Universidade Estadual de Campinas
FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

Mecanismo de Monitoramento do Uso de Recursos Web para Apoio à Avaliação de Ambientes

Autora: Maria Angélica Calixto de Andrade Cardieri
Orientador: Prof. Dr. Ivan Luiz Marques Ricarte

Tese de Mestrado apresentada à Faculdade de Engenharia Elétrica e de Computação como parte dos requisitos para a obtenção do título de Mestre em Engenharia Elétrica. Área de concentração: **Engenharia de Computação.**

Banca Examinadora

Ivan Luiz Marques Ricarte, Ph.D.....FEEC/Unicamp
Heloisa Vieira da Rocha, Ph.D.....IC/Unicamp
Mário Jino, Ph.D.....FEEC/Unicamp

Campinas – SP
Dezembro/2004

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

C179m

Cardieri, Maria Angélica Calixto de Andrade
Mecanismo de monitoramento do uso de recursos Web para apoio à avaliação de ambientes / Maria Angélica Calixto de Andrade Cardieri. --Campinas, SP: [s.n.], 2004.

Orientador: Ivan Luiz Marques Ricarte
Dissertação (Mestrado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Ensino a distância. 2. Tecnologia educacional. 3. Internet na educação. I. Ricarte, Ivan Luiz Marques. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Título em Inglês: Mechanism to monitor interactions between students and pages in a Web based learning environment
Palavras-chave em inglês (keywords): Distance education; Educational technology e Internet (Computer network) in education
Área de concentração: Engenharia de Computação
Titulação: Mestre em Engenharia Elétrica
Banca Examinadora: Heloisa Vieira da Rocha e Mario Jino
Data da defesa: 10/12/2004

Resumo

Este trabalho apresenta um mecanismo que permite monitorar as ações realizadas pelo aluno em um ambiente de ensino na Web. A motivação para este desenvolvimento foi a constatação de que a maioria dos ambientes de ensino a distância, disponíveis na Web, não possuem uma forma eficiente para registro das interações dos usuários com as páginas. Este registro é importante, pois pode fornecer suporte para a avaliação do comportamento do aluno e de seu aprendizado. Além disso, possibilita que o próprio ambiente seja avaliado. O trabalho iniciou-se com a condução de um estudo de caso sobre um curso de ensino a distância já existente. Para este, foi construído um mecanismo específico de monitoramento a partir do qual foram identificados os problemas e adaptações necessárias. Foi então proposto um novo mecanismo para solucionar os problemas identificados. Este foi implementado e os resultados indicam um mecanismo flexível, eficiente e confiável de coleta de dados. Embora tenha sido voltado para ambientes educacionais, este mecanismo pode ser utilizado em qualquer aplicação na Web para a qual registros mais precisos sejam necessários.

Abstract

This work presents a mechanism to monitor interactions between students and pages in a Web based learning environment. There is a need for this type of mechanism due to the fact that most environments for distance education available on the Web, as well as their usage assessment tools, do not keep appropriate records about such interactions. This type of information is fundamental to support evaluation of students behavior and learning, adapt the system according to different learning models, and also to evaluate the environment itself. The tool development was motivated by a case study of a previously implemented Web based course. In this study, a specific monitoring mechanism was built, in which some problems and required changes were identified. Therefore, a new architecture was proposed for the monitoring tool, which was then implemented. Results presented in this work indicate a flexible, efficient, and reliable way to collect data. Even though this engine has been implemented in an educational environment, it can be used in any Web based application that requires more precise information about user interactions.

Aos meus pais José Paulo e Alba Maria,
por me ensinarem a arte de ser feliz.
A meu marido César e minha filha Giulia,
por todo o amor a mim dedicado.
Aos meus irmãos José Paulo, Márcio e Maurício,
sempre presentes, embora em terras distantes.

Agradecimentos

Ao professor Dr. Ivan Luiz Marques Ricarte pela orientação, incentivo e conhecimentos transferidos, essenciais para o desenvolvimento deste trabalho.

À Dra. Denise Bertolli Braga, pela oportunidade de trabalho concedida, que fundamentou o início deste projeto.

Aos meus colegas da FATEC-Sorocaba, Dimas Ferreira Cardoso e Samuel Oliveira pela ajuda em programação.

À minha família, amigos, colegas de mestrado, ao pessoal da "rede Recope", enfim, a todos que me incentivaram e colaboraram para a elaboração deste trabalho.

"Renova-te.
Renasce em ti mesmo.
Multiplica os teus olhos, para verem mais.
Multiplica os teus braços, para semeares tudo.
Destrói os olhos que tiverem visto.
Cria outros, para as visões novas"

Cecília Meireles

ÍNDICE

ÍNDICE DE FIGURAS

Capítulo 1 - Introdução

Com a evolução das tecnologias computacionais, um número cada vez maior de indivíduos tem acesso a Internet e seus serviços. Os recursos da Web (*World Wide Web*) têm sido utilizados para disponibilizar diversos tipos de aplicações, na área de comércio eletrônico, em ambientes de apoio à aprendizagem, sítios de pesquisas, consultas e outros.

As organizações responsáveis pelo desenvolvimento destas aplicações têm grande interesse em avaliá-las, buscando garantir a eficiência e a qualidade de seus produtos em um mercado cada vez mais competitivo. Pela identificação e registro das interações dos usuários na Web, torna-se possível avaliar o padrão de comportamento do usuário e extrair informações que podem ser utilizadas para guiar a manutenção das páginas, auxiliar a personalização de serviços, auxiliar estratégias de marketing e melhorar a estrutura e conteúdo da própria aplicação (Murtagh & Tao, 2000).

Existem atualmente vários mecanismos que permitem realizar o processo de monitoramento do uso da Web. Grande parte desses mecanismos possuem restrições em sua utilização pois o processo de monitoramento utilizado ou mesmo a tecnologia empregada nem sempre são adequados às diferentes aplicações na Web. Devido à natureza diferenciada dessas aplicações, podem surgir necessidades distintas em relação ao tipo de informação que é preciso coletar para que se possa obter uma base de dados adequada.

Este trabalho situa-se no contexto de ambientes de Educação a Distância Mediada por Computador – EDMC, induzido pelo estudo de caso que iniciou esta pesquisa. Este estudo teve como objeto uma aplicação desta categoria. Porém, apesar deste trabalho ter sido direcionado para a aplicação em um ambiente de ensino a distância mediado por computador, este mecanismo pode ser adequado a qualquer aplicação na Web onde seja necessário monitorar o acesso aos recursos disponíveis.

1.1 Motivação

Na área da educação, o desenvolvimento das tecnologias associadas à Internet aliadas à busca de uma alternativa ao ensino presencial e a redução de custos motivou a Educação a Distância Mediada por Computador – EDMC (Tarouco, 2000). Os ambientes de ensino a distância diferenciam-se dos ambientes tradicionais de ensino principalmente pelo aspecto não-presencial, pois não há uma interação face-a-face entre aluno e professor.

O processo de avaliação dos sistemas de EDMC deve ser capaz de fornecer parâmetros que permitam analisar o ambiente educacional tecnológico envolvendo não somente o material instrucional e aspectos computacionais, mas também o processo de aprendizado dos alunos.

A avaliação da aprendizagem é um dos grandes problemas encontrados nestes ambientes. Na modalidade tradicional, onde o ambiente é presencial, os mecanismos de avaliação são mais claros tais como: participação, freqüência, comportamento e atividades escritas ou orais. Nos ambientes de ensino a distância os instrumentos de avaliação tradicionais devem ser adequados, tendo em vista que os alunos não estão em contato direto com o professor.

Um significativo número de trabalhos têm sido publicados relativos à área de avaliação de ambientes educacionais mediados por computador. Segundo Hack (2000), o número de instituições de ensino que usam a Web aumenta a cada ano, porém na educação mediada por computador muitos materiais educacionais são pobres em componentes que permitam a avaliação do processo ensino e aprendizagem. Há, portanto a necessidade de mecanismos complementares para auxiliar a avaliação do aluno, do material e de todo o processo envolvido.

Constatando-se os fatores citados, este trabalho apresenta uma ferramenta que permite monitorar o aluno em um ambiente de ensino na Web, gerando uma base de dados representativa das interações do usuário com as páginas, procurando investigar não só o comportamento do usuário, mas também buscar informações que permitam a melhoria do projeto dos sistema elaborados.

O trabalho foi desenvolvido em duas fases. A primeira envolveu um estudo de caso baseado em um curso de ensino a distância já implementado: o Read in Web, cujo desenvolvimento foi coordenado pelo Instituto de Estudos da Linguagem da Universidade Estadual de Campinas – Unicamp. As características pedagógicas deste curso permitem caracterizá-lo como um "ensino auto-direcionado de leitura em inglês" (ReadWeb, 2002) e também como "auto-instrucional", isto é, o aluno desenvolve estratégias que lhe permitam resolver os problemas propostos. Por ocasião deste estudo, o curso já se encontrava em uso por vários alunos, porém não continha qualquer tipo de controle que provesse informações sobre a sua utilização. Devido às características pedagógicas do curso, o monitoramento das tarefas executadas pelo aluno tornava-se fundamental para apoiar o processo de avaliação do comportamento do aluno assim como o próprio curso.

Neste estudo de caso, foi desenvolvida uma ferramenta simples, que utilizou parcialmente a estrutura gerada automaticamente pelos servidores Web, os arquivos de registro de acessos (*http log file*), associada a um processo de filtragem e conversão dos dados. Esta ferramenta ofereceu alguma forma de controle sobre as ações executadas pelos alunos como solicitado. No entanto, alguns problemas e restrições foram diagnosticados.

Baseada nos resultados obtidos nesse estudo de caso, foi desenvolvida a segunda fase da pesquisa, a qual apresenta a proposta e o desenvolvimento de um sistema mais elaborado, utilizando um mecanismo desvinculado da estrutura dos arquivos de *logs* dos servidores Web. Esta solução agregou independência, flexibilidade e escalabilidade à ferramenta desenvolvida.

1.2 Organização do documento

Inicialmente, no Capítulo 2 é apresentado um panorama da educação mediada por computador, a problemática envolvida com o processo de rastreamento do uso dos recursos computacionais neste ambiente e em outras aplicações na Web assim como alguns trabalhos relacionados à área da pesquisa.

O Capítulo 3 apresenta o estudo de caso realizado e a descrição do processo utilizado para obtenção e análise de dados. A partir deste estudo, foram identificadas dificuldades e limitações, as quais motivaram a proposta de uma arquitetura para um novo mecanismo de coleta de registros de interação.

O Capítulo 4 descreve a solução proposta para os problemas identificados no capítulo anterior. São discutidos os requisitos, a arquitetura e detalhes da implementação. É também apresentada a proposta de uma ferramenta de visualização dos dados coletados como complemento ao trabalho desenvolvido. Finalmente é realizada uma avaliação da ferramenta.

O Capítulo 5 conclui este trabalho, apresentando um resumo dos principais aspectos abordados, juntamente com a proposta de trabalhos futuros.

Capítulo 2 - Estratégias de avaliação do uso de ambientes de educação via Web.

Nos últimos anos, a Internet vem se estabelecendo como grande meio de comunicação entre as pessoas envolvendo diversas áreas tais como entretenimento, negócios e educação. A proliferação de ambientes baseados na Web vem gerando uma preocupação crescente relativa à avaliação de sua utilização. Esta avaliação pode envolver tanto fatores relacionados ao desempenho como também a análise de conteúdo das páginas e das ações realizadas pelo usuário.

Particularmente na área educacional, a introdução de ambientes de educação a distância via Web introduz muitos desafios relacionados à avaliação, pois esta deve ser capaz de oferecer informações que auxiliem a conferir o grau de aprendizado ou as competências adquiridas pelos alunos.

Atualmente, as tecnologias empregadas na Internet permitem coletar um grande volume de dados associados à sua utilização. Masand e Spilioupoulou (2000) consideram que embora esses dados possam constituir uma "mina de ouro" a ser utilizada como suporte à avaliação de ambientes da Web, a qualidade destas informações é duvidosa devido ao próprio processo indisciplinado de desenvolvimento de aplicações na Internet, além do que pode ser gerada uma quantidade de informações difícil de se gerenciar. Desta forma, torna-se necessária a utilização de ferramentas ou mecanismos que permitam registrar de forma confiável as interações do usuário em ambientes da Web possibilitando posterior análise e extração do conhecimento associado a esses dados.

O uso de ferramentas para facilitar a extração, registro e posterior análise de grandes volumes de dados referentes à utilização da Web é uma área vastamente pesquisada com trabalhos disponíveis principalmente na área de educação a distância.

Alguns trabalhos concentram-se em técnicas para coleta e registro de informações empregando tecnologias disponíveis tais como CGI (*Common Gateway Interface*), *applets,* e agentes (Pereira & Geyer 2000; Souto *et al.,* 2000), enquanto outros se estendem além das técnicas de extração e pré-processamento incluindo a aplicação de técnicas e algoritmos de extração de dados (*data mining*) e posterior visualização das informações extraídas (Mobasher *et al*., 2000; Murtagh & Tao, 2000; Spilioupoulou & Faulstich, 1998).

Este capítulo apresentará conceitos relativos à área na qual este trabalho está inserido. Inicialmente será explorado o conceito de educação a distância mediada por computador e na seqüência serão abordados alguns tópicos relacionados ao processo de extração e registro de informações associadas ao uso de ferramentas educacionais na Web visando a análise do comportamento do usuário.

2.1 Educação a distância mediada por computador

Educação a distância (EAD) é considerada qualquer forma de estudo em que professor e aluno não estejam em contato direto, isto é, estejam geograficamente distantes. Sherry (1996) aponta como característica principal dos ambientes EAD a separação entre instrutor e aprendiz no espaço e/ou tempo onde a comunicação entre ambos se processa através de material impresso ou outra forma de tecnologia.

Segundo Barcellos (1998), a educação a distância se caracteriza por quatro fatores, sendo: 1- o professor e alunos estão separados a maior parte do processo de instrução; 2- uma instituição educacional influencia no processo, provendo alguma forma de avaliação; 3- meios de comunicação educacional são usados para unir professores e alunos e transmitir o conteúdo do curso; 4- comunicação bi-direcional entre professor e aluno.

A EAD não é uma técnica nova, ela tem sido usada como modo de ensino e aprendizagem durante pelo menos os últimos 100 anos (Barcellos, 1998). A forma mais antiga de educação a distância foi estabelecida por meio de cursos por correspondência na Europa utilizando-se textos escritos. Entre 1960 e 1980, a televisão e o rádio se popularizaram e passaram a ser utilizados de uma forma mais intensa como meios de comunicação de massa em EAD. Recursos tecnológicos de multimídia tais como áudio e vídeo, também passaram a ser usados como complemento a textos impressos. A partir dos anos 80, novas tecnologias na área computacional permitiram o desenvolvimento de ferramentas diferenciadas de apoio ao aprendizado. As tecnologias interativas associadas ao computador tais como: CD-Rom, hipertexto, hipermídia, realidade virtual e programas simuladores, passaram a ser integradas aos ambientes de ensino e aprendizagem.

A consolidação da Internet como meio de comunicação possibilitou uma inovação na forma de se oferecer cursos a distância, a EDMC – Educação a Distância Mediada por Computador, que se originou a partir da união das tecnologias interativas com a educação (Tarouco, 2000). Neste contexto, a educação a distância é entendida como o apoio ao aprendizado realizado de maneira on-line, baseado em tecnologia que inclui tanto o treinamento assistido por computador como o treinamento baseado na Web.

A EDMC vem se expandindo rapidamente em todo o mundo. Muitas instituições vêm desenvolvendo programas educacionais a distância procurando aumentar as oportunidades educacionais oferecidas a seus estudantes ou mesmo atrair novos alunos entre pessoas que trabalham ou que possuam outros tipos de restrições de tempo ou mobilidade (Hentea *et al.*, 2003).

Outros motivos que concorrem para estimular o avanço da EDMC, segundo Souto *et al*. (2000) são: a possibilidade de realizar aprendizado continuado, possibilidade de adquirir novas habilidades e competências, a possibilidade de reduzir custos e a disponibilidade de recursos tecnológicos a custos reduzidos. Alguns pontos importantes também constituem vantagens da EDMC sobre os outros modos de educação tais como: a

rapidez e abrangência das informações, pois estas podem ser acessadas de toda parte do mundo através da Internet de modo rápido; possibilidade de maior interação entre alunos e professores em relação à educação presencial, uma vez que elimina a inibição ou timidez dos alunos e a grande variedade de ferramentas para comunicação disponíveis como por exemplo a comunicação por textos ou em tempo real (Barcellos, 1998).

Enquanto o sistema de educação convencional totalmente síncrono, calcado somente no uso da sala de aula, exige a presença física do instrutor e do aluno simultaneamente, a EDMC oferece condições assíncronas de aprendizado por meio do vasto ferramental pedagógico atualmente disponível, que também pode ser combinado com o ferramental do sistema convencional trazendo eficiência no aprendizado final e flexibilidade ao ambiente educacional (Loyolla & Prates, 1998). Podemos considerar como ferramentas convencionais, utilizadas no ensino tradicional: os textos didáticos em papel; aulas expositivas presenciais; avaliação de trabalhos e seminários com presença simultânea de professor e alunos; e orientações de pesquisas e dissertações também com a presença dos envolvidos. As ferramentas não convencionais utilizadas pela EDMC são; textos didáticos e aulas expositivas disponibilizadas em páginas na Internet; orientação de pesquisas, trabalhos e seminários através de correio eletrônico ou diálogo remoto (*chats*), além de outras ferramentas complementares tais como CD-ROM e sistemas de tele-conferência.

Apesar das vantagens da EDMC, vários problemas ainda impedem a implementação com sucesso deste modo de aprendizado. Estes problemas afetam alunos, professores e também as instituições. Muitos estudantes que utilizaram esta forma de treinamento apontaram problemas. As críticas são relativas à insuficiência de tempo ou recursos alocados às tarefas requisitadas, frustrações com o hardware ou software empregado, sentimento de isolamento em relação a outros estudantes ou o professor e insatisfação com suas notas (Hentea *et al.*, 2003).

A preparação inadequada do estudante também é um ponto frágil. Devido a fracos critérios de seleção permite-se, muitas vezes, a matrícula de alunos sem o perfil adequado para freqüentar o curso, seja por falta de conhecimentos ou por falta de disponibilidade de hardware e software em seu local de estudo.

A falta de treinamento dos professores para o ambiente de ensino a distância pode impedir o desenvolvimento de um material de apoio adequado prejudicando o aprendizado. Segundo Souza (2002), em qualquer sistema de ensino-aprendizagem a mediação pedagógica é indispensável. Nos sistemas presenciais, o professor é o mediador pedagógico entre a informação a oferecer e a aprendizagem por parte dos alunos. No caso dos sistemas de ensino a distância esta mediação, feita por meio de textos e outros materiais e ferramentas de comunicação, tem que ser pedagogicamente diferente dos meios usados nos sistemas presenciais. Isto faz com que o desenvolvimento de um ambiente computacional para o apoio à aprendizagem não seja trivial. Os profissionais envolvidos com a área de ensino a distância têm, na maioria das vezes, dificuldades em produzir um material instrucional adequado independentemente do meio de comunicação empregado (Souza, 2002). Este problema se agrava também pelo fato de não ser fácil obter-se um retorno em relação à avaliação do próprio curso.

Hentea *et al.* (2003) argumentam que as soluções para os problemas da EDMC envolvem vários aspectos, como o treinamento adequado de professores, seleção mais criteriosa dos alunos, a utilização de novos métodos ou tecnologias e uma forma mais efetiva de avaliação do aprendizado. Segundo os autores, a avaliação é a mais importante e difícil parte da educação a distância .

Em um ambiente tradicional de ensino presencial, métodos formais e informais podem ser utilizados para avaliação do aluno. Os métodos formais incluem as provas, testes e listas de exercícios. As avaliações informais são realizadas por meio de técnicas tais como a observação das expressões faciais do aluno, participação em sala de aula ou mesmo realização de perguntas que possibilitem verificar o comportamento individual ou coletivo

dos alunos (Menezes *et al*.,1998). Em EDMC, mecanismos de avaliação informais são muito pouco utilizados, fazendo com que professor tenha dificuldades avaliar o estudante baseado em suas reações (Musa e Oliveira, 2000).

Perosa e Santos (2003) consideram que a avaliação, em qualquer forma de processo educacional, deve ser encarada como um processo contínuo e sistemático, presente em todas as etapas do trabalho. Analisando de forma mais ampla, esta avaliação deve ser mais qualitativa do que quantitativa, pois devem ser considerados além da aquisição de conhecimentos, a capacidade de observação, reflexão, criação, julgamento, comunicação, cooperação, decisão e ação no processo educativo.

Para que sejam incluídos os aspectos qualitativos no processo de avaliação do aluno, diversas atividades devem ser consideradas tais como as interações do aluno com o texto, com o material adicional disponibilizado, consulta a resposta-modelo ou mesmo a utilização de atividades colaborativas que venham a integrar o aluno com os demais participantes estimulando discussões e troca de conhecimentos.

Neste contexto o trabalho de tutoria, isto é, o acompanhamento das atividades desenvolvidas pelo aluno passo a passo, surge como fundamental suporte à avaliação do aluno, assim como do ambiente de ensino a distância .

Para Hack (2000), o rastreamento ou monitoramento do aluno no curso é importante porque permite ao professor monitorar o progresso dos alunos, mostrando o tipo de acesso que o aluno tem feito e o tempo gasto em cada acesso. Serve também de guia ao aluno, pois a forma não-linear com que os cursos são estruturados na Web pode fazer com que o aluno se perca, necessitando de orientação sobre onde está e para onde deve ir.

Mas não é somente a avaliação do aluno a fonte de desafios. As dúvidas relativas ao processo de avaliação de um ambiente educacional na Web situam-se em um nível mais amplo onde despontam outras questões tais como: O que realmente deve ser avaliado? O

que é possível ser avaliado? Quando avaliar? A avaliação do ambiente deve ser contínua ou em fases específicas? Como medir se o material instrucional é adequado? Estamos utilizando a tecnologia adequada? Como rastrear as interações do aluno com o material?

Em relação a quando um sistema de ensino a distância deve ser avaliado, Souza (2002) considera que este processo deve ocorrer em pelo menos três momentos básicos: antes do início do curso para validação do material; durante a oferta através do trabalho de tutoria e após a oferta para analisar resultados.

O rastreamento do uso dos recursos Web, além de ser útil para a avaliação do aluno, também pode ser associado a outras atividades tal como ser agregado a ferramentas que permitam a personalização de serviços como por exemplo um tutorial individualizado ou exercícios suplementares para reforço de aprendizado.

Pereira e Geyer (2000) apresentam um agente desenvolvido para selecionar estratégias de ensino para um ambiente educacional na Internet. Este agente observa a interação do aluno com o curso e avalia as táticas que devem ou não ser aplicadas ao aluno devido ao seu desempenho. Musa *et al.* (2001) consideram que, mesmo no caso de sistemas já consolidados, o monitoramento das interações é uma necessidade que se faz presente, pois também possibilita analisar a forma pela qual um ambiente de ensino a distância pode ser aperfeiçoado.

Torna-se evidente que um ambiente educacional mediado por computador deve contar com mecanismos que permitam a avaliação de todo o processo de ensino e aprendizagem. Esse tipo de avaliação requer registros da interação dos alunos com o material desenvolvido que permitem que os autores verifiquem se o seu uso ocorre conforme o previsto em sua concepção. Assim é possível contemplar a análise individual do aluno ou do grupo; avaliação técnica do ambiente; avaliação de técnicas de aprendizagem e interesses pedagógicos. Os resultados provenientes destas avaliações

podem fornecer subsídios que possibilitem não somente checar o desempenho do aluno, mas também buscar a melhoria da qualidade dos ambientes desenvolvidos.

2.2 Aquisição de dados da interação via Web

Os dados coletados relativos ao uso da Web podem auxiliar a análise do comportamento do usuário. Estes dados constituem uma vasta e rica fonte de informações. Esta afirmação é válida tanto para aplicações de comércio e consultas na Internet, como para ambientes de ensino e aprendizagem.

O processo de obtenção e análise de dados extraídos da Web não é um processo isolado, uma vez que a simples coleta não é suficiente para se obter informações consistentes. Na maioria das vezes, é componente de um processo mais amplo denominado "mineração de dados".

O termo mineração do uso da Web pode ser definido como sendo "a aplicação de técnicas de *data mining* a grandes repositórios de dados da Web com o objetivo de extrair padrões de uso" (Cooley *et al*., 1999). Esta técnica pode utilizar diferentes tipos de fontes de dados classificados em:

- Conteúdo: é o dado real da página e consiste geralmente de textos e figuras.
- Estrutura: é o dado que descreve a organização do conteúdo. A informação da estrutura inclui *tags* de vários tipos nas linguagens HTML (*Hypertext Markup Language*) ou XML (*Extensible Markup Language*).
- Uso: é o dado que descreve o padrão de uso das página na Web, tal como identificação da origem, páginas referenciadas, dia e hora de acesso.

A análise do conteúdo concentra-se na descoberta de informações úteis na Web para analisar, classificar ou definir categorias de documentos. A análise do documento inclui não somente o conteúdo da página mas também sua estrutura, isto é, as ligações entre as páginas e o relacionamento semântico entre elas (Masand & Spilioupoulou, 2000). Estas duas formas de análise são chamadas de "mineração do conteúdo da Web" (*Web Content Mining).*

A análise do uso focaliza a descoberta do conhecimento sobre as pessoas que utilizam a Web, seus interesse e expectativas, os problemas que são encontrados e os requisitos implícitos associados aos mesmos e é normalmente denominada "mineração do uso da Web" (*Web Usage Mining*).

Grande parte das análises se concentra no terceiro tipo de fonte citado acima, isto é, na análise do uso que representa o comportamento do usuário. Este trabalho se enquadra neste tipo de análise e dará ênfase a técnicas que explorem a obtenção e preparação dos dados coletados.

Ao realizarmos a análise do uso da Web (Cooley *et al.*, 1997) podemos descobrir informações sobre como reestruturar um sítio da Web para aumentar a produtividade, gerenciar melhor a comunicação de um trabalho e analisar padrões de acesso dos usuários. O uso da mineração de dados também está sendo utilizado para suporte ao desenvolvimento de sítios adaptativos com personalização das aplicações Web. Diversos trabalhos de pesquisa dão enfoque à personalização como alvo principal. Mobasher *et al.* (2000), por exemplo, sugerem um *framework* para obter uma personalização mais efetiva da Web, associando as informações do uso com informações do perfil contidas no própria página e mostra como a junção destes pode ser usada para prover personalização em tempo-real.

Ceri e*t al.* (1999) classificam o suporte à personalização como um dos dez princípios básicos para o desenvolvimento de sítios da Web que disponibilizam grandes quantidades de dados a vários usuários. Um exemplo típico é o comércio eletrônico, onde

pode ser interessante apresentar versões diferentes das páginas de acordo com perfis específicos.

Outro ponto em discussão é a amplitude da análise do comportamento do usuário que engloba dois aspectos: 1- o interesse do usuário e a informação que ele acessa; 2- o modo como ele acessa a informação (Spiliopoulou *et al.*, 1999). O primeiro aspecto pode ser alcançado por técnicas que estabelecem o perfil do usuário tais como tutores inteligentes e não consiste portanto em uma análise específica do uso da Web. O segundo aspecto é, na maioria das vezes, atingido por meio de técnicas de análises dos *logs* dos servidores Web.

Na ferramenta denominada WUM (*Web Utilization Miner*), Spiliopoulou e Faulstich (1998) exploram a análise do comportamento navegacional do usuário, onde se procura a especificação, descoberta e visualização de padrões que reflitam o interesse do usuário. Esta ferramenta utiliza o arquivo de *log* dos servidores Web. Murtagh e Tao (2000) também utilizam os arquivos de *log*, os quais servem como base para a construção de um modelo transacional onde é registrado um histórico dos acessos do usuário para posteriormente serem aplicadas técnicas de mineração de dados.

Outros trabalhos apresentam ferramentas que utilizam um processo mais elaborado de extração de dados, pela implementação de técnicas especiais tais como *cookies* ou agentes (Shahabi *et al*., 2001).

Várias questões importantes estão associadas ao paradigma da Web, principalmente quando há a necessidade de análises com técnicas sofisticadas para serem feitas em dados coletados do lado servidor. Dentre estas questões podemos citar a necessidade de integração de várias fontes de dados tais como: *logs* de acesso dos servidores Web, dados adicionais referentes ao usuário e informações provenientes de perfis (Cooley *et al.,* 1997b). A dificuldade na identificação do usuário também constitui uma questão a ser resolvida, pois nem sempre o acesso é realizado por meio do nome do usuário e senha, o

que pode causar a falta de atributos chave nos dados coletados. Por fim, a identificação de sessões e transações a partir dos dados de uso e modelos de comportamento do usuário é outro aspecto que necessita atenção.

Independentemente da ferramenta ou mecanismo utilizado para extrair o conhecimento, torna-se necessário um processo prévio de coleta e preparação dos dados, processo este que pode ser constituído de várias etapas uma vez que a fonte de dados utilizada apresenta na maioria das vezes dados brutos, muitas vezes irrelevantes, que necessitam ser tratados para encontrar uma informação confiável.

Segundo Shahabi *et al*. (2001), um típico sistema voltado à "mineração do uso da Web" consiste de duas camadas:

1- Aquisição, na qual a interação com o usuário é requisitada,
2- Análise, na qual os padrões de acesso do usuário são descobertos e interpretados pela aplicação de técnicas de mineração aos dados coletados.

A camada de aquisição de dados é um componente necessário em qualquer sistema de mineração de dados de uso. Nestes sistemas esta camada deve ser confiável, eficiente e permitir escalabilidade (Shahabi *et al*., 2001). Várias técnicas podem ser usadas para implementar esta camada, como relatado nas seções seguintes.

As técnicas de análises e descoberta de padrões não serão discutidas de forma específica, pois não fazem parte do objetivo deste trabalho, embora a utilização de tais técnicas seja indicada como seqüência natural após a coleta dos dados.

2.2.1 Arquivos de *log*

Como citado anteriormente, os arquivos de *log* gerados pelos próprios servidores (*http log files*) são usados como base em vários trabalhos. Estes arquivos normalmente seguem o padrão *Common Log Format File* (CLF), definido como parte do protocolo HTTP, conforme apresentado na Figura 2.1. Dentro desse padrão, qualquer requisição de serviços feita ao servidor é registrada, sendo que esta normalmente inclui: o endereço IP de onde a requisição se originou, a identificação do usuário (caso requisitada), data e hora da requisição e o recurso solicitado, identificado por seu localizador uniforme de recursos (URL).

Atualmente, existem diversas ferramentas comerciais que permitem obter informações sobre a interação do usuário com páginas na Web. Estas ferramentas normalmente exploram estes arquivos de *log* gerados automaticamente pelos servidores Web.

Formato: **"%h %l %u %t \"%r\" %s %b"**

%h É o endereço IP do cliente o que fez a requisição ao servidor
%l Corresponde à identificação da máquina cliente. É válida apenas para redes internas onde a máquina tem uma identificação específica. O hífen significa falta de informação.
%u É a identificação do usuário (*userid)* de acordo com o procedimento de autenticação do http. Se o documento não é protegido por uma senha, então um hífen é colocado nesta posição.
%t Corresponde à hora que o servidor terminou de processar a requisição. O formato é (dia/mes/ano:hora:minuto:segundo).
%r Corresponde à linha requerida pelo cliente. Indica o método, o recurso requisitado e o protocolo.
%s Código do status que o servidor envia para o cliente. Revela se houve sucesso (código começando com 2) ou erro (código começando com 4 ou 5).
%b Indica o tamanho em bytes do objeto retornado ao cliente.

Figura 2.1 - Padrão *Common Log Format File* do protocolo HTTP

Ferramentas comerciais, tais como *Webaliz*er (Barret, 2001), *WebTrends* (Webtrends Corporation, 2001) e outras, geram relatórios estatísticos baseados no *log* padrão. As informações que estas disponibilizam incluem totais de visitas a páginas, páginas mais acessadas, data e hora de utilização, origem das requisições e tráfego total. Também é possível, na maioria dos casos realizar algum tipo de filtragem selecionando URLs e nomes de domínios. Um exemplo de relatório gerado pela ferramenta Webalizer, relativo ao arquivo de *log* do servidor Apache instalado no Centro de Computação da Unicamp, pode ser visualizados na Figura 2.2.

Daily Statistics for December 2000

Day	Hits		Files		Pages		Visits		Sites		KBytes	
22	598	20.54%	328	21.49%	162	21,29%	8	10,13%	8	11,27%	1721	21.12%
23	112	3.85%	100	6.55%	26	3,42%	27	34,18%	14	19,72%	411	5.04%
24	242	8.31%	183	11.99%	49	6,44%	4	5,06%	4	5,63%	768	9.42%
25	217	7.45%	151	9.90%	75	9,86%	18	22,78%	11	15,49%	872	10,7%
26	293	10.07%	148	9.70%	68	8,94%	6	7,59%	6	8,45%	737	9.04%
27	181	6.22%	113	7.40%	40	5,26%	6	7,59%	5	7,04%	472	5.79%
28	339	11.65%	191	12.52%	92	12,09%	9	11,39%	9	12,68%	907	11.13%
29	679	23.33%	300	19.66%	177	23,26%	8	10,13%	9	12,68%	2192	26.90%
30	228	7.83%	8	0.52%	62	8,15%	3	3.80%	3	4,23%	45	0.55%
31	22	0.76%	4	0.26%	10	1,31%	3	3.80%	4	5,63%	25	0.31%

Figura 2.2 - Exemplo de relatórios gerados pela ferramenta Webalizer.

A figura acima apresenta algumas informações estatísticas relativas aos acessos realizados, onde os totais exibidos representam:

Hits: um *hit* é qualquer requisição feita ao servidor tal como páginas HTML, imagens, áudio, etc. Esta coluna apresenta o total de requisições feitas ao servidor durante o período de tempo selecionado que, no exemplo, refere-se ao período de 22 a 31 de dezembro de 2000.

Files: total de respostas enviadas de volta do servidor ao cliente como resultado das requisições (hits).

Pages: representa o número total de páginas requisitadas ao servidor. Neste caso é considerada uma página apenas documentos em HTML ou qualquer coisa que gere um documento HTML. São desconsiderados documentos do tipo imagens, áudio, etc.

Sites: cada requisição feita ao servidor é proveniente de um único endereço IP (*Internet Protocol*), aqui denominado *site*. Esta coluna mostra quantos endereços IP diferentes fizeram requisições ao servidor durante o período.

Visits: Uma requisição feita ao servidor a partir de um *site* (como definido acima) é considerada uma visita se a diferença de tempo entre a requisição atual e a anterior for maior que um tempo pré-determinado (*default* 30 minutos, pode ser alterado). Só são consideradas as requisições de documentos do tipo *page*.

As informações oferecidas pela maioria das ferramentas que utilizam o *log* do servidor são úteis em um cenário onde os dados a serem analisados são resultados de fatores quantitativos e se restringem a contagem e quantificação de uso, não sendo indicadas para traçar o comportamento do usuário, definir padrões de uso ou para qualquer outra análise qualitativa.

Porém, as limitações relativas a essas ferramentas não se baseiam apenas no fato de fornecerem informações resumidas ou inadequadas, mas principalmente por utilizarem os arquivos de *log* dos servidores como fonte de informação.

Shahabi *et al.* (2001) consideram os *logs* da Web pouco confiáveis e ineficientes. Não são confiáveis pelo fato de não registrarem acessos a várias páginas devido à existência de vários níveis de *caches* embutidos na Web, como por exemplo a *cache* do navegador e a *cache* dos servidores *proxies.* São ineficientes, pois precisam de um pré-processamento, por vezes exaustivo, antes de serem utilizados porque carregam um grande número de itens irrelevantes como arquivos de figuras, sons, animações, etc.

Devido aos problemas citados com os arquivos de *log*, várias técnicas alternativas são usadas para aquisição de dados, a maioria delas baseadas em algum mecanismo implementado pelo navegador do lado cliente que faça o registro direto das ações utilizando para isto *cookies*, agentes, registros via Javascript, CGI e outros mecanismos.

2.2.2 *Cookies*

Os *cookies* são pequenos arquivos de texto que são armazenados por um servidor da Web no cliente pelo navegador. Sempre que um navegador solicita um URL cujo servidor e diretório correspondem àqueles dos *cookies* armazenados, os *cookies* correspondentes são enviados de volta ao servidor (somente para o servidor que originou os *cookies*) na forma de cabeçalhos de solicitação.

A principal motivação para o uso de *cookies* é o gerenciamento da sessão, pois o protocolo HTTP não tem informações de estado, isto é, não mantém qualquer informação sobre os navegadores que estão conectados a eles de uma solicitação para outra. O gerenciamento da sessão é implementado por *cookies*, gerando um número de identificação

de sessão único para cada usuário e armazenando esta identificação em um *cookie.* Estes dados são enviados de volta ao servidor a cada solicitação subseqüente por uma página da Web que tem como origem aquele servidor.

A abordagem baseada em *cookies* apresenta algumas desvantagens (IBM-Developers, 2004). A principal delas é associada à privacidade do usuário, que muitas vezes inibe a permissão de instalação de *cookies* em suas máquinas com receio de que informações confidenciais, tais como número do cartão de crédito, sejam armazenados indevidamente. Desta forma não é possível depender somente dos *cookies* para obter informações sobre a sessão. Além disto, a implementação pode tornar-se mais complicada quando se tem vários *cookies* envolvidos e há um limite para o número de *cookies* que podem ser gerados para cada domínio.

2.2.3 CGI – Common Gateway Interface

O CGI (*Common Gateway Interface*) foi o primeiro padrão para conteúdo dinâmico da Web. Ele especifica um mecanismo para servidores Web passarem as informações da solicitação para programas externos, que são executados para gerar respostas em tempo de execução.

O *script* CGI é um programa que reside no computador servidor e convencionalmente localiza-se no diretório `cgi-bin`. Na configuração do servidor Web é possível especificar que um ou mais diretórios são reservados para o CGI. Sempre que for recebida uma solicitação para um URL contendo referência a esses diretórios, o servidor Web sabe que esta é uma solicitação CGI e que ele precisa executar um aplicativo ao invés de ler um arquivo HTML. O servidor Web ativa o *script* CGI, o qual processa os dados e produz outra página HTML, que é enviada de volta para o navegador.

O CGI é um mecanismo simples, bem estabelecido e com o qual os desenvolvedores estão familiarizados (Horstmann & Corner, 2000). No entanto, a abordagem tradicional de CGI tem algumas ineficiências que limitam sua aplicabilidade de uso em aplicações de grande escala baseadas na Web. Um novo processo deve ser iniciado cada vez que for solicitada a execução de um programa de CGI. Isto ocasiona uma sobrecarga associada com a criação e comunicação com este, além do que cada processo precisa de sua própria cota de recursos de memória de máquina local. Estes fatores podem trazer significativos problemas de desempenho, principalmente porque muitos sítios da Web tratam de milhares de solicitações simultâneas.

A segurança desses *scripts* também é difícil de ser controlada. Os *scripts* CGI podem ser escritos em qualquer linguagem que possa ler a entrada padrão e escrever para a saída padrão. A escolha mais comum é o Perl. No entanto o código Perl é de difícil compreensão para os não acostumados com a linguagem.

2.2.4 Agentes

Um agente pode ser definido como "uma entidade autônoma capaz de perceber o ambiente através de sensores e atuar neste ambiente através de atuadores" (Russel & Norvig, 1995). Um agente é uma entidade de software que funciona de forma contínua e autônoma em um ambiente em particular. Ele deve ser capaz de se comunicar com outros agentes, intervir no seu ambiente sem necessidade de orientação humana constante.

A idéia de empregar agentes para delegar certas tarefas baseadas em computador foi inicialmente introduzida por Negroponte e Kay (Maes, 1994). O termo agente foi inicialmente definido em trabalhos preliminares na área de inteligência artificial, onde pesquisadores dedicavam-se em tentar reproduzir uma entidade artificial que imitasse as habilidades humanas.

Muitos trabalhos têm sido elaborados utilizando agentes. Segundo Morreale (1998), inicialmente os agentes desenvolvidos estavam restritos a um único computador ou no máximo a uma rede homogênea, como uma plataforma Unix por exemplo e limitado a tarefas pré-estabelecidas. Atualmente os agentes estão quebrando este confinamento a um só ambiente apresentando-se móveis e aprendendo a executar tarefas baseados em sua própria experiência.

A utilização de um agente para auxílio ao processo de avaliação dos alunos em ambientes de ensino a distância na Web é proposto em Musa *et al.* (2001). Este agente monitora as interações do aluno na tentativa de descobrir e tratar comportamentos fora do previsto. Espera-se com isto assegurar que o aluno aproveite o curso de forma satisfatória. Este agente está integrado em uma sociedade de agentes, isto é, um conjunto de agentes que se comunicam trabalhando de forma cooperativa.

A utilização de agentes para aquisição de dados relativos ao uso da Web também é proposto em Pereira e Geyer (2000). Neste trabalho é apresentado um agente pedagógico[1] que tem como objetivo a seleção automática de estratégias de ensino adaptadas às características de um modelo de aluno individual. A modelagem do aluno é baseada em um perfil previamente atribuído ao aluno. Este perfil pode ser modificado ao longo do curso. Neste caso, o agente pedagógico não tem o papel de registrar o uso mas de selecionar estratégias de ensino e aprendizagem para o ambiente. O agente modifica a estratégia caso receba uma mensagem de baixo desempenho do aprendiz. Com isto, espera-se que o aluno possa alcançar uma aprendizagem mais efetiva.

Shahabi *et al. (*2001) propõem um mecanismo de aquisição de dados de uso que coleta os dados do lado cliente e, através de um agente remoto implementado em Java, envia informações referentes ao uso ao servidor Web. Este agente é carregado pelo

[1] Agentes pedagógicos são aqueles que estão ligados a um ambiente onde existe uma sociedade de agentes que compõem um sistema de ensino-aprendizagem (Pereira e Geyer, 2000).

navegador no lado cliente e captura todas as características da interação do usuário com o *sítio* tais como acessos, número de vezes de visualização das páginas e transfere estes dados para um módulo do servidor que grava a informação diretamente em um banco de dados. Este agente é implementado por um código Java, embutido em uma página HTML.

A abordagem utilizando agentes não apresenta desvantagens associada à privacidade do usuário tal como os *cookies* ou mesmo a segurança das páginas acessadas, como no caso de scripts CGI. No entanto, um fator que pode restringir a sua utilização é a complexidade envolvida em sua implementação.

2.2.5 Preparação dos dados extraídos

Independentemente da técnica que originou os dados a serem analisados, torna-se necessário uma fase de preparação para o processamento pois, não raramente, o volume recolhido é muito grande e as informações significativas encontram-se misturadas a outras sem importância. Segundo Cooley *et al.* (1999b), a tarefa de pré-processamento de dados se constitui em uma fase de extrema importância e inclui:

- Desenvolver um modelo de dados dos arquivos de *log*.
- Desenvolver técnicas de filtragem e limpeza dos dados para eliminar itens irrelevantes.
- Agrupar páginas individuais em unidades semânticas tais como transações.
- Interligar vários tipos de fontes de dados, tais como os obtidos via *scripts* CGI, *log* de servidores Web, agentes, *cookies*, informações previamente cadastradas, etc.
- Identificar o usuário e identificar a transação.

A utilização de técnicas que permitam a limpeza de dados provenientes de arquivos de *log* dos servidores é um item importante em qualquer tipo de análise dos *logs* da Web e não somente para a mineração de dados.

No caso dos arquivos de *log* provenientes dos servidores, como o protocolo HTTP solicita requisições separadas para cada arquivo requisitado ao servidor Web, várias entradas são registradas e muitas destas não têm utilidade pois não são significativas para rastrear o comportamento do usuário. Incluem-se neste caso entradas com arquivos de sufixos tais como jpg ou gif, as quais representam arquivos requisitados pelas páginas HTML. Estas entradas irrelevantes devem ser excluídas.

2.2.6 Identificação de uma transação

O conceito do que realmente significa uma transação pode assumir vários aspectos. Uma simples transação pode referenciar várias páginas ou várias transações podem compor uma única página.

Como apresentado por Cooley *et al*. (1999b), o objetivo de se identificar uma transação é criar conjuntos significativos de informações sobre cada usuário. A tarefa de se identificar transações pode tanto dividir grandes transações em pequenas transações como agrupar pequenas transações em grandes. Este processo pode ser desenvolvido em vários passos, buscando-se identificar transações que permitam a aplicação consistente de técnicas de mineração de dados.

Murtagh e Tao (2000) propõem um modelo de transação o qual interpreta os registros de acessos do usuário e, baseado no período de tempo decorrido e na identificação do usuário, define uma transação.

Cooley *et al.* (1997b) consideram que a seqüência dos dados correspondente aos registros de acessos às páginas Web não é representativa do comportamento do usuário, pois *caches* e *proxies* podem distorcer a análise. Também considera que nem todas as páginas são significativas, pois é necessário fazer uma distinção entre páginas acessadas apenas com propósito de navegação e páginas acessadas para visualização de conteúdo. Além disso, uma página pode ser navegacional para um usuário e ser de conteúdo para outro.

Cooley *et al.* (1997) apresentam três métodos que podem ser usados para identificar uma transação: duração da referência, referência posterior máxima e janelas de tempo.

O conceito de "duração de referência" da transação pode ser considerado como o total de tempo que o usuário gasta na página. Este tempo é utilizado para definir se a página é de conteúdo ou navegacional. Uma página navegacional é aquela utilizada com o propósito de achar *links* para a informação procurada, ou seja, é uma página de passagem. Página de conteúdo é a que contém a informação. Espera-se que o tempo de acesso a uma página navegacional seja menor que o tempo de referência de uma página de conteúdo.

Este método de identificação por duração de referência calcula o tempo despendido em uma página pela diferença entre o tempo da página corrente e a chamada da próxima. No entanto, este algoritmo traz um problema referente ao registro de tempo da última página, pois este não pode ser calculado, uma vez que não existe a próxima página. Neste caso é assumido que a última página é sempre de conteúdo. Esta premissa pode introduzir erros.

O método de referência máxima posterior considera que uma transação é definida como uma seqüência de páginas visitadas em um caminho, desde a primeira página visitada até a última página antes de uma referência reversa. Uma referência reversa é uma página pertencente ao caminho já visitado. Uma nova transação é iniciada quando uma nova página ainda não visitada é acessada. Esta nova página é denominada referência posterior.

A última página acessada antes de se iniciar o processo reverso é chamada de referência posterior máxima e é considerada como uma página de conteúdo.

O método de janela de tempo simplesmente divide a sessão em intervalos com duração determinada por um parâmetro. Não é possível diferenciar entre páginas de conteúdo ou navegacional.

O sistema WebMiner (Cooley *et al*., 1997) permite a escolha de uma ou várias das formas de identificação de transações citadas acima. Este sistema inclui a preparação dos dados, identificação de transações, integração de fontes de dados, formatação e disponibilização para consultas por meio de uma interface.

A ferramenta WUM (*Web Utilization Miner*) apresentada por Spilioupoulou e Faulstich (1998) assume que consecutivos acessos provenientes da mesma máquina durante um certo período de tempo pertencem ao mesmo usuário. Uma vez definida a origem de cada acesso, estas entradas são agrupadas conceitualmente como uma transação. A definição de uma nova sessão obedece a dois critérios: uma nova sessão começa quando a duração de um grupo de requisições excede um tempo estipulado ou quando o tempo entre dois acessos consecutivos excede o tempo previsto.

2.2.7 Identificação do usuário

É importante compreender o porquê da necessidade de identificação de um usuário na Web. Esta identificação justifica-se pelo fato de que os usuários que analisam os dados desejam identificar a seqüência de um usuário individual em seu navegador dentro e fora de um sítio de forma a definir padrões de comportamento do usuário que possam ser usados para análise ou previsão de uso. Para este tipo de pesquisa nem sempre é importante definir

a identidade de um usuário (seu nome, por exemplo), mas apenas identificar que um conjunto de ações pertencem ao mesmo usuário (Broder, 2000).

Por outro lado, pode haver pessoas interessadas em saber não o que um usuário faz mas quem o usuário é. Neste caso é interessante ter acesso a um perfil individual do usuário onde podem constar nome, endereço, telefone e outros dados pertinentes à natureza do conhecimento que se deseja buscar.

O endereço IP é o elemento principal de identificação de um computador na Internet. No entanto, este endereço não identifica a pessoa, mas sim a máquina na qual o usuário está.

Cooley *et al.* (1999b) apresentam as dificuldades de se identificar um usuário a partir dos arquivos de *log* dos servidores Web. Dentre os problemas, o autor cita a existência de *cache* local dos navegadores e os servidores *proxy.* A *cache* local possibilita que quando o usuário aperte o botão "voltar" a página seja novamente exibida sem que este acesso seja registrado no *log* do servidor. Já os servidores *proxy* introduzem um nível intermediário de *cache* e para o arquivo de *log* do servidor todas as requisições provenientes do *proxy* têm o mesmo identificador, mesmo que sejam de diversos usuários.

Na maioria dos ambientes de apoio a educação a distância este problema não existe, uma vez que o usuário normalmente tem acesso ao sistema por meio de uma identificação e de uma senha.

2.3 Ambientes de apoio a cursos a distância

Vários ambientes de apoio a cursos a distância têm sido desenvolvidos, tanto no meio acadêmico como no âmbito comercial. Esses ambientes possuem uma estrutura pré-

definida onde a criação do curso pode ser feita por usuários não especialistas em computação. A seguir são apresentados alguns ambientes desenvolvidos, suas características principais e os métodos utilizados para o processo de avaliação do ambiente.

2.3.1 WebCT

O WebCT é um programa para criação de ambientes educacionais pela Internet desenvolvido pela Universidade de Bristish Columbia, Canadá (Goldberg & Salari, 1997) e atualmente comercializado pela empresa WebCT *inc.* (www.webct.com). Este ambiente permite criar cursos a distância ou disponibilizar material complementar como apoio aos cursos presenciais. Não exige que o usuário possua grande experiência técnica em computadores seja do lado do desenvolvedor ou do aluno.

O WebCT reside em um servidor permitindo que o estudante o acesse via navegador da Web. Uma vez instalado em um servidor, o WebCT pode abrigar inúmeros cursos e treinamentos. Tem interface usável e é dotado de um conjunto de ferramentas que facilitam o aprendizado, a comunicação e a colaboração. Também possui um conjunto de ferramentas administrativas que auxiliam o professor na entrega, manutenção e desenvolvimento do material para o curso.

O WebCT possui quatro classes de usuários: administrador, desenvolvedor (professor), monitor e aluno. Cada classe trabalha com funções previamente definidas (Franco & Barbetti,1998).

O administrador é o responsável por inicializar e remover cursos além de alterar senhas dos desenvolvedores. Não é permitido ao administrador alterar conteúdo dos cursos ou mesmo alterar senhas dos estudantes.

O desenvolvedor é o instrutor do curso. Ele poderá manipular o conteúdo, criar provas, controlar freqüência, acompanhar o desempenho dos alunos e criar contas para os estudantes.

O monitor é um usuário que tem os mesmos privilégios do aluno, porém pode acessar as avaliações, testes e dados sobre o desempenho dos estudantes.

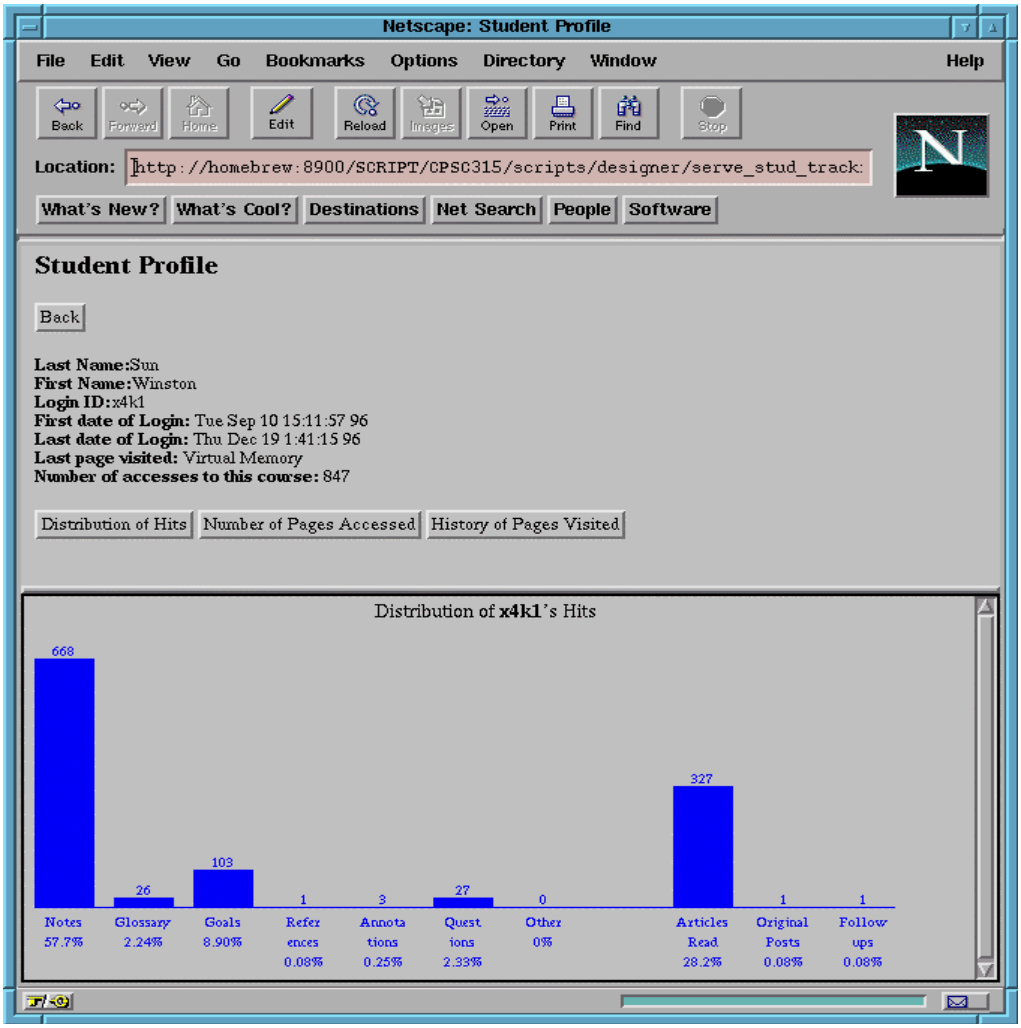

Figura 2.3 - Página de visualização de acessos do WebCT

Fonte: Goldberg & Salari, 1997

O aluno é quem acessa o material do curso. Ele não pode alterar o conteúdo do curso. Deve possuir uma conta e senha. Para o estudante são disponibilizadas ferramentas de comunicação como correio eletrônico, *chat* e listas de discussão.

Como parte das ferramentas de administração, o WebCT permite monitorar e avaliar o estudante. A ferramenta permite acessar um resumo onde são informados: primeiro acesso, último acesso e número total de acessos. Também é disponibilizada uma página com o perfil do estudante. Nesta página, além das informações já incluídas no resumo, tem-se também informações mais detalhadas tais como: distribuição de acessos, número de páginas acessadas em relação ao número total de páginas que deverá acessar, histórico das páginas visitadas e o tempo gasto em cada acesso. Estas informações podem ser visualizadas através de uma interface gráfica. A Figura 2.3 apresenta uma página relativa à distribuição de acesso para um determinado aluno.

Também é mantido um histórico do desempenho do aluno nas questões relacionadas às avaliações. Estas questões são armazenadas em um banco de dados podendo ser do tipo verdadeiro/falso, múltipla escolha ou mesmo resposta curta. Em qualquer um dos tipos citados é provido um retorno imediato ao estudante. Todo acesso é permitido somente pelo nome do usuário e senha.

2.3.2 AulaNet

AulaNet é um ambiente que permite a criação, aplicação e administração de cursos na Web. Foi desenvolvido no Laboratório de Engenharia de Software do Departamento de Informática da PUC-Rio. O AulaNet baseia-se nas relações de trabalho cooperativo onde as interações entre os membros do grupo são mais valorizadas que o estudo individual dos conteúdos do curso (Fuks *et al.,* 2001). É um software distribuído gratuitamente a qualquer instituição pública ou privada.

O AulaNet considera os seguintes atores envolvidos no processo de criação e aprendizado (Aulanet, 2003):

Administrador: Trata de questões de natureza operacional, como inscrição de docentes, admissão de aprendizes em cursos, etc. Atua como facilitador da integração entre docente, curso e aprendiz.

Aprendiz: Representa o público alvo para quem o curso se destina, ou seja, é o usuário final.

Coordenador: Participa desde a descrição inicial do curso até a entrada dos conteúdos do mesmo. É o criador do curso. Pode contar ou não com o auxílio de um docente co-autor.

Docente co-autor: Responsável por ajudar o coordenador na criação e fornecimento de conteúdos educacionais para um curso.

Mediador: Responsável pela aplicação do curso.

Os cursos oferecidos pelo AulaNet são dotados de elevado grau de interatividade com a participação intensiva do estudante com o professor, com os colegas e com o material didático. Essa participação pode ser realizada pelos vários mecanismos oferecidos pelo ambiente. O C*ontato com Docentes* é um destes mecanismos e permite que os aprendizes se comuniquem diretamente com os docentes por mensagens. Já as C*onferências* permitem a comunicação entre participantes por meio de uma discussão estruturada. Todas as mensagens ficam armazenadas no ambiente. O AulaNet conta também com a *Lista de Discussão* que possibilita a cada participante do curso enviar uma mensagem diretamente para todos os outros participantes; com o *Mecanismo de Debate* que provê comunicação por *chat* e com *Mensagem para os Participantes* onde é possível a comunicação síncrona por meio de mensagens entre os participantes do curso.

O AulaNet separa o conteúdo da navegação. Desta forma, é possível a migração dos conteúdos desenvolvidos para o ambiente AulaNet para outros ambientes similares. Não existem ferramentas de autoria no AulaNet. O conteúdo pode ser elaborado por qualquer editor de textos que o usuário preferir.

Em relação ao processo de coordenação do curso, o AulaNet oferece os seguintes mecanismos: *Avisos, Plano de Aulas*, *Tarefas*, *Avaliação* e *Acompanhamento da Participação*. O mecanismo de *Avisos* permite a divulgação de eventos, o *Plano de Aulas* permite organizar conteúdos de um curso em aulas, o mecanismo de *Tarefas* permite submeter trabalhos ou exercícios para os aprendizes resolverem e a *Avaliação* possibilita a criação de exames para a avaliação dos alunos. O mecanismo de *Acompanhamento da Participação* permite o acompanhamento e avaliação das contribuições dos aprendizes durante o curso que pode ser visualizado em um relatório.

Para o suporte à avaliação informal, conforme discutido na Seção 2.1, Menezes *et al.* (1998) propõem um modelo para o ambiente AulaNet baseado em agentes assistentes de tarefas. Estes agentes seriam responsáveis pelo monitoramento do processo de interação do aluno com o ambiente, registrando quais os caminhos percorridos, quais as fontes consultadas, qual a sua contribuição em atividades conjuntas ou mesmo qual sua assiduidade em tarefas em grupo (*chats*, videoconferências, etc).

No entanto, o uso de agentes como proposto por Menezes *et al.* não foi o caminho escolhido pelos desenvolvedores. Em trabalhos mais recentes, Fuks *et al.* (2002) demonstram como o uso da categorização e da estruturação de mensagens em ferramentas de comunicação textuais assíncronas pode ser utilizada para melhoria da qualidade do trabalho do grupo e também para fornecer subsídios para classificação e agrupamento automático das mensagens por meio de relatórios que os docentes utilizam para acompanhar a participação dos estudantes nas discussões ou mesmo identificar seus elementos centrais. A Figura 2.4 apresenta um relatório de avaliação de participação em debates.

Figura 2.4 - Relatório administrativo de participação em debates
Fonte: suporte on-line Aulanet em 18/06/2003

A categorização de mensagens foi implantada no AulaNet em 2001 e observou-se maior participação dos estudantes e melhoria da qualidade das contribuições. Esta solução segundo o autor veio ao encontro da premissa do AulaNet que incentiva metodologias de avaliação que preconizem a produção dos aprendizes interagindo em grupo por meio de debates, palestras e conferências.

2.3.3 TelEduc

O TelEduc é um ambiente de suporte para ensino e aprendizagem em desenvolvimento pelo Instituto de Computação e pelo Núcleo de Informática Aplicada à Educação da Unicamp desde 1997 (Teleduc, 2004).

A metodologia proposta pelo TelEduc pressupõe a proposição gradativa de atividades a serem resolvidas pelos alunos, isto é, as atividades propostas abrangem todo o conteúdo do curso de forma incremental.

A ferramenta Agenda é um componente fundamental no TelEduc pois por meio dela todo o processo de aprendizagem é organizado. Ela apresenta a programação de um período do curso onde o professor pode descrever os objetivos a serem alcançados, as atividades a serem planejadas e os recursos disponibilizados ao estudante para alcançar os objetivos no período. Além desta ferramenta várias outras como: Atividades, Material de Apoio, Leituras, Fóruns de Discussões, Bate-Papo e o Mural, foram criadas para apoiar o aprendizado. O TelEduc dispõe ainda de ferramentas de autoria e gerenciamento do curso.

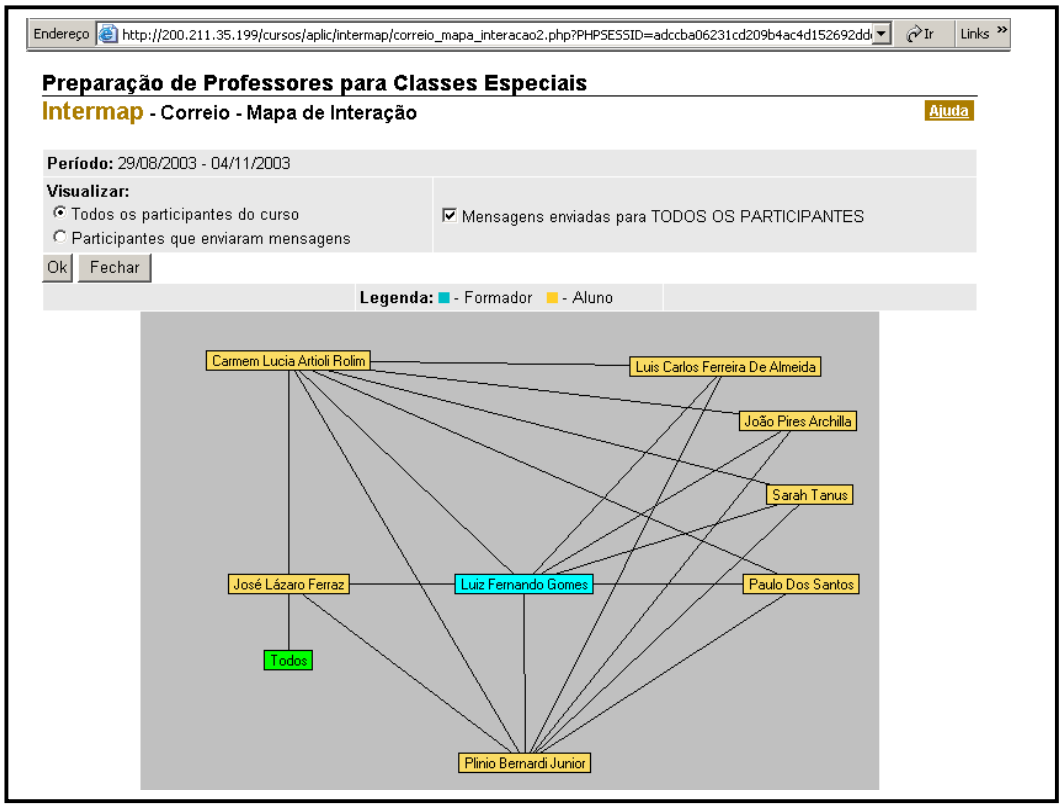

Figura 2.5 - Grafo criado pela ferramenta InterMap
(fonte: Uniso[2])

[2] Universidade de Sorocaba. Página cedida pelo administrador do curso em Nov/2004.

Todas as ferramentas de comunicação do TelEduc possuem registro das interações (Otsuka *et al.,* 2002). Este processo auxilia a avaliação formativa, pois permite a análise e orientação contínua das participações dos aprendizes durante as atividades, sejam elas individuais ou em grupo.

Os dados quantitativos das interações realizadas no ambiente podem ser visualizados pela ferramenta InterMap (Romani, 2000), que permite a visualização do fluxo de mensagens de correio trocadas entre os participantes apresentada na forma de um grafo ou de tabelas com cores diferenciadas (Figura 2.5).

A ferramenta Acessos também permite o acompanhamento do curso pela geração de relatórios contendo o número de acessos, data e hora do último acesso de cada participante ao curso, a freqüência de acessos de cada participante e os acessos a cada ferramenta do ambiente. A Figura 2.6 apresenta o relatório de acessos gerado pela ferramenta.

Embora o Teleduc permita o registro de todas as interações dos alunos ao longo do curso, a forma como estas informações são tratadas e apresentadas não facilita a análise de dados qualitativos das participações dos alunos (Lachi *et al.*, 2002). Para suprir esta deficiência, três projetos estão em andamento no TelEduc. O *redesign* das ferramentas é um deles e visa facilitar o registros das avaliações e sua posterior recuperação e análise. Um sistema baseado em agentes de interface constitui o segundo projeto e tem como finalidade o suporte à análise e seleção de mensagens relevantes em sessões de bate-papo. O terceiro projeto engloba os anteriormente citados e visa "prover suporte às funções principais desempenhadas pelo professor no processo de avaliação *on-line* formativa" (Lachi *et al.*, 2002).

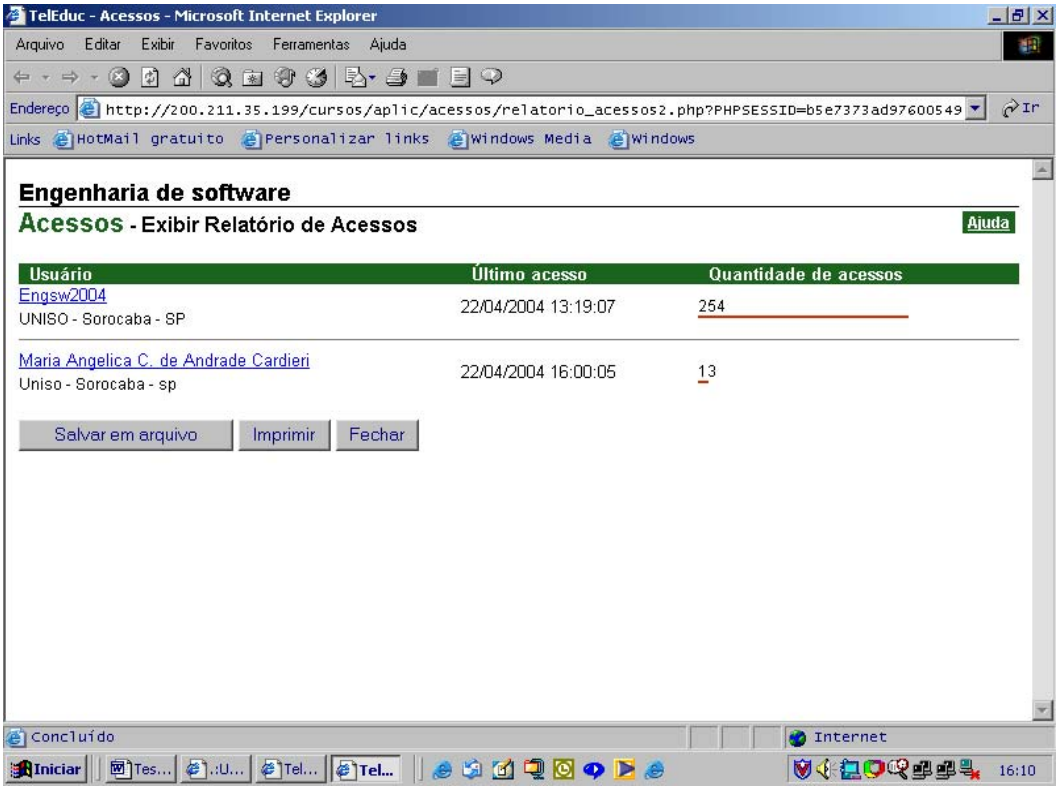

Figura 2.6 - Relatório gerado pela ferramenta Acessos (fonte: Uniso)

2.4 Técnicas de aquisição de dados em ambientes de apoio a EDMC

Na seção anterior foram apresentados três ambientes de apoio ao desenvolvimento de cursos a distância. Em todos, há uma grande preocupação em oferecer ferramentas que permitam o acompanhamento do progresso do estudante, pelo rastreamento das ações do aluno em atividades individuais ou colaborativas. Para a implementação destas funcionalidades é necessário a existência de um mecanismo de aquisição de dados e armazenamento.

Na Seção 2.2, foram discutidas algumas técnicas que podem ser usadas para se implementar uma camada de aquisição dos dados em aplicações na Web. As técnicas

citadas incluem a utilização de arquivos de *log* gerados pelos servidores (*http log files*), *cookies,* CGI ou agentes. Nos ambientes analisados algumas destas técnicas são utilizadas.

O ambiente WebCT, por exemplo, utiliza um servidor HTTP Apache e as páginas da aplicação são escritas utilizando HTML e JavaScript (Locatelli, 2004). Os dados são manipulados por CGI com programação na linguagem Perl. O sistema de rastreamento das atividades do aluno permite a identificação do progresso do estudante, porém informa dados quantitativos como percentual do tempo gasto em acessos a conteúdo, glossário, referências, etc. Há também um histórico de páginas visitadas e informações sobre o número de acessos, tempo total e médio. Não há a visualização detalhada das ações executadas pelo estudante que permita uma avaliação qualitativa.

A arquitetura do AulaNet é baseada em um modelo cliente-servidor na Web e foi desenvolvida utilizando a tecnologia Scriba, que é um *servlet* para intermediar a comunicação do cliente com o servidor (Blois *et al.,* 1999). O Scriba é uma ferramenta que permite a criação de páginas HTML dinâmicas e a definição de classes em Java voltadas a aplicações na Web. Esta tecnologia oferece uma linguagem própria, que pode ser embutida em páginas HTML e possibilita a invocação de métodos de classes escritas em Java e o acesso a banco de dados através de uma ponte JDBC-ODBC. O Scriba tem funcionalidades semelhantes ao JSP (Java Server Pages) desenvolvido pela Sun Microsystems, o qual ainda não havia sido lançado quando o Scriba foi desenvolvido. O servidor utilizado é o IIS (Internet Information Service) da Microsoft ou Apache (Apache Organization, 2004).

A versão atualmente disponível do AulaNet é a 2.0. Segundo Fucks *et al*. (2003), muitas funcionalidades foram acrescentadas à versão atual na medida em que se tornavam necessárias, sendo agora preciso uma reestruturação do código. Outro fator que justifica uma nova versão é o fato do Scriba ser uma tecnologia proprietária, o que dificulta a integração de novos membros à equipe de desenvolvimento e por conseqüência o desenvolvimento de novos produtos.

Fucks também considera como um grave problema a disseminação de código em arquivos HTML e classes Java. Isto configura uma arquitetura de baixa modularidade pois as classes Java são específicas de cada página e se comportam como uma biblioteca de funções, apesar de terem sido desenvolvidas em uma linguagem orientada a objetos. Uma nova versão, o AulaNet 3.0, está sendo estudada, a qual deve utilizar técnicas de desenvolvimento baseado em componentes visando maior flexibilidade e escalabilidade.

Quanto às ferramentas para acompanhamento das atividades dos alunos, o AulaNet se baseia em uma abordagem cooperativa portanto não possui ferramentas para acompanhamento individual da navegação entre as páginas do curso e sim acompanhamento das interações do indivíduo com o grupo como por exemplo o Relatório de Participação em Debates mostrado na Figura 2.4.

O Teleduc, a partir da versão 3.0, foi desenvolvido em linguagem PHP, utiliza MySQL para o armazenamento de dados e o servidor HTTP Apache (Teleduc, 2004). Quanto ao rastreamento das atividades, o Teleduc possui ferramentas para a visualização dos registros das interações entre os participantes de um curso. As ferramentas InterMap e Acessos auxiliam a análise quantitativa das interações. Porém, para a análise qualitativa os dados apresentados por estas ferramentas ainda não são ideais pois demandam muito tempo e trabalho dos formadores (Otsuka *et al*., 2002). A versão atual do ambiente Teleduc (v. 3.3.3) já incorpora outros mecanismos de avaliação que não foram analisados neste trabalho.

2.5 Considerações finais

A literatura indica que a educação a distância é uma área em expansão. Muitas pesquisas têm sido feitas em direção a melhoria de ambientes educacionais para esse tipo

de aplicação. Estas melhorias agregam necessidades distintas tais como facