WebMedia 2009 - XV Simpósio Brasileiro de Sistemas Multimídia e Web, 2009, Fortaleza, CE, pg. 91-98
http://www.archive.org/details/santanche2009_paper_componere_webmedia

Componere

Autoria na Web baseada em Componentes

André Santanchè
UNIFACS, Av Cardeal da Silva, 747, Salvador, BA, Brazil
santanche@unifacs.br

Matheus Mota
UNIFACS, Av Cardeal da Silva, 747, Salvador, BA, Brazil
matheus.motta@gmail.com

Diego Costa
UNIFACS, Av Cardeal da Silva, 747, Salvador, BA, Brazil
diego.costa.unifacs@gmail.com

Nícolas Oliveira
UNIFACS, Av Cardeal da Silva, 747, Salvador, BA, Brazil
nk.oliveira@gmail.com

Christianne Orrico Dalforno
UNIFACS, Av Cardeal da Silva, 747, Salvador, BA, Brazil
profcod@gmail.com

ABSTRACT

O crescimento da largura de banda para acesso à Internet, associado ao aumento na capacidade dos computadores, tem viabilizado a migração de diversas ferramentas, que antes operavam isoladamente no computador do usuário, para o ambiente Web. A plataforma básica deixa de ser um hardware e sistema operacional específicos e passa a ser o navegador Web. Os usuários se beneficiam com o espaço de trabalho colaborativo da rede que dispensa instalação e atualização de software especializado no cliente. Este artigo apresenta uma ferramenta de autoria baseada em componentes chamada Componere que segue esta abordagem que está sendo desenvolvida por nós. O artigo relata desafios enfrentados que envolvem: a produção de um modelo de componentes em JavaScript, o uso de Microformats para a representação de composições em HTML e a criação de uma ferramenta que permite a usuários finais contribuírem na expansão da biblioteca de componentes através de ferramentas que já lhe são familiares.

Categories and Subject Descriptors

H.5.1 [**Multimedia Information Systems**]; H.3.7 [**Digital Libraries**]

Keywords

autoria, casa mágica, dcc, componere

1. INTRODUÇÃO

Em 1999 Roschelle e outros [12] preconizam que no futuro atividades educacionais vão converter-se em software de forma tão simples quanto a escrita de um texto. Ao invés de palavras e imagens, autores criam pela combinação e edição de objetos computacionais de alto nível – os componentes de software educacionais. Essa poderosa metáfora remete a uma ambição antiga, na qual os usuários finais ganham a capacidade de criar muito mais do que conteúdo passivo, através da combinação de componentes de software, que permitem a construção de aplicações interativas.

Durante este mesmo período se desenvolvia um projeto para a construção de um ambiente de autoria de aplicações educacionais, denominado Casa Mágica. O sistema Casa Mágica propicia a colaboração entre professores e alunos na criação de projetos de ensino/aprendizagem [20]. Sua abordagem para a construção de aplicações se baseia na instanciação, customização e combinação de componentes, e seu ambiente foi projetado para ser operado por autores não especialistas em informática.

Nos últimos anos o crescente poder computacional associado ao aumento da largura de banda para acesso à Internet tornaram possível a construção de aplicações complexas que são executadas diretamente no navegador Web. Tais aplicações exploram a universalidade da linguagem JavaScript, disponível na quase totalidade dos navegadores modernos, no intuito de transferir à máquina cliente (navegador) a grande carga de processamento exigida por aplicações interativas. Essa estratégia possibilitou a execução, diretamente a partir da Web, de uma categoria de aplicações que antes estavam disponíveis exclusivamente no ambiente local, tais como, processadores de texto, planilhas eletrônicas, programas de edição de imagens etc. Essa categoria de aplicações não requer instalação de software localmente – e conseqüentemente se atualiza automaticamente – e proporciona um ambiente naturalmente colaborativo pela rede. Neste artigo tal categoria de aplicações será referenciada como "*100% Web*".

Nos últimos anos o sistema Casa Mágica sofreu um processo de reformulação. O novo ambiente – o qual passamos a denominar *Componere* – passa a atuar em um cenário de autoria mais amplo, não limitado a educação, e tem os seguintes diferenciais em relação ao original: (i) tornar-se um ambiente de autoria 100% Web; (ii) possibilitar a inserção de

produtos da autoria diretamente em páginas Web na forma embedded; (iii) aumentar a inserção e contribuição do autor (usuário final) que passa a ser capaz também de contribuir na expansão da biblioteca de componentes.

Como será detalhado adiante, cada um destes diferenciais constitui individualmente uma contribuição deste artigo. Este artigo está organizado da seguinte maneira: na Seção 2 apresentamos trabalhos anteriores do grupo, que forneceram as bases para o Componere; na Seção 3 apresentamos trabalhos relacionados; na Seção 4 apresentamos o ambiente de autoria Componere e seus diferenciais; na Seção 5 apresentamos as considerações finais e trabalhos futuros.

2. TRABALHOS ANTERIORES

Dado que este artigo apresenta uma evolução de um trabalho anterior – o sistema de autoria Casa Mágica – nas Subseções 2.1 e 2.2 apresentamos trabalhos anteriores que forneceram as bases para a criação do Componere.

2.1 Casa Mágica

Usualmente o termo autoria remete a sistemas de autoria no contexto multimídia, que possibilitam a autores a criação de produtos multimídia combinando artefatos e inserindo-os em uma estrutura narrativa [3]. O modo como sistemas multimídia organizam e apresentam a estrutura subjacente de seus produtos define um paradigma de autoria (e.g., baseado em: estrutura, linha do tempo, grafo ou script) [3]. O sistema Casa Mágica é baseado em uma combinação do paradigma de autoria baseado em grafos com abordagens visuais de modelagem baseada em componentes de software.

