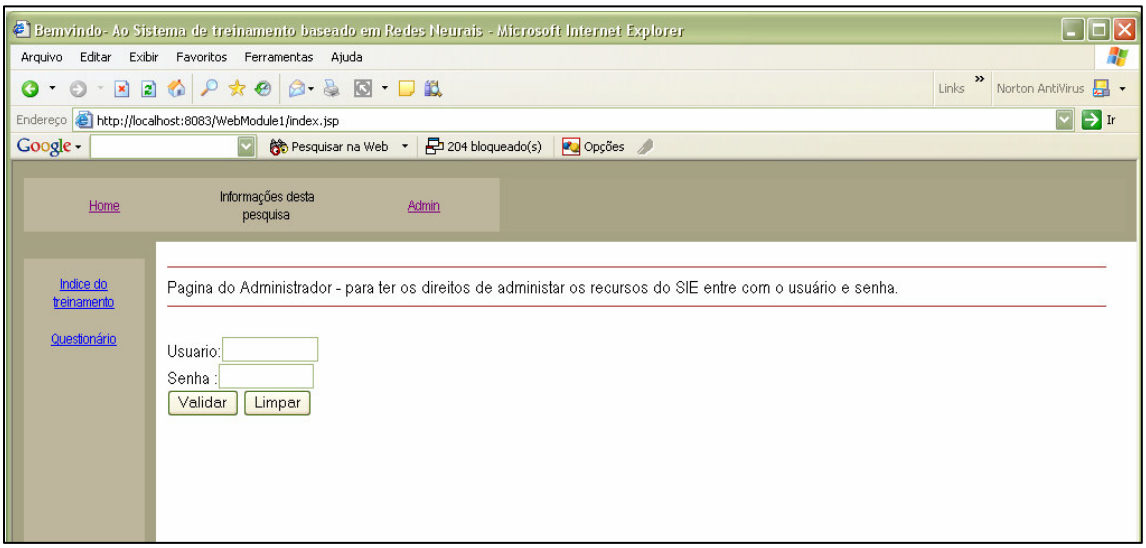

Figura 29 - Página de administração do STI

Esta página faz a validação do usuário e senha, previamente cadastrados no sistema e libera através da página “adm_tools.jsp” o acesso aos “Parâmetros da rede neural” e a “Consulta de desempenho dos alunos”.

Quando ativado o “hiperlink” “Parâmetros da rede neural” a página “adm_sie.jsp” é executada e permite a consulta aos padrões utilizados durante o treinamento da RNA através do “hiperlink” “Consultar todos os padrões” e aos alunos que foram rastreados pelo sistema quando não estavam sendo monitorados pela RNA através do “hiperlink” “Rastrear aluno(s) não monitorado(s)”.

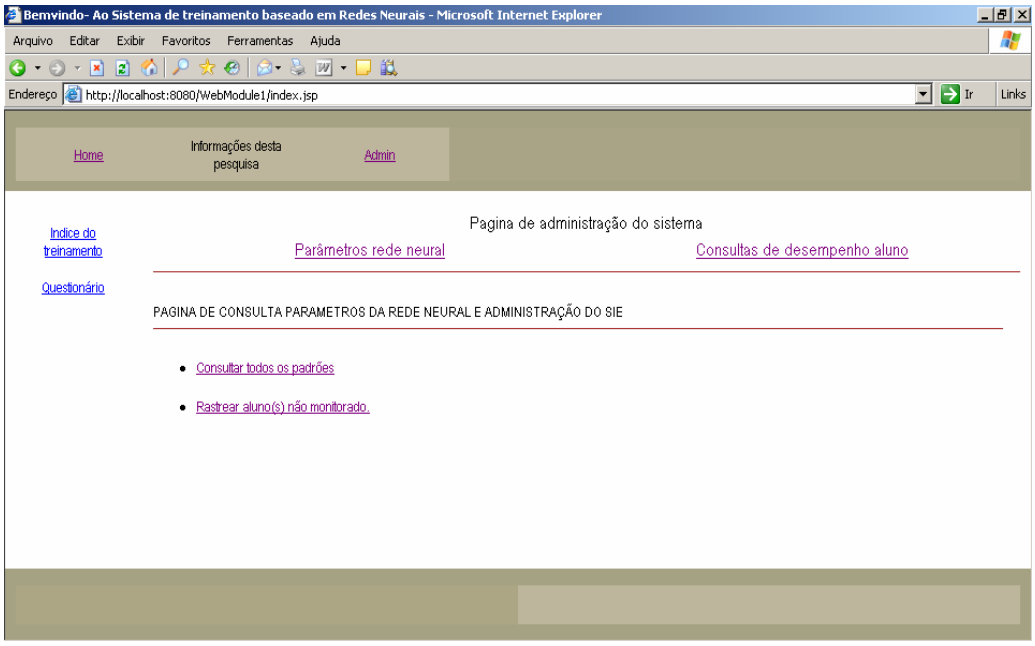

Figura 30 - Página de administração do STI

Ao ativar o “hiperlink” “Consultar todos os padrões” a página chamada “adm_rna.jsp” executa a consulta e retorna os dados para visualização.

