DOI: 10.19180/978-85-99968-49-9.4

As etapas da construção de objetos de aprendizagem em Flash e HTML5 a serviço dos professores de matemática para apoiar ações de ensino e aprendizagem

Leandro Pires de Souza
Instituto Federal Fluminense [leandro.pires.souza@outlook.com]
Graduando em Bacharelado em Sistemas de Informação
Arilise Moraes de Almeida Lopes
Instituto Federal Fluminense [arilise@iff.edu.br]
Doutora em Informática na Educação/UFRGS

Diante dos avanços tecnológicos que permeiam todas as áreas do conhecimento, o uso de recursos pedagógicos na Educação apresenta-se como uma possibilidade para favorecer processos de ensino e aprendizagem na sala de aula.

No entanto, ao expandir o repertório tecnológico dos docentes, como meio de instrumentalizá-los para uma prática pedagógica diferente da tradicional, há de se desenvolver nestes uma reflexão crítica e competente para o uso das Tecnologias de Informação e Comunicação (TIC) (GARCIA et al., 2011).

Para que haja uma educação de qualidade, o uso das TIC deve ser apoiado por uma proposta pedagógica coerente com os processos de ensino e aprendizagem desenvolvidos na sala de aula. Dentre as TIC, em apoio ao ensino e aprendizagem, estão os Objetos de Aprendizagem (OA) que, na última década, vêm sendo discutidos por pesquisadores de diversas áreas de ensino como uma alternativa de recurso pedagógico, possibilitando o estímulo do raciocínio e o pensamento crítico dos alunos, quando trabalhados na sala de aula ou em espaços extraclasse (MACEDO; LAUTERT; CASTRO-FILHO, 2008).

Várias são as definições para OA pesquisadas. A proposta por Wiley (2000) é a mais citada. O autor define OA como qualquer recurso digital que pode ser reutilizado para assistir à aprendizagem. Não existe um consenso entre os autores, sobre o termo OA, havendo, na literatura, muitos outros termos utilizados, tais como objetos educacionais (TAROUCO; FABRE; TAMUSIUNAS, 2003), objetos instrucionais (GIBBONS; NELSON; RICHARDS, 2000), objetos espertos (ABDULMOTALEB et al., 2000), objetos funcionais (GOMES et al., 2005) e objeto do conhecimento (GLUZ; XAVIER, 2011).

Essas definições resultam de diferentes visões acerca de sua aplicação pedagógica e tecnológica. Embora algumas delas sobre OA tenham um

enfoque mais operacional, este artigo baseia–se na definição de Lopes (2012), que considera OA um recurso digital ou não digital a ser usado em ações de ensino e aprendizagem, composto por processos de mediação do conhecimento entre professor-aluno e aluno-aluno, na utilização do objeto, de forma a possibilitar novos conhecimentos.

No Instituto Federal de Educação, Ciência e Tecnologia Fluminense, entre os núcleos de pesquisa existentes, há o Núcleo de Tecnologias Educacionais e Educação a Distancia (NTEAD) e Núcleo de Informática na Educação (NIE), dos quais os autores deste trabalho participam. Entre as linhas de pesquisa, desenvolvem-se os OA em Flash e HTML5 voltados para o ensino de Matemática, para serem oferecidos a professores e a alunos do Ensino Médio da rede pública.

Diante dos avanços nas pesquisas com a linguagem HTML5, uma das linhas de pesquisa do NIE se volta para o "Desenvolvimento de Objetos de Aprendizagem Digitais Acessíveis para alunos com Deficiência Visual". O grupo pertencente a este núcleo desenvolveu um OA em HTML5.

Assim, este artigo apresenta os procedimentos metodológicos no desenvolvimento de um OA voltado para o estudo do conteúdo de Adição e Subtração de Matrizes para o Ensino Médio, usando a linguagem HTML5 e a comparação com a implementação do mesmo objeto desenvolvido com o *software* Macromedia Flash 8, destacando-se as vantagens e as desvantagens observadas na implementação em HTML5. Para esse OA, foi proposto também implementar requisitos de acessibilidade para atender pessoas com deficiência visual.

O artigo está estruturado em seis seções, a saber: "Histórico do HTML5", "Procedimentos Metodológicos para o desenvolvimento do Objeto de Aprendizagem em HTML5: *Adição e Subtração de Matrizes"*, "Ambientes de desenvolvimento", "Implementação dos Requisitos de Acessibilidade", "Comparações entre os OA em HTML5 e Flash" e "Considerações Finais".

1. HISTÓRICO DO HTML5

HTML (*HyperText Markup Language*) é uma linguagem de marcação para hipertextos criada por Tim Bernes-Lee, tendo logo se tornado a principal linguagem utilizada na construção de páginas para a *Web,* sendo seus padrões definidos pela *World Wide Web Consortium* (W3C, 2008).

Em 1997, a W3C publicou o HTML4.0 e, em 1999, apenas dois anos após, o HTML4.01 surgiu como uma revisão do HTML4.0, tendo saído ainda uma errata quanto ao HTML4.01 em 2001. Ambos traziam diversos recursos

como o suporte a folhas de estilos e a possibilidade de implementar objetos que não fossem em HTML como, por exemplo, aplicações em Flash ou Java *Applets*. Também criou-se um novo modelo para tabelas, além de novos parâmetros que visavam aumentar a acessibilidade da página.

Em 2004, surge o WHATWG (*Web Hypertext Application Technology Work Group*), grupo de trabalho não oficial com colaboração aberta dos desenvolvedores dos principais *Browsers* do mercado, com o objetivo de elaborar especificações baseadas em HTML e tecnologias relacionadas, para facilitar o desenvolvimento de aplicações *Web* interoperáveis e posteriormente submeter seus resultados à W3C. Esse grupo de trabalho deu origem ao HTML5. Em 2008, a W3C anuncia o HTML5, com base nas especificações entregues pela WHATWG (PILGRIM, 2010).

HTML5 é a nova versão da linguagem de marcação para a *Web* HTML, provendo novos recursos para a criação de aplicações *Web* moderna e padronizando recursos já utilizados por desenvolvedores há anos, sem que os mesmos tivessem sido vetados ou documentados por um comitê de padronização (PILGRIM, 2010).

As expectativas quanto ao HTML5 para o futuro na Web vêm crescendo a cada dia. Houve uma preocupação com os OA desenvolvidos no núcleo de pesquisa, que, até o ano de 2012, eram desenvolvidos exclusivamente em Flash. Diante da possibilidade de se utilizar a linguagem HTML 5, surgiu a ideia de se desenvolver um objeto com essa linguagem, descrito na seção a seguir.

2. PROCEDIMENTOS METODOLÓGICOS PARA O DESENVOLVIMENTO DO OBJETO DE APRENDIZAGEM EM HTML5: *ADIÇÃO E SUBTRAÇÃO DE MATRIZES*

O desenvolvimento do OA pelo grupo de pesquisa está baseado em Polsani (2003), que define quatro etapas, a saber: concepção do projeto, planificação, implementação e validação. Descrevem-se essas etapas percorridas para o desenvolvimento do OA em HTML5: Adição e Subtração de Matrizes nas próximas subseções.

