

Servidor Web Adaptativo

P. R. M. Cereda

Resumo— A navegação do usuário na Internet gera uma procura por métodos que possam melhorar a experiência do usuário em um determinado website. Este artigo apresenta uma proposta de um servidor web adaptativo que assume um papel mais ativo durante a navegação do usuário, fornecendo técnicas de personalização e recomendação aos websites que estão hospedados em seu domínio, além de poder tratar de modo mais consistente e amigável os erros que eventualmente acontecem. São apresentados os aspectos de implementação e um experimento para verificação e análise dos módulos utilizados no servidor web.

Palavras-chave:— Servidor Web, Autômato Adaptativo, Sistema de Recomendação, Personalização.

I. INTRODUÇÃO

A navegação do usuário na Internet tornou-se um parâmetro fundamental para analisar a qualidade dos serviços disponibilizados. A grande maioria dos websites procura obter informações acerca dessa navegação e oferecer recursos personalizados e funcionalidades que possam melhorar essa experiência de navegação dos usuários visitantes.

Os servidores web são responsáveis por disponibilizar conteúdo para o usuário na Internet. De modo geral, o servidor web tem apenas uma participação passiva no processo de melhoria da experiência da navegação do usuário. A aplicação web quase sempre assume a tarefa de personalizar recursos e oferecer características inerentes aos interesses dos usuários.

Este artigo apresenta uma proposta de um servidor web adaptativo, chamado *Apache HTTPD Adaptativo*. O servidor em questão possui seis módulos que buscam melhorar a experiência de navegação dos usuários e utilizam autômatos adaptativos para oferecer tais funcionalidades. O autômato adaptativo foi escolhido para representar as características de cada módulo devido à sua simplicidade, capacidade de modificação autônoma e poder computacional.

A organização deste artigo é a seguinte: a Seção I contextualiza a navegação dos usuários na Internet, apresentando técnicas para melhoria da navegação e alguns problemas comuns encontrados. A Seção II apresenta algumas considerações sobre a adaptatividade em servidores web. O servidor web adaptativo é apresentado na Seção III. Os aspectos de implementação e implantação são apresentados na Seção IV. A Seção V apresenta um experimento realizado para

analisar as possíveis melhorias na navegação do usuário. As discussões sobre o servidor web proposto neste artigo são apresentadas na Seção VI. As conclusões são apresentadas na Seção VII.

II. NAVEGAÇÃO DO USUÁRIO NA INTERNET

Com o advento e popularização da Internet, os usuários passaram a interagir de modo mais ativo em websites, buscando cada vez mais um conteúdo inerente aos seus interesses. Assim, a forma de navegação do usuário na Internet mobilizou companhias de comércio eletrônico e grupos de pesquisa acadêmica na procura de métodos que pudessem melhorar a experiência do usuário em um determinado website.

Um dos métodos de melhoria da experiência de navegação é a personalização de serviços. De modo geral, a personalização pode ser descrita como tornar algo pessoal, individual, dependente das características e dos interesses humanos. Ao personalizar um objeto de acordo com um usuário, cria-se uma relação de afinidade e interesse. Grande [1] afirma que “um produto ou serviço pode atender as necessidades fundamentais de uma pessoa por suas funcionalidades e características primárias. Além disso, um serviço, através da personalização, pode possuir determinadas características que o torna mais parecido com um indivíduo. Essas qualidades secundárias são consideradas tão importantes que em muitos casos a escolha do produto ou serviço é regida somente através delas.” No âmbito do comércio eletrônico, a personalização é um recurso indispensável para tais websites.

Kimball e Merz [2] listam algumas funcionalidades de personalização, imprescindíveis para a melhoria da navegabilidade e experiência do usuário em um determinado website. Tais funcionalidades incluem: *reconhecimento de re-visitantes*, que possibilita a identificação do usuário como um novo visitante ou um visitante recorrente; *interface de usuário e personalização de conteúdo*, que permite ao usuário ter vários tipos de menus de acesso rápido, de acordo com sua preferência; *vendas colaterais*, que são recomendações a partir de conhecimentos prévios obtidos do usuário; *vendas por impulso*, que são sugestões geralmente relacionadas a um determinado grupo de interesse, como por exemplo, se o usuário comprou um determinado livro em um website de comércio eletrônico, o website poderá relatar outros produtos comprados por outros usuários que também compraram o mesmo livro; *filtragem colaborativa ativa*, que são sugestões e

indicações de outros usuários; *eventos de calendário*, no qual o conteúdo de um website pode ser relacionado de acordo com eventos baseados em feriados ou datas importantes, como por exemplo, dia das Mães, Páscoa ou Natal; *eventos de estilo de vida*, no qual o conteúdo pode ser relacionado de acordo com situações comuns do dia-a-dia, por exemplo, nascimento de um filho, casamento ou formatura; e *localização*, que utiliza um conhecimento prévio acerca do usuário para personalizar um conteúdo de acordo com o seu idioma ou localização geográfica.

Para que a personalização exista, é necessário que exista a coleta de algumas informações do usuário. Quanto mais informações são obtidas, maior será a capacidade de oferecer personalização ao usuário. Entretanto, existem vários questionamentos sobre a privacidade do usuário e a exposição do mesmo; assim, quanto mais informações obtidas, menos privacidade esse usuário terá [3], [4]. A melhor opção, na maioria dos casos, é oferecer uma personalização mínima sem, entretanto, macular a privacidade do usuário [5].

Existem diversos mecanismos para coleta de dados na Internet com o intuito de oferecer personalização. Geralmente, são utilizadas análises de dados em formulários e sobre a navegação do usuário [3]. Com os dados obtidos, são aplicadas técnicas de mineração de dados para determinar preferências, tendências e outras informações relevantes.

Um dos mecanismos mais interessantes para a análise da navegação do usuário em um website é o *clickstream*. Também conhecido como sequência de cliques ou *clickpath*, o *clickstream* representa o caminho que o usuário percorre enquanto está visitando um determinado website. A sequência de cliques obtida, quando aplicada e analisada de forma correta, pode proporcionar um conjunto de informações muito valioso para empresas e organizações [3].

Os dados de *clickstream* também podem ser coletados por Provedores de Serviço de Internet (ISPs), painéis de acesso ou mesmo manualmente, apesar da coleta através dos arquivos de log dos servidores ser a mais comum [6].

O *clickstream* é muito importante, do ponto de vista comercial, pois é possível identificar as preferências e os padrões de comportamento do usuário; isso inclui qual área lhe interessa, a frequência que a procura e quais as informações úteis para criar estratégias de marketing mais direcionadas e com maior chance de sucesso [7], [5].

Outra forma de melhorar a experiência de navegação é através dos chamados *sistemas de recomendação*. O *sistema de recomendação* é um método tradicional para retornar informações relevantes ao usuário. De um modo geral, um sistema de recomendação tenta identificar quais são os recursos mais importantes para um determinado usuário e recomendá-los [8], [9], [10].