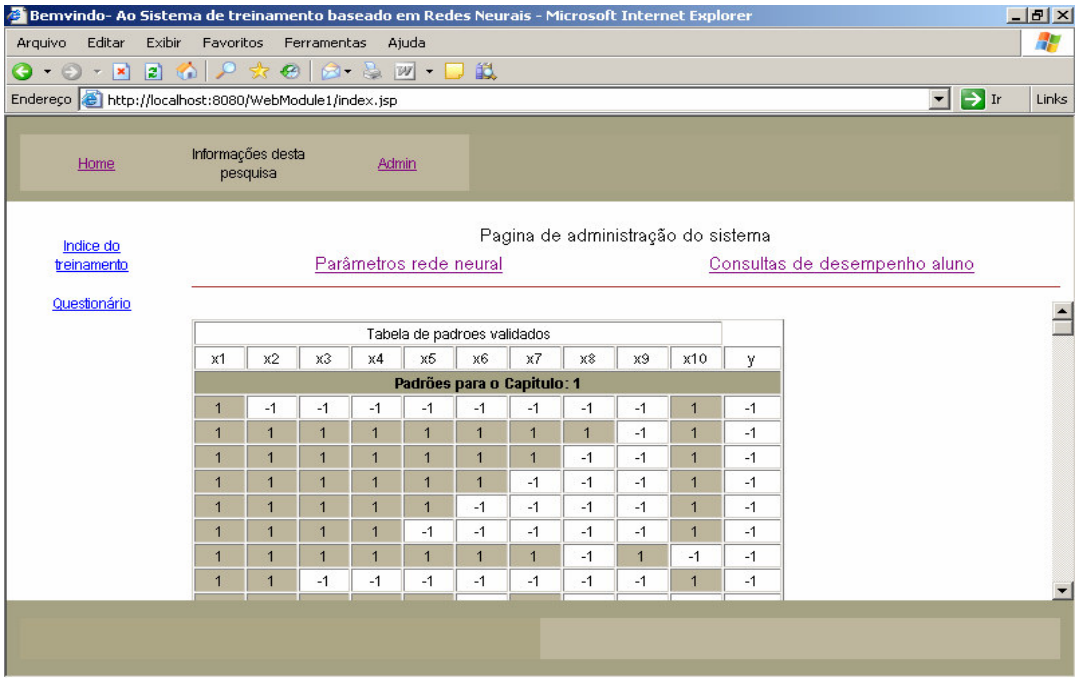

Figura 31 – Interface para consulta aos padrões de treinamento da RNA.

Quando o instrutor ativar o “hiperlink” de texto “Rastrear aluno(s) não monitorado(s)” a página “adm_raster.jsp” exibe uma lista dos aprendizes que optaram por estudar por um caminho alternativo.

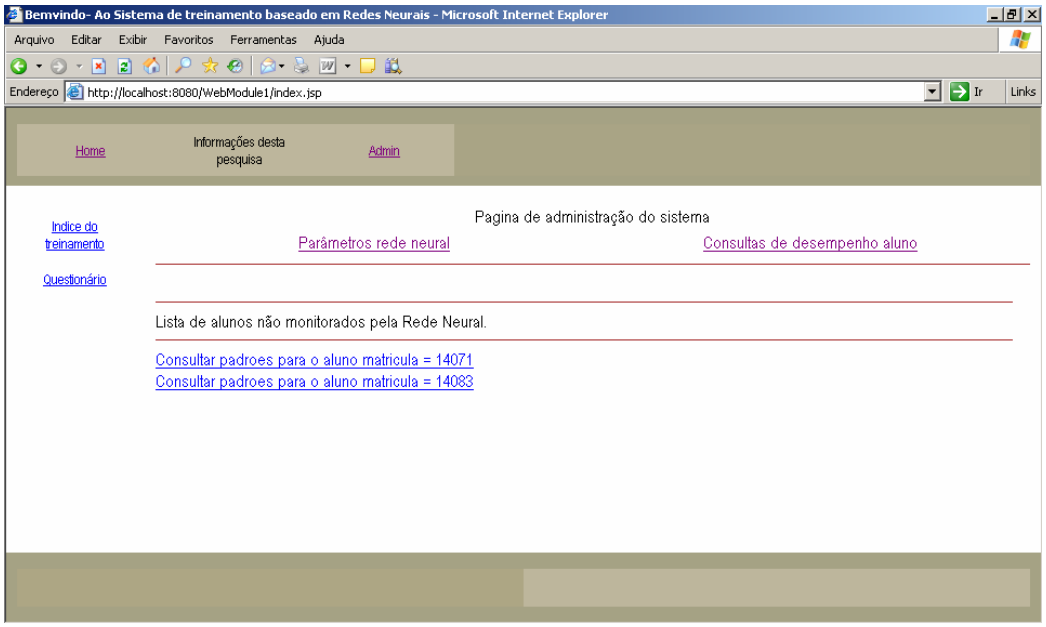

Figura 32 – Lista de aprendizes não monitorados.

Ao “clicar” sob o número de matrícula na lista exibida o instrutor realiza a consulta do caminho percorrido por aquele aprendiz.

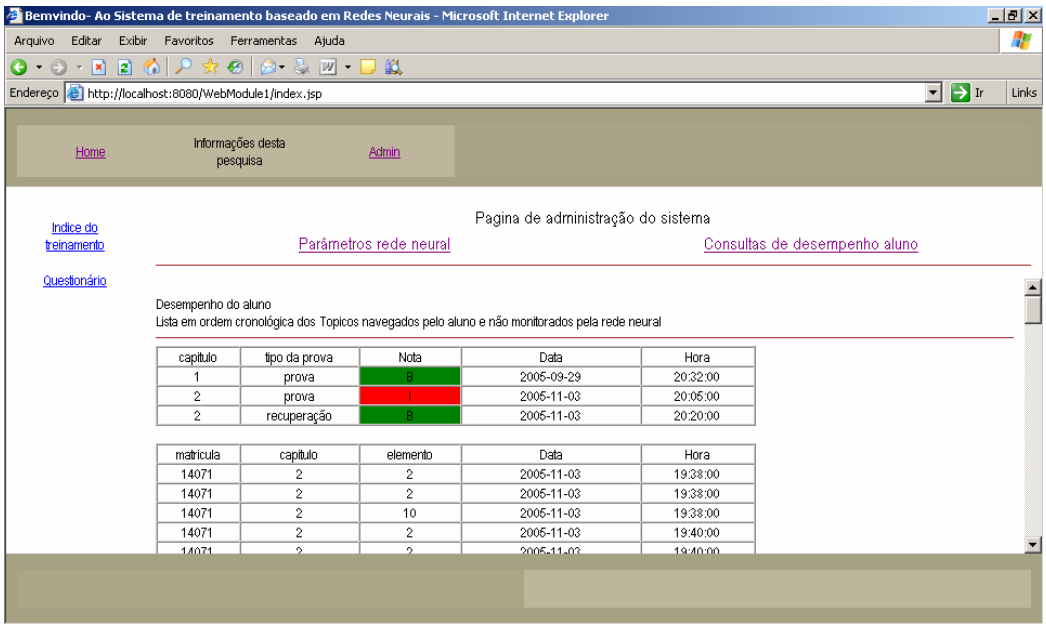

Figura 33 – Lista de caminho não monitorado por aprendiz.

Uma vez tendo consultado o caminho alternativo, através de “adm_raster.jsp”, se o instrutor desejar atualizar o treinamento da RNA, com este novo padrão, deverá ativar o “hiperlink” de texto “(Avançar) na validação dos padrões” mostrado na parte inferior da mesma página e então invocará a página “valida_padrao.jsp” que permite gravar no banco de dados estes novos padrões de entrada para treinamento da RNA.