2.1 Concepção

A concepção do OA iniciou-se com a equipe do NTEAD/NIE composta por uma professora com formação em Matemática, duas bolsistas de

iniciação científica do curso de Licenciatura em Matemática, uma bolsista da área de Design Gráfico e dois bolsistas da área de Ciência da Computação.

Definiu-se que, diante do objetivo de se desenvolver um OA em HTML5 e fazer uma análise comparativa com o Flash, seria escolhido um OA já desenvolvido em Flash. Dessa forma, decidiu-se pelo OA Adição e Subtração de Matrizes.

Os recursos gráficos do objeto em HTML5 e os conteúdos elaborados foram os mesmos do objeto já desenvolvido em Flash, utilizando o critério de reusabilidade[1]. Como já estava definido o tema "Matrizes", o público-alvo, que são alunos do Ensino Médio e os conteúdos de Adição e Subtração de Matrizes, avançou-se para a etapa de planificação.

2.2 Planificação

A etapa de planificação inicia-se com a criação de um Mapa Conceitual (Figura 1) que detalha como o conteúdo será dividido em nós ou em unidades. Como os nós serão exibidos, quais as mídias serão utilizadas e como o usuário vai interagir com a aplicação. É a organização das informações e das mídias (FALKEMBACK, 2005).

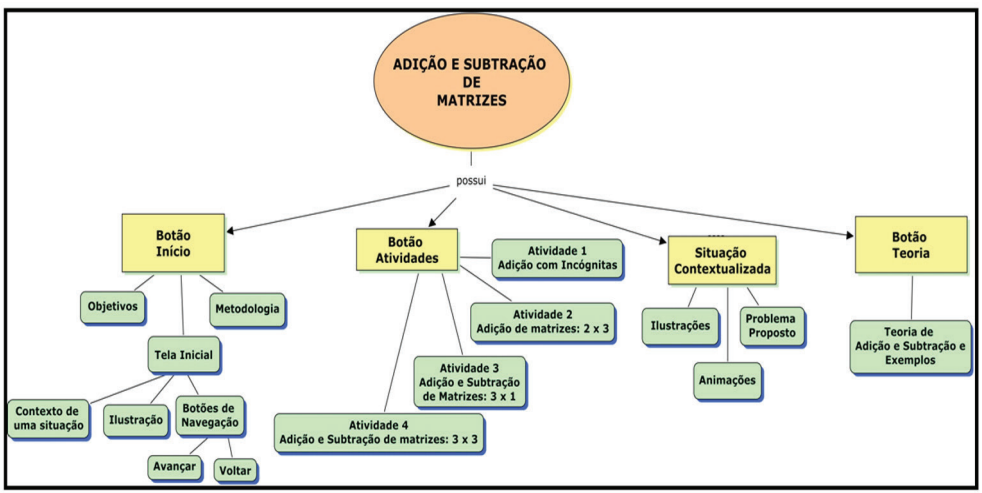

Figura 1 – Mapa conceitual
Fonte: Elaboração própria (2014).

Após a construção do Mapa Conceitual, foi elaborado um *storyboard*, de acordo com os estudos de (FALKEMBACK, 2005) que propõe o uso de

[1] Reusabilidade: diversas possibilidades de se adaptar um determinado de um mesmo recurso a diferentes unidades de aprendizagem (ÁVILA et al., 2013).

storyboards[2] para a representação das telas de um OA. Tomando por base um objeto já desenvolvido em Flash, o OA foi composto de uma tela inicial, telas da situação contextualizada, tela de teoria e tela de atividades. A resolução das telas manteve-se em 800 x 600, com 480.000 pixels.

Discutiram-se os mecanismos de navegação do objeto. Este foi composto de 17 telas. Todas têm a mesma estrutura (barra de navegação e zona de conteúdo), de maneira que quando se navega pelas telas, há a possibilidade de uma navegação não linear em algumas rotas, enquanto em outras definiu-se uma navegação linear.

2.3 Implementação

Nessa etapa, foram utilizadas ferramentas como bibliotecas e linguagens, além do HTML, todos os projetos de *Software* Livre ou código aberto, que permitiram a construção do OA em HTML5. Entre essas ferramentas, cabe destacar *jQuery*, que é uma biblioteca JavaScript *cross-browser*. Nela, há uma ampla API criada para facilitar e agilizar o desenvolvimento de códigos JavaScript. Alguns recursos de sua API foram utilizados para desenvolver animações no OA. Todos os recursos gráficos utilizados no desenvolvimento desse objeto foram exportados do projeto original em .fla para arquivos de imagem comuns de extensão .png.

Utilizou-se, como referência estrutural, um projeto desenvolvido pelo Google que consiste em uma apresentação de slides construído inteiramente em HTML5. Como os OA desenvolvidos pelo NTEAD possuem certa semelhança estrutural com o projeto do Google, ambos se baseiam em um grupo de telas pelo qual o usuário navega, pode-se tirar proveito da estrutura dos slides de modo que cada tela fosse composta por um elemento HTML do tipo *article* (Quadro 1). Esse elemento, implementado junto com o HTML5, faz parte dos novos recursos semânticos, nos quais novos elementos têm significado próximo de sua implementação, tal como os elementos *footer* (para rodapé) e *header* (para cabeçalho) ou *nav* (para navegação), por exemplo.

[2] *Storyboard* pode representar um esboço do modelo de uma aplicação e mostra como seus elementos estão organizados. Possibilita um planejamento do conteúdo de cada unidade e a disposição das mídias. Caracteriza-se por um "rascunho" da aplicação, permitindo aos responsáveis pelo projeto visualizarem sua estrutura de navegação, ou seja, discutirem a sequência do conteúdo e fazerem as revisões e o acompanhamento necessários para o bom andamento do trabalho (FALKEMBACK, 2005).

```
<section id="telas">
   <article>
   </ article>
</section>
```

Quadro 1 - Estrutura básica em HTML do conjunto de telas
Fonte: Elaboração própria (2014).

Tendo a estrutura básica das telas pré-definida, pode-se utilizar o *CSS* (*Cascading Style Sheet,* ou Folha de Estilos em Cascata). É uma linguagem que complementa o HTML de forma a definir o aspecto visual dos elementos. Junto com o HTML5, vieram vários novos recursos também para o CSS, a fim de nominar uma série de características estéticas comuns a todas as telas (Quadro 2) como suas dimensões, seu posicionamento e margem interna, além de parte da interface padrão das telas, com sua borda e seu painel de rodapé característico, sendo estes últimos implementados por meio de uma imagem como plano de fundo.

```
section#telas > article{
   display: none;
   position: absolute;
   width: 800px;
   height: 600px;
   Background-color: #FFFFFF;
   left: 50%;
   top: 50%;
   margin-left: -400px;
   margin-top: -300px;
   Background-image: url(../imagens/painel/quadro2.png);
   padding: 15 30 15 30;
}
```


Quadro 2 – Uso do CSS: definição de características estéticas
Fonte: Elaboração própria (2014).

