Dedalus - Educação a Distância e Inclusão Digital através de Aplicações Auto-hospedadas

Dalila Paiva, Danielle Parga, Carlo Tola Oliveira

Núcleo de Computação Eletrônica - Universidade Federal do Rio de Janeiro (UFRJ)
CCMN Bloco C - Caixa Postal 2324 - Rio de Janeiro - RJ - Brazil

dalilamp@posgrad.nce.ufrj.br, {parga,carlo}@nce.ufrj.br

Resumo. *A internet vem se destacando como mídia de comunicação. Entretanto, a tecnologia cliente-servidor centraliza a publicação de conteúdo nas mãos de poucos, impedindo os usuários comuns de exporem livremente suas idéias. O Dedalus é uma plataforma federada que propõe uma alternativa a esse modelo. Ele consiste de uma rede de sites gráficos onde um usuário pode reunir e veicular suas informações com pouco conhecimento tecnológico. A própria natureza do modelo federado proporciona um ambiente perfeito para a publicação de sites. Uma interface drag-and-drop básica permite uma imediata customização, além de apoio a expansões mais avançadas utilizando a linguagem Java. O dedalus é um passo à frente no sentido da democratização da internet, amenizando a exclusão digital.*

Palavras-chave: (Computação Federada, Inclusão Digital, Educação a Distância)

Abstract. *Internet has been playing a major role as communication media. However, client-server technology centralizes publishing on major players, excluding the average user from exposing his ideas. Dedalus is a peer-to-peer platform that proposes an alternative to this model. It is a web of graphical sites where a user can assemble his information with little knowledge of computer science. The peer-to-peer nature precludes the need of arrangements to obtain a publishing site. A basic drag-and-drop interface allows immediate site customization, but there is support for more advanced extensions with Java applications. Dedalus is a step towards Internet democratization and relief from the digital exclusion.*

Key words: (Peer-to-Peer, e-learning, digital inclusion)

Dedalus - Uma Plataforma Java para Aplicações Ponto-a-Ponto

1 Introdução

A sociedade moderna se vê em meio a uma grande revolução tecnológica. Os computadores pessoais e principalmente a rede mundial de computadores (a internet) modificaram a vida das pessoas. A maior parte das tarefas sociais está envolvida com estas tecnologias, obrigando cada indivíduo buscar por conhecimento tecnológico para obter uma maior integração social. Mas, ironicamente, a mesma tecnologia que facilita a vida das pessoas se encontra em um estágio de inércia no que diz respeito a ajudar com que mais pessoas façam parte deste mundo digital.

Poucas pessoas possuem conhecimentos e recursos necessários à produção e publicação de conteúdo neste emergente meio de comunicação que é a internet. Grande parte dos usuários precisa se contentar em apenas receber informações. Enquanto que a capacidade de desenvolver um papel ativo na internet, fica restrita a poucos. A comunidade ainda é carente quanto aos recursos e conhecimentos tecnológicos necessários à concretização da publicação de idéias na rede, constituindo uma barreira para uma participação mais ativa e livre das pessoas na internet.

O caráter comercial que impera na internet requer toda informação veiculada seja sujeitada a um aporte monetário. Mesmo quando não é cobrado diretamente do detentor da informação, o modelo de "gratuidade" se remuneram fazendo o uso de informação sem repassar os direitos autorais devidos. Além de não se responsabilizarem pela disponibilidade permanente destas informações. Como não existe nenhum compromisso por parte dos provedores, muitas vezes acontece deles simplesmente retirarem os sites do ar sem nem mesmo consultar ou avisar seus proprietários.

Uma tecnologia que vem se destacando no mundo da informática em resposta a isto é o modelo federado de comunicação entre nós da internet. Nele, todos os pontos conectados à rede se comunicam diretamente para obter e oferecer recursos. Cada recurso pode ser distribuído por vários pontos, aumentando a probabilidade de sucesso na busca por uma informação. Assim, o sistema é todo mantido por computadores pessoais descartando os sites comercias.

O sistema Dedalus oferece toda uma infra-estrutura para a criação de uma rede federada de sites gráficos onde os usuários podem expor suas informações com pouco conhecimento tecnológico. Com isto, visa conseguir uma maior democratização do uso da internet nos mais diversos grupos de interesse. O esforço investigativo e criativo das pessoas é canalizado em benefício da criação de novas aplicações mais complexas e específicas. Sua principal idéia é proporcionar a capacidade de cada ponto na rede poder ser seu próprio "lugar" na internet. Será o lugar onde as pessoas poderão trocar idéias e recursos entre si.

