OBJETOS DE APRENDIZAGEM PARA O ENSINO DE PROGRAMAÇÃO EM ENGENHARIA E COMPUTAÇÃO

Ivan Carlos Alcântara de Oliveira[1], Carlos Fernando de Araújo Jr.[2] e Manuel Fernández Paradela Ledón[3]

Resumo *— Este trabalho descreve um framework para ambientes de aprendizagem baseado na Web que integra um objeto de interface para inserção de conteúdos digitais, a modelagem deste conteúdo e um interpretador de algoritmos com recursos para o aluno exercitar a solução de problemas representados em pseudocódigo. O objeto de interface foi desenvolvido com HTML (Hypertext Markup Language), JavaScript e CSS (Cascading Styles Sheets) permitindo a criação de arquivos pequenos e interfaces interativas seguindo critérios de design com o objetivo de facilitar a procura de partes do conteúdo modelado. O interpretador de algoritmos foi desenvolvido com a tecnologia Java, possibilitando a análise léxica e sintática de um programa escrito em uma linguagem algorítmica estruturada denominada PoStWEL (Português Estruturado do Projeto WEL). Sua interface simples e robusta possibilita ao aluno escrever e compilar facilmente seus exercícios.*

Palavras Chaves — conteúdo digital, ensino de programação, interpretador, objetos de aprendizagem.

1. INTRODUÇÃO

Nos últimos anos, o constante crescimento do número de cursos que se utilizam das tecnologias baseadas na *web*, vêm provocando uma verdadeira revolução nas mais diversas áreas da educação [1]. A crescente popularização de tecnologias hipermídia vem provocando uma demanda cada vez maior por cursos a distância, cuja qualidade é medida não somente pelo conteúdo, mas também por seu *design*, navegabilidade, organização e clareza das informações [2].

Um fator relevante é que uma vasta gama de recursos computacionais podem ser utilizados na preparação de materiais e em geral não o são. Exemplos disso são: simulações em tempo real, modelos tridimensionais, avaliações eletrônicas, animações e gráficos, recursos de grande importância no processo de aprendizagem e que dificilmente são reproduzidos em sala de aula no ensino tradicional [3]. Isto Posto, o Núcleo de Pesquisa em Computação e Tecnologia da Informação (NCTI) da UNICSUL (Universidade Cruzeiro do Sul) propôs um projeto temático, chamado Projeto WEL (*Web Engineering for Learning*) que engloba basicamente a produção de conteúdo digital para algumas disciplinas dos cursos de graduação em Engenharia e Computação e a construção de alguns componentes de software, denominados objetos de aprendizagem, com o intuito de facilitar o entendimento dos alunos em algumas disciplinas científicas e técnicas [4][5][6]. Os objetos de aprendizagem desenvolvidos neste trabalho são:

- Um objeto de interface para disponibilizar conteúdo digital. Para este objeto também foi modelado o conteúdo de uma disciplina básica relacionada a programação dos cursos de Computação e Informática chamada Estruturas de Dados [7].
- Um objeto de simulação que envolve um interpretador de algoritmos baseado na *Web* onde o aluno pode testar os algoritmos descritos no material das disciplinas e/ou desenvolvê-los na linguagem algorítmica PoStWEL (*Português Estruturado do Projeto Wel*) [8].

Este artigo está organizado em seções conforme segue:

- Na seção 2 são fornecidos detalhes sobre o projeto temático WEL e onde os objetos de aprendizagem descritos estão inseridos.
- Os conceitos relacionados a objetos de aprendizagem e as tecnologias que foram utilizadas na criação destes objetos são introduzidos na seção 3.
- A proposta e os resultados obtidos com a implementação são descritos na seção 4.
- A conclusão obtida com o desenvolvimento e a respectiva continuidade em trabalhos futuros são especificados na seção 5.