Apesar da existência de tais melhorias, a navegação na Internet também pode ser prejudicada de modo significativo. Com a disseminação de programas maliciosos através de websites suspeitos, o usuário sente-se ameaçado e, por muitas vezes, interrompe sua navegação [11]. Em uma escala de risco muito menor, porém muito mais frequente, erros comuns de recursos não encontrados, *hyperlinks* inválidos ou servidores

web indisponíveis comprometem a permanência do usuário em um determinado website e sua navegação em geral.

É importante que o usuário possa ter uma navegação eficiente e consistente, minimizando os erros e oferecendo subsídios para melhoria da qualidade da mesma.

III. ADAPTATIVIDADE EM SERVIDORES WEB

Um *servidor web* é um programa de computador responsável por atender requisições e retornar conteúdo utilizando o protocolo HTTP⁶. O objetivo principal de um servidor web é retornar páginas web e todo o conteúdo associado a elas (imagens, folhas de estilo, entre outros). Um cliente (um *browser*⁷ ou um *crawler*⁸) faz uma requisição através do protocolo HTTP para o servidor, que recebe esta requisição, processa-a e retorna o conteúdo.

A quase totalidade das aplicações web busca fornecer para o usuário os subsídios necessários para personalização e recomendação de conteúdo e serviços. O servidor web tem uma participação passiva no processo, apenas servindo a seu propósito mais geral – responder requisições – e lançar erros quando estes acontecem.

A adição de uma camada adaptativa em servidores web justifica-se pelo fato de que a qualidade da navegação do usuário está intimamente relacionada com a qualidade das técnicas de personalização e recomendação presentes na aplicação web. Em outras palavras, uma aplicação web que não implementa tais técnicas – ou as implementa de modo precário – não pode oferecer uma experiência adequada ao usuário. Assim, propôs-se que o servidor web assumisse um papel mais ativo no processo, fornecendo técnicas de personalização e recomendação aos websites que estão hospedados em seu domínio, além de poder tratar de modo mais consistente e amigável todos os erros que eventualmente acontecem. Além disso, tal característica permite que as aplicações web sofram o mínimo de ajustes necessários para poder incorporar as funcionalidades oferecidas pelo servidor web adaptativo.

A. Autômato Adaptativo

O *autômato adaptativo*, proposto por Neto [12], é uma extensão do formalismo do autômato de pilha estruturado que permite o reconhecimento de linguagens do tipo 0, segundo a Hierarquia de Chomsky. O termo *adaptativo*, neste contexto, pode ser definido como a capacidade de um dispositivo em alterar seu próprio comportamento. Logo, um autômato adaptativo tem como característica a possibilidade de provocar alterações em sua própria topologia durante o processo de reconhecimento de uma dada cadeia [13].

Essa capacidade de alteração do autômato faz-se possível através da utilização de ações adaptativas, que podem ser executadas antes e/ou depois de uma transição. A cada

⁶ HTTP é o acrônimo de *Hypertext Transfer Protocol*, ou Protocolo de Transferência de Hipertexto.

⁷ *Browser* ou navegador é um programa de computador que permite ao usuário navegar na Internet. Os *browsers* mais conhecidos são *Internet Explorer* e *Mozilla Firefox*.

⁸ *Crawler* é uma aplicação que navega automaticamente por páginas web, indexando seu conteúdo e seguindo seus *hyperlinks*.

execução de uma ação adaptativa, o autômato tem sua topologia alterada, obtendo-se uma nova configuração. O objetivo de uma ação adaptativa é lidar com situações esperadas, mas ainda não consideradas, detectadas na cadeia submetida para reconhecimento pelo autômato [14]. Uma transição pode ter ações adaptativas associadas, que permitam a inclusão ou eliminação de estados e transições.

Um *autômato adaptativo* M é definido por $M = (Q, S, \Sigma, \Gamma, P, q_0, Z_0, F)$, tal que Q é o conjunto finito de estados, $Q \subseteq Q^A$, Q^A é o conjunto de todos os estados possíveis, Q^A é enumerável, S é o conjunto finito de sub-máquinas, Σ é o alfabeto de entrada, $\Sigma \subseteq \Sigma^A$, Σ^A é o conjunto enumerável de todos os símbolos possíveis, Γ é o alfabeto da pilha, $\Gamma \subseteq \Gamma^A$, $\Gamma = Q \cup \{Z_0\}$, Γ^A é o conjunto enumerável de todos os símbolos possíveis da pilha, $\Gamma^A = Q^A \cup \{Z_0\}$, P é um mapeamento $P: Q^A \times \Sigma^A \times \Gamma^A \rightarrow Q^A \times (\Sigma^A \cup \{\varepsilon\}) \times (\Gamma^A \cup \{\varepsilon\}) \times H^0 \times H^0$, H^0 definido a seguir, $q_0 \in Q$ é o estado inicial, $Z_0 \in \Gamma$ é o símbolo inicial da pilha, e $F \subseteq Q$ é o conjunto de estados finais. Os conjuntos “ A ” (“All” – para todos) são convenientes porque as *funções adaptativas* podem a) inserir novos estados q , $q \in Q$ mas $q \notin Q^A$, e b) usar novos símbolos de pilha $\gamma \in \Gamma$ mas $\gamma \notin \Gamma^A$. Em resumo, as *funções adaptativas* podem modificar o autômato, mas os novos símbolos que elas introduzem estão todos nos conjuntos “ A ” [15].

H^0 é o conjunto de todas as funções adaptativas no autômato adaptativo M . Define-se $H^0 = \{f \mid f: E \times G_1 \times G_2 \times \dots \times G_k \rightarrow E\}$, onde f é uma função, $k \in \mathbb{N}$ é o número de argumentos em f , e $G_i = Q^A \cup \Sigma^A \cup \Gamma^A$ [15].

E é o conjunto de todos os autômatos adaptativos que têm o estado inicial q_0 , o símbolo inicial de pilha Z_0 e o conjunto de estados finais F iguais aos do autômato adaptativo M . Define-se $E = \{N \mid N \text{ é um autômato adaptativo } N = (Q', \Sigma', \Gamma', P', q_0, Z_0, F), \text{ onde } Q' \subset Q^A, \Sigma' \subset \Sigma^A, \Gamma' \subset \Gamma^A, P': Q^A \times \Sigma^A \times \Gamma^A \rightarrow Q^A \times (\Sigma^A \cup \{\varepsilon\}) \times (\Gamma^A \cup \{\varepsilon\}) \times H^0 \times H^0\}$. Observe que q_0, Z_0 e F são os mesmos em qualquer $N \in E$ [15].

O conjunto de todas as sub-máquinas do autômato adaptativo M é representado por S . Cada *sub-máquina* s_i é definida como $s_i = (Q_i, \Sigma_i, P_i, q_{i0}, F_i)$, onde $Q_i \subseteq Q$ é o conjunto de estados da sub-máquina s_i , $\Sigma_i \subseteq \Sigma$ é o alfabeto de entrada da sub-máquina s_i , $P_i \subseteq P$ é o mapeamento da sub-máquina s_i , $q_{i0} \in Q_i$ é o estado inicial da sub-máquina s_i , e $F_i \subseteq Q_i$ é o conjunto de estados finais da sub-máquina s_i .