2 Revisão Literária

O Paper AirPlane foi a única aplicação encontrada com os mesmos objetivos do Dedalus: permitir a auto-publicação de conteúdos através de redes peer-to-peer. O Paper Airplane é uma extensão para o navegador Mozilla[1] para integração a um servidor Web. Ele possui ferramentas para a criação de comunidades colaborativas on-line que são armazenadas na máquina do próprio usuário. Através do arcabouço JXTA da Sun, uma rede peer-to-peer é criada entre todos os nós Paper Airplane que estão sendo executados para realizar buscas por nomes de grupos, réplica de conteúdo e alcançar nós, normalmente inalcançáveis por causa de dispositivos de NAT´s e *firewalls*.

O Paper Airplane é a primeira aplicação a utilizar o arcabouço P2PSockets. Ele funciona como cliente e servidor em uma mesma máquina. O servidor é uma coleção de Servlets[2] e JSPs[3] construídos para

1 Mozilla - é uma suíte de aplicativos para Internet, livre, multi-plataforma, cujos componentes incluem um navegador, um cliente de correio eletrônico, um editor HTML e um cliente de bate-papo IRC (Internet Relay Chat).

rodarem sobre o P2PSockets. Os arquivos JSP/Servlets irão implementar serviços simples de Wiki[4] e de mensagens instantâneas.

A aplicação cliente foi construída utilizando as tecnologias da plataforma Mozilla (XUL, JavaScript, XPInstall etc) que formam uma coleção de ferramentas multi-plataforma criadas para que a construção de aplicações possa ser realizada da mesma forma que são criadas páginas Web. A ferramenta cliente funciona como uma extensão para o navegador Mozilla/FireBird. É uma barra de ferramentas com funcionalidades como: "Novo Site", "Editar Página", entre outros. Tais funcionalidades correspondem às ações que podem ser realizadas pelos usuários na criação e manutenção de seus próprios conteúdos localmente.

A ferramenta cliente conta ainda com tutores (*wizards*) que guiam o usuário na criação de seu sítio e escolha de um nome de domínio. Estes sítios são armazenados utilizando o sistema de domínio de nome fornecido pelo P2PSockets na própria máquina do usuário.

As aplicações cliente e servidor se conectam através de um *proxy* que executa na máquina local. Este *proxy* traduz requisições normais de navegadores em requisições para a rede ponto-a-ponto e vice-versa. O cliente é "induzido" a acreditar que está lidando com um servidor Web normal, quando na verdade tudo está sendo tratado por uma rede JXTA ponto-a-ponto. Domínio de nomes atualmente são nomes de grupos de pontos e as portas são arquivos pipe do JXTA.

O Paper Airplane tem como principal objetivo: trazer as características e benefícios das redes ponto-a-ponto para os navegadores Web. Isto fará com que os navegadores exibam arquivos HTML a partir de compartilhamentos de um outro usuário, assim como atuar como um servidor, atendendo requisições de páginas locais. As características citadas aliadas à possibilidade de escolha de qualquer padrão de caracteres (desde que não esteja em uso) como "pseudo-domínio" e ferramentas simples para geração de sítios fazem com que o Paper Airplane seja a ferramenta estudada mais próxima aos objetivos do Dedalus. Até mesmo os desafios encontrados são muito semelhantes.

Mas o Paper AirPlane não resolve o problema da auto-hospedagem por completo. Ele permite apenas a auto-hospedagem de conteúdo, pois não suporta a persistência de comportamentos. Com isto, não engloba todos os recursos já oferecidos pela Web hoje. O Dedalus parte do princípio que o usuário não vai aceitar perder funcionalidades que ele já possui. Outra questão á a abordagem utilizada pelo Dedalus para a construção de conteúdo: enquanto o Paper Airplane é uma extensão do navegador Mozzila, o Dedalus é uma aplicação *desktop* Java.

3 O Propósito Dedalus

Promover a cooperação mútua, seja ela formal ou informal, entre grupos de todas as idades e interesses pode proporcionar uma vasta gama de tipos de utilitários. Pode ser na área de informática na educação, ensino à distância, entretenimento, publicação de idéias, informativos e afins. No ensino à distância, vários cursos poderão ser conduzidos remotamente, interativamente e em tempo real. Permitindo uma maior divulgação e abrangência de novas tecnologias, assim como baixar o custo de capacitação pessoal em empresas, alunos de escolas técnicas, ensino primário, entre outros.

O Dedalus estimula o auto-aprendizado das pessoas possibilitando que elas mesmas construam novas aplicações ou que estendam as funcionalidades do sistema. Através da própria ferramenta, será possível adaptar ou criar comportamentos. À medida que a necessidade e o interesse do usuário cresçam, o sistema oferece recursos gradativos para isso, contribuindo para uma elevação no nível da cultura digital de seus usuários.