2. PROJETO WEL (*WEB ENGINNERING FOR LEARNING*)

O projeto WEL (*Web Engineering for Learning*) é um projeto temático financiado pela UNICSUL (Universidade Cruzeiro do Sul) que propõe o desenvolvimento de software baseado na Web, com uso das metodologias, técnicas e métricas de qualidade da área de *Web Engineering*, para disponibilizar material "instrucional" direcionado ao aprendizado e apoio às disciplinas técnicas e científicas nas áreas de Computação e Engenharia [2]. Este projeto foi

[1] Ivan Carlos Alcântara de Oliveira, Universidade Cruzeiro do Sul - UNICSUL, Av. Dr. Ussiel Cirilo, 225 – 08060-070, São Paulo, SP, Brazil, ivan.oliveira@unicsul.br, icalcan@dglnet.com.br
[2] Carlos Fernando de Araújo Jr., Universidade Cruzeiro do Sul - UNICSUL, Av. Dr. Ussiel Cirilo, 225 – 08060-070, São Paulo, SP, Brazil, carlos.araujo@unicsul.br
[3] Manuel Fernandéz Paradela Lédon, Universidade Cruzeiro do Sul - UNICSUL, Av. Dr. Ussiel Cirilo, 225 – 08060-070, São Paulo, SP, Brazil, manuel.ledon@unicsul.br

subdividido em quatro projetos de pesquisa conforme Fig. 1. Da figura observa-se o item "Engenharia da Informação e Modelagem de conteúdos" onde se encaixa o objeto de interface e a modelagem do conteúdo e o item "Construção de Programa de Simulação para Apoio ao Aprendizado" onde o interpretador está inserido. Neste projeto foram selecionadas disciplinas de graduação relacionadas ao desenvolvimento de algoritmos, programação, física e cálculo numérico. A disciplina, associada a programação, Estruturas de Dados foi a escolhida como objeto inicial para a modelagem de conteúdo.

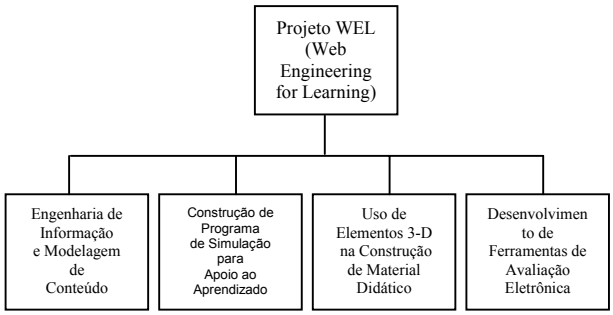

FIGURA. 1
SUBDIVISÃO DOS PROJETOS DE PESQUISA DENTRO DO PROJETO TEMÁTICO WEL

3. OBJETOS DE APRENDIZAGEM E TECNOLOGIAS PARA A WEB: ESTADO ATUAL

3.1 Objetos de Aprendizagem

Objetos de aprendizagem são elementos de um novo tipo de aprendizagem baseado em computador utilizando conceito de orientação à objetos como, por exemplo, a criação de componentes. Estes objetos são, normalmente, pequenos, específicos e podem ser reutilizados em contextos diferentes. [6]

Objetos de aprendizagem podem ser definidos como entidades digitais com acesso via Web, de forma que inúmeras pessoas possam usufruir dos seus benefícios simultaneamente [6]. Para que um objeto possa ser reusável deve ser flexível, interoperável e possuir facilidade de atualização e busca.

Alguns exemplos de objetos de aprendizagem são:

- Fotos ou imagens digitais, animações e pequenas aplicações.
- Páginas *web* que combinam texto, imagens e outros meios ou aplicações.
- Sistemas de treinamento por computador, ambientes de aprendizagem interativos, sistemas inteligentes de instrução com o auxílio do computador.

Algumas organizações internacionais voltadas a padronização, como o LTSC (*Learning Tecnology Standard Commitee*), estão desenvolvendo padrões abertos para os objetos de aprendizagem, focando a sua reusabilidade, permitindo que ocorra a redução de tempo no desenvolvimento e facilidade na sua distribuição e adaptação [9].