A linguagem aceita por um autômato adaptativo M é dada por $L(M) = \{w \in \Sigma^* \mid (q_0, w, Z_0) \vdash^* (q_f, \varepsilon, Z_0), \text{ onde } q_f \in F\}$. A capacidade de auto-modificação confere ao autômato adaptativo o poder computacional de uma Máquina de Turing [16].

Os autômatos adaptativos [12] [13] constituem um mecanismo formal para a descrição de linguagens recursivamente enumeráveis; sua simplicidade e facilidade de entendimento em relação ao modelo clássico de reconhecimento destas linguagens, a Máquina de Turing, fazem com que sua utilização seja muito ampla [17].

B. Servidor Web Adaptativo

Espera-se que um servidor web adaptativo contemple algumas funcionalidades desejadas para que ocorra uma melhoria significativa na experiência de navegação do usuário. Algumas destas funcionalidades incluem:

- a capacidade de oferecer personalização e recomendação de conteúdo ao usuário sem requerer modificações drásticas no código-fonte das aplicações web existentes;
- a verificação de erros de recursos não encontrados no servidor;
- a detecção de possíveis causas de erro e eventual tentativa de correção;
- a possibilidade de uma estrutura de navegação modificável ao longo do tempo; e
- a tentativa de gerenciamento eficiente de processos.

A Figura 1 ilustra uma possível arquitetura de um servidor web adaptativo. Essa arquitetura foi utilizada no desenvolvimento desta proposta.

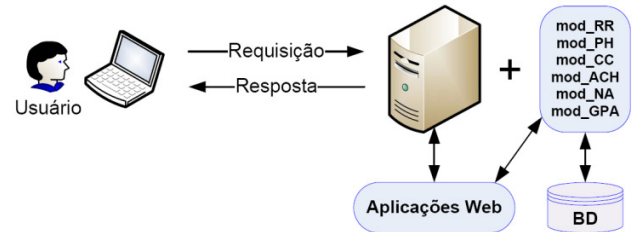


Figura 1 Uma possível arquitetura de um servidor web adaptativo.

De acordo com a Figura 1, espera-se que um servidor web adaptativo possua um conjunto de funções (módulos) incorporadas ao processo principal que proporcione as funcionalidades mencionadas anteriormente. Eventualmente, algumas destas funcionalidades podem requerer a persistência dos dados processados, justificando a utilização de um banco de dados ou algum arquivo de configuração disponibilizado. Além disso, algumas funcionalidades podem requerer também a alteração de alguns arquivos disponíveis no espaço das aplicações web. Deseja-se ainda que esse conjunto de funções esteja disponível de modo transparente ao usuário.

É importante que o servidor web adaptativo possa realizar tais funcionalidades de modo consistente e eficiente, para que a navegação do usuário não seja prejudicada, tanto em tempo de resposta quanto de qualidade de serviço.

Ao utilizar um servidor web com características de adaptatividade para oferecer melhorias na navegação do usuário, alguns questionamentos importantes surgem a partir dessa escolha. A Seção VII trata desse assunto, apresentando tais questionamentos em maiores detalhes.

IV. APACHE HTTPD ADAPTATIVO

O programa de computador *Apache HTTPD* é atualmente o servidor web mais comum existente, abrangendo em torno de 60% do mercado de servidores. Desenvolvido pela *Apache*

Software Foundation⁹, possui seu código-fonte disponível livremente para download, na página do projeto¹⁰.

O servidor Apache HTTPD foi projetado para oferecer um nível muito alto de flexibilidade. Uma de suas grandes características é a capacidade de permitir que usuários escrevam seus próprios módulos (também chamados de *Apache Modules*). Internamente, o Apache HTTPD é construído sobre módulos independentes projetados para realizar tarefas específicas. Em outras palavras, diversos módulos podem ser incorporados ao processo principal do servidor para estender suas funcionalidades. No geral, os módulos do Apache HTTPD abrangem suporte à CGI¹¹, controles de acesso, negociação de conteúdo, depuração, monitoramento, entre outros. Alguns módulos são incorporados por padrão; outros, entretanto, são adicionados à medida que são necessários [18].

A característica modular do Apache HTTPD mostrou-se adequada para um estudo sobre adaptatividade direcionada aos servidores web em diversos níveis de cobertura. Optou-se pelo desenvolvimento de módulos que utilizem elementos da tecnologia adaptativa e que possam contribuir para uma análise acerca da experiência de navegação do usuário. A Seção VII apresenta uma discussão sobre a implementação dessas funcionalidades sob a forma de módulos do Apache HTTPD.

Essa versão modificada do servidor Apache HTTPD recebeu o nome de *Apache HTTPD Adaptativo* e possui seis módulos disponíveis, a saber: recomendação de recursos, promoção de *hyperlinks*, criação de conteúdo, auto-correção de *hyperlinks*, navegação adaptativa e gerenciamento de processos adaptativo. Tais módulos são apresentados em detalhes a seguir.

A. Módulo de Recomendação de Recursos

O *Módulo de Recomendação de Recursos (mod_RR)* é responsável por verificar a existência de alguma relação entre os recursos disponíveis no servidor. Tais recursos podem ser representados por páginas web ou mesmo conteúdo multimídia, devidamente classificados e rotulados. O *mod_RR* utiliza o sistema *RecomAA* [10], [19] para recomendar os recursos disponíveis no servidor e apresentá-los sob a forma de *layers* durante a visualização de uma determinada página web.

O *RecomAA* atua da seguinte forma sobre os recursos disponíveis: dados dois recursos, o sistema verifica se existe alguma relação entre eles. Se a relação existe, o segundo recurso pode ser recomendado em função do primeiro e vice-versa. Por exemplo, dados os recursos *agenda* e *livro*, deseja-se saber se são relacionados; em outras palavras, o sistema responde *sim* ou *não* à seguinte pergunta: *dado o recurso agenda, é possível recomendar o recurso livro?*, ou *dado o recurso livro, é possível recomendar o recurso agenda?* [10].

Para auxiliar a execução do *mod_RR*, as páginas web recebem anotações especiais, inseridas em comentários no código-fonte, que são interpretadas em tempo de execução:

```
<!-- resource("Página inicial") -->
<!-- resource("Downloads") -->
```

Essas anotações especiais são submetidas ao *mod_RR* para verificação de relação entre possíveis recursos. Caso exista um recurso compatível com a recomendação, o servidor web adicionará um *layer* contendo a recomendação ao código-fonte da página em questão e retornará a requisição normalmente, conforme ilustra a Figura 2.