Em sua primeira versão, o ambiente dispunha de um catálogo de objetos, os quais o usuário instanciava, configurava e combinava dentro de espaços de trabalho denominados Unidades [14]. Em seguida, o modelo de objetos foi substituído por um modelo de componentes, ampliando as possibilidades de extensão e compartilhamento do sistema. O modelo de componentes no sistema Casa Mágica extrapola o domínio de especialistas de informática de dois modos: tornando-se perceptível ao usuário final – autor de composições no sistema de autoria – e estabelecendo uma ponte de colaboração entre esses autores e profissionais de informática.

De um lado estão os usuários autores, que no nosso contexto são professores que planejam atividades de ensino/aprendizagem e demandam componentes de software para a sua execução. Do outro estão profissionais de informática, que implementam, testam e entregam os componentes aos autores. Novamente entram em cena os professores (autores) que preparam atividades e ambientes compondo componentes; alunos interagem com tais composições experimentado-as, modificando seus componentes e recombinando-os, tornam-se desse modo não apenas usuários, mas co-autores. Esta estratégia amplia o papel dos componentes, que deixam de estar restritos ao domínio de especialistas em informática e se tornam peças reconhecidas por usuários não especialistas, tornando-os capazes de criar sobre elas. Componentes se tornam unidades de expressão através das quais usuários projetam e encomendam software a especialistas; se tornam unidades que podem ser compartilhadas em bibliotecas e reusadas.

Esta abordagem para desenvolvimento foi aplicada na prática no contexto educacional em projetos envolvendo o sistema Casa Mágica. Relatos de experiências bem sucedidas e contribuições do sistema estão descritos em [18, 15, 19]. O projeto que adquiriu maior destaque se desenvolveu no contexto do ensino de matemática, no qual componentes de software representam abstrações de máquinas, utilizadas como metáforas no ensino desta disciplina. Este projeto específico resultou em uma publicação relatando a metodologia aplicada e detalhes da experiência [19].

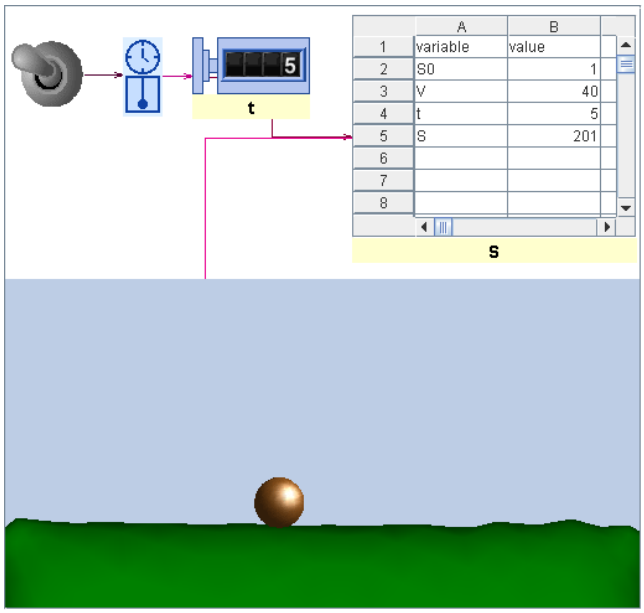

Figure 1: Exemplo de aplicação produzida no sistema Casa Mágica.

A Figura 1 ilustra uma típica aplicação construída no sistema Casa Mágica, cujo propósito é simular uma esfera em movimento retilíneo uniforme (MRU). A aplicação é formada pela composição de seis componentes: interruptor, relógio, contador, planilha, cenário e esfera. Como mostra a figura, os componentes são interligados entre si e cada ligação é uma quádrupla (origem, evento, mensagem, destino), indicando que ao ocorrer um evento na origem, será enviada uma mensagem ao destino. Neste exemplo, ao se acionar o interruptor, ele envia uma mensagem para o relógio requisitando que inicie o trabalho; a cada ciclo do relógio ele envia uma mensagem de incremento para o contador; cada vez que o contador recebe uma mensagem, envia outra mensagem com seu valor atualizado para uma célula específica do componente planilha (célula à direita do rótulo t). O componente planilha mantém uma fórmula para cálculo de MRU e, a cada vez que o valor da célula t é atualizado, recalcula a posição da esfera. A cada atualização a planilha envia uma mensagem para a esfera que atualizará a sua posição.

O sistema Casa Mágica sofreu duas evoluções significativas. Na primeira, passou a trabalhar com padrões abertos da Web e Web Semântica (XML e RDF) na representação de suas composições e descrição de seus componentes [20]. Seu ambiente de execução também foi remodelado, de modo que os componentes passaram a se comunicar com mensagens XML, permitindo a interoperabilidade entre linguagens e

plataformas diferentes. Na segunda, adotou o modelo de componente de conteúdo digital (Digital Content Component – DCC) que será detalhado na subseção a seguir.

2.2 Digital Content Component

O modelo Digital Content Component (DCC) amplia o antigo modelo de componentes do sistema Casa Mágica para uma esfera mais ampla. Os componentes não estão limitados a encapsular código executável, mas podem encapsular qualquer tipo de conteúdo. Este conteúdo pode ser um código executável – DCC de Processo – ou um objeto digital complexo – DCC Passivo. Tal como os objetos digitais complexos, um DCC agrega e encapsula um ou mais artefatos digitais e representa internamente as suas relações; tal como os componentes de software, um DCC esconde detalhes internos da sua representação e expõe uma interface pública.