A consciência da existência de diferentes tipos de usuário levou à idealização de níveis de aplicações que poderão ser criadas no Dedalus. O primeiro tipo engloba as aplicações do nível I, que são as que podem

2 Servlet – Tecnologia Java para construção de conteúdos dinâmicos para a Web.

3 JSP – Java Server Pages – Tecnologia Java para geração dinâmica de páginas Web.

4 WikiWiki - software colaborativo que permite a edição coletiva dos documentos.

ser criadas através da própria interface gráfica do sistema. O segundo tipo é o nível II, elas são arquitetadas através da construção e integração de scripts escritos na linguagem python. As do nível III são as construídas através da criação de novas classes Java que estendam as classes e interfaces existentes no Dedalus.

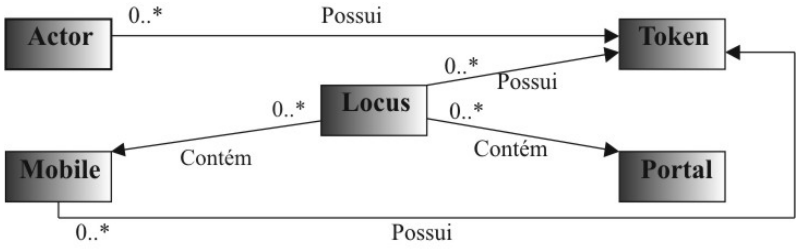
Quanto à utilização do Dedalus, a representação metafórica dos elementos de modelagem facilita o entendimento do sistema. Está mais que comprovada a eficácia da transmissão de conhecimento através da linguagem cultural das pessoas. Por esta razão acreditamos que a inclusão de pessoas no mundo digital pode ser feita através da associação de conceitos da informática com a vida cotidiana das pessoas. A associação do conceito com a metáfora aprendida será quase imediata, pois será transmitido através de uma linguagem mais natural.

Logo, a intenção do Dedalus é ser uma ferramenta de interação direta entre os diversos níveis sociais e intelectuais de pessoas. Assim como um viabilizador de novas idéias que estimula o aprendizado e a criação de novos serviços e funcionalidades. É uma ferramenta que procura ajudar as pessoas a transporem as barreiras sociais e tecnológicas que envolvem a utilização de recursos digitais a favor da liberdade de expressão.

4 Arquitetura

O Dedalus visa proporcionar uma melhor interação do usuário com o sistema. Pessoas que possuem pouco conhecimento tecnológico são capazes de ter suas próprias páginas, com componentes sofisticados, de uma forma direta e fácil. Os objetos e sites foram construídos com um forte apelo gráfico. Esse novo mundo pode ser utilizado em vários setores sociais, facilitando o relacionamento usuário-computador.

A implementação do sistema Dedalus é baseada em um mundo virtual composto por interfaces que descrevem o comportamento de cada elemento participante. São elas: Locus, Actor, Mobile, Portal e Token. Os usuários são os atores da aplicação, são quem sofrem e realizam ações , logo eles implementam a interface Actor. A interface Mobile representa os objetos que podem ir de um lugar para outro. Actors e mobiles devem estar em algum lugar. Este lugar é caracterizado pela interface Locus. Para que um ator possa ir de um Locus para outro, ele precisa de uma estrutura que lhe dê acesso , como para ir de um cômodo a outro de uma casa precisa-se de uma porta. É a interface Portal. Todos os elementos podem tomar ou sofrer ações que representarão estados ou indicadores para outras ações. Tais ações serão objetos que terão suas características gerais cobertas pela interface Token. Resumidamente, atores navegam por locus que são como um repositório de mobiles e portais. Os mobiles são seus objetos que servem para os mais diversos propósitos, desde mostrar um simples texto, até uma programação de eventos (uma aula, por exemplo). Os portais levam os atores até outros locus e tokens são as ações tomadas (como um ator "entrar" em um locus, por exemplo). O relacionamento entre os comportamentos é representado pela Figura 1.



. Figura 1. Interfaces do Dedalus

A arquitetura do sistema foi projetada para alcançar os seguintes objetivos: simplicidade e versatilidade. Com isto, o sistema é capaz de ser moldado de acordo com cada necessidade específica, além de prover liberdade e facilidade para a criação de novas funcionalidades ou remodelagem de funcionalidades já existentes. Tais modificações podem ser feitas de três maneiras diferentes. Isto porque o Dedalus possui mecanismos de interação que variam de acordo com o conhecimento tecnológico de cada usuário. Assim, podemos dividir as aplicações que podem ser criadas no Dedalus em três níveis:

Nível I – São as aplicações que podem ser criadas utilizando somente a interface gráfica do próprio Dedalus. Com isto, usuários com pouco conhecimento tecnológico poderão, de forma intuitiva, criar seus sites, publicar informações e criar lógicas de acordo com suas necessidades.