3.2 Tecnologias para Desenvolvimento Web

O desenvolvimento de objetos de aprendizagem para a Internet pode ser realizado utilizando vários tipos de tecnologias. Pode-se gerar poderosas aplicações com CGI (*Common Gateway Interface*) e Servlets (*Dynamic WebPages in Java*) do lado servidor com a inclusão de Applets Java e JavaScript/CSS (*Cascading Styles Sheets*) do lado cliente permitindo a modificação ou adaptação do conteúdo a ser apresentado ao aluno dependendo de suas ações. O foco deste trabalho está no desenvolvimento de dois objetos de aprendizagem utilizando principalmente as tecnologias JavaScript e Java.

JavaScript é uma linguagem interpretada que associada ao CSS possibilita desenvolver páginas dinâmicas com maior interação permitindo a criação de interfaces mais atraentes com recursos de navegação mais simples e interativos que podem motivar o aprendizado e prender a atenção do aluno[10][11].

Java pode ser executado do lado cliente na forma de um *applet,* inserido em páginas *web*, ser carregado e processado em browsers a partir da solicitação do usuário[12]. O uso de *applets* tem como vantagens a criação de interfaces mais complexas, a independência de plataforma, não necessita instalação e atualização pelo aluno além de ser removido depois do término do seu processamento. Há algumas restrições de segurança que impedem ler ou escrever arquivos no disco local e a comunicação fica restrita somente ao servidor de origem do *applet*. Para contornar esta restrição esquemas de autenticação e modificação da política de segurança da JVM (*Java Virtual Machine*) podem ser implementados. Além das classes Java comuns para geração do *applet* foram utilizados duas ferramentas JFlex e Cup que são distribuídos gratuitamente e são escritas em Java [13][14]. A importância destas duas ferramentas auxiliares está ligada ao código em Java que elas geram para os analisadores léxico e sintático. Atualmente estas ferramentas se encontram nas versões CUP 0.10, para gerar o analisador sintático, e JFlex 1.3.5, para gerar o analisador léxico. Na geração dos analisadores é necessária a definição de dois arquivos de configuração, um com extensão ".flex" e outro com a extensão ".cup" onde são especificadas as informações necessárias à criação destes analisadores.

O desenvolvimento do objeto de interface foi baseado em HTML, JavaScript e CSS e o interpretador em *applet* Java com o Jflex e o Cup.

4. PROPOSTAS E RESULTADOS

4.1 Objeto de Interface e o Conteúdo Digital Modelado

O objeto de aprendizagem para inserção de conteúdo digital foi construído para possibilitar a modelagem de conteúdos de cursos e disciplinas, subdivididos em módulos e

submódulos pequenos utilizando uma interface padronizada [2]. Esta padronização é uma tentativa de facilitar a navegação e a procura de informação pelo aluno dentro do conteúdo.

A Fig. 2 ilustra um modelo do arquivo CSS utilizado para padronizar determinados itens de interface como menus, títulos, tipos e cores das fontes do conteúdo digital [10][11].

```
<style type="text/css">
<!--
.curso {position:absolute; top:0px; left:334px;
visibility:visible; width:58%;}
.nav1 {position:absolute; top:25px; left:334px;
...
.variavel{position:absolute; top:0px; left:0px;
visibility:hidden;}
li.inside{list-style-position:inside;}
body {background-color:lightblue;}
a,font {color:black; font-family:verdana; font-
size:12pt; text-decoration:none;}
a:hover {color:blue;}
-->
</style>
```

FIGURA. 2
ARQUIVO CSS ANEXADO A UMA PÁGINA WEB PARA PADRONIZAR COMPONENTES DO OBJETO DE INTERFACE.

Como forma de testar este objeto foi modelado o assunto "árvore" da disciplina Estruturas de Dados. Este assunto possui um conteúdo relativamente complexo e um caráter de abstração considerável onde nem sempre o professor consegue que os alunos compreendam de forma clara as operações ocorridas em cada estrutura.