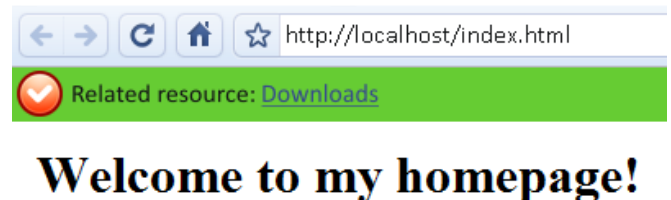


Figura 2 Exibição de uma recomendação.

O *mod_RR* também é responsável pelo monitoramento das escolhas do usuário em relação a um recurso recomendado. De acordo com a escolha do usuário (aceitar ou não uma determinada recomendação), o autômato adaptativo utilizado pelo *RecomAA* sofre alterações em sua topologia. Essas alterações buscam refletir de modo mais consistente as recomendações disponíveis no servidor web.

B. Módulo de Promoção de Hyperlinks

O *Módulo de Promoção de Hyperlinks (mod_PH)* é responsável pelo gerenciamento de *hyperlinks* em uma página web. O *mod_PH* avalia os *hyperlinks* disponíveis e registra o número de acessos a cada um. Eventualmente, os links mais acessados são promovidos, isto é, ascendem um nível hierárquico em relação aos demais. Para desempenhar tal tarefa, o módulo utiliza um autômato adaptativo, representado na Figura 3.

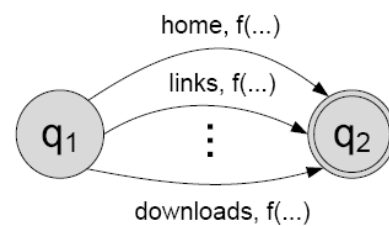


Figura 3 Autômato Adaptativo do Módulo de Promoção de Hyperlinks.

O autômato da Figura 3 representa a quantidade de acessos a cada *hyperlink* de uma determinada página web. Cada *hyperlink* α é representado como um símbolo na transição q_1 q_2 , e a função adaptativa f é responsável por inserir um novo estado q_i entre q_1 e q_2 , tal que $(q_1, \alpha) \rightarrow (q_i, \epsilon)$ e $(q_i, \epsilon) \rightarrow (q_2, \epsilon)$. Assim, a cada acesso ao *hyperlink* α adiciona-se um

⁹ Página oficial: <http://www.apache.org>.

¹⁰ Página oficial: <http://httpd.apache.org>.

¹¹ CGI é o acrônimo de *Common Gateway Interface*.

estado ao trecho do autômato que representa o número de acessos de α até o momento.

Após a atualização da configuração corrente do autômato adaptativo, o *mod_PH* realiza a contagem dos estados de cada trecho do autômato que representa um *hyperlink* e gera uma lista ordenada. Após a contagem, o *mod_PH* altera a ordem dos *hyperlinks* no código-fonte da página web de acordo com os números de acesso da lista obtida, do maior até o menor, e retorna a requisição ao usuário.

O *mod_PH* também faz uso de anotações especiais nas páginas web, inseridas em comentários no código-fonte, que são interpretadas em tempo de execução:

```
<!-- process-ranking -->
<a href="links.html">Links</a>

<!-- process-ranking -->
<a href="downloads.html">Downloads</a>

<!-- process-ranking -->
<a href="research.html">Research</a>
```

As anotações assinalam quais *hyperlinks* podem participar de promoções na página web. Esse tipo de marcação faz-se necessário devido ao fato de que alguns *hyperlinks* são imutáveis na página web, ou seja, não podem – e não devem – mudar de posição ao longo do tempo. Assim, optou-se por anotar explicitamente quais *hyperlinks* podem efetivamente sofrer promoções.

A Figura 4 exibe uma página web contendo uma lista de *hyperlinks* disponíveis.

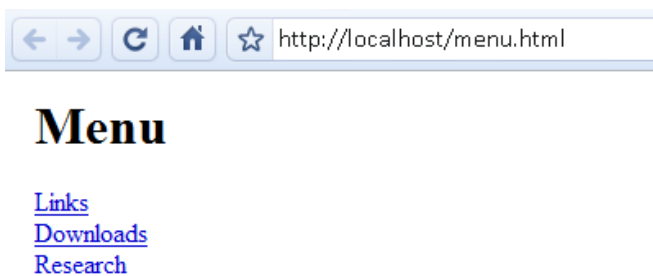


Figura 4 Página web contendo uma lista de *hyperlinks*.

Suponha que o *hyperlink* *Downloads* seja o mais acessado entre os três. A Figura 5 ilustra a promoção dos *hyperlinks* da página web, realizada pelo *mod_PH*.

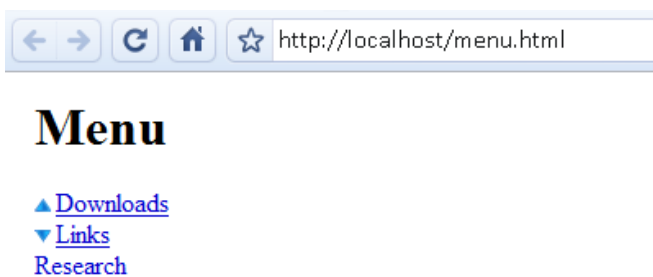


Figura 5 Página web contendo os *hyperlinks* já promovidos.

É importante observar que essa personalização de conteúdo baseada na promoção de *hyperlinks* não atua no código-fonte original da página web armazenada no servidor web, mas na instância a ser exibida em função de uma determinada requisição. O *mod_PH* encarrega-se de construir o autômato, analisá-lo e alterar o código-fonte da página web antes de enviá-la.

C. Módulo de Criação de Conteúdo

O *Módulo de Criação de Conteúdo* (*mod_CC*) é responsável pela criação de conteúdo no espaço de aplicações do servidor web. Em algumas situações, os websites apresentam *hyperlinks* inválidos (*broken hyperlinks*) para documentos, conteúdo multimídia ou mesmo para outras páginas web. Ao seguir um *hyperlink* que não pode ser resolvido, o servidor web lança o já conhecido erro 404 – *Não encontrado* (em inglês, *404 error – Not found*). O *mod_CC* analisa uma situação de erro 404 e verifica se o *hyperlink* procurado é acessado com frequência. Em caso positivo, o módulo cria a referência não encontrada, tornando o *hyperlink* válido. O autômato adaptativo responsável pela tomada de decisão é apresentado na Figura 6.

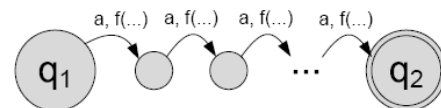


Figura 6 Autômato Adaptativo do Módulo de Criação de Conteúdo.

De acordo com a Figura 6, o autômato verifica se o recurso não encontrado a deve ou não ser criado. Entre os estados q_1 e q_2 , existe um caminho de estados (representados por estados menores) que define a ocorrência da procura pelo *hyperlink* procurado e não encontrado a ; à medida que tal recurso a é acessado, a função adaptativa f remove um dos estados intermediários, até que exista uma única transição $q_1 \rightarrow q_2$. Quando o estado final $q_2 \in F$ é atingido, isto significa que recurso não encontrado a já possui um número de ocorrências suficientemente grande para que tal recurso seja criado e disponibilizado no servidor web.