No rodapé da tela, encontram-se elementos de navegação (avançar e voltar), de teoria, de atividades e um botão chamado *inicial*. Fazem parte da interface padrão e estão presentes em todas as telas. A inclusão desses elementos foi feita a partir de um código JavaScript com o auxílio da biblioteca jQuery (Quadro 3). Dessa forma, eles foram incluídos, automaticamente, em todas as telas sem a necessidade de adicionar manualmente cada um deles.

```
$('section#telas > article').append("<div class='painel'></div>");

$('section#telas > article').append("<div id='btnContainer'><input
type='button' class='btnNavVo' title='Voltar'><input type='button'
class='btnNavAv' title='Avançar'></div>");


$('section#telas > article').append("<input type='button'

class='btnTeoria'>");
```

Quadro 3 – Código JavaScript exemplificando elementos de navegação do objeto
Fonte: Elaboração própria (2014).

Posteriormente esses elementos tiveram suas características estéticas manipuladas, utilizando-se do CSS. Nos elementos denominados por botões de avançar, voltar, teoria, atividades e inicial, foi atribuído o código JavaScript para que os esses botões pudessem realizar suas respectivas funções. O CSS foi utilizado a fim de definir características diferenciais de cada tela, imagens de planos de fundo diferentes.

A interface implementada na primeira tela descreve o objetivo do estudo. Ressalta-se que todas as telas tiveram seu conteúdo inteiramente baseado nas telas correspondentes da versão em Flash do mesmo OA. Sua estrutura HTML (Quadro 4) é descrita a seguir.

```
<article>
   <img src="oa/imagens/telas/quadro.png" class="quadro" alt="quadro">
   <p class="tela1texto1">
       O nosso estudo tem por objetivo a aprendizagem de alguns conceitos de
Matrizes.
   </p>
   <p class="tela1texto2">
       O que vamos estudar?
       <br/>- Adição de Matrizes.
       <br/>- Subtração de Matrizes.
   </p>
   <p class="tela1texto3">
        O que difere este estudo dos modelos tradicionais de ensino?
        <br/>- Oferecer a possibilidade da interação com o objeto de estudo.
   </p>
   <p class="tela1texto4">
       Vamos conhecer?
   </p>
   <p class="tela1texto5">
       Clique no botão "avançar" para irmos aos conceitos.
   </p>
   <img src="oa/imagens/telas/ob1.png" class="objetos_na_mesa" alt="objetos
sobre a mesa">
</article>
```

Quadro 4 - Estrutura HTML da tela de apresentação do OA
Fonte: Elaboração própria (2014).

O código CSS dessa tela e de alguns de seus elementos encontram-se no Quadro 5.

```
section#telas > article:nth-child(1){
    background-image: url(../imagens/painel/quadro2.png),
url(../imagens/telas/cen1.png);
    background-position: 0px 0px, 15px 0px;
    background-size: 100% 100%, 770px 560px;
}
img.quadro{
   position: absolute;
   width: 515px;
   height: 327px;
   top: 95px;
   left: 66px;
}
img.objetos_na_mesa{
   position: absolute;
   top: 465px;
   left: 100px;
   width: 300px;
}
p.tela1texto1{
   width: 443px;
   position: absolute;
   top: 124px;
   left: 102px;
   color: #FFFFFF;
   font-size: 14px;
   font-family: verdana;
   line-height: 19px;
   text-align: justify;
}
```


Quadro 5 - Código CSS de alguns dos elementos da tela 1
Fonte: Elaboração própria (2014).

Na implementação da tela de apresentação em HTML5, como nas demais telas, teve-se o cuidado de manter a mesma interface gráfica para que se pudesse fazer uma comparação entre a versão em Flash e a versão em HTML5. Apresenta-se a tela inicial (Figura 2) com as duas versões implementadas.

Figura 2 – Tela de apresentação do objeto implementada em HTML5 e Flash
Fonte: Elaboração própria (2014)

Em uma das telas, após a contextualização de uma situação, é proposta uma animação, em que frutas de determinadas espécies de uma caixa, mais as mesmas espécies de frutas de outra caixa são transportadas para uma terceira caixa. O objetivo dessa dinâmica é apresentar o conceito de Adição de Matrizes, que mostra espécies semelhantes de frutas transportadas para determinada localização na caixa. Dessa forma, os alunos podem refletir sobre esse o conceito.

Para realizar essa animação, foi utilizada a função animate() da API jQuery. Na implementação, basta apenas ter descrito como parâmetro as propriedades CSS desejadas para o elemento, a duração e uma função, caso se queira, a ser executada ao fim da animação. No caso, trata-se de duas animações de dois grupos de elementos. A segunda animação sucedendo à primeira, e a primeira sucedendo à segunda de modo que se crie um loop. Foi utilizada, ainda, a função .delay() também da API jQuery para atrasar a execução das animações, evitando, assim, que elas acontecessem simultaneamente. O desejado seria que uma animação sucedesse à outro como pode ser visto no Quadro 6.

```
$('.f1').animate({
   'left': '+470'
},5000, function(){
   $('.f1').css({
       left:'118px'
   });

   $('.f2').animate({
       'left': '+470'
   },3000,moveFrutas).delay(5000);
}).delay(3000);
```

Quadro 6 - Programação para a animação da tela
Fonte: Elaboração própria (2014).

A tela implementada por essa programação é apresentada na Figura 3.

Figura 3 – Interface da tela que apresenta a animação descrita na implementação
Fonte: Elaboração própria (2014).

Na implementação da tela há três matrizes: as duas primeiras são matrizes vazias e a terceira matriz possui valores. O usuário terá que preencher esses campos. Sugere-se que o usuário faça uma soma de matrizes, preenchendo as matrizes vazias, elemento por elemento de modo que a soma destas resultem nos valores apresentados pela terceira matriz.

Ao clicar no botão *responder*, é executada uma função que guarda cada valor de cada elemento das matrizes anteriormente vazias em suas respectivas variáveis. Após essa ação, é feita uma comparação para verificar se a soma de cada dupla de elementos resulta no valor desejável. Caso todas as condições sejam satisfeitas, um determinado elemento exibirá uma mensagem, indicando o êxito do usuário e modificando uma variável de controle posteriormente utilizada para verificar se o usuário poderá avançar para a próxima tela ou não (Quadro 7).

```
$('input#responder').click(function (){
    var a1 = parseInt($('input#a1').attr('value'));
    var b1 = parseInt($('input#b1').attr('value'));
    var a2 = parseInt($('input#a2').attr('value'));
    var b2 = parseInt($('input#b2').attr('value'));
    var a3 = parseInt($('input#a3').attr('value'));
    var b3 = parseInt($('input#b3').attr('value'));
    var a4 = parseInt($('input#a4').attr('value'));
    var b4 = parseInt($('input#b4').attr('value'));
    if ((a1+b1 == 12)&&(a2+b2 == 14)&&(a3+b3 == 10)&&(a4+b4 == 9)){
        $('span#resposta').text('Parabéns! Valores corretos.');
            tela9correcao = 1;
    }else{
        $('span#resposta').text('INCORRETO! Tente outra vez.');
        tela9correcao = 0;
    }
});
```

Quadro 7 - Código do botão responder
Fonte: Elaboração própria (2014).

Apresenta-se a interface dessa atividade implementada (Figura 4).