A Fig. 3 ilustra três modelos de interface obtidos com o desenvolvimento deste objeto de aprendizagem. No modelo de interface (1) observa-se: o logo da instituição, título da disciplina, navegação entre módulos (anterior, próximo), título do módulo, quantidade de submódulos com possibilidade de navegação seqüencial ou aleatória pelo aluno, ligações para exemplos e exercícios relacionados a disciplina, possibilidade de retornar ao índice. O modelo de interface (2) demonstra a organização do conteúdo na forma de um índice. No modelo (3) observa-se um exemplo que permite ao aluno interagir com o conteúdo obtendo informações sobre o desenho da árvore como por exemplo, grau de parentesco, grau de um nó, altura da árvore e outros.

A modelagem da interface e do conteúdo seguiu critérios de *design* envolvendo, principalmente, usabilidade, comunicabilidade e navegação, como tentativa de gerar um ambiente com conteúdo capaz de prender a atenção do aluno, possibilitar uma maior interação com o material e deixar a navegação mais intuitiva e simples [2].

4.2 Interpretador de Algoritmos: Um objeto de Simulação

O interpretador é um objeto de aprendizagem que lê um conjunto de símbolos descrito em uma determinada linguagem, analisa e informa se a junção destes símbolos está léxico, sintático e semanticamente corretos. Não gera um programa executável mas deseja-se que ele seja capaz de executar o programa gerando algum tipo de resultado. No estágio atual de desenvolvimento foi implementado somente os analisadores léxico e sintático [15].

Inicialmente foi criada uma gramática para uma linguagem algorítmica, denominada PoStWEL, desenvolvido um *applet* Java constando de um editor de texto e dos analisadores léxico e sintático [12].

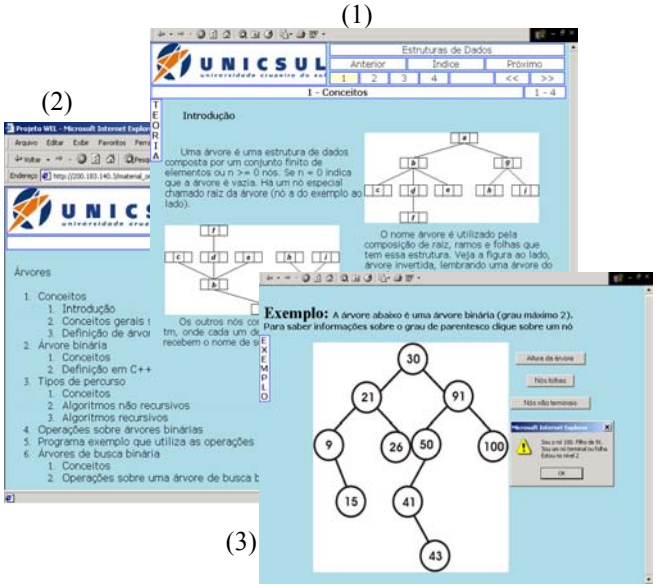

FIGURA. 3
OBJETO DE INTERFACE E A MODELAGEM COM CONTEÚDO E EXEMPLO INTERATIVO.

4.2.1 Gramática da linguagem PoStWEL (Português Estruturado do Projeto WEL)

A gramática da PoStWEL segue a estrutura de uma gramática livre de contexto, como definida pelo lingüista Noam Chomsky [8], onde uma determinada cadeia segue regras para sua formação, como descrito abaixo:

$$A \rightarrow \beta \qquad (1)$$

Em (1) "A" representa um conjunto de símbolos denominados não terminais e "β" uma combinação de símbolos terminais e não terminais. Podemos dizer que um símbolo terminal é qualquer palavra reservada, símbolo especial, operador que pertença a linguagem descrita. E os não terminais são os símbolos utilizados durante o processo de derivação das produções no reconhecimento de uma dada cadeia.