O *mod_CC* avalia se o estado final q_2 do autômato adaptativo foi alcançado e, em caso positivo, cria o recurso e o disponibiliza no espaço de aplicações do servidor web. A Figura 7 ilustra uma situação de erro em relação à tentativa de exibição do recurso *new.html*.



The requested URL /new.html was not found on this server.

Figura 7 Erro lançado pelo servidor web durante a tentativa de acessar um recurso inexistente.

Suponha que a procura pelo recurso inexistente *new.html* tenha atingido um número considerável de ocorrências. Dessa forma, ao tentar novamente exibir tal recurso, o *mod_CC* verificará esse número de ocorrências e criará o recurso, conforme ilustra a Figura 8.



Figura 8 Recurso recém-criado pelo Módulo de Criação de Conteúdo.

No momento da escrita deste artigo, o *mod_CC* ainda possui algumas limitações. O conteúdo criado limita-se apenas às páginas web, sem nenhum contexto válido; as páginas web recém-criadas contêm apenas um texto informando que não há conteúdo efetivo definido. Espera-se que, no futuro, este módulo seja capaz de obter dados oriundos de diversas fontes e popular tais conteúdos com informações válidas e pertinentes ao domínio da aplicação. Testes preliminares com sistemas de *question answering* [20] mostraram-se promissores para popular os novos recursos; por exemplo, o recurso recém-criado *futebol.html* poderia gerar uma pergunta a um sistema de *question answering* – “O que é futebol?” – e este se encarregaria de buscar a melhor resposta e acrescentá-la como conteúdo válido ao novo recurso.

D. Módulo de Auto-Correção de Hyperlinks

O Módulo de Auto-Correção de Hyperlinks (*mod_ACH*) é responsável pela verificação dos *hyperlinks* e uma eventual auto-correção dos mesmos. Em algumas situações do cotidiano, o erro 404 não é proveniente apenas de referências não existentes, mas também de erros de digitação dos *hyperlinks*. Por exemplo, a palavra *downloads* pode ser grafada erroneamente de várias formas, entre elas, *downkloads* (proximidade das letras K e L no teclado), *downlaods* (inversão das letras O e A), e *dowloads* (ausência da letra N). Esse tipo de erro inevitavelmente lançará um erro de recurso não encontrado, apesar de não ser este o caso (grafia errada).

O *mod_ACH* analisa o *hyperlink* da requisição e verifica se existe algum erro de grafia. Em caso positivo, tenta identificar qual é o *hyperlink* correto mais provável para a requisição e o

retorna para o cliente. Para cada *hyperlink* a ser analisado, o *mod_ACH* cria um autômato adaptativo correspondente. Por exemplo, o autômato adaptativo responsável pela identificação do *hyperlink downloads* é apresentado na Figura 9.

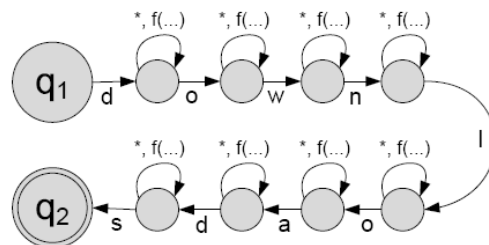


Figura 9 Autômato Adaptativo do Módulo de Auto-Correção de Hyperlinks, responsável pela identificação do *hyperlink downloads*.

O autômato da Figura 9 tenta identificar possíveis variações do *hyperlink downloads*. Caso a grafia seja diferente de *downloads*, o autômato tenta alterar sua topologia através da função adaptativa *f*. Se a cadeia for aceita, o *hyperlink* grafado erroneamente corresponde ao *hyperlink downloads*. O módulo *mod_ACH* então tratará a requisição como se fosse originalmente ao *hyperlink downloads*. Caso o autômato não aceite a cadeia, o erro 404 será lançado.

A Figura 10 apresenta a situação onde o *hyperlink downloads* foi grafado erroneamente (*downkloads*). O *mod_ACH* corrigiu o erro e informou o usuário acerca de seu erro, através de um *layer* adicionado ao código-fonte da página web.



Downloads

[Fedora](#)
[Slackware](#)
[Ubuntu](#)

Figura 10 *Hyperlink* corrigido através do Módulo de Auto-Correção de Hyperlinks.

O *mod_ACH* pode ser utilizado tanto para verificação automática de *hyperlinks* com erros de grafia nas páginas web, quanto para a digitação manual dos mesmos na barra de endereços do navegador. Em caso de lançamento do erro 404, o Módulo de Criação de Conteúdo pode trabalhar conjuntamente para evitar problemas oriundos de recursos não encontrados ou mal grafados.

E. Módulo de Navegação Adaptativa

O Módulo de Navegação Adaptativa (*mod_NA*) é responsável pela análise da navegação dos usuários e eventual alteração do mapa de *hyperlinks*. Em outras palavras, a

navegação pode ser facilitada, reduzindo-se o número de cliques necessários até alcançar algum recurso em particular.

Um exemplo de uma instância do autômato adaptativo do módulo *mod_NA* é apresentado na Figura 11. As possíveis configurações do autômato refletem os diversos mapas de navegação possíveis.

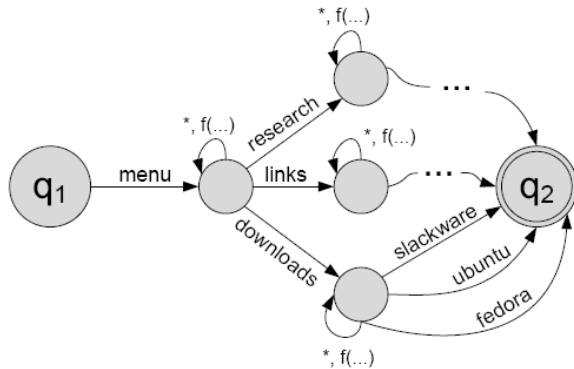


Figura 11 Autômato Adaptativo do Módulo de Navegação Adaptativa.

De modo simplificado, o autômato da Figura 11 pode ser analisado como um mapa de navegação por *hyperlinks*, onde cada caminho possível representa a ordem de cliques até atingir um determinado recurso. A função adaptativa *f* é responsável por encurtar um caminho, de acordo com a ocorrência de caminhos similares obtidos a partir de um histórico de navegação.

Por exemplo, suponha que o *hyperlink Fedora*¹² seja um dos mais acessados, e que o caminho *Página inicial* *Menu* *Downloads* *Fedora* possua um grande número de ocorrências. O módulo *mod_NA* poderá promover o *hyperlink Fedora*, reduzindo em um nível o caminho até ele. Dessa forma, o caminho mais curto passará a ser *Página inicial* *Menu* *Fedora*, conforme ilustrado na Figura 12.