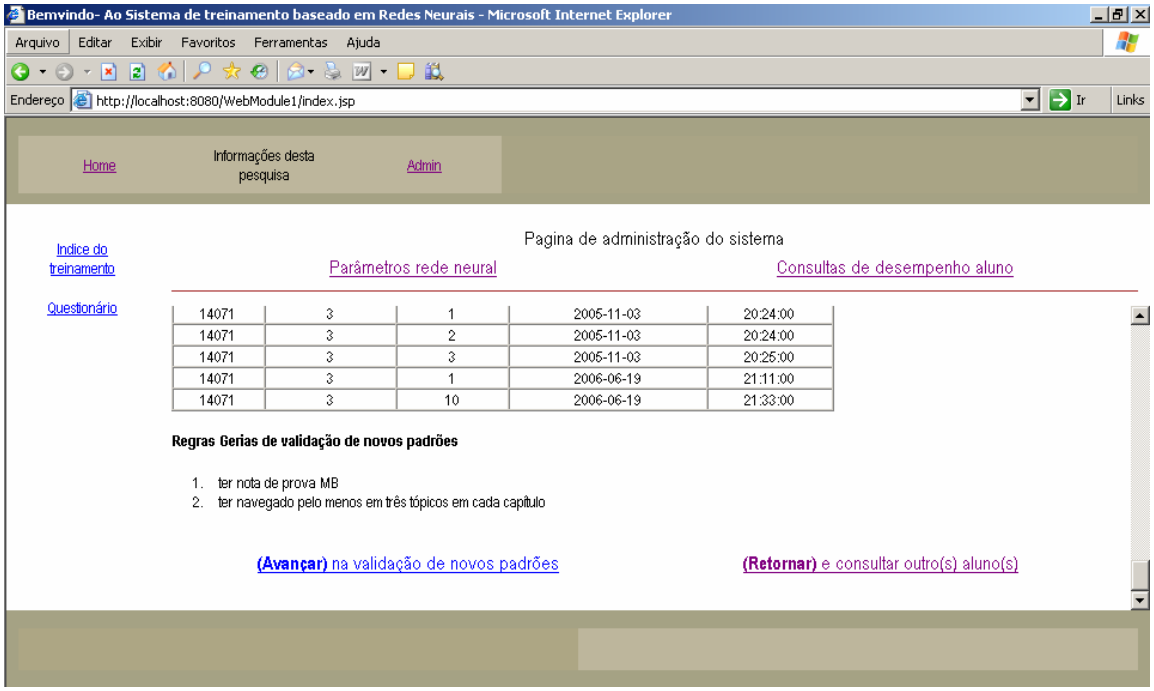

Figura 34 – Interface para validação de um novo padrão de treinamento.

O STI solicitará a confirmação do instrutor antes de efetivar a gravação dos novos valores. Uma vez confirmada a gravação, uma nova tela surgirá, permitindo ao instrutor que invoque o arquivo executável que inicia um novo treinamento da RNA.

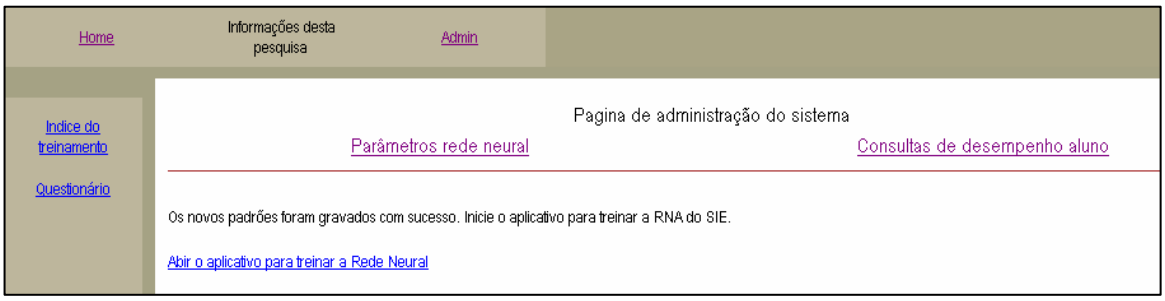

Figura 35 – Chamada do aplicativo de treinamento da RNA através do STI.

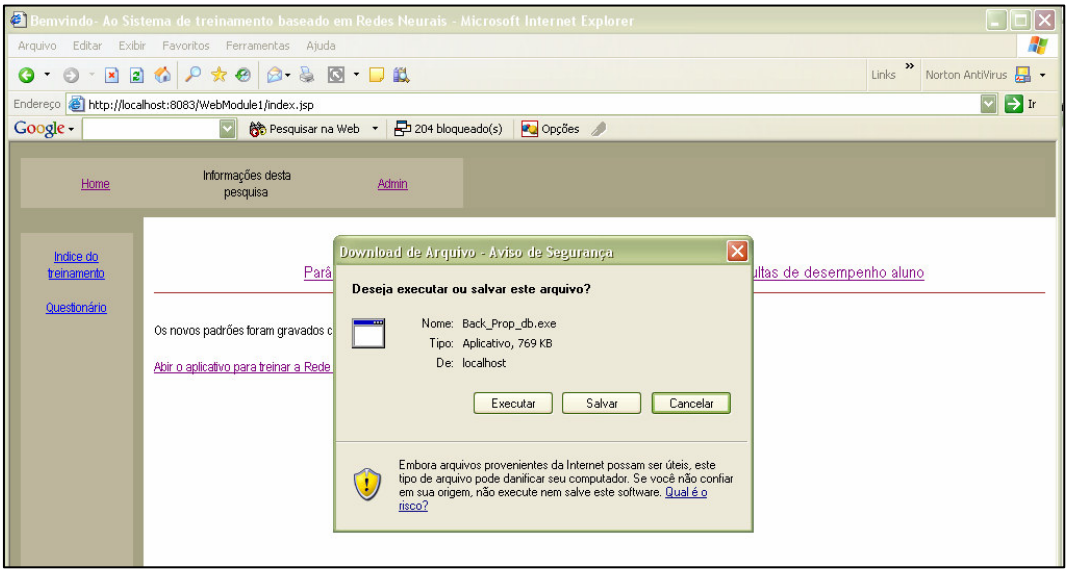

Figura 36 – Tela para autorização de execução.