Através de um mecanismo que denominamos "execução dirigida pelo tipo de conteúdo" um DCC Passivo, encapsulando conteúdo, é associado automaticamente a um DCC de Processo contendo software executável, capaz de manipular seu conteúdo [17]. Por exemplo, a Figura 1 ilustra um DCC Passivo contendo uma planilha. Esse DCC não encapsula o software da planilha propriamente dito. Em tempo de execução, um DCC de Processo é associado dinâmicamente ao DCC Passivo contendo a planilha a fim de processá-la e apresentá-la.

3. TRABALHOS RELACIONADOS

Os trabalhos relacionados estão organizados da seguinte maneira: a Subseção 3.1 apresenta um resumo do cenário de trabalhos relacionados para a construção da ferramenta de autoria Casa Mágica; as Subseções 3.2 e 3.3 apresentam trabalhos relacionados com a evolução da ferramenta.

3.1 Autoria e Componentes Educacionais

Nos meados da década de 90 um promissor campo de pesquisa combinando a tecnologia dos componentes de software com a criação de software educacional recebe o nome de componentes educacionais.

Mesmo sendo considerado por alguns autores um termo pouco preciso [9], os componentes de software são uma tecnologia madura. Existem diferentes interpretações e implementações da noção de componente de software, mas elas sempre se baseiam em um conjunto fundamental de características [1, 2, 21]. Um componente, como indica o próprio termo, é uma unidade feita para compor algo. Componentes publicam a sua funcionalidade através de interfaces, que guiam a maneira como ele se relaciona com as demais entidades do sistema. As interfaces escondem detalhes de implementação, permitindo que o usuário abstraia detalhes internos do componente e se concentre na sua funcionalidade.

Os componentes não são apenas um poderoso modelo de programação, mas representam uma poderosa metáfora que nos liga à forma como percebemos coisas no mundo real. Através da metáfora dos componentes é possível se pensar na composição de software de forma equivalente ao modo como utilizamos componentes do mundo real para montar coisas. Por exemplo, Roschelle et al. [13] dizem que poucas pessoas têm condições de montar o seu próprio aparelho estéreo combinando peças eletrônicas (transistores, resistores etc.), mas muitas conseguem fazê-lo comprando e ligando unidades maiores (tocador CD, caixa de som, amplificador etc.), que nesse caso representam os componentes. A metáfora de componentes do mundo real produz um interessante modelo abstrato computacional, no qual autores não especialistas em computação são capazes de perceber artefatos de software e a forma como combiná-los.

A tecnologia dos componentes de software motivou diversos projetos a lançarem ferramentas nas quais os autores poderiam combinar peças para a construção de aplicações educacionais. O projeto ESCOT - Educational Software Components of Tomorrow - foi um passo além, trabalhando na construção de uma solução para possibilitar interoperabilidade entre componentes educacionais de diversas origens, de modo que eles possam se integrar em aplicações que reúnam vários componentes [12]. Dentre os integrantes do ESCOT estão: o E-Slate, um ambiente para a construção de micromundos por meio da combinação de componentes de software; o SimCalc e o MathWorlds, ambientes voltados ao domínio de matemática, em que o autor pode configurar um cenário onde são dispostos atores cujo movimento está descrito por funções matemáticas e pode ser acompanhado em gráficos cartesianos; o AgentSheets, sistema destinado à construção de simulações, baseado em uma metáfora de agentes autônomos e animados, que se comportam conforme algoritmos independentes de decisão e ação. Cada um dos integrantes adaptou sua solução para o modelo de componentes do ESCOT. Desse modo, se criaram projetos em que componentes dos diferentes universos são combinados para, por exemplo, visualizar em um gráfico do MathWorlds os resultados de uma simulação no AgentSheets [12].

O movimento na direção de tornar o usuário final autor, possibilitando a criação de ambientes de autoria para o desenvolvimento de atividades educacionais, vai além da composição de peças. Do mesmo modo que peças de software podem assumir a forma de componentes, ações e variáveis também podem ser representadas como blocos manipuláveis que combinados resultam em programas. Esta abordagem remete ao ambiente Boxer [6], que utiliza a metáfora de caixas em que são encapsulados os blocos. A estratégia adotada pelo AgentSheets [11] e mais recentemente pelo ambiente de programação Scratch [8] possibilita que usuários não especialistas em computação elaborem seus próprios programas, o que os torna ambientes aptos ao desenvolvimento de atividades de ensino-aprendizagem.

Ainda que alguns desses ambientes proporcionem plug-ins para a execução dos resultados finais no ambiente Web, até onde sabemos, nenhum ambiente deste tipo oferece uma ferramenta de edição diretamente na Web e mais especificamente 100% Web, tal como o Componere.

3.2 Microformats

O Microformats[7] é uma estratégia que permite atribuir uma maior semântica a documentos HTML, através de padrões de anotação, com o uso de tags HTML já existentes. Dessa forma, fica mais fácil a interpretação destes documentos por máquinas e humanos.