Nível II – Aumentando um pouco a dificuldade, temos expansões criadas através de scripts que poderão ser incorporados ao sistema. Uma linguagem de script, geralmente, é menos complexa que as linguagens de programação, mas possui grande parte de suas funcionalidades. Permitindo assim, uma elevação na complexidade da aplicação criada.

Nível III – São as aplicações que necessitam de mais funcionalidades ou possuem uma lógica mais complexa. Sendo assim, necessitam de programação direta em código Java.

Na fase de desenvolvimento que o dedalus se encontra, o foco maior é nas aplicações do nível I. São elas que melhor representam o ataque direto ao problema de segregação descrito na introdução deste documento. A facilidade de criar novos ambientes com lógicas simples e de editar ambientes e elementos já criados, permite uma grande liberdade de criação aos usuários, esteja ele em qualquer nível de conhecimento tecnológico. Mais adiante, a criação de aplicações deste tipo será ilustrada com um exemplo de jogo educativo (ver seção 6).

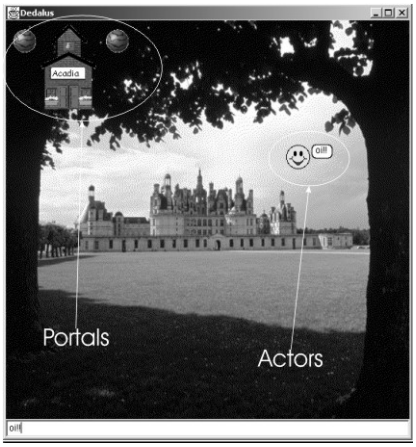
5 Implementação

Nesta sessão serão ressaltadas as questões mais relevantes da codificação propriamente dita. A implementação será descrita seguindo a ordem da arquitetura MVC do sistema. Primeiramente será apresentado como o modelo foi implementado, seguido da Vista e do Controle. Destacando principalmente os mecanismos criados para a edição e criação de elementos e seus comportamentos via interface gráfica do sistema, além da camada de persistência e da criação de locus que renderizam textos.

Modelo

Todo usuário possui sites e um inventário onde seus objetos móveis podem estar. Para conseguir estes objetos móveis, o usuário precisa ir até um site especial chamado Armazém (é um repositório de mobiles) - Figura 2a, copiá-los para seu inventário e depois colocá-los em outros sites. Os usuários também poderão criar portais em seus sites para sites de outros usuários e assim, poderão se visitar.

a) 

b) 

Figuras 2. Armazém e Home Site

Quando o usuário se *loga* no sistema, é criado um usuário Dedalus com o status de *logado* (para futuras verificações), seu site nativo (seu home site - Figura 2b) e um inventário de objetos. Seu home site irá, automaticamente possuir portais para os sites especiais Armazém, Dedalus e para seu próprio Inventário. Tendo o seu home site criado, o usuário poderá modificar suas propriedades, assim como adicionar ou excluir Mobiles e Portais. Todo Locus possui uma coleção de Mobiles e outra de Portals que refletem o que o Locus possui naquele determinado momento.

Um usuário pode visitar um site desde que possua um portal para ele. O usuário indica que quer passar por um portal e este se encarrega de pedir o seu site destino ao Pantheon, que é um repositório local de sites. O ator é removido do site em que estava e adicionado ao site destino. O site destino terá adicionado à sua coleção de portais um portal para o HomeSite daquele ator (visto apenas por aquele ator) para que ele possa sempre voltar.

No caso da classe User, quando um usuário visita um site, ele é adicionado à coleção de atores daquele site para que ele seja pintado e avisado caso ocorra alguma atualização. Mas uma instância da classe User possui muitas informações que não são pertinentes ao site em que ele está entrando. Por este motivo, ao invés de se passar o usuário propriamente dito, é passado um proxy com o nome do usuário, a sua imagem e o seu endereço remoto.

Para que os usuários do Dedalus se comuniquem, é preciso um mecanismo de navegação por sites. Quando um site não é encontrado no Pantheon, O módulo responsável por localizar e buscar sites remotos, chamado de Olimpo, envia uma requisição remota a todos os outros Olimpos que estão conectados ao grupo Dedalus. Para isto, ele verifica se o Pantheon associado àquele Olimpo possui o site procurado. Se sim, o Olimpo empacota o site procurado e o envia como resposta para o Olimpo requisitante.