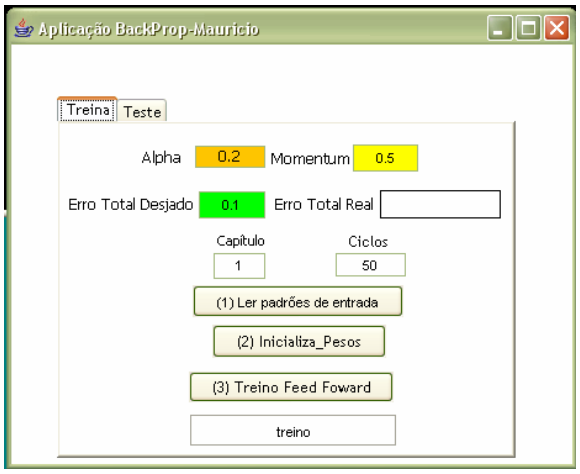

Figura 37 – Programa para execução de treinamento da RNA.

Outro item acessível através do recurso de administração é a consulta referente ao desempenho do aprendiz feito através do “hiperlink” “Consulta do desempenho do aluno”, executado pela página “adm_aluno.jsp”.

Esta opção permite ao instrutor consultar como foi o desempenho individual ou do grupo de aprendizes durante a etapa de estudo. Através do número de matrícula, digitado no campo do formulário, o instrutor pode consultar um único aprendiz, ou então ativar o “hiperlink” “Consultar todos os alunos” e realizar a consulta em um grupo. Este tipo de

consulta também permite efetuar comparações entre aprendizes e auxilia o instrutor na criação de novos caminhos de estudo.

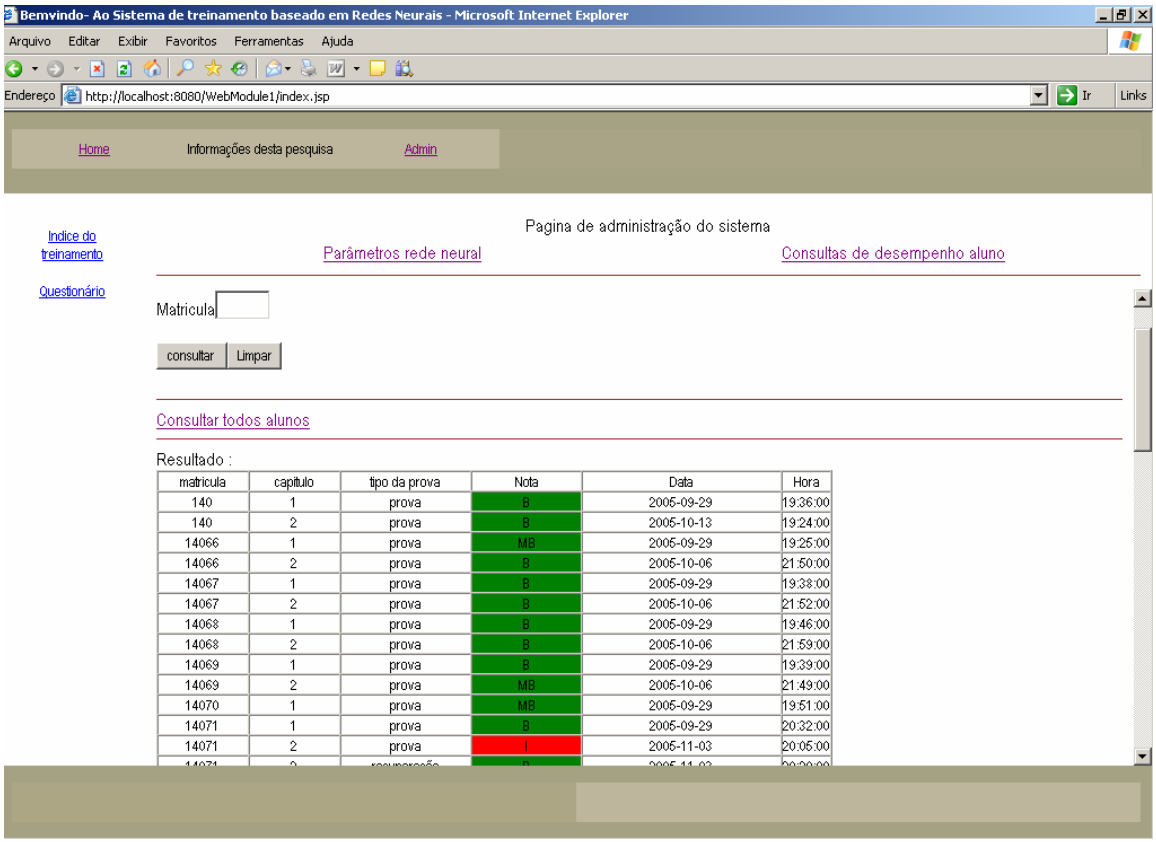

Figura 38 – Lista de desempenho por grupo de aprendizes.

Um diagrama completo de todos os estados percorridos durante a utilização do sistema de administração é mostrado na figura 39.