A Figura 2, por exemplo, apresenta um trecho de código que descreve uma organização através dos Microformats hCard,

fazendo uso apenas de tags nativas do próprio HTML para anotar o conteúdo. Este tipo de descrição é possivel pois cada Microformat possui um fim específico e bem definido [7].

```
<div>
<div>Bio-Core</div>
<div>UNICAMP</div>
<a href="http://www.ic.unicamp.br">
http://www.ic.unicamp.br/</a>
</div>
```

```
<div class="vcard">
<div class="fn">Bio-Core</div>
<div class="org">UNICAMP</div>
<a class="url" href="http://www.ic.unicamp.br">
http://www.ic.unicamp.br/</a>
</div>
```

Figure 2: Exemplo de uso do Microformat

3.3 Extração de Informação

Extração de Informação ou Information Extraction (IE) é definida como um processo que, a partir de uma entrada de texto livre ou de conteúdo semi-estruturado, filtra e seleciona dados relevantes e gera resultados estruturados e relacionados ao conteúdo [4]. A extração de informação pode partir de um texto que tenha sido manualmente anotado pelo autor ou não. Estes sistemas podem também identificar automaticamente anotações utilizando técnicas de análise de padrões ou inferindo semântica a partir de ontologias (anotações automáticas e semiautomáticas) [5]. Em suma, IE produz dados estruturados prontos para pós-processamento, o que é fundamental para aplicações de mineração de dados, ferramentas de pesquisa, recuperação de dados específicos etc.

4. COMPONERE

Esta seção apresenta transformações realizadas no sistema de autoria Casa Mágica para convertê-lo no novo sistema Componere que opera 100% Web. O cenário geral de trabalho do Componere é ilustrado na Figura 3.

Como está ilustrado na figura o processo de autoria no ambiente Componere é dividido em três momentos: (i) produção de componentes; (ii) autoria e (iii) apresentação. Cada um destes momentos inclui uma contribuição apresentada neste artigo em uma das subseções a seguir respectivamente. Na subseção 4.1 será apresentado um processo desenvolvido por nós que permite que autores possam contribuir na expansão da biblioteca de componentes usando ferramentas familiares. Na subseção 4.2 será apresentado como o modelo de componentes do sistema de autoria está sendo transformado em uma versão JavaScript, permitindo a construção de uma ambiente de autoria no navegador. Na subseção 4.3 será apresentada como a representação das composições foi adaptada, de modo que possa ser embutida em páginas HTML usando a tecnologia dos Microformats. A nova plataforma Componere ainda se encontra em processo de construção, entretanto, cada um de seus novos diferenciais, apresentados a seguir, representa isoladamente uma contribuição independente.

4.1 Criação de Componentes centrada em Documentos

A principal estratégia de uma ferramenta de autoria, como o Componere, é a combinação de recursos pré-existentes. O sistema Componere pode combinar uma ampla variedade de componentes, considerando-se que estejam encapsulados em DCCs. Como conseqüência, surge a questão crucial: como ampliar a biblioteca de recursos básicos que podem tomar diversas formas (e.g., animações, planilhas, artefatos multimídia etc.). Nesse sentido foi criada uma estratégia para ampliar a inserção do usuário final no incremento da biblioteca. A esta estratégia denominamos: criação de componentes centrada em documentos.

Nessa estratégia, ao invés de se criar uma ferramenta específica para cada tipo de conteúdo, optou-se pela utilização de ferramentas de produtividade (processador de texto, planilha etc.) às quais o usuário está familiarizado. Ferramentas de produtividade apresentam um ambiente amigável e robusto, o que possibilita a produção de documentos ricos em conteúdo de maneira rápida e simples. Através de uma metodologia desenvolvida por nós chamada Semântica In Loco, o autor anota o conteúdo na medida em que o produz e é possível extrair DCCs Passivos diretamente do conteúdo.

Semântica In Loco

A Semântica In Loco é uma estratégia para a produção de anotações associadas a recursos digitais, cuja semântica segue padrões abertos da Web [16]. Do ponto de vista do processo de anotação de conteúdo digital, a grande dificuldade é justamente permitir que, ao mesmo tempo em que o conteúdo seja produzido, este possa ser anotado.

Normalmente o processo de anotação requer uma ferramenta diferente daquela utilizada para a produção de conteúdo. As metáforas da ferramenta de anotação são usualmente diferentes das que o autor está habituado a usar nas ferramentas de produção. Desta forma o processo de anotação acaba sendo oneroso e exige o aprendizado de diferentes ferramentas e metáforas.

No intuito de resolver esta separação, a Semântica In Loco propõe que o mesmo ambiente de produção utilizado pelo autor seja utilizado para inserir anotações ao conteúdo, seguindo as mesmas metáforas utilizadas na produção. No caso de processadores de texto, por exemplo, o autor faz uso da própria solução de estilos oferecidos pelo ambiente para formatar o conteúdo (e.g., Título, Sub Título etc.), inserindo anotações cuja semântica direciona uma futura extração.

Esta metodologia segue três princípios. O primeiro estabelece que as anotações sejam feitas paralelamente à produção, logo, o usuário faz uso de uma ferramenta de produção já conhecida para anotar o conteúdo. Cada ambiente de produção de conteúdo possui modelos e metáforas próprias. Fazer uso destas para anotar o conteúdo que está sendo produzido é o segundo princípio, o de integração de metáforas. Finalmente, a interoperabilidade é a união dos dois outros princípios, permitindo que uma ferramenta de extração – tal como o DDEx apresentado a seguir – converta o conteúdo e as anotações para um padrão aberto, interoperável, extensível e escalável.

No que se refere à implantação da Semântica In Loco, primeiro é necessário selecionar os metadados utilizados na anotação, em seguida deve-se definir um padrão de anotação que possa ser identificado automaticamente por uma ferramenta de extração e, finalmente, desenvolver uma ferramenta de extração e conversão. Mais detalhes sobre a Semântica In Loco podem ser encontrados em [16].