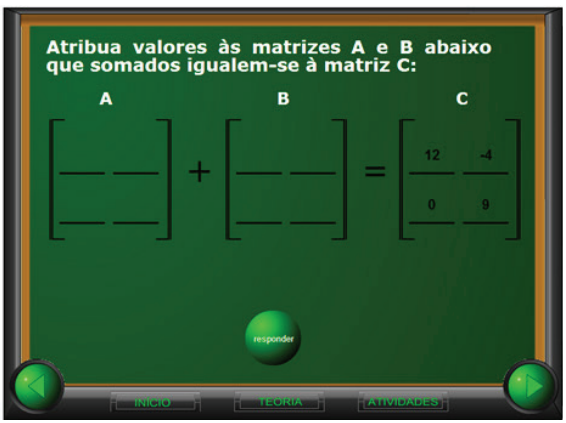

Figura 4 – Proposta de atividade de adição de Matrizes.
Fonte: Elaboração própria (2014).

Uma segunda situação contextualizada foi implementada. Na implementação da tela que descreve a situação, o usuário deverá fazer uma subtração de matrizes de modo a descobrir a diferença nas idades. De maneira semelhante, a implementação anterior (Quadro 7), ao se clicar no botão *responder*, o valor presente no campo de resposta deverá ser alocado em uma variável que, em seguida, é comparada ao valor desejado . Caso o valor satisfaça essa condição, um elemento exibirá uma mensagem indicando o êxito do usuário e, em seguida, uma variável de controle usada para permitir ou negar o avanço para a próxima tela será alterada. Do contrário, a mensagem indicará que o usuário errou a resposta e o campo de resposta ficará vazio; receberá então o foco para que o usuário tente novamente até que o mesmo chegue à resposta correta (Quadro 8).

```
$('9input#respondert15').click(function (){
   var resposta = parseInt($('input#quanto').attr('value'));
   if (resposta == 28){
       $('span#respostat15').text('Parabéns! valores corretos.');
       tela15correcao = 1;
   }else{
       $('span#respostat15').text('INCORRETO! Tente outra vez.');
       $('input#quanto').attr('value','').focus();
       tela15correcao = 0;
   }
});
```

Quadro 8 – Implementação da funcionalidade do botão responder
Fonte: Elaboração própria (2014).

A interface da implementação acima é apresentada na Figura 5.

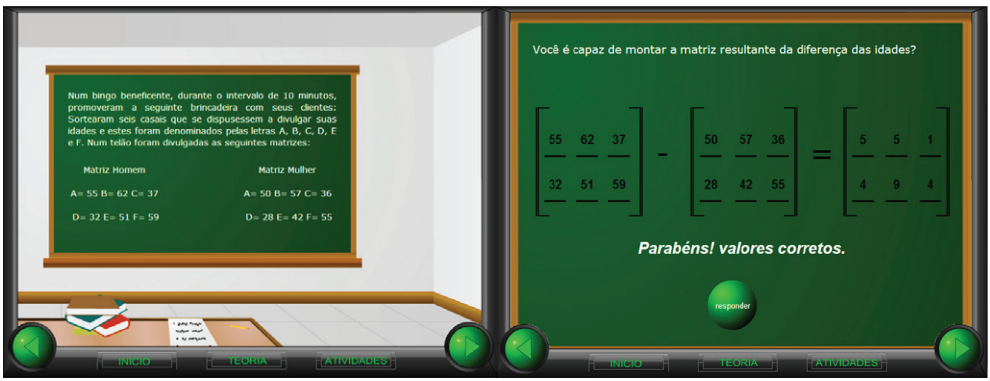

Figura 5 – Tela da segunda situação contextualizada: Subtração de Matrizes
Fonte: Elaboração própria (2014).

Como já descrito, o botão teoria é proposto para aprofundamento da situação proposta. Descreve-se uma das telas implementadas da Teoria (Quadro 9).

```
<article class="teoria">
   <h1>
        Teoria
   </h1>
   <h2>
        Adição
   </h2>
   <p>
   As matrizes envolvidas na adição devem ser da mesma ordem e o resultado
dessa soma será também outra matriz com a mesma ordem.
   </p>
   <p>
       Assim podemos concluir que:
   </p>
   <p>
              Se somarmos a matriz A com a matriz B de mesma ordem, A +<span
class="invisivel">mais</span> B =<span class="invisivel">igual a</span> C,
teremos como resultado outra matriz C de mesma ordem e para formar os
elementos de C somaremos os elementos correspondentes de A e B, assim:
a<sub>1<span class="invisivel"> </span>1</sub> +<span
class="invisivel">mais</span> b<sub>1<span
class="invisivel"> </span>1</sub> =<span class="invisivel">igual
a</span> c<sub>1<span class="invisivel"> </span>1</sub>.
   </p>
</article>
```

Quadro 9 – código HTML da tela Teoria
Fonte: Elaboração própria (2014).

Essa implementação é apresentada na interface da Figura 6.

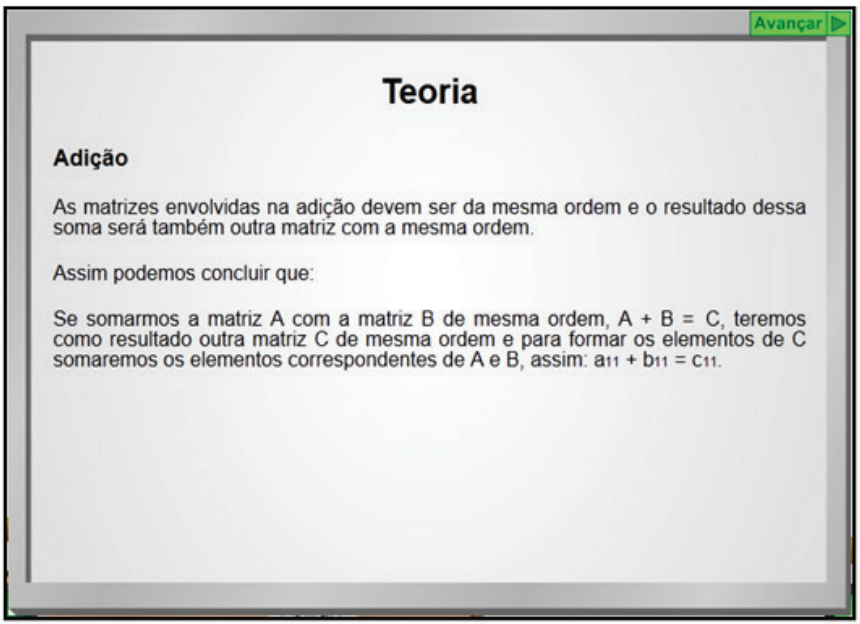

Figura 6 - Tela de Teoria do OA
Fonte: Elaboração Própria (2014).

Entre as atividades propostas, no botão *atividade*, exemplifica-se a implementação de uma das atividades. Assim, como as demais telas do OA, cada tela do botão a*tividades* é representada por elementos *article* e dentro de cada um desses elementos se encontram os parágrafos e as matrizes que compõem cada tela.