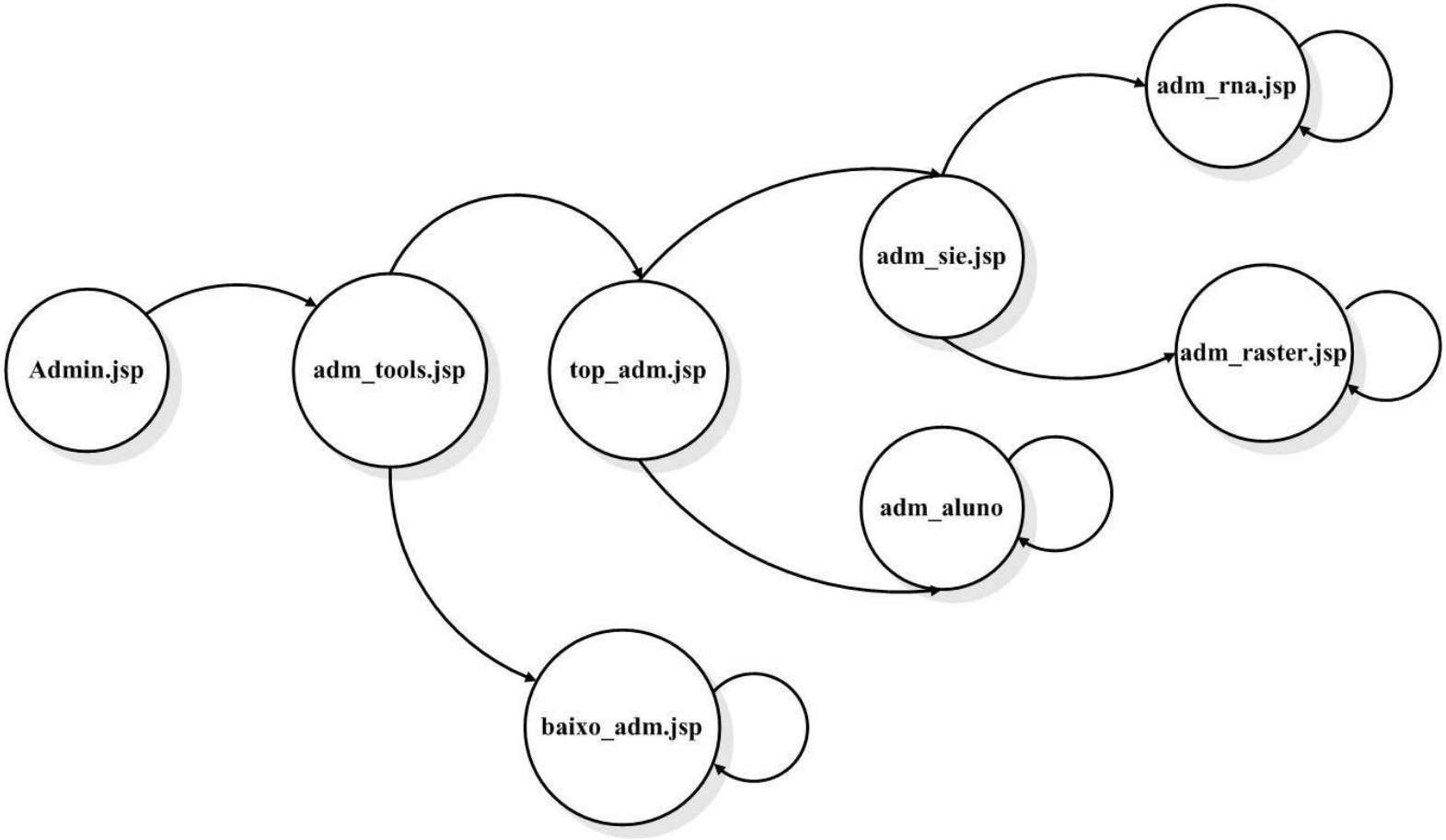

Figura 39 – Diagrama de estados do sistema de gerenciamento do STI.

6.6. Critérios para auto-aprendizado do STI

O critério de auto-aprendizado do STI foi avaliado sob dois pontos de vista antes de sua implementação nesta pesquisa. O primeiro ponto de vista seria implementar o auto-aprendizado da RNA de modo automático, ou seja, assim que um novo caminho de sucesso fosse realizado pelo aprendiz este novo padrão seria automaticamente considerado pelo STI sem nenhum tipo de avaliação pelo instrutor ou administrador do sistema. Para tal seria necessário criar um programa que executaria em modo multi-tarefa juntamente com o STI e que verificaria regularmente se algum padrão novo havia sido criado. Isto provocaria automaticamente um novo aprendizado da RNA. A vantagem deste sistema é que o mesmo ficaria totalmente autônomo para acrescentar novos padrões.

Algumas das desvantagens são: aumento no custo de computação e erro na execução da multi-tarefa.

O aumento do custo de computação reside no fato do servidor executar o programa do STI e ao mesmo tempo suportar a execução do programa para treinamento da RNA.

A outra desvantagem seria no caso de ocorrência de um problema na execução da multi-tarefa durante o treinamento da RNA. Este erro poderia acarretar uma parada do servidor e consequentemente parar a execução do STI. Uma parada inesperada de execução pode corromper ou comprometer o banco de dados onde estavam gravados os pesos para uma RNA já treinada.

Algumas soluções poderiam ser implementadas para contornar esta situação, uma delas seria a criação de uma réplica do banco de dados toda vez que fosse dado início há um novo treinamento e a sincronização desta réplica ao final de um novo treinamento.

Tudo isto envolveria um gasto de tempo no desenvolvimento de um programa a prova de falhas para algumas das situações expostas anteriormente e mesmo assim estaria sendo assumido um risco. Neste primeiro ponto de vista exposto haveria também o risco do sistema iniciar a instrução dos aprendizes com um novo padrão totalmente sem sentido pois o mesmo não foi julgado pelo instrutor. Devido a todos estes fatores relevantes para o funcionamento correto do STI foi tomada a decisão de que novos padrões de treinamento somente seriam submetidos ao treinamento da RNA após a validação do instrutor. Isto assegura e permite que um novo treinamento só ocorra após análise do instrutor e no momento em que o STI não esteja sendo utilizado pelos aprendizes, o que diminui os riscos de erro e os custos de computação para o servidor.

Capítulo 7. Conclusão

O sistema tutor inteligente mostrou-se eficiente quando utilizado de modo complementar aos estudos regulares, aumentando em 10% o índice de alunos aprovados.

Isto significa um ganho de recursos tais como: energia elétrica, horas aula do instrutor e alocação física de sala de aula, permitindo assim, que o instrutor possa dedicar mais atenção aos alunos com maior dificuldade de aprendizado.

A rede neural artificial, através do algoritmo de retropropagação, foi totalmente capaz de responder corretamente aos valores aplicados na entrada durante a execução do sistema tutor.

Uma vantagem na utilização de RNA foi o reduzido número de padrões utilizados para treinamento. A rede neural foi treinada para os capítulos 1,3,4 e 5 com 52 padrões de entrada e para o capítulo 2 com mais 46 padrões. Assim a rede neural foi capaz de generalizar todas as possibilidades de navegação dentro de cada capítulo.

Comparativamente se o mesmo sistema tutor tivesse sido projetado para utilizar lógica combinacional, teríamos que considerar 1024 (2^{10}) possibilidades, uma vez que cada capítulo possui 10 tópicos. A utilização da abordagem combinacional, conduz a recompilação do software toda vez que uma ou mais regras forem alteradas. Isto pode levar o programador a cometer erros durante a alteração do software, uma vez que o mesmo deve percorrer as 1024 possibilidades, toda vez que houver alteração de regras no caminho de estudo. Em contrapartida quando a rede neural artificial é utilizada, faz-se necessário, acrescentar somente as novas regras e executar um novo treinamento.