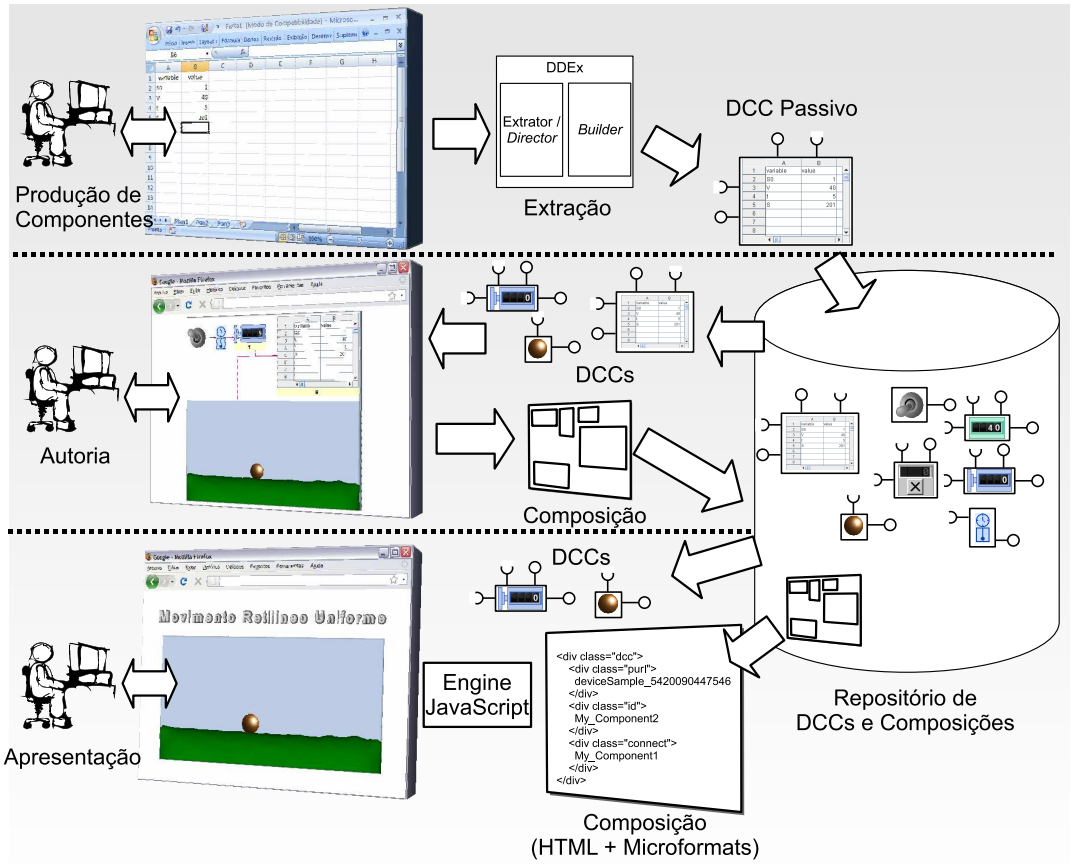

Figure 3: Diagrama geral de funcionamento do ambiente Componere.

DDEx

A maioria dos documentos produzidos estão em formato proprietário, o que dificulta a reusabilidade e interoperabilidade com outros contextos. A fim de permitir o reuso e interoperabilidade das informações presentes nos documentos, produzidos por programas de produtividade, foi construído o DDEx.

O DDEx – Document Data Extractor – é um framework open-source e autônomo que busca oferecer suporte aos mais conhecidos formatos de documentos. O DDEx faz uso de APIs para manipular os documentos e permitir que a extração e manipulação ocorram de modo transparente de maneira que, independente do formato de arquivo, possa se gerar o mesmo resultado baseado num template preenchido com conteúdo. Este template é um documento que foi previamente construído e está preparado para ser utilizado pelo usuário, de modo que as metáforas do ambiente sejam utilizadas pelo autor para anotar o conteúdo e, conseqüentemente, oferecer diretivas que facilitem uma posterior extração pelo framework.

Neste trabalho o DDEx foi utilizado para a extração de conteúdo anotado em documentos e para o seu encapsulamento em DCCs Passivos. Dentro do processo geral para criação DCCs, um sistema faz uso do DDEx para extrair e reconhecer, baseado em padrões de anotação, os dados dos documentos. Uma vez gerado um DCC correspondente ao documento, os dados passam a ser reutilizáveis e interoperáveis com outros sistemas computacionais. O diagrama da Figura 4, além de sintetizar o funcionamento do DDEx, identifica o papel do mesmo dentro do contexto geral do processo de geração de DCCs Passivos. Ela corresponde ao primeiro nível identificado como Produção de Componentes na Figura 3.

Inicialmente, na metodologia Semântica In Loco, é feita a seleção dos metadados e em seguida a definição de um padrão de anotação. Uma vez selecionados os metadados e definido o padrão de anotação, o resultado será um template que será utilizado pelo usuário para produzir o documento. Considere o exemplo em que se deseja produzir uma animação sobrepondo-se várias imagens das etapas de crescimento de uma muda de tangerina (Figura 4). A partir de um template o usuário utiliza um processador de texto para inserir as imagens que compõem a animação, bem como rótulos e outros dados a ela associados. Na medida em que insere os dados, utiliza metáforas do próprio ambiente de produção de conteúdo (estilos para editores de texto, rótulos de células para planilhas eletrônicas etc.) para anotar e associar mais semântica ao conteúdo, permitindo que este seja extraído, identificado e convertido. Na figura o texto contendo as imagens é convertido em um DCC Passivo.