Sendo essa tela composta por diversas matrizes, foi decidido que para representá-las seriam utilizados elementos do tipo *table* que, segundo suas especificações, são geralmente usados para implementar tabelas comuns. Esse elemento foi escolhido, porque cumpre bem o papel de representar uma matriz visualmente. Também se comporta muito bem, quando lido por um leitor de tela, de modo a permitir ao usuário com deficiência explorar tabelas, elemento por elemento ou linha por linha, dando ao mesmo essa mesma possibilidade com as matrizes, quando se utiliza o *table* para representá-las. Apresenta-se o código de uma das matrizes implementadas na tela (Quadro 10).

```
<article>
   <h1>Atividades</h1>
       <p>
           1) Determine a matriz C, resultado da soma das matrizes A e B.
       </p>
       <p>
           A =<span class="invisivel">igual a</span>
       </p>
           <div class="a1">
               <table>
                   <tr>
                        <td>
                            -3
                        </td>
                           <td>
                                   5
                           </td>
                           <td>
                                   2
                           </td>
                   </tr>
                   <tr>
                           <td>
                                   6
                           </td>
                           <td>
                                   4
                           </td>
                           <td>
                                   8
                           </td>
                   </tr>
               </table>
           </div>
           <p>
```

Quadro 10 - Parte do código da primeira tela Atividades
Fonte: Elaboração própria (2014).

Nesta atividade, a interface é representada pela figura 7.

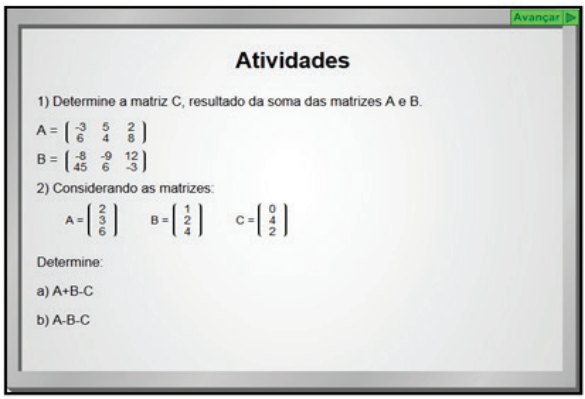

Figura 7- Primeira tela da seção Atividades
Fonte: Elaboração própria (2014).

Finaliza-se, assim, a descrição de implementação de algumas telas do OA "Adição e Subtração de Matrizes", sendo que todas as demais telas tiveram o mesmo procedimento.

2.4 Validação

Ao término da implementação pelo bolsista da Área da Ciência da Computação, o OA foi apresentado à equipe. Discutiu-se, com os membros da equipe, as dificuldades encontradas na implementação do objeto. Em todos os testes de validação feitos pela professora pesquisadora e pelas duas bolsistas do curso de Licenciatura em Matemática, foram feitos relatórios para que se pudessem registrar as recomendações e reformulações, de forma a rever o processo de implementação e corrigir as distorções verificadas.

3. AMBIENTES DE DESENVOLVIMENTO

No desenvolvimento do OA em Flash, foi utilizado um *software* próprio da Adobe para este fim, com uma interface rica, permitindo a importação direta de recursos gráficos de outros *softwares* da Adobe, incluído o Adobe Illustrator, utilizado pela Designer da equipe para a criação dos recursos gráficos. No desenvolvimento do OA em HTML5, foram utilizados editores de textos comuns, tendo como ferramentas a própria linguagem para a importação dos recursos gráficos. Foi necessário exportar esses recursos

em formato bitmap. Todo o desenvolvimento em HTML5 foi realizado em contato direto com os códigos HTML, CSS e JavaScript.

O Macromedia Flash Professional 8 (que nesta versão ainda pertencia à Macromedia, sendo posteriormente adquirido pela Adobe Systems) apresenta, em sua interface, diversas áreas (Figura 8). O palco (também conhecido como *stage*) é inicialmente uma grande área branca. Nele, ficam todos os elementos que serão mostrados, quando a aplicação for reproduzida.

A barra de ferramentas, apesar de poder ser arrastada para qualquer lugar da tela, inicialmente, localiza-se no canto esquerdo. Nessa barra, estão várias ferramentas úteis à manipulação gráfica na aplicação a ser desenvolvida. Na parte superior da tela, está a linha do tempo (ou *timeline*), onde ficam os frames que são unidades que definem a temporalidade ou o encadeamento de eventos da aplicação em uma aplicação já pronta.

Ao se navegar pelos frames, é possível ver que o palco muda, apresentando o que será mostrado no momento em que aquele frame será executado. Na biblioteca (ou *library*) ficam guardados os recursos utilizados na aplicação como imagens, botões ou *movieclips* (objetos multimídia). Para utilizá-los em um determinado momento, basta arrastá-los da biblioteca para o palco. Há, também, uma área de propriedades onde é possível manipular as definições de um recurso, bem como a formatação de textos, entre outras propriedades.

Existe, ainda, uma área bastante importante, porém inicialmente oculta, que é a área chamada *Actions*, onde é possível inserir código *ActionScript,* para controlar os recursos ou frames, receber e tratar eventos dos usuários (como o clique em um botão, por exemplo).

Apresenta-se a interface (Figura 8) do programa Macromedia Flash Professional 8.

Figura 8 - Interface do programa Macromedia Flash Professional 8
Fonte: Elaboração própria (2014).

No ambiente de desenvolvimento do OA Adição e Subtração de Matrizes em HTML5, foi utilizado o editor de textos Brackets. A interface desse editor de textos voltado para o desenvolvimento do OA, é bastante simples, semelhante a editores de texto comuns, como o Microsoft Word, o Microsoft WordPad ou o Bloco de Notas do Windows, porém sem qualquer recurso de formatação de textos, recuo de página, uma vez que nenhuma linguagem de programação leva em conta nenhuma dessas características.

O editor de texto utilizado possui adaptações relevantes para quem programa, por exemplo, contagem de linhas, atalhos no teclado ou uma barra lateral, que permite a navegação rápida por arquivos, já que é bastante frequente, durante o desenvolvimento do OA, ter que modificar diversos arquivos.

Foram usados os principais *browsers* do mercado (Internet Explorer, Mozilla Firefox e o Google Chrome, sendo este último mais compatível por utilizar o motor de renderização Webkit, que é mais alinhado às definições da W3C), com o objetivo de avaliar como o OA iria se comportar em cada um desses navegadores e, assim, garantir uma maior compatibilidade.

Utilizaram-se, ainda, ferramentas para desenvolvedor providas pelos próprios browsers, com o fim de validar código JavaScript e, no Mozilla Firefox, visualizar, em tempo real, mudanças no código CSS. Apresenta-se o editor de texto Brackets da Adobe (Figura 9) utilizado durante o desenvolvimento em HTML5.