O sistema tutor mostrou-se eficiente no rastreamento de 100 % das ações de navegação do aprendiz, uma vez que, o mesmo opta-se por ser monitorado, não permitindo de maneira alguma que as regras impostas pela RNA fossem burladas.

Também houve eficiência do sistema tutor, nas ações dedicadas ao registro dos alunos, que optaram por não serem monitorados.

Este sistema foi capaz de armazenar no banco de dados o caminho percorrido durante o estudo, resgatar estes caminhos quando requisitados pelo instrutor para análise e possibilitar o retreinamento da RNA quando foi necessário.

Este foi um fator determinante para que o sistema pudesse ser chamado de inteligente, sendo capaz de aprender com novos padrões gerados pelo aprendiz e ratificados pelo instrutor.

Com referência a autoria de conteúdo para instrução, a estrutura deste sistema tutor, foi criada de modo a facilitar o reaproveitamento do material didático previamente produzido pelo instrutor. Para inclusão de conteúdo produzido originalmente em modo texto ou formato eslaide, no diretório do sistema tutor, somente é necessário a conversão dos arquivos para formato HTML. Existem vários programas editores de texto ou dedicados à criação de apresentações, que exportam seu conteúdo para formato HTML, o que se enquadra no conjunto de ferramentas das quais os instrutores tem domínio, tornando-se assim uma ferramenta de uso fácil e barato.

O modo como o sistema foi projetado mostrou-se eficiente e estável, possibilitando sua execução em qualquer um dos cinco laboratórios instalados na escola.

É importante evidenciar, que alguns dos computadores utilizados para realização deste estudo de caso, eram Pentium 100 MHz com 64MB de memória RAM, demonstrando assim, que este sistema pode ser utilizado em ambientes onde não estejam disponíveis os últimos recursos em termos de “hardware” do computador.

Tratando dos resultados em relação aos aprendizes, durante o período de estudo com o STI, cabe ressaltar a preferência deles por seguir o treinamento de acordo com as regras estabelecidas pelo sistema tutor, sem mostrar o desejo de buscar caminhos alternativos. Do total de aprendizes que concluíram o semestre, menos de 5%, buscaram caminhos próprios de estudos.

Aplicando o sistema tutor, durante um semestre letivo com 31 aprendizes, foram estudados três capítulos de um total de cinco e realizadas 88 avaliações.

O resultado foi uma queda de 14% no índice de alunos em processo de recuperação e um aumento de 10% no índice de aprovação.

Tabela 4 – Tabela de resultados obtidos na pesquisa

Critério Avaliado	Método Regular	Método utilizando o STI
Média de aprendizes em recuperação	27 %	**13 %**
Porcentagem de aprendizes aprovados	67 %	**77 %**

Para calcular o índice de aprendizes em recuperação e aprovados nas turmas anteriores, foi utilizado a média dos respectivos índices, nos últimos cinco anos para esta mesma disciplina.

Os aprendizes das turmas anteriores, que não usaram o sistema tutor, possuíam um perfil similar aos aprendizes que se submeteram aos estudos utilizando o STI. Com relação aos aspectos de instrução, cabe ressaltar, as respostas dadas pelos aprendizes, através de questionário aplicado ao final do semestre, após a utilização do STI.

Elementos importantes foram levantados tais como:

- 50 % já fizeram algum tipo de treinamento virtual utilizando um CD de curso, Internet ou outro meio;
- 56 % acreditam que o sistema ajudou muito nas aulas de C/C++;
- 48 % dos alunos se tivessem tempo estudariam somente pelo tutor;
- 50 % deram nota 8-10 para a disciplina que usou sistema tutor, comparada com as outras que não utilizaram; 36% deram nota de 5-7; 16% deram nota de 2-4;
- 92% recomendariam o sistema tutor para outras disciplinas;
- 72% avaliaram como excelente poder realizar a prova no STI e receber seu resultado logo após o final da avaliação.

Sendo assim é possível considerar o sistema tutor como uma importante ferramenta no auxílio à instrução.

Para considerações relativas ao desenvolvimento deste sistema tutor, podemos destacar a maneira como foi implementada a solução de rastreamento e armazenamento, através dos links de hipertexto, o que permitiu criar um núcleo, para ser aprimorado em outros trabalhos de pesquisa. Sendo assim recomendo alguns trabalhos para os colegas pesquisadores, que tenham interesse em aperfeiçoar os sistemas tutores, tomando como ponto de partida o trabalho desenvolvido para esta pesquisa.

Para aprofundar a pesquisa na área de educação sugiro o desenvolvimento de ambientes de aprendizagem em sistemas distribuídos que estimulem os processos metacognitivos por meio de agentes inteligentes.

Para o aprimoramento dos sistemas tutores, sugiro aplicação de técnicas de inteligência artificial, para prover ao aprendiz uma navegação não linear e permitir a implementação de outros modos de ensino. Os dados para busca destes novos modos, podem ser extraídos dos links de hipertexto navegados e armazenados no banco de dados por este sistema, quando o aprendiz não está sendo monitorado pela RNA.

Uma possível solução seria a adição de outras redes neurais, para monitoração de novos caminhos de navegação orientados ao aprendizado.

Dando continuidade a este projeto, voltado à idéia de navegação linear, surge uma alternativa para desenvolvimento de pesquisa, focada na utilização de algoritmos genéticos para seleção do melhor caminho alternativo de estudo. O novo algoritmo trabalharia em função da navegação não monitorada pela RNA para aqueles aprendizes que mesmo tendo optado por um caminho não previsto inicialmente, obtiveram o melhor rendimento na avaliação. O resultado da seleção do melhor caminho, feito pelo algoritmo genético, seria utilizado para alimentar a rede neural com um novo conjunto de dados para retreinamento.

Para melhoria do método de avaliação, deste sistema, sugiro que a apresentação dos resultados da avaliação seja seguida de conteúdo e tópico a serem estudados, tomando como base, os links que deixaram de ser navegados pelo aprendiz durante o estudo do capítulo.

Neste método de avaliação o sistema tutor pode estipular um mínimo de navegação através do capítulo, antes da realização da avaliação.