O diferencial deste modelo é justamente a possibilidade do

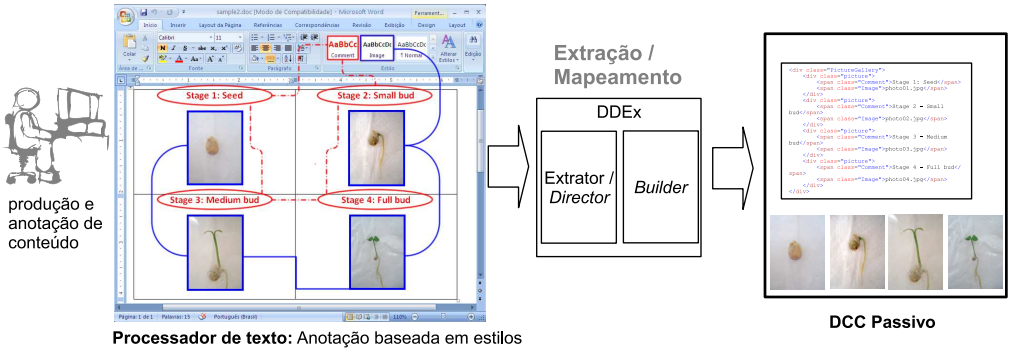

Figure 4: Etapas de produção de um DCC a partir de um documento anotado.

usuário utilizar um ambiente já conhecido para produzir um documento que posteriormente será um componente de conteúdo.

4.2 Componentes na Web

Um desafio para alcançar um solução 100% Web é a adoção da linguagem JavaScript para a implementação do ambiente de autoria e conseqüentemente do modelo de componentes. Desse modo, o modelo DCC foi convertido para a linguagem JavaScript, de modo que configurações com DCCs feitas no ambiente Componere pudessem ser executadas diretamente no navegador, sem a necessidade de software extra.

Para isto contribuiu o configuração da infra-estrutura dos DCCs, conforme está apresentado na Figura 5. Concebida inicialmente em Java, a infra-estrutura dos DCCs é divida em três contextos. No primeiro contexto, componentes têm que ser escritos na mesma linguagem de programação e se ligam por conectores via código. No segundo contexto os componentes se comunicam através de um barramento de mensagens, ainda no contexto local. Essa estratégia dá mais flexibilidade que o contexto 1 e simplifica a criação de conexões dinâmicas em tempo de execução. O contexto 3 permite interoperabilidade entre diferentes plataformas e linguagens. Para isso existe um barramento distribuído que opera sobre padrões abertos da Web. Os barramentos do contexto 2 possuem um componente especial chamado Proxy capaz de capturar mensagens endereçadas para outra plataforma, convertê-la e colocá-la no barramento distribuído. Essa mensagem será capturada por um componente Proxy da plataforma destino que converte a mensagem e a coloca no barramento local endereçada ao componente destino.

A arquitetura apresentada permite que o modelo DCC aplicado a cada linguagem de programação funcione como uma especificação lightweight, ou seja, comparada aos padrões de componentes COM [22] e XPCOM [10] o modelo DCC não faz exigências de padronização binária de comunicação entre componentes de diferentes linguagens na esfera individual. Ao contrário, a responsabilidade pela interoperabilidade é transferida para o componente Proxy, simplificando a implementação de cada DCC individualmente e permitindo que o modelo DCC possa aproveitar especificidades de cada linguagem de programação em que ele é aplicado. Seguindo esta estratégia, a segunda contribuição apresentada neste artigo foi a conversão da infra-estrutura de DCCs para operar em JavaScript.

Até onde sabemos o único modelo de componentes amplamente adotado no contexto JavaScript é o XPCOM [10], entretanto, seu framework é fortemente dependente de plataformas específicas de execução, que no caso dos navegadores Web são aqueles produzidos pela Mozilla. Por esse motivo, independente da plataforma de autoria, nosso modelo de componentes (DCC) em JavaScript é uma infra-estrutura que pode ser usada para o desenvolvimento de aplicações baseadas em componentes.

A implantação dos DCCs em JavaScript envolveu vários desafios, dado que a implementação de referência dos DCCs adota diversas primitivas disponíveis em linguagens orientadas a objetos não disponíveis em linguagem JavaScript, tais como interfaces, sobrecarga de métodos e processo de anotações de código.

Os componentes JavaScript permitirão criar o segundo e terceiro níveis apresentados na Figura 3 (identificados como Autoria e Apresentação).

4.3 Composições na Web

O terceiro desafio para a criação de um ambiente de autoria 100% Web foi a possibilidade de se inserir tanto componentes individualmente, quanto composições inteiras, diretamente dentro de páginas HTML na forma *embedded*, sem que o usuário necessite de um plug-in especial. Desse modo, foi adotada a estratégia dos Microformats [7] para se inserir descrições de instâncias de componentes ou composições inteiras diretamente dentro de páginas Web, sem se faça necessária qualquer modificação no padrão HTML.

Os DCCs de processo visuais em JavaScript são autônomos, de modo que cada componente instanciado tem a capacidade de desenhar a si próprio no local especificado pelo usuário. DCCs Passivos podem ser apresentados de forma visual em um navegador Web através de sua associação com um DCC de Processo desenvolvido em JavaScript que é capaz de ler e apresentar seu conteúdo.