Além da busca por sites, o módulo remoto também é o responsável por garantir que um site seja sempre o mesmo para todos os usuários que o visite. Com isto é preciso se ter um controle das atualizações que ocorrem no site. Mas como uma cópia do site modificado pode estar em vários lugares, é preciso notificar todos os Pantheons que possuírem uma cópia daquele site para que estas sejam sempre iguais. Na implementação deste controle de atualizações, foram utilizados os Tokens de entrada e de saída em um Locus. Estes tokens possuem a ação tomada e o objeto envolvido. Assim, quando um ator ou um mobile exerce uma ação em um Locus, um token é criado e pede-se ao Olimpo que o envie para todos do grupo sinalizando que alguma coisa naquele site foi modificada. Assim, quem possuir uma cópia do Locus, irá processar o Token, descobrir qual foi a ação tomada e repeti-la em sua cópia. Como em um mecanismo de réplica.

Commands

No sistema Dedalus os commands são responsáveis por controlar a edição de elementos e para a simulação do mundo virtual, onde existem condições como: um usuário ter que possuir um determinado objeto em seu inventário para efetuar alguma ação (entrar em um outro site, por exemplo). O Command funciona como um

método que é um objeto e, assim, pode ser passado como parâmetro para outros objetos ou métodos para indicar que determinada ação seja executada por eles. O objeto Command faz com que seja construída uma camada de abstração entre um objeto que solicita a ação e o que a executa e será o responsável por receber a requisição e passá-la ao receptor certo.

Tokens

Outra função dos commands é trabalhar com a interface Token para a simulação do mundo virtual do Dedalus. Os tokens são estruturas que podem produzir um estado de um determinado elemento como, por exemplo, um usuário precisar possuir uma determinada propriedade (uma chave) para efetuar alguma ação (entrar em um site). Todos os elementos (Locus, Actor, Mobile e Portal) podem tomar ou sofrer ações que representarão estados ou indicadores para outras ações. Tais ações, por serem tão complexas, foram representadas por objetos e possuem suas características representadas pela interface Token.

Token é um atributo (propriedade) que o usuário pode ter. São ações dos objetos em geral. Sua função é simular o comportamento dos objetos (mobiles, portais, locus e actor), bem como permitir que os usuários escolham que ações serão restringidas pelo token adicionado, isto é, um token é adicionado ao objeto de forma a restringir a execução de alguma ação neste objeto.

Um usuário possui uma coleção de tokens. Ao escolher o item de menu *Add Token* do popup, o usuário será que selecionar que tipo de token deseja adicionar ao objeto que sofreu a ação. São três os tipos de tokens que podem ser criados no Dedalus.:

Quantitativo – Uma ação só pode ser executada pelo usuário se este possuir a quantidade do token exigido. Exemplo: O usuário só pode entrar no Site Armazém se possuir uma quantidade igual a 10 (dez) do token milhagem. Este tipo de token é composto pelos campos nome do token, valor e ação. Somente o campo ação possui valores pré-definidos e que serão descritos mais adiante.

Condicional – O usuário só pode executar uma determinada ação se possuir o token exigido. Exemplo: Para pegar o mobile gato o usuário deve ter o token dinheiro. Este tipo de token é composto pelos campos nome e ação.

Adicional – Uma certa quantidade do token definido é adicionado ao usuário toda vez que ele executa uma determinada ação. Exemplo: Toda vez que o usuário entrar no Site Inventário receberá uma quantidade igual a 20 (vinte) do token milhagem. Esse tipo de token é composto pelos mesmos campos que o token quantitativo.

No caso dos mobiles, as ações a serem restringidas são: adicionar token, pegar mobile, remover mobile, salvar para inventário, trocar hint e trocar background. Já os portais possuem as ações adicionar token e entrar no locus. O ator possui apenas a ação de adicionar token. Um usuário não conseguirá executar a ação desejada caso não possua o token exigido. A figura 3 mostra como adicionar um token a um elemento.

a)

 **b)**

Figura 3. Como adicionar um token a um elemento.

Vista

Commands e Interação com os sprites

Todos os elementos citados na composição do mundo virtual Dedalus utilizam como suas representações gráficas estruturas chamadas de sprites. Sprites são elementos que possuem uma figura associada e que sabem como renderizá-las na tela do computador. Os sprites do Dedalus são herdados da biblioteca gráfica utilizada [Baklava]. Sendo assim, todo Locus, Portal ou Mobile possui seus respectivos representantes gráficos que por sua vez são do tipo Sprite.

Os sprites de um locus podem responder a diversos estímulos externos. O usuário pode clicar sobre um sprite, clicar com botão direito, arrastar um sprite em direção a outro. Em todos estes casos, um evento correspondente a este estímulo é gerado e repassado ao modelo. Cabe ao modelo, tomar as decisões correspondentes ao evento ocorrido.