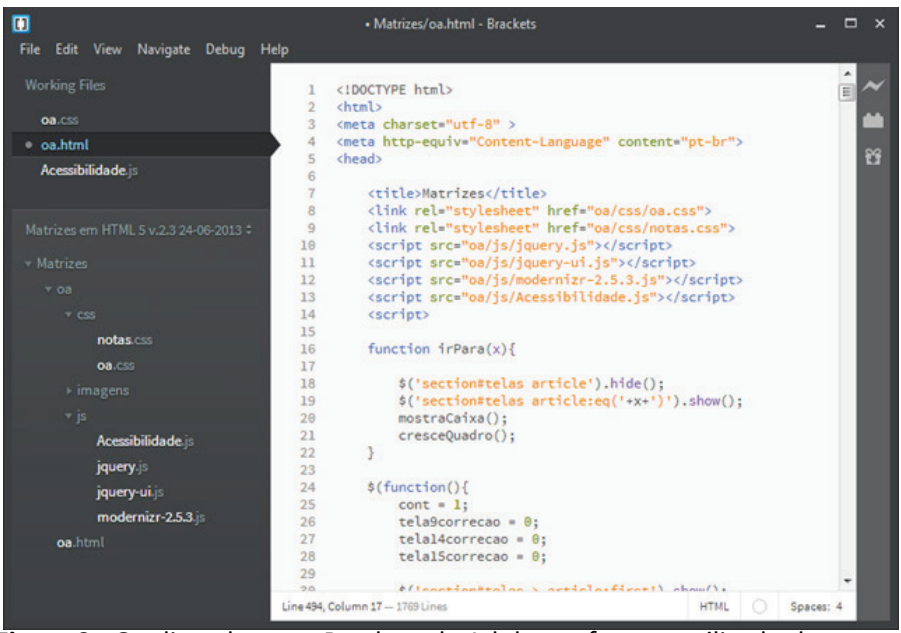

Figura 9 - O editor de texto Brackets da Adobe: *softwares* utilizado durante o desenvolvimento em HTML5

Fonte: Elaboração própria (2014).

É possível perceber que os ambientes de desenvolvimento do OA em Flash e de sua versão em HTML5 são bastante diferentes.

Enquanto o Flash provê uma interface rica na qual cada elemento pode ser inserido em um palco, utilizando comandos já conhecidos ao usuário como o arrastar e soltar (*drag and drop*), o desenvolvimento em HTML5 ficou limitado à descrição direta de cada cenário através de códigos HTML, CSS e JavaScript, utilizando-se diretamente de browsers, para verificação de cada nova implementação.

A seguir, descrevem-se, na próxima seção, as ferramentas que possibilitam tornar acessível um objeto de aprendizagem em Flash e HTML5.

4. IMPLEMENTAÇÃO DOS REQUISITOS DE ACESSIBILIDADE

É possível tornar acessível um OA tanto no Flash quanto no HTML5, porém usando métodos diferentes. Os recursos providos nas duas linguagens quanto à acessibilidade serão pouco efetivos, se o objeto a ser desenvolvido não for pensado desde seu planejamento quanto a sua usabilidade, de modo a possuir uma interface fácil de ser manipulada.

O processo de tornar acessível o objeto de aprendizagem Adição e Subtração de Matrizes iniciou-se com a escolha dos elementos que deveriam ter requisitos de acessibilidade em cada tela do objeto. Para Lopes et al., (2012), tal escolha foi decorrente de uma análise de que nem todas as composições visuais que foram implementadas no objeto de aprendizagem devem ter requisitos de acessibilidade, de forma a evitar sobrecarregar o usuário com informações que não sejam relevantes.

No processo da extensão FLA do *software* Macromedia Flash 8, este possui uma ferramenta denominada Accessibility, que possibilita tornarem acessíveis elementos existentes nas telas de um objeto de aprendizagem. Para deixar esses elementos do objeto de aprendizagem com requisitos de acessibilidade, usou-se o campo *Name* da ferramenta *Acessibility* e marcou-se a opção *Make object accessible.*

A leitura de cada elemento é feita por um leitor de tela. No OA, utilizou-se o leitor de telas NVDA, que normalmente os alunos com deficiência visual usam na instituição, por ser um *software* de domínio público. Para ordenar uma leitura do leitor de telas, foi inserido um valor numérico no campo *Tab índex.* O usuário com deficiência visual, ao abrir o objeto, primeiramente usa a tecla TAB e o leitor de telas descreve o que a tela apresenta. Essa descrição é inserida no campo *Name* (Figura 10).

Figura 10 – Implementação de requisito de acessibilidade em uma tela em Flash
Fonte: Elaboração própria (2014).

Para o aluno com deficiência visual, a exploração deve seguir uma sequência não necessariamente ordenada, mas de forma que o aluno seja conduzido aos elementos que têm requisitos de acessibilidade para serem explorados devido a sua importância para a aprendizagem esperada.

Todos esses elementos que compõem o OA foram escritos através de um texto estático, texto esse que apresenta informações, que são digitadas durante o desenvolvimento e que o usuário final não pode alterar.

O campo *Tab index* possibilita uma sequência ordenada de leitura dos elementos que compõem cada tela, possibilitando ao leitor de telas uma navegação por todos os elementos da tela do OA que receberam requisitos de acessibilidade.

No caso do OA desenvolvido em HTML5, existe certa facilidade quanto a tornar acessíveis páginas HTML. Isso é devido ao fato de que os leitores de tela já estão preparados para ler páginas HMTL comuns. Basta ao desenvolvedor cumprir corretamente os padrões da Web e as especificações adicionais sugeridas pela W3C quanto à acessibilidade de conteúdos Web, como a API WAI-ARIA. Esta permite ao desenvolvedor lidar diretamente com o leitor de tela, indicando, por exemplo, ignorar certo elemento ou tratar esse elemento como se fosse outro, de modo a facilitar bastante a definição do fluxo que o leitor de tela irá seguir.

Existe, ainda, o WCAG (*Web Content Acessibility Guidelines*) que é um conjunto de especificações sobre como construir páginas da Web (ou objetos de aprendizagem) de modo a terem seu conteúdo acessível.

A nova versão do HTML (HTML5) traz ainda mais especificações que auxiliam o desenvolvedor nesse quesito, uma vez que possui uma nova gama

de elementos com semântica mais clara, de modo a possibilitar ao leitor de tela interpretar melhor o conteúdo a ser lido.

Na aplicação dessas especificações, quanto à acessibilidade na primeira tela foi utilizado um atributo do elemento *img*, que permite inserir um texto alternativo à imagem, para que este seja lido, por exemplo, por leitores de tela, quando encontrar essa imagem.

Utilizou-se esse atributo a fim de descrever, para o usuário, o cenário no qual ele se encontra, de modo que ainda que ele não enxergue, consiga perceber melhor o contexto do OA.

Toda estrutura HTML é lida pelo leitor de tela que o interpreta em forma de áudio para o usuário com deficiência visual. Por isso é importante que, nessa estrutura, a ordem em que a informação está disposta seja a ordem em que a mesma deva ser lida.

É possível realocar a ordem dos elementos por meio do CSS de modo que a ordem em que cada elemento está disposto no HTML, pouco importe ao usuário. Porém, essa prática não é recomendada, uma vez que o leitor de tela irá ignorar a disposição segundo o CSS e pode apresentar a informação numa ordem que confunda o usuário.

Assim, em todo o OA, a disposição do conteúdo, em sua estrutura HTML, foi pensada de modo a facilitar o entendimento do usuário que se utiliza de leitores de tela (Quadro 11).


```
<article>
   <img src="oa/imagens/telas/quadro.png" class="quadro" alt="Nesta tela há
uma sala de aula com uma mesa do professor e livros sobre ele e um quadro de
giz no qual está ecrito o objetivo do OA">
    <p class="tela1texto1">
            O nosso estudo tem por objetivo a aprendizagem de alguns
conceitos de Matrizes.
    </p>
    <p class="tela1texto2">
        O que vamos estudar?
        <br/>- Adição de Matrizes.
        <br/>- Subtração de Matrizes.
    </p>
    <p class="tela1texto3">
        O que difere este estudo dos modelos tradicionais de ensino?
        <br/>- Oferecer a possibilidade da interação com o objeto de estudo.
    </p>
    <p class="tela1texto4">
        Vamos conhecer?
    </p>
    <p class="tela1texto5">
        Clique no botão "avançar".
    </p>
    <img src="oa/imagens/telas/ob1.png" class="objetos_na_mesa">
</article>
```

Quadro 11 – Código HTML acessibilizado da primeira tela
Fonte: Elaboração própria (2014).