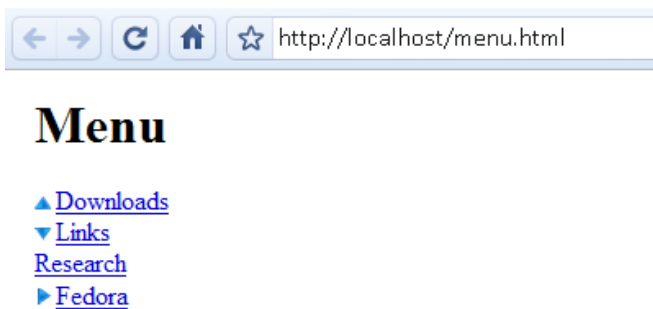


Figura 12 Promoção do *hyperlink Fedora*, reduzindo em um nível o caminho até ele.

Da mesma forma que o Módulo de Promoção de *Hyperlinks* (*mod_PH*), o *mod_NA* também faz uso de anotações especiais nas páginas web, inseridas em

comentários no código-fonte, que são interpretadas em tempo de execução:

```
<!-- process-browsing -->
<a href="fedora.iso">Fedora</a>

<!-- process-browsing -->
<a href="slackware.iso">Slackware</a>

<!-- process-browsing -->
<a href="ubuntu.iso">Ubuntu</a>
```

As anotações assinalam quais *hyperlinks* podem participar de alterações na navegação. Esse tipo de marcação faz-se necessário devido ao fato de que alguns *hyperlinks* são imutáveis na página web, ou seja, não podem – e não devem – mudar de posição ao longo do tempo. Assim, optou-se por anotar explicitamente quais *hyperlinks* podem efetivamente sofrer alterações na navegação.

A personalização de conteúdo sob o aspecto de uma navegação adaptativa não atua no código-fonte original da página web armazenada no servidor web, mas na instância a ser exibida em função de uma determinada requisição. O *mod_NA* encarrega-se de construir o autômato, analisá-lo e alterar o código-fonte da página web antes de enviá-la.

F. Módulo de Gerenciamento de Processos Adaptativo

O servidor Apache HTTPD utiliza um *pool* de processos em *prefork* para reduzir o atraso de tempo e custos que são associados à criação de novos processos. Existe um processo principal que monitora a combinação de portas e *sockets* nas quais as requisições via TCP/IP são recebidas. O processo principal nunca manipula tais requisições, mas as distribui para os processos subordinados. Cada um desses processos atua como um servidor em série, tratando cada cliente por vez. Quando um processo termina seu trabalho, volta para o *pool* [18].

Além de ser responsável pela distribuição de carga, o processo principal também é responsável pelo ajuste do número de processos subordinados. Esse ajuste é extremamente importante, pois poucos processos subordinados atrasam o atendimento aos clientes, enquanto que muitos processos desperdiçam recursos do sistema.

O Módulo de Gerenciamento de Processos Adaptativo (*mod_GPA*) tem como objetivo utilizar um autômato adaptativo para o gerenciamento do *pool* de processos. Este módulo ainda é experimental e apresenta alguns desafios de implementação.

O autômato adaptativo utilizado pelo *mod_GPA* é apresentado na Figura 13.

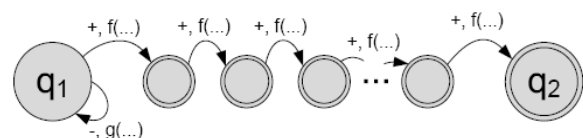


Figura 13 Autômato Adaptativo do Módulo de Gerenciamento de Processos Adaptativo.

¹² *Fedora* é uma distribuição Linux muito popular para *desktop*. Sua página oficial é: <http://www.fedoraproject.org>.

De modo simplificado, o autômato adaptativo do *mod_GPA* (Figura 13) tem o mesmo comportamento do processo principal do Apache HTTPD, no qual distribui as requisições para os processos subordinados (função adaptativa *f*, removendo estados intermediários) e gerencia os processos que retornam ao *pool* (função adaptativa *g*, inserindo estados intermediários). Os estados intermediários do autômato representam os processos subordinados disponíveis.

No momento da escrita deste artigo, o *mod_GPA* ainda possui limitações. Espera-se que, no futuro, este módulo seja capaz de gerenciar plenamente o *pool* de processos de modo consistente e eficiente.

V. IMPLEMENTAÇÃO E IMPLANTAÇÃO

Para a implementação de um servidor web adaptativo, foi utilizado o Apache HTTPD em sua versão 1.3.41¹³. As funcionalidades apresentadas na seção anterior foram escritas em C como módulos do Apache e carregadas junto ao servidor web durante sua inicialização.

De acordo com a proposta de arquitetura para o servidor web adaptativo apresentada na Seção III-B (Figura 1), seria interessante utilizar um banco de dados para realizar a persistência dos dados obtidos e tratados pelos módulos. Como primeiro protótipo, optou-se pela utilização do *engine* de banco de dados *SQLite*¹⁴, devido à sua simplicidade.

O servidor web adaptativo foi implantado em um computador dedicado e populado com uma quantidade considerável de páginas web, documentos e conteúdo multimídia. O computador em questão foi utilizado exclusivamente para atender requisições.

VI. EXPERIMENTO E ANÁLISE

Para verificar as funcionalidades do servidor web adaptativo, foi realizado um experimento com 30 alunos de graduação e pós-graduação de instituições de ensino superior da região de São Carlos. A primeira parte do experimento consistiu na definição das tarefas que cada aluno realizaria durante o período de quatro semanas. A Tabela I apresenta as tarefas definidas.

Tabela 1
TAREFAS DEFINIDAS PARA O EXPERIMENTO

Índice	Descrição da tarefa
t_1	Navegar aleatoriamente pelos websites
t_2	Verificar os <i>hyperlinks</i> disponíveis
t_3	Procurar por determinado conteúdo disponível
t_4	Procurar por conteúdo inexistente
t_5	Errar o nome de <i>hyperlinks</i> propositalmente
t_6	Realizar diversas requisições simultaneamente
t_7	Aceitar ou não as recomendações sugeridas
t_8	Acessar conteúdo multimídia disponível

Cada aluno poderia realizar qualquer tarefa apresentada na Tabela I quantas vezes desejasse, e sem uma ordem definida. A única exigência do experimento foi que todas as tarefas apresentadas fossem realizadas.

A segunda parte do experimento consistiu na interação dos alunos com o ambiente. O experimento durou quatro semanas, com tarefas diárias realizadas pelos alunos. Dois servidores contendo as mesmas aplicações web foram disponibilizados: o Apache HTTPD convencional e sua versão adaptativa. Os 30 alunos participantes foram divididos em dois grupos, e cada grupo interagiu com um dos servidores.

Durante todo o experimento, ao longo de quatro semanas, a navegação de cada aluno e o comportamento de cada servidor foram monitorados. A partir destas informações, os resultados foram obtidos.

A Figura 14 ilustra o número de ocorrências do erro 404 lançado pelos servidores durante as quatro semanas do experimento.