Por fim, recomendo mais uma modificação no modo de navegação linear, onde o sistema tutor apresenta primeiramente uma pré-avaliação pertinente ao capítulo e a partir das respostas dadas pelo aprendiz cria um caminho de estudos personalizado.

ANEXO A1 – Padrões de treinamento.

A.1 - Padrões de treinamento capítulo 1, 3, 4, 5.

padrão	Entradas										saída
	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	y
1	1	1	-1	-1	-1	-1	-1	-1	-1	-1	1
2	1	1	1	-1	-1	-1	-1	-1	-1	-1	1
3	1	1	1	1	-1	-1	-1	-1	-1	-1	1
4	1	1	1	1	1	-1	-1	-1	-1	-1	1
5	1	1	1	1	1	1	-1	-1	-1	-1	1
6	1	1	1	1	1	1	1	-1	-1	-1	1
7	1	1	1	1	1	1	1	1	-1	-1	1
8	1	1	1	1	1	1	1	1	1	-1	1
9	1	-1	1	-1	-1	-1	-1	-1	-1	-1	-1
10	-1	1	1	1	1	1	1	1	1	1	1
11	1	1	-1	1	-1	-1	-1	-1	-1	-1	-1
12	1	1	1	-1	1	-1	-1	-1	-1	-1	-1
13	1	1	1	1	-1	1	-1	-1	-1	-1	-1
14	1	1	1	1	1	-1	1	-1	-1	-1	-1
15	1	1	1	1	1	1	-1	1	-1	-1	-1
16	1	1	1	1	1	1	1	-1	1	-1	-1
17	1	1	1	1	1	1	1	1	-1	1	-1
18	1	1	1	1	1	1	1	-1	-1	1	-1
19	1	1	1	1	1	1	-1	-1	-1	1	-1
20	1	1	1	1	1	-1	-1	-1	-1	1	-1
21	1	1	1	1	-1	-1	-1	-1	-1	1	-1
22	1	1	1	-1	-1	-1	-1	-1	-1	1	-1
23	1	1	-1	-1	-1	-1	-1	-1	-1	1	-1
24	1	-1	-1	-1	-1	-1	-1	-1	-1	1	-1
25	1	-1	-1	-1	-1	-1	-1	-1	1	-1	-1
26	1	1	-1	-1	-1	-1	-1	-1	1	-1	-1
27	1	1	1	-1	-1	-1	-1	-1	1	-1	-1
28	1	1	1	1	-1	-1	-1	-1	1	-1	-1
29	1	1	1	1	1	-1	-1	-1	1	-1	-1
30	1	1	1	1	1	1	-1	-1	1	-1	-1

A.1 - Padrões de treinamento capítulo 1, 3, 4, 5(cont.).

padrão	Entradas										saída
	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	y
31	1	1	1	1	1	1	1	-1	1	-1	-1
32	1	-1	-1	-1	-1	-1	-1	1	-1	-1	-1
33	1	1	-1	-1	-1	-1	-1	1	-1	-1	-1
34	1	1	1	-1	-1	-1	-1	1	-1	-1	-1
35	1	1	1	1	-1	-1	-1	1	-1	-1	-1
36	1	1	1	1	1	-1	-1	1	-1	-1	-1
37	1	1	1	1	1	1	-1	1	-1	-1	-1
38	1	-1	-1	-1	-1	-1	1	-1	-1	-1	-1
39	1	1	-1	-1	-1	-1	1	-1	-1	-1	-1
40	1	1	1	-1	-1	-1	1	-1	-1	-1	-1
41	1	1	1	1	-1	-1	1	-1	-1	-1	-1
42	1	1	1	1	1	-1	1	-1	-1	-1	-1
43	1	-1	-1	-1	-1	1	-1	-1	-1	-1	-1
44	1	1	-1	-1	-1	1	-1	-1	-1	-1	-1
45	1	1	1	-1	-1	1	-1	-1	-1	-1	-1
46	1	1	1	1	-1	1	-1	-1	-1	-1	-1
47	1	-1	-1	-1	1	-1	-1	-1	-1	-1	-1
48	1	1	-1	-1	1	-1	-1	-1	-1	-1	-1
49	1	1	1	-1	1	-1	-1	-1	-1	-1	-1
50	1	1	-1	1	-1	-1	-1	-1	-1	-1	-1
51	1	-1	1	-1	-1	-1	-1	-1	-1	-1	-1
52	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1

A.2 - Padrões de treinamento capítulo 2.

padrão	Entradas										saída
	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	y
1	1	-1	1	-1	-1	-1	-1	-1	-1	-1	1
2	1	-1	-1	1	-1	-1	-1	-1	-1	-1	-1
3	1	-1	-1	-1	1	-1	-1	-1	-1	-1	-1
4	1	-1	-1	-1	-1	1	-1	-1	-1	-1	-1
5	1	-1	-1	-1	-1	-1	1	-1	-1	-1	-1
6	1	-1	-1	-1	-1	-1	-1	1	-1	-1	-1
7	1	-1	-1	-1	-1	-1	-1	-1	1	-1	-1
8	1	-1	-1	-1	-1	-1	-1	-1	-1	1	-1
9	1	1	-1	1	-1	-1	-1	-1	-1	-1	1
10	1	1	-1	-1	1	-1	-1	-1	-1	-1	-1
11	1	1	-1	-1	-1	1	-1	-1	-1	-1	-1
12	1	1	-1	-1	-1	-1	1	-1	-1	-1	-1
13	1	1	-1	-1	-1	-1	-1	1	-1	-1	-1
14	1	1	-1	-1	-1	-1	-1	-1	1	-1	-1
15	1	1	-1	-1	-1	-1	-1	-1	-1	1	-1
16	1	1	1	-1	1	-1	-1	-1	-1	-1	1
17	1	1	1	-1	-1	1	-1	-1	-1	-1	-1
18	1	1	1	-1	-1	-1	1	-1	-1	-1	-1
19	1	1	1	-1	-1	-1	-1	1	-1	-1	-1
20	1	1	1	-1	-1	-1	-1	-1	1	-1	-1
21	1	1	1	-1	-1	-1	-1	-1	-1	1	-1
22	1	1	1	1	-1	1	-1	-1	-1	-1	1
23	1	1	1	1	-1	-1	1	-1	-1	-1	-1
24	1	1	1	1	-1	-1	-1	1	-1	-1	-1
25	1	1	1	1	-1	-1	-1	-1	1	-1	-1
26	1	1	1	1	-1	-1	-1	-1	-1	1	-1
27	1	1	1	1	1	-1	1	-1	-1	-1	1
28	1	1	1	1	1	-1	-1	1	-1	-1	-1
29	1	1	1	1	1	-1	-1	-1	1	-1	-1
30	1	1	1	1	1	-1	-1	-1	-1	1	-1