Descrevem-se, a seguir, as comparações julgadas pertinentes na análise entre os OA implementados.

5. COMPARAÇÕES ENTRE OS OA EM HTML5 E FLASH

A partir da experiência adquirida por meio da implementação do OA em Flash e de estudos sobre HTML5, na qual se desenvolveu um objeto de aprendizagem, o grupo de pesquisa envolvido no desenvolvimento dos dois objetos Flash e HTML5, percebeu algumas características bastante distintas entre as duas implementações.

Discussões foram levantadas a respeito dessas características e verificou-se a necessidade de apresentar vantagens e desvantagens percebidas na implementação dos dois objetos. Apresenta-se (Quadro 12) um resumo dos principais aspectos analisados durante a comparação do objeto construído em Flash e reconstruído em HTML5, destacando-se características de cada objeto e vantagens e desvantagens observadas.

HTML5	Flash
Conjunto de Arquivos (desvantagem)	Arquivo único
Software Livre	*Software* Proprietário (desvantagem)
Suportado nas versões mais recentes dos *Browsers* mais utilizados (desvantagem)	Adobe Flash é suportado em mais de 98% dos computadores
Foi utilizado um bloco de notas específico para o desenvolvimento web e auxiliado por ferramentas para desenvolvedores nos próprios *Browsers* (desvantagem)	Foi utilizado um *software* multimídia para a criação dinâmica de aplicações interativas
Foram encontradas dificuldades na implementação de animações mais complexas (desvantagem)	Animações complexas foram mais facilmente implementadas
Código mais longo (desvantagem)	Código mais curto
Implementação do HTML5 não quebra com os padrões Web	Implementação do Flash quebra com os padrões Web (desvantagem)
Rica documentação	Rica documentação
Baixo custo no desenvolvimento	Desenvolvimento com elevado custo, sendo necessário pagar pelas ferramentas da Adobe, como Flash Professional (desvantagem)
Foram utilizados gráficos bitmap, o que resultou em uma aplicação mais pesada (desvantagem)	Foram utilizados gráficos vetoriais, o que resultou em uma aplicação mais leve
Apresenta facilidade na implementação de acessibilidade	Apresenta acessibilidade limitada (desvantagem)
Grande compatibilidade com dispositivos móveis modernos	Compatibilidade com dispositivos móveis limitadas (desvantagem)

Quadro 12 - Comparativo de OA em HTML5 e FLASH
Fonte: Elaboração própria (2014).

Enquanto no Flash conta-se com apenas um arquivo executável tendo todos os recursos utilizados compactados dentro do próprio executável, o que se entendeu ser mais vantajoso, no HTML5 todos os recursos referenciados são externos, necessitando, assim, no mínimo, de uma pasta na qual esses recursos sejam alocados. Isso apresentou uma desvantagem, uma vez que o ideal para um objeto de aprendizagem é que o mesmo seja constituído de um bloco único para que possa ser facilmente veiculado à Internet sem a dependência de programas externos que o descompacte.

Para o desenvolvimento de aplicações em Flash, é necessário um *software* proprietário, geralmente com um custo alto, além de necessitar de *plugins* também proprietários, para que a aplicação em Flash seja

executada. O HTML5, por ser um padrão aberto, não necessita de *softwares* proprietários para seu desenvolvimento, o que resulta em vantagem em relação ao Flash. O Plugin Flash Player, atualmente, encontra-se instalado em mais de 98% dos computadores ligados à Internet (MILWARD BROWN, 2011), sendo considerada uma plataforma acessível. O HTML5 tem, pelo menos, grande parte de suas atuais especificações implementadas apenas nas versões mais recentes de todos os *browsers* mais utilizados (Internet Explorer, Safari, Google Chrome, Firefox e Opera).

A utilização do Flash 8 possui vantagens de promover um desenvolvimento mais dinâmico, voltado para a construção da interface gráfica do aplicativo, sendo fácil manipular elementos em uma interface amigável e completa.

Quanto ao HTML5, ainda existem poucos *softwares* especializados em trabalhar com a interface gráfica de aplicações. Porém, a cada dia vêm surgindo novos *softwares* e aplicações pagos e gratuitos que visam tornar o desenvolvimento gráfico de aplicações em HTML5 muito mais dinâmico, assim como no Flash.

Um exemplo de *software* pago que serve de ferramenta no desenvolvimento dessas interfaces gráficas é o conjunto Adobe Edge, que são *softwares* que auxiliam o desenvolvedor Web e que vão desde a criação de animações até a manipulação de fontes a serem utilizadas na aplicação.

No Flash 8, foi possível desenvolver animações em uma interface gráfica com ferramentas específicas para esse fim, de modo que o desenvolvedor precisou se preocupar pouco com o código ActionScript relacionado, enquanto nas ferramentas utilizadas para o desenvolvimento em HTML5, o desenvolvedor precisou descrever toda a animação. Utilizou-se JavaScript sobre o elemento *canvas* ou a função .animate() da biblioteca JavaScript jQuery diretamente sobre elementos da página. Desse modo, inicialmente, foram encontradas dificuldades em lidar com essas animações no HTML5. Porém, com pesquisas às documentações disponibilizadas pela comunidade de desenvolvimento, pôde-se, então, concluir as animações.

É da natureza do Macromedia Flash que o código seja dedicado às funcionalidades dos elementos da aplicação, deixando que o estilo e *layout* sejam definidos em uma interface gráfica rica. Desse modo, é natural que o HTML5 possua mais *scripts* uma vez que toda a interface é definida a partir dos códigos HTML, JavaScript/jQuery e CSS.

Ambos possuem rica documentação acessível por toda a Internet. O Flash, devido a sua larga utilização e tempo no mercado. Já o HTML5, por conta do engajamento da comunidade por todo o mundo e larga utilização, seja em diversos tipos de aplicações como para diversos meios, além de sua natureza como código livre, o que atrai desenvolvedores e, consequentemente, documentação. As bibliotecas do HTML5, tornam fácil a inclusão de novos recursos, aumentando as possibilidades nas ações de

desenvolvimento, principalmente para aqueles que ainda não possuem um extenso conhecimento da linguagem JavaScript.

O Macromedia Flash possui integração com um ambiente de criação gráfica vetorial e aceita muito bem vetores como recurso gráfico. Aproveitando-se disso, praticamente todos os gráficos do objeto foram construídos em forma de vetores, o que garantiu que a aplicação em Flash pesasse apenas 86KB, enquanto a mesma aplicação em HTML5, 863KB, uma vez que todos seus recursos gráficos eram do tipo bitmap.