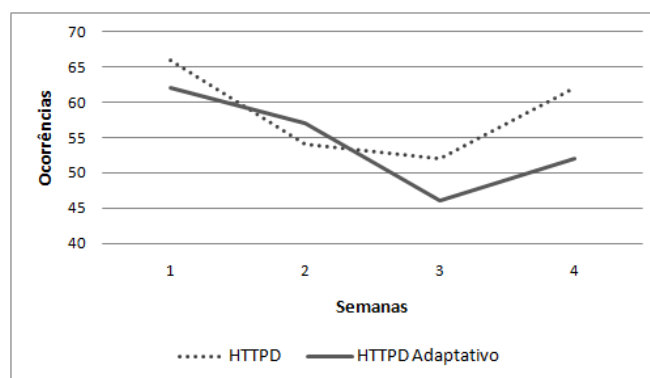


Figura 14 Número de ocorrências do erro 404 lançado pelos servidores durante as quatro semanas do experimento.

Algumas das tarefas consistiam em gerar propositalmente erros 404 (tarefas t_4 e t_5 , Tabela I). De acordo com a Figura 14, ao longo das quatro semanas, o servidor web adaptativo apresentou um número relativamente menor de erros 404. Essa diminuição de erros deve-se aos módulos que tratam da criação de conteúdo e da auto-correção de *hyperlinks* (*mod_CC* e *mod_ACH*, respectivamente). Tais módulos minimizam alguns problemas inerentes à navegação dos usuários.

A Figura 15 ilustra o total de cliques processados pelos servidores durante as quatro semanas do experimento.

¹³ Nota de lançamento disponível em <http://tinyurl.com/rdvtg>.

¹⁴ Página oficial disponível em <http://www.sqlite.org>.

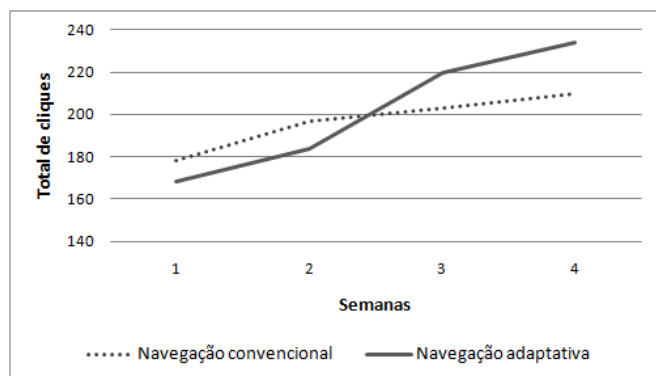


Figura 15 Total de cliques processados pelos servidores durante as quatro semanas do experimento.

De acordo com a Figura 15, é possível observar que, ao longo do tempo, a navegação adaptativa proporcionou um número maior de cliques nos *hyperlinks* disponíveis nas páginas. O módulo de navegação adaptativa (*mod_NA*) proporcionou uma navegação mais inerente aos interesses dos usuários durante o acesso ao conteúdo disponível.

A Figura 16 apresenta o total de cliques dos *hyperlinks*, promovidos (na versão adaptativa) ou não (na versão convencional), durante as quatro semanas do experimento.

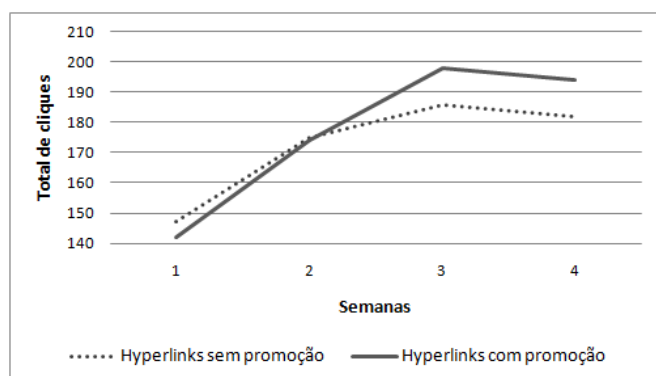


Figura 16 Total de cliques dos *hyperlinks* promovidos durante as quatro semanas do experimento.

É possível observar, de acordo com a Figura 16, que os *hyperlinks* promovidos proporcionaram um número maior de cliques do que seus equivalentes não promovidos. O módulo de promoção de *hyperlinks* (*mod_PH*) proporcionou ao usuário os *hyperlinks* mais relevantes à sua navegação, aumentando portanto o número de cliques.

A Figura 17 apresenta o total de cliques dos recursos recomendados durante as quatro semanas do experimento.

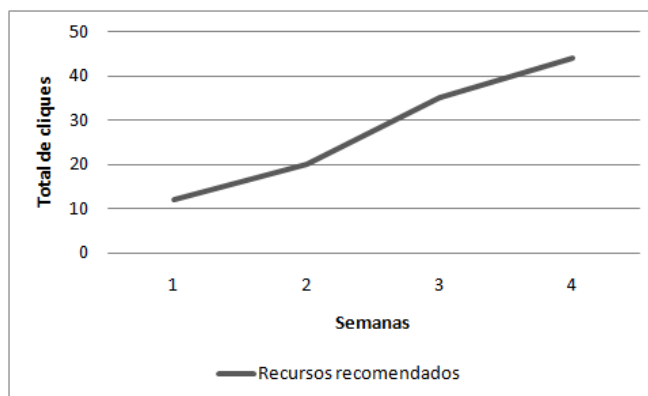


Figura 17 Total de cliques dos recursos recomendados durante as quatro semanas do experimento.

De acordo com a Figura 17, o total de cliques dos recursos recomendados aumentou ao longo das quatro semanas do experimento, ilustrando a importância da recomendação de recursos para os usuários. Com o tempo, as recomendações tornaram-se mais consistentes, o que justifica o aumento mais significativo nas duas últimas semanas. O módulo de recomendação de recursos (*mod_RR*) proporcionou uma navegação direcionada aos interesses do usuário, apresentando recursos semelhantes e sugerindo sua visualização.

VII. DISCUSSÕES

Esta seção trata das discussões acerca da utilização de um servidor web com características de adaptatividade para oferecer melhorias na navegação do usuário. A seguir, são apresentados alguns questionamentos.

A motivação para a proposta de um servidor web adaptativo iniciou-se com uma análise do comportamento do usuário ao navegar na Internet e as implicações no ambiente web. Constatou-se que a utilização de serviços personalizados e um tratamento mais adequado aos erros inerentes à navegação, da forma como são tratados atualmente, estão intimamente relacionados com as técnicas de implementação e suporte da aplicação web disponível. Em outras palavras, a qualidade da navegação do usuário está relacionada com a qualidade do serviço disponibilizado na aplicação web.

Na tentativa de remover a dependência de serviços de personalização e recomendação da aplicação web, propôs-se que o servidor web assumisse essa tarefa. Apesar de remover essa dependência da aplicação web, surgiram questionamentos referentes à viabilidade do servidor web como componente para tratar dos requisitos apresentados na Seção III-B. É importante destacar que o Apache HTTPD, mesmo tendo como objetivo principal e função básica atender requisições, devida à sua arquitetura modular, permite que tais requisições possam ser tratadas nos mais diversos níveis de abstração, desde a manipulação de pacotes de rede até interpretação de arquivos disponíveis no espaço de aplicações web. Essa possibilidade de tratamento em vários níveis motivou a utilização do servidor web para tratar das características de personalização e recomendação das aplicações web hospedadas em seu domínio.