No contexto desta proposta, os Microformats são usados de forma que eles possam identificar instâncias dos DCCs de Processo e Passivos dentro de páginas HTML, bem como as suas conexões. A inserção e utilização dos DCC em páginas Web se dá de forma simples. O autor, um desenvolvedor Web, por exemplo, especifica a inserção do DCC no local desejado através de marcações Microformats no documento

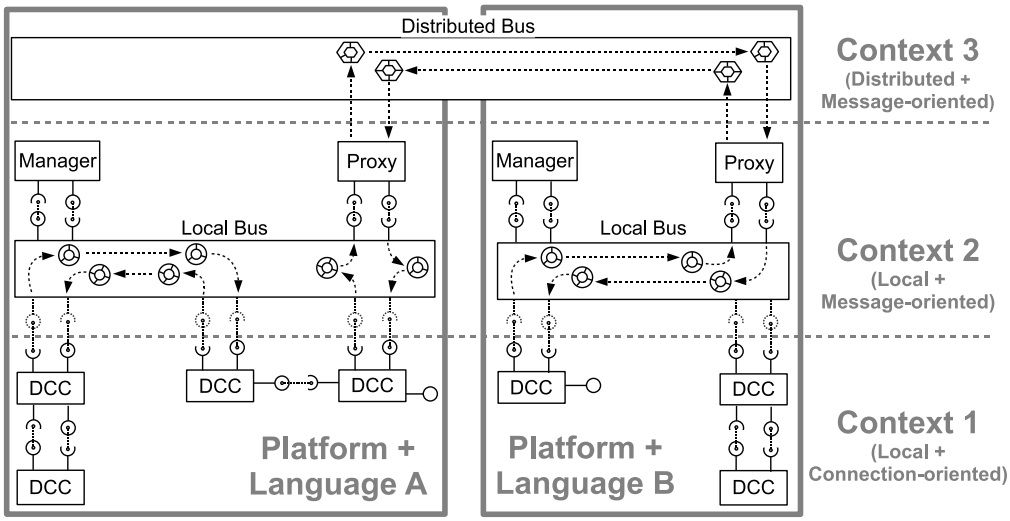

Figure 5: Infra-estrutura de runtime dos DCCs.

HTML. Nessa marcação o usuário informará, dentre outras coisas, o identificador único do DCC Passivo desejado. Uma composição pode também ser gerada automaticamente em HTML+Microformats a partir de uma representação em XML, tal como está apresentado na Figura 3.

A Figura 6 ilustra o processo de transformação das marcações Microformats em código JavaScript que instancia e conecta os DCCs. Esse processo ocorre completamente no cliente. Para o momento da apresentação, foi implementado um mecanismo em JavaScript que percorre todo o código HTML em busca dessas anotações Microformats (1) e (2). Para cada marcação, o mecanismo verifica, através do identificador único informado, a existência desses DCCs em um repositório de componentes, caso exista, é recuperada uma instância do componente solicitado. Caso seja um DCC Passivo, ele será automaticamente associado a uma instância de um DCC de Processo em JavaScript que é responsável por sua apresentação (3). Logo depois o mecanismo, faz a préconfiguração junto ao DCC de Processo, de acordo com as características descritas no HTML. Por fim um método inicia a execução da composição que se apresenta visualmente no navegador (4).

5. CONCLUSÃO

O sistema Casa Mágica é resultado de anos de pesquisa em autoria no contexto educacional. O desafio agora é como aproveitar as vantagens oferecidas pelo ambiente Web para a criação de um ambiente de autoria 100% Web.

Este trabalho apresenta a migração do software de autoria Casa Mágica para acompanhar esta nova tendência. Para a nova versão se adotou o nome *Componere*, que abrange um universo mais amplo, não restrito ao domínio de educação. Este novo ambiente e seus componentes, por serem executados diretamente no navegador, diminuem complexidade e resistência à sua adoção, uma vez que não exigem instalação de softwares ou plug-ins no cliente.

O ambiente ainda não está completamente implementado. A ferramenta de autoria completamente em JavaScript ainda se encontra em fase de produção. Entretanto, este artigo apresentou três pesquisas que estão completamente implementadas e operacionais, que fazem parte do processo de migração do sistema Casa Mágica para a Web. Cada uma delas constitui individualmente uma contribuição.

O *Componere*, fazendo uso da ferramenta DDEx, facilita a criação de componentes e objetos complexos de conteúdo, uma vez que permite a utilização de ferramentas de produtividade como editores de texto, por exemplo, para este fim. Desta forma sistemas que já fazem parte das atividades diárias de muitas pessoas podem ser utilizados dispensando o esforço inicial de aprendizagem de uma nova ferramenta. Assim, os autores poderão alimentar a biblioteca de componentes, não ficando mais restritos ao uso de componentes preexistentes ou à colaboração de desenvolvedores de software para realizar as atividades educacionais por eles elaboradas.

Uma vez inseridos os componentes no repositório, estes podem ser resgatados e utilizados em páginas Web através da utilização de anotações Microformats e componentes JavaScript. Os componentes em JavaScript também serão a base no ambiente de autoria 100% na Web.

Os trabalhos futuros incluem a conclusão do ambiente do módulo de autoria, sua disponibilização como um serviço na Web. Em versões futuras, essa ferramenta oferecerá suporte à criação colaborativa. Outro trabalho futuro consiste na evolução das possibilidades de inserção de componentes em material Web, de modo a se construir materiais mistos que envolvam conteúdo Web e composições.

6. AGRADECIMENTOS

Este trabalho foi parcialmente financiado pelos projetos: BioCORE (CNPq), MediaBank (HP Digital Publishing), WebMaps II (CNPq) e LabMultiflex (FAPESB).