O HTML5 possui total compatibilidade com os padrões de acessibilidade desenvolvidos pela W3C, de modo a, inclusive, incorporar padrões semânticos, o que já é um recurso pró-acessibilidade. Além disso, os próprios padrões Web preveem um desenvolvimento que leve em conta as mais variadas limitações possíveis aos seus usuários. O Flash também possui recursos pró-acessibilidade. Consiste em atribuir um texto alternativo aos elementos de cada tela, porém não inclui recursos semânticos ou outros padrões de acessibilidade.

Após a recusa da Apple em incluir suporte ao Flash em seus dispositivos que viriam guiar a indústria dos dispositivos móveis, o Flash teve sua participação cada vez menor nesse segmento, sendo substituído, no desenvolvimento de conteúdos multimídia para a Web, pelo HTML5. Apesar disso, ainda era possível utilizar versões antigas do Flash em dispositivos Android que ocupavam, e ocupam até hoje, um grande lugar no mercado. No final de 2011, a Adobe anunciou a descontinuação do suporte e desenvolvimento do Flash para dispositivos móveis deixando que o HTML5 tomasse seu lugar. Portanto, ao desenvolver em HTML5, o OA pode ser disponibilizando para uma gama bem maior de usuários já que agora, para o usuário, há a possibilidade de acessar o OA de um dispositivo móvel com menos limitações do que no Flash.

6. CONSIDERAÇÕES FINAIS

Como descrito anteriormente, o NTEAD/NIE tem, entre seus projetos de pesquisa, o desenvolvimento de objetos de aprendizagem em Flash. Esses OA sempre foram programados em Flash, pois se buscava uma ferramenta que pudesse oferecer conteúdos com animações e interações com o usuário. Depois de desenvolvidos, foram disponibilizados em um repositório do Ambiente Moodle e instalados em um laboratório da Instituição, sendo acessado por meio de um *plugin.*

Embora não fossem programados para serem usados via dispositivos móveis, podiam ser acessados por aqueles, cujo sistema operacional fosse inferior ao Android 4.0.

Um dos fatores que levou a equipe a rever o desenvolvimento

em Flash foi a opção da Adobe em descontinuar seu suporte ao mesmo em dispositivos móveis. Nesse sentido, a equipe, diante do avanço das tecnologias, percebeu a inviabilidade do acesso pelos alunos a essa tecnologia por meio de dispositivos móveis, na sala de aula.

Assim, a decisão de conceber um OA em HTML5 decorreu dos avanços tanto das tecnologias como dessa linguagem de programação.

No quadro comparativo, buscou-se analisar as vantagens e desvantagens entre um e outro, para que, no futuro, tome-se uma decisão quanto à viabilidade do desenvolvimento dos objetos em uma das duas linguagens. O tempo de implementação não foi levado em conta, tendo em vista ter sido o primeiro OA desenvolvido em HTML5, e, ao longo do processo, foi preciso aprofundar estudos para avançar na implementação. Esse objeto será oferecido a professores do Ensino Médio para ser aplicado na sala de aula.

Um novo objeto está em fase de finalização para o estudo de determinantes, tendo demandado um tempo menor desde a concepção até a validação, diante dos avanços no estudo em HTML5. Continua-se a avaliar as vantagens e desvantagens do desenvolvimento de OA em HTML5 e aprofundar estudos quanto a sua viabilidade.

REFERÊNCIAS

ABDULMOTALEB, E. S. et al. Metadata for smart multimedia learning objects. In: AUSTRALASIAN COMPUTING EDUCATION CONFERENCE, 40., 2000, Melbore. *Proceedings...* Melbore, Australia, ACM-CSE, dec. 2000.

ÁVILA, B. G. et al. Construção de Objetos de Aprendizagem a partir de um software de Geometria Dinâmica: uma proposta de capacitação para professores de Matemática. *Revista Novas Tecnologias na Educação (RENOTE)*, Porto Alegre, v.11, n.3, p. 1-10, dez. 2013.

FALKEMBACH, G. A. M. Concepção e Desenvolvimento de Material Educativo Digital. *Revista Novas Tecnologias na Educação (RENOTE)*, Porto Alegre, v.3, n.1, p. 1-10, maio. 2005.

GARCIA, M. F.; RABELO, D. F.; SILVA, D.; AMARAL, S.F. Novas competências docentes frente às tecnologias digitais interativas. *Revista Teoria e Prática da Educação*, v. 14, n. 1, p. 79-87, jan-abr., 2011.

GIBBONS, A. S.; NELSON, J.; RICHARDS, R. The nature and origin of instructional objcts. In: WILEY, D. A. (Ed.). *The instructional use of learning objects. Bloomington*. [S.l.: S.n.], 2000.

GLUZ, J. C.; XAVIER, A. AutoEduMat: uma Ferramenta de Apoio a Catalogação de Objetos de Aprendizagem de Matemática do Ensino Médio Compatíveis com o Padrão OBAA. In: LATIN AMERICAN CONFERENCE ON LEARNING OBJECTS, 6., 2011, Montevideo, Uruguai. *Anais...*Montevideo: Universidad de la República, 2011.

GOMES, S. R.; GADELHA, B. F.; MENDONÇA, A. P.; AMORETTI, M. S. M. Objetos de aprendizagem funcionais e as limitações dos metadados atuais. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 16., 2003, Juiz de Fora. *Anais...* Juiz de Fora: Universidade Federal de Juiz de Fora, 2005.

LOPES, A. M. A. *Estratégias de mediação para o ensino de matemática com objetos de aprendizagem acessíveis:* um estudo de caso com alunos com deficiência visual. 2012. 290 f. Tese (Doutorado em Informática na Educação) – Universidade Federal do Rio Grande do Sul, Porto Alegre, 2012.

LOPES, A. M. A.; PASSERINO, L. M.; VICCARI, R. M. Requisitos de acessibilidade: Objeto de aprendizagem para a Educação Especial no estudo de Matemática. In: CONGRESO INTERNACIONAL DE INFORMÁTICA EDUCATIVA, 17., 2012, Santiago de Chile. *Anais...* Chile: Universidad de Chile, 2012.

MACEDO, L. N.; LAUTERT, S. L.; CASTRO-FILHO, J. A. Análise do uso de um OA digital no ensino de álgebra. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 19., 2008, Fortaleza. *Anais*... Fortaleza: Universidade Federal do Ceará, 2008.

PILGRIM, M. HTML5: Up and running. *O`Reilly Media*: Sebastopol, 2010.

POLSANI, P. Use and abuse of reusable learning objects. *Journal of digital information*, Canadá, v.3, n.4. p.1-10, fev. 2003.

TAROUCO, L. M. R.; FABRE, M-C J. M.; TAMUSIUNAS, F. R. Reusabilidade de objetos educacionais. *Revista Novas Tecnologias na Educação (RENOTE),* Porto Alegre, v. 1, n. 1, p. 1-10, fev., 2003.

W3C (World Wide Web Consortium). Sobre o consórcio W3C, 2008. Disponível em: <http://www.w3c.br/sobre/>. Acesso em: 12 dez. 2013.

WILEY, D. A. Connecting learning objects to instructional design theory: A definition, a metaphor, and a taxonomy. In: WILEY, D. A. (Org.). *The instructional use oflLearning objects*: online version. [S.l.: S.n.], 2000. Disponível em: <http://reusability.org/read/chapters/wiley.doc>. Acesso em: 18 jan. 2014.