Um sprite pode ter também um menu de opções. Este menu é montado em resposta ao clique com o botão direito sobre o sprite. No modelo, as opções de menu são armazenadas como uma coleção de PopupCommand. Esta classe possui um texto, que será o texto exibido, e um método execute().Vários PopupCommands podem ser adicionados a um mobile. Na vista, um PopupCommand é representado com um item de menu: DedalusMenuItem. Ao clicar em um item do menu, é chamado execute() do PopupCommand correspondente. Entre os comandos criados, temos: Largar um mobile em um locus, pegar um mobile de um locus, criar um novo mobile, portal, token ou locus de texto, trocar a imagem do mobile, trocar o nome do mobile, mudar a imagem de fundo de um locus e editar um texto. Assim, os commands são responsáveis por controlar a edição de elementos.

Controle

O controle é composto por duas classes: a DedalusClient e a User. A DedalusClient é a responsável pelo login do usuário e pelo registro do usuário e seus sites no site especial Dedalus. Já a classe User é a responsável pelo controle de visita e manutenção de sites e mobiles.

Locus Texto

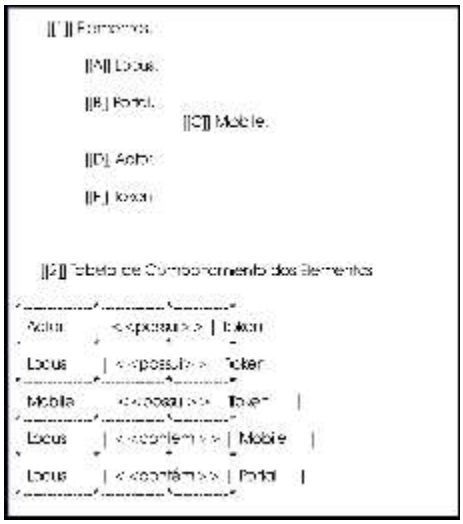
Para proporcionar uma melhor divulgação de conteúdos, de forma rápida e fácil, sem que se tenha a necessidade de conhecimentos da linguagem HTML, foi introduzido no Dedalus um novo tipo de locus, o Locus Texto. Sua principal vantagem é a publicação de textos já existentes, bem como a criação de novos, possibilitando uma melhor simulação da Internet tradicional. Além disso, permite que textos já publicados possam ser editados, o que é uma outra grande vantagem, pois não desperdiça todo o trabalho até então

empregado. Esse tipo de locus utiliza uma linguagem de marcação chamada APT (*Almost Plain Text*) para a formatação final das páginas de texto.

APT é uma linguagem de marcação simples (como o HTML) que pode ser utilizada para escrever um artigo como documento [APT]. Funciona como um filtro e ao contrário do HTML, usa poucas *tags* para expressar a estrutura do documento. Os documentos APT podem ser convertidos em muitos formatos (atualmente LaTeX, PS, PDF, HTML, SGML ou XML/DocBook, RTF), usando o comando de linha *aptconvert*.

O Locus Texto facilita a criação de páginas HTML, mesmo para aqueles que não possuem o conhecimento de tal linguagem, pois é necessário apenas que o usuário digite o seu texto, utilizando as marcações pré-definidas no APT (Figura 4a) e salve o conteúdo deste texto. O texto resultante é um texto no formato HTML sem que o usuário precise conhecer as diversas *tags* da linguagem de marcação (Figura 4b). Neste tipo de locus existe um link de retorno para o locus anterior. Um exemplo deste tipo de locus é mostrado nas figuras abaixo.

a)



b)

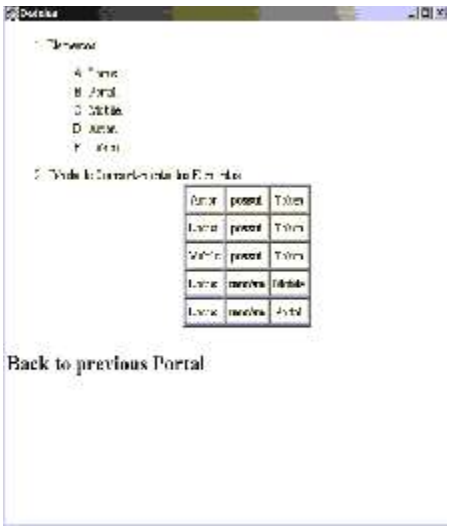


Figura 4. Notação Apt

Prevalência Distribuída

A persistência foi implantada utilizando um engenho de armazenamento de comandos em arquivos que serão retomados toda vez que o aplicativo for reinicializado. O armazenamento em arquivo utiliza a capacidade de serialização que a linguagens orientadas a objetos possuem (Java é uma delas).