7. REFERENCES

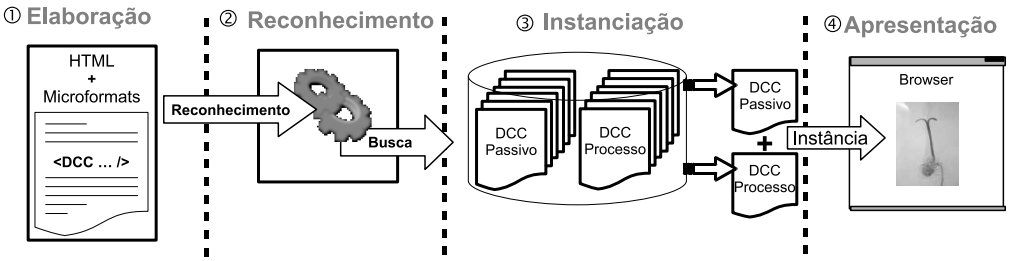

Figure 6: Etapas do processo de apresentação do componente

[1] F. Bachmann, L. Bass, C. Buhman, S. Comella-Dorda, F. Long, J. Robert, and R. S. K. Wallnau. Volume II: Technical Concepts of Component-Based Software Engineering, 2nd Edition. Technical Report CMU/SEI-2000-TR-008, Carnegie Mellon University, July 2000.

[2] M. Broy, A. Deimel, J. Henn, K. Koskimies, F. Plášil, G. Pomberger, W. Pree, M. Stal, and C. Szyperski. What characterizes a (software) component? *Software – Concepts & Tools*, 19(1):49–56, June 1998.

[3] D. Bulterman and L. Hardman. Structured multimedia authoring. *ACM Transactions Multimedia Computing, Communications and Applications*, 1(1):89–109, 2005.

[4] C.-H. Chang, M. Kayed, M. R. Girgis, and K. F. Shaalan. A survey of web information extraction systems. *IEEE Transactions on Knowledge and Data Engineering*, 18(10):1411–1428, 2006.

[5] J. Davies, R. Studer, and P. Warren. *Semantic Web Technologies: Trends and Research in Ontology-based Systems*. Wiley, 2006.

[6] A. A. diSessa. *Changing Minds: Computers, Learning, and Literacy*. MIT Press, Cambridge, MA, 2000.

[7] R. Khare. Microformats: The next (small) thing on the semantic web? *IEEE Internet Computing*, 10(1):68–75, 2006.

[8] J. Maloney, L. Burd, Y. Kafai, N. Rusk, B. Silverman, and M. Resnick. Scratch: A sneak preview. In *C5 '04: Proceedings of the Second International Conference on Creating, Connecting and Collaborating through Computing*, pages 104–109, Washington, DC, USA, 2004. IEEE Computer Society.

[9] G. Olsen. From com to common. *Queue*, 4(5):20–26, 2006.

[10] R. Parrish. XPCOM Part 1: An introduction to XPCOM, February 2001. `http://www-128.ibm.com/developerworks/webservices/library/co-xpcom.html`, accessed on 12/2004.

[11] A. Repenning and J. Ambach. Tactile programming: A unified manipulation paradigm supporting program comprehension, composition and sharing. In *Proc. of the IEEE Symposium of Visual Languages*, pages 102–109, Boulder, CO, 1996.

[12] J. Roschelle, C. DiGiano, M. Koutlis, A. Repenning, J. Phillips, N. Jackiw, and D. Suthers. Developing educational software components. *Computer*, 32(9):50–58, 1999.

[13] J. Roschelle, C. Digiano, R. Pea, and J. Kaput. Educational Software Components of Tomorrow (ESCOT), 1998. `http://www.escot.org/docs/MSET_ESCOT.html`, accessed on 10/2007.

[14] A. Santanchè. Sistema para construção de aplicações educacionais. In *Actas do IV Congresso Ibero-Americano de Informática na Educação – RIBIE98*, 1998.

[15] A. Santanchè. Explorando metáforas na elaboração de composições hipermídia. In *Anais do XXII Congresso da Sociedade Brasileira de Computa cão – VIII Workshop de Informática na Escola*, pages 389–396, 2002.

[16] A. Santanchè. Otimizando a anotação de objetos de aprendizagem através da semântica in loco. In *Anais do XVIII Simp. Brasileiro de Informática na Educação*, pages 526–535, 2007.

[17] A. Santanchè, C. B. Medeiros, and G. Z. Pastorello Jr. User-author centered multimedia building blocks. *Multimedia Systems*, 12(4):403–421, March 2007.

[18] A. Santanchè and C. A. C. Teixeira. Integrando instrucionismo e construcionismo em aplicações educacionais através do Casa Mágica. In *Anais do XIX Congresso Nacional da SBC – V Workshop de Informática na Educação – WIE 1999*, pages 805–817, 1999.

[19] A. Santanchè and C. A. C. Teixeira. Máquinas e componentes de software na aprendizagem matemática. In *Anais do XXII Congresso da Sociedade Brasileira de Computação – VIII Workshop de Informática na Escola*, pages 293–301, 2002.

[20] A. Santanchè and C. A. C. Teixeira. Representando e comunicando objetos hipermídia heterogêneos em documentos Web. In *Proc. of 8th Brazilian Symposium on Multimedia and Hypermedia Systems*, pages 116–131, 2002.

[21] C. Szyperski. *Component Software: Beyond Object-Oriented Programming*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2002.

[22] S. Willliams and C. Kindel. The Component Object Model: A Technical Overview, October 1994. `http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dncomg/html/msdn_comppr.asp`, accessed on 11/2004.