A.2 - Padrões de treinamento capítulo 2(cont.).

padrão	Entradas										saída
	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	y
31	1	1	1	1	1	1	-1	1	-1	-1	1
32	1	1	1	1	1	1	-1	-1	1	-1	-1
33	1	1	1	1	1	1	-1	-1	-1	1	-1
34	1	1	1	1	1	1	1	-1	1	-1	1
35	1	1	1	1	1	1	1	-1	-1	1	-1
36	1	1	1	1	1	1	1	1	-1	1	1
37	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1
38	1	1	-1	-1	-1	-1	-1	-1	-1	-1	1
39	1	1	1	-1	-1	-1	-1	-1	-1	-1	1
40	1	1	1	1	-1	-1	-1	-1	-1	-1	1
41	1	1	1	1	1	-1	-1	-1	-1	-1	1
42	1	1	1	1	1	1	-1	-1	-1	-1	1
43	1	1	1	1	1	1	1	-1	-1	-1	1
44	1	1	1	1	1	1	1	1	-1	-1	1
45	1	1	1	1	1	1	1	1	1	-1	1
46	1	1	1	1	1	1	1	1	1	1	1

Referências Bibliográficas

[1] Beting, G. Sem passar pelo controle de passaporte. **Guia de Educação a distancia 2006, São Paulo,** nº3**,** p.8-14, 2005.

[2] URBAN, M. **Intelligent Tutoring Systems: An Historical Review in the Context of the Development of Artificial Intelligence and Educational Psychology**, Michigan, USA.1996.

Disponível em: <http://www.cse.msu.edu/rgroups/cse101/ITS/its.htm>, Acesso em: 20 de Abril de 2004.

[3] MCARTHUR, D.; LEWIS, M.; BISHAY, M. **The Roles of Artificial Intelligence in Education: Current Progress and Future Prospects**. Santa Monica, USA. 1993. Disponível em: < http://www.rand.org/education/mcarthur/Papers/role.html>, Acesso em: 10 de Março de 2004.

[4] Silva, D. R.; Vieira, M. T. P. **Modelo para acompanhamento do aprendizado em educação à distância.** In: Anais do VII Workshop de Informática na Escola, Fortaleza, Brasil, julho de 2001.

Disponível em: <http://www.dc.ufscar.br/~marina/ >. Acesso em: mar. 2004.

[5] Fonseca, J.J. S da. **Metodologia da Pesquisa Científica**, Ceára: Curso de especialização em comunidades virtuais de aprendizagem – Informática Educativa, [2002]. Apostila.

[6] RICH, E.; KNIGHT, K., **Inteligência Artificial** ,1993.

[7] FAUSETT, L.V., **Fundamentals of neural networks: architectures, algorithms and applications**. EUA: Prentice Hall, 1994, 461 p.

[8] TODD, N;SZOLKOWSKI., **Java Server Pages : O guia do Desenvolvedor.** Tradução de Edson Furmanckiewicz. Rio de Janeiro: Elsevier, 2003, 621 p.

[9] **Definition of ODBC**. Disponível em: <http://www.webopedia.com/term/o/odbc.html>. Acesso em: 26 nov. 2005.

Revisão Bibliográfica

SILVA, A.M.; PINHEIRO, M.S.F.; FREITAS, N.E. **Guia para normalização de trabalhos técnico – científicos : Projetos de pesquisa, monografias, dissertações, teses**. 2. ed. Uberlândia: Edufu, 2002, 159 p.

MASTERS, T. **Practical Neural Network Recipes in C++**.EUA : [s.n.], 1993, 493 p.

ROGERS, J. **Object-Oriented Neural Networks in C++**. EUA : Academic Press, 1997, 310 p.

LOZANO, F. **Segurança no J2EE**, Java Magazine, Rio de Janeiro, n.22, p. 24-30, março, 2005.

MARTINS, R.N. **Aprendizagem cooperativa via internet - a implantação de dispositivos computacionais para a viabilidade técnica de cursos on-line.** Florianópolis, 2000. 146 p. Dissertação. Disponível em: <http://teses.eps.ufsc.br/Resumo.asp?1287>.

VARNA, A. **ICARUS: Design and Deployment of a Case-Based Reasoning System for Locomotive Diagnostics. 1999.** Disponível em: <www.ai-cbr.org/ iccbr99/ presents/varma_presentation.pdf>. Acesso em: 25 mar. 2004.

AAMODT, A; PLAZA, E. **Case – Based Reasoning: Foundation Issues, Methodological Variations, and System Approaches**. IOS Press, V.7, n.1, p. 39-59.1994.

STANKOV, S.; GLAVINIĆ, V.; ROSIĆ, M. **On Knowledge Representation in an Intelligent Tutoring System.** Disponível em: <http://scholar.google.com/url?sa=U&q=

http://www.pmfst.hr/~stankov/zn_radovi/2000/INES2000.pdf>. Acesso em: 10 mar. 2004.

KOLODNER, J. L., GUZDIAL, **M. Theory and Practice of Case-Based Learning Aids**, 2000. 27 p. Georgia Institute of Technology, EUA. Disponível em: < http://coweb.cc.gatech.edu/guzdial/uploads/18/>. Acesso em: 25 mar. 2004.

ARAUJO Jr., C.F. A. et al. **Desenvolvimento de Material Didático Digital baseado na Web para Área de Computação e Informática**. 12 p. UNICSUL, São Paulo, SP, 2002.

THRUN, S et al. **Automated Learning and Discovery: State Of The Art And Research Topics in a Rapidly Growing Field**. 1998. 30 p. Carnegie Mellon University, Pittsburgh.

MACKAY, D.J.C. **A Practical Bayesian Framework for Backprop Networks**. 1991. 11 p. California Institute of Technology. Pasadena, CA.

Publicação

ALMEIDA, C. M.; YAMANAKA, K. **Sistema tutor monitorado por rede neural artificial.** In: WCCSETE 2006 – World Congress on Computer Science, Engineering and Technology Education, 2006, Santos, São Paulo. p. 1246-1250.