Para isto, foi utilizada a biblioteca de terceiros Prevayler [Prevayler] que já possui todo o mecanismo de serialização de comandos. Sendo assim, basta que toda operação que necessite de persistência seja realizada através de um command que herde do command da biblioteca Prevayler. O command é então serializado no seu atual estado em um arquivo e é recuperado posteriormente, fazendo com que o sistema volte ao seu estado exatamente antes da aplicação ser finalizada.

O mecanismo de comandos para a persistência pode ser extensivo à replicação das informações e serviços pela rede distribuída do Dedalus. Para garantir uma maior disponibilidade das informações e dos serviços disponíveis, as redes ponto-a-ponto os replicam pelos diversos pontos. Desta forma, o conteúdo que está sendo servido por uma máquina de um usuário que tem a necessidade de se desconectar da rede, pode ser servido por outro usuário que permanecerá conectado.

Esta redundância provê uma solução de alta disponibilidade a um custo baixo. Para que isto seja possível, é preciso um mecanismo de replicação e, conseqüentemente, um de sincronização dessas informações. Sincronização por que, como o conteúdo dos sítios Dedalus são dinâmicos, existe a possibilidade de edição do conteúdo por outros usuários. Logo, quando o conteúdo que estava desconectado, conectar-se ou quando um conteúdo que está hospedado em mais de um lugar for modificado, terá que ocorrer uma sincronização das informações que garanta a consistência das mesmas.

Este mecanismo é o mesmo utilizado pelo Prevayler, ou seja, cada modificação realizada em um sítio Dedalus é encapsulada na forma de um comando. Este comando é enviado pela rede aos pontos que possuem cópia do sítio. O gerenciador de modificações do Dedalus recebe o comando e se encarrega de reproduzi-lo no seu sítio local, para que o mesmo se torne sincronizado com as demais cópias. Exatamente da mesma forma como o Prevayler realiza a carga dos dados de um sistema ao inicializá-lo.

6 Avaliação

As aplicações do nível I são muito úteis para que educadores consigam criar estruturas de ensino que trabalhem com a imaginação de seus alunos. O exemplo abaixo explora uma estória infantil como ferramenta de ensino de estruturas de dados. A associação da estória com os conceitos de árvore binária e busca, facilita o aprendizado. Primeiramente será descrita a estória criada para ensinar os alunos e depois a sua implementação segundo as metáforas do Dedalus. A aplicação foi desenvolvida totalmente no Dedalus através dos recursos do nível I por uma professora de estrutura de dados, aluna do mestrado do NCE/UFRJ.

Três Perguntas para Sete Portas

Era uma vez, um lugar distante chamado Verve onde habitavam quatro raças: os pequenos Aufnul, um povo de pequena estatura, trabalhador e alegre, mesmo em horas difíceis; a comunidade de comerciantes de Gnark, que visava o enriquecimento acima de qualquer coisa; um grupo de estudiosos chamados Lembas, donos de sabedoria e bondade, embora reduzidos a uns poucos devido à repressão do povo de Gnark; e os ferozes Grifnul, servos fiéis dos Gnark, criaturas dotadas de imensa força física e intelecto bem reduzido: só sabiam dizer "sim" e "não".

Muito embora gostassem de sua vida pacata e isolada dos demais povos, os Aufnul, de tempos em tempos, necessitavam ir à cidade Gnark adquirir suprimentos que, em suas terras, eram bem escassos. A cidade Gnark era rodeada de muros por toda volta, exceto por uma saída na parte posterior e a entrada, um local na frente da cidade onde havia um grande saguão com sete enormes portas de ferro e bronze numeradas de um a sete no verso. Dessa forma, ninguém era capaz de ver os números, a não ser o da porta que utilizou para entrar, pois apenas uma das portas abrigava a passagem para a cidade. Cada uma delas era guardada por um enorme Grifnul trajando uma couraça de ferro e portando uma enorme clava de madeira, do tamanho de um tronco de árvore. A cada acesso à cidade, um enorme mecanismo barulhento, manejado por dez Grifnul ainda maiores que os guardas, era acionado a fim de garantir que a passagem nunca ficasse detrás da mesma

porta por duas vezes seguidas. Assim, quem desejasse entrar na cidade deveria adivinhar por trás de qual porta o acesso era permitido. Somente os guardas sabiam qual a porta certa.

Para abrir uma porta, o visitante devia colocar-se na frente da porta escolhida e dizer ao guarda:

- Esta porta guarda a passagem?

Sendo a resposta "sim" ou "não", a porta era aberta e três moedas de ouro deveriam ser entregues ao Grifnul da porta, senão....