A tabela I ilustra algumas produções desta gramática utilizando a notação BNF (*Backus Normal Form*) onde o símbolo inicial é <postwel>.

TABELA I
ALGUMAS PRODUÇÕES DA POSTWEL

Simbolos não-terminais (A)	Regras de produção (β)
<postwel> ::=	ALGORITMO <nome> ; <bloco>
<bloco> ::=	INICIO <corpo> FIM
<corpo> ::=	<declaraçãoCorpo> | <comds>
<comds> ::=	<atribuição> ; <comds> |
	<seCondicional> <comds> |
	...
<seCondicional> ::=	SE (<expressaoLogica>) ENTAO <acao> |
	SE (<expressaoLogica>) ENTAO <acao>
	SENAO <acao>
<acao>::=	<comandoSimples> | <blocoComandos>
<blocoComandos> ::=	INICIO <comds> FIM

Um exemplo de programa escrito em PoStWEL que calcula o fatorial de um número é ilustrado na Fig. 4.

```
algoritmo "Calcula fatorial de um Número" ;
inicio
     inteiro num, fat, i;
     escreva ( "Digite um Numero : " );
     leia ( num );
     fat = 1;
     i = 1;
     enquanto (i <= num) faca
     inicio
         fat = fat * i;
         i = i + 1;
     fim
     escreva ("O Fatorial deste numero é: ", fat);
fim
```

FIGURA. 4
ALGORITMO EXEMPLO NA LINGUAGEM POSTWEL QUE CALCULA O FATORIAL DE UM NÚMERO.

4.2.2 Processo de Interpretação: Análise Léxica e Sintática

O mecanismo de interpretação representado na Fig. 5 segue com o pedido de verificação pela *applet* que envia o arquivo (*reader*) a ser avaliado ao analisador sintático, este segue com a captação dos símbolos ou *tokens* do arquivo de entrada pelo analisador léxico (*scanner*) que envia token a *token* ao analisador sintático (*parser*) que reconhece uma sentença através do conjunto de símbolos definidos pela linguagem algorítmica ao passo que a cada verificação de um conjunto de símbolos (*yybuffer*) interpretados poderá surgir inconsistências com a gramática fazendo com que o analisador sintático gere erros sintáticos mostrando o local através da linha (*yyline*) onde existe a inconsistência [15].

Um manipulador de erros é ativado sempre que for detectado um erro no programa fonte, avisando ao programador da ocorrência do erro e emitindo uma mensagem. O procedimento construído para o tratamento de erros foi interromper a análise léxica e sintática quando um erro é encontrado.

Um trecho do arquivo relativo ao analisador léxico que ilustra algumas definições de *tokens* no formato de expressões regulares são exibidos na Fig. 6 e outro trecho do arquivo para o analisador sintático mostrando as regras para a formação dos elementos sintáticos podem ser visualizados na Fig. 7.

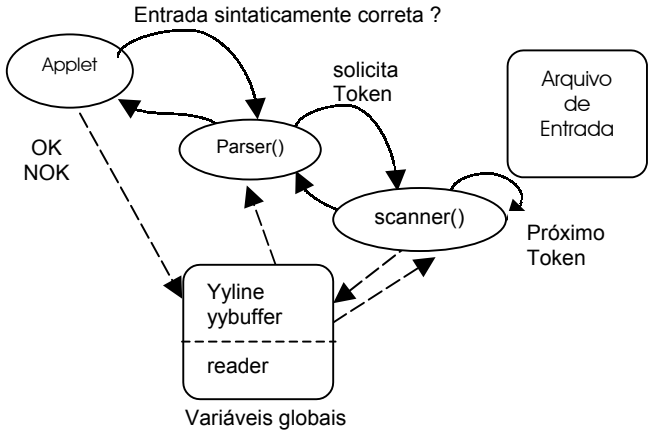

FIGURA. 5
ALGORITMO EXEMPLO NA LINGUAGEM POSTWEL QUE CALCULA O FATORIAL DE UM NÚMERO.

```
...
/* identifiers */
Identifier = [:jletter:][:jletterdigit:]*
/* integer literals */
DecIntegerLiteral = 0 | [1-9][0-9]*
%state STRING, CHAR
%%
/* ------------------------Lexical Rules Section-------*/
<YYINITIAL> {
/* palavras chaves */
"algoritmo"                    { return symbol(ALGORITMO); }
"inicio"            { return symbol(INICIO); }
"fim"                          { return symbol(FIM); }
…
```