O desenvolvimento de módulos Apache se mostrou interessante porque permitiu que as funcionalidades desejadas

fossem disponibilizadas de forma consistente no servidor web. O Apache HTTPD invoca os módulos em vários momentos durante a configuração e em vários pontos durante o processamento da requisição, mas apenas nas atividades nas quais os módulos tenham sido registrados. Por exemplo, o Apache HTTPD permite que um módulo atue na fase de manipulação de conteúdo (também conhecida como *content handling*), utilizando atributos de geração ou de transformação. Essa característica de gerenciamento permitiu que cada módulo atuasse em seu devido momento e realizasse sua tarefa específica.

Os resultados obtidos a partir do experimento apresentado na Seção VI são importantes, pois indicam que o servidor web adaptativo proposto neste artigo realmente ofereceu uma melhoria significativa na experiência de navegação do usuário. Entretanto, ainda não foi possível realizar testes de desempenho do servidor web nas mais diversas situações – aumento de demanda, latência de rede, latência de módulos, entre outros – e verificar seu comportamento. Tais testes podem permitir uma análise mais detalhada e fornecer subsídios para otimizações no modelo arquitetural. A obtenção destas métricas poderá ratificar o servidor web adaptativo como uma opção viável para melhoria da navegação do usuário.

VIII. CONCLUSÕES

Este artigo apresentou uma proposta de um servidor web adaptativo que utiliza autômatos adaptativos em seus módulos de controle. O autômato adaptativo, devido à característica de auto-modificação, é uma escolha interessante para auxiliar na tarefas de personalização, recomendação e otimização.

A adaptatividade em servidores web pode tornar-se uma tendência muito interessante, pois ao mesmo tempo em que fornece uma experiência nova de navegação ao usuário, também torna o processo de disponibilização e atualização de conteúdo já existente quase que transparente para o administrador do servidor. A utilização da tecnologia adaptativa na área de servidores web proporciona soluções consistentes que atendem às necessidades inerentes a esse domínio de aplicação. Além disso, o usuário beneficia-se com um conteúdo mais direcionado aos seus interesses.

AGRADECIMENTOS

O autor agradece ao prof. José de Oliveira Guimarães, do Departamento de Computação da Universidade Federal de São Carlos, *campus* de Sorocaba, pelo auxílio na revisão deste artigo.

REFERÊNCIAS

- [1] R. E. Grande, “Sistema de integração de técnicas de proteção de privacidade que permitem personalização”, Dissertação de Mestrado, Departamento de Computação, Universidade Federal de São Carlos, 2006.
- [15] R. Kimball and R. Merz, *Data Webhouse: construindo o data warehouse para a Web*. Rio de Janeiro: Campus, 2000.
- [16] S. D. Zorzo, R. A. Gotardo, P. R. M. Cereda, B. Y. L. Kimura, R. A. Rios and R. E. Grande, “Web privacy controlled by user: an approach to treat the user’s preferences about personal data”. In *European Computing Conference 2007*, 2007.
- [17] S. D. Zorzo and P. R. M. Cereda, “Fatores de privacidade e confiança em websites,” *Revista de Computação e Tecnologia da PUC-SP*, vol. 1, pp. 35–43, 2009.
- [18] P. R. M. Cereda and S. D. Zorzo, “Formalismo com autômato adaptativo em mecanismo de privacidade e personalização,” in *XXXIII Conferencia LatinoAmericana en Informatica*, vol. 1, San Jose, Costa Rica, 2007.
- [19] A. L. Montgomery, “Using clickstream data to predict www usage”, University of Maryland, Tech. Report, 2003.
- [20] A. L. B. Nogueira and L. R. Oliveira Jr., *Uma análise da aplicabilidade de Data Warehouse em ambientes empresariais*, Faculdade Ruy Barbosa, Salvador, 2004.
- [21] R. A. Gotardo, C. A. C. Teixeira and S. D. Zorzo, “Predicting user’s interests in web-based educational systems using a collaborative filtering weighted method”. In *11th IEEE International Conference on Computational Science and Engineering – Workshop on Computational Science and Engineering – CSE 2008*, 2008.
- [22] S. Sae-Tang and V. Esichaikul, “Web personalization techniques for e-commerce”. In *Active Media Technology: 6th International Computer Science Conference*, 2005.
- [23] P. R. M. Cereda, R. A. Gotardo, and S. D. Zorzo, “Resource recommendation using adaptive automaton,” in *16th International Workshop on Systems, Signals and Image Processing*, Chalkida, Greece, 2009.
- [24] W. Aiello and P. McDaniel, *Lecture 1, Intro: Privacy*, Stern School of Business, NYU, 2004.
- [25] J. J. Neto, “Contribuições à metodologia de construção de compiladores”, Tese de Livre Docência, Escola Politécnica da Universidade de São Paulo, São Paulo, 1993.
- [26] ———, “Adaptive automata for context-dependent languages”, *SIGPLAN Notices*, vol. 29, no. 9, PP. 115-124, 1994.
- [27] ———, “Solving complex problems efficiently with adaptive automata,” in *CIAA '00: Revised Papers from the 5th International Conference on Implementation and Application of Automata*. London, UK: Springer-Verlag, 2001, pp. 340–342.
- [28] P. R. M. Cereda, “Modelo de controle de acesso adaptativo”, Dissertação de Mestrado, Departamento de Computação, Universidade Federal de São Carlos, 2008.
- [29] R. L. A. Rocha and J. J. Neto, “Autômato adaptativo, limites e complexidade em comparação com máquina de Turing”. In *Proceedings of the Second Congress of Logic Applied to Technology – LAPTEC*. São Paulo, Brasil: Faculdade SENAC de Ciências Exatas e Tecnologia, 2001.
- [30] J. J. Neto, “Um levantamento da evolução da adaptatividade e da tecnologia adaptativa,” *IEEE Latin America Transactions*, vol. 5, no. Num. 7, pp. 496–505, Novembro 2007.
- [31] N. Gray, *Web Server Programming*. Wiley, 2003.
- [32] P. R. M. Cereda, R. A. Gotardo, and S. D. Zorzo, “Recomendação de recursos utilizando autômato adaptativo,” in *Terceiro Workshop de Tecnologia Adaptativa*, São Paulo, Brasil, 2009.
- [33] Lehnert, *The Process of Question Answering*. Erlbaum, 1978.



Paulo Roberto Massa Cereda obteve o mestrado em Ciência da Computação pela Universidade Federal de São Carlos. Tem experiência na área de Ciência da Computação, atuando nas áreas de Inteligência Artificial, Privacidade e Personalização de Serviços, Teoria da Computação e Tecnologia Adaptativa.