Antes, porém, de realizar a pergunta para abrir a porta, o visitante tinha direito de, sem nenhum ônus, fazer unicamente o seguinte questionamento:

- A passagem está numa porta com um número maior?

O objetivo do jogo é encontrar a porta da cidade gastando a menor quantidade de moedas de ouro possível.

Implementação segundo as metáforas do Dedalus

O site do jogo é um locus Dedalus que tem como fundo as muralhas da cidade de Gnark, com sua entrada composta pelas 7 portas (portais). Temos o usuário como sendo um actor Dedalus, representado por um Aufnul (boneco pequeno e não aparentando muita inteligência). Num primeiro momento, o cenário seria composto ainda por alguns mobiles: um guarda para cada porta, representado por um Grifnul (bonecos maiores e mal encarados) e um porteiro Gnark. Esses mobiles e o próprio actor, teriam alguns tokens de ação associados a eles. O Aufnul e os guardas se encarregariam do diálogo a respeito de qual porta está a passagem. O Aufnul possuiria o token de ação perguntar, que poderia ser dirigido aos guardas ou ao porteiro. Os guardas teriam associados a eles o token de ação responder (sempre com sim ou não) e o porteiro quando acionado teria o token de ação repetir a frase enigmática.

Além disso, haveria também um outro mobile, representando o narrador, que se encarregaria de contar a estória. O narrador poderia ser ativado ou desativado, de acordo com o desejo do usuário, pois após algumas jogadas, não será mais necessário ter de escutar toda a estória para iniciar a tentativa de abertura das portas.

Cada actor começaria com um número determinado de moedas (21 por exemplo), que será representado por um token de propriedade quantitativo. A cada pedido de abertura de porta o número de moedas seria diminuído de 3 unidades. O objetivo é conseguir entrar na cidade gastando o mínimo possível de moedas. No momento que o actor não possuir mais tokens moedas em número suficiente, ele estará incapacitado de tentar abrir as portas.

Caso necessitasse de ajuda, o usuário poderia adicionar ao seu locus um mobile Lembas (bonecos com aspecto idoso e sábio) que através de um token de ação se encarregaria de fornecer ao actor a explicação a respeito de como encontrar a porta de maneira mais rápida e eficiente (iniciando pelo meio e utilizando a pergunta sem ônus).

Apenas uma das portas seria um portal para um novo locus, representando o interior da cidade de Gnark. Ao abrir uma porta, seria exibido o número dessa porta e um fundo preto ou a visão do interior da cidade (dependendo se foi aberta a porta correta ou não). O site possuiria um outro portal que daria acesso a outro site, destinado ao ensino de estruturas de dados que conteria, entre outras coisas, explicações mais detalhadas a respeito de árvores binárias de busca e o mecanismo de pesquisa utilizado na solução do problema.

7 Conclusão

O modelo apresentado neste artigo, propõe uma plataforma base para outras aplicações que desejam usufruir as oportunidades da arquitetura federada. Entre elas, podemos destacar as aplicações de suporte ao ensino. Elas poderão se beneficiar da grande abrangência e cooperação proporcionadas pelo Dedalus. Outra utilização

seria na divulgação de serviços e informações por toda a comunidade e na área de entretenimento, como a criação de jogos.

O Dedalus proporciona uma maior independência ao usuário no sentido dele ter total controle das suas informações veiculadas. Eles podem criar suas próprias páginas, personalizando-as de uma forma fácil e rápida. A versatilidade do sistema é ampliada através dos mecanismos de expansão em níveis. À medida que o usuário obtém um certo amadurecimento digital, o Dedalus oferece novas funcionalidades que atendem ao anseio por conhecimento que eles desenvolvem.

Com estas possibilidades, foi percebido que o fato do usuário poder interagir com o funcionamento da aplicação, podendo moldá-la de acordo com suas necessidades, fez com que fossem estimulados seu raciocínio lógico computacional e a sua imaginação. Assim, pode-se vislumbrar a utilização da aplicação para o aprendizado intuitivo de computação para o mais variado tipo de usuários e aproveitando as facilidades que um ambiente distribuído e cooperativo pode oferecer.

8 Referências

[1] **Prevayler - Prevalence layer for Java**. URL: http://www.prevayler.org

[2] WILSON, B. J. "**JXTA**", 2001 . 1ª Edição - New Riders Publishing

[3] APT, http://reality.sgi.com/employees/jam_sb/mocap/MoCapWP_v2.0.html, 2003.

[4] Baklava, http://www.boutell.com/baklava/documentation.html, 2003.

[5] Paper AirPlane, https://paperairplane.dev.java.net/, 2004