FIGURA. 6
TRECHO DO ARQUIVO DE CONFIGURAÇÃO ".JFLEX" (ANALISADOR LÉXICO).

```
...
non terminal var_caracter;
non terminal var_logica;
non terminal var_numerica;
non terminal vetor;
/* start with algoritmo; */
start with postwel;
/****** gramatica ******/
postwel ::= ALGORITMO nome PONTO_VIR  bloco;
nome  ::= _STRING_ ;
bloco ::= INICIO corpo FIM;
bloco_comandos ::= INICIO comds FIM;
corpo  ::=   declaracao corpo
          | comds;
comds ::= | atribuicao PONTO_VIR comds
          | entradaSaida comds
          | se_condicional comds
...
```

FIGURA. 7
TRECHO DO ARQUIVO DE CONFIGURAÇÃO ".CUP" (ANALISADOR SINTÁTICO).

4.2.3 Modelagem, desenvolvimento e resultado do Interpretador

O interpretador foi modelado utilizando a linguagem UML (*Unified Modeling Language*). Uma parte do diagrama obtido é descrito na Fig. 8 [16]. O desenvolvimento foi realizado com Java, utilizando Jflex e Cup, a interface gráfica foi implementada utilizando o pacote Swing e um componente do pacote AWT [12][13][14].

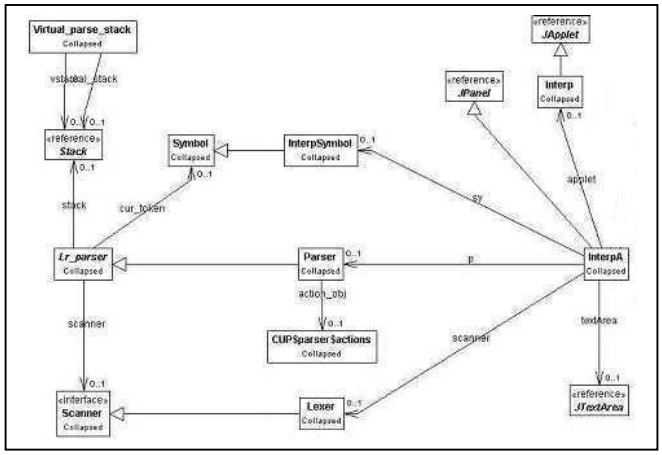

FIGURA. 8

DIAGRAMA DE CLASSES EM UML QUE MODELA O INTERPRETADOR DE ALGORITMOS.

O resultado da interface para o interpretador e a execução de um algoritmo é ilustrada na Fig. 9.

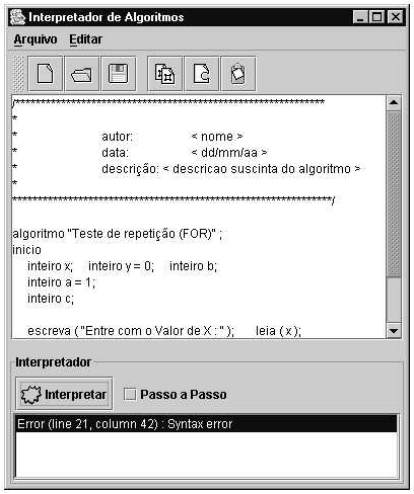

FIGURA. 9

INTERFACE DO INTERPRETADOR , NO FORMATO DE APLICAÇÃO, ILUSTRANDO A COMPILAÇÃO DE UM ALGORITMO.

5. CONCLUSÕES E TRABALHOS FUTUROS

A utilização dos objetos em disciplinas presenciais ou semi-presenciais possibilitará ao professor ilustrar determinadas situações nem sempre tão claras quando a aula é apresentada de maneira tradicional [3].

A possibilidade de maior interação, facilidade de navegação e simplicidade permitirá que o aluno fique mais estimulado a estudar conteúdos de difícil compreensão.

Os objetos construídos poderão ser reusados como um *framework* em disciplinas técnicas e/ou científicas com pouca ou nenhuma alteração interna.

Como trabalhos futuros pretende-se:

- Acrescentar a analise semântica ao interpretador.
- Possibilitar a conversão dos algoritmos criados em PoStWEL para linguagens de programação tipo C/C++ e Java de forma automática.
- Possibilitar a execução passo a passo do algoritmo previamente compilado dentro do interpretador.
- Aplicar o material modelado em disciplinas presenciais avaliando o desempenho dos alunos e comparando ao ensino tradicional.

AGRADECIMENTOS

Os autores agradecem à Coordenadoria de Informática da UNICSUL e o apoio financeiro dado pelo Programa de Qualificação Docente da UNICSUL.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] Dipaolo, A., "Developing University/Industry Partnerships in Distance Education: the Stanford experience", *In: Simpósio Internacionalde Redes e Educação à Distância*, Rio de Janeiro, 1998.

[2] Powell, T.A., "Web Site Engineering; Beyound Web Page Design", *Prentice Hall,,*1998.

[3] Valente, J.A., "Computadores e conhecimento: repensando a eduação", Campinas, Unicamp, 1997.

[4] Educational objects economy website [On-line]. Disponível em: http://www.eoe.org/eoe.htm.

[5] Araújo Jr., C.F., Amaral, L.H., Oliveira, I.C.A., Silveira, I.F. "Novas Tecnologias de Informação e Comunicação e Educação a Distância no Ensino Superior: experiências na área de Computação e Informática". *VI Congreso Internacional en Educación a Distancia - Mercosur/CREAD*. Antofagasta, Chile, Agosto, 2002.

[6] Wiley, D. A. "Connecting learning objects to instructional design theory: A definition, a metaphor, and a taxonomy", *In D. A. Wiley, Ed.*, The Instructional Use of Learning objects, 2001 [On-line]. Disponível em: http://reusability.org/read/chapters/wiley.doc.

[7] Diretrizes Curriculares para os Cursos na área de Computação e Informática [On-line]. Disponível em: http://www.inf.ufrgs.br/ceeinf.

[8] Hopcroft, J. E., Motwani, R., Ullman, J. D. "Introduction to Automata Theory, Languages, and Computation" 2ª Ed., *Addison Wesley*, New York, USA, 2001.

[9] Learning Object Metadata Working Group of the LTSC website [On-line]. Disponível em: http://ltsc.ieee.org.

[10] Goodman, D., "JavaScript Bible", 3rdEd., *IDG Books Worldwide Inc.*, NewYork, USA., 1998.

[11] Teague, J.C. "DHTML e CSS para a World Wide Web", 2ª Ed., *Ed. Campus*, Rio de Janeiro, RJ., 2001.

[12] Deitel, H.M., Deitel, P.J. "How To Program in Java", 3rd.Ed., *Prentice Hall Inc.* ,New Jersey, 2000.

[13] JFLEX The Fast Scanner Generator for Java website [On-line]. Disponível em: http://www.thegateway.org/.

[14] CUP LALR Parser Generator for Java website [On-line]. Disponível em: http://cs.princeton.edu/~appel/modern/java/CUP/index.html.

[15] Watt, D. A., Brown, D. F. "Programming Language Processors in Java: Compilers and Interpreters", *Prentice Hall, Inc.*, New Jersey, 2000.

[16] Booch, G., Rumbaugh, J., Jaccobson, I. "The Unified Modeling Language User Guide", *Addison-Wesley*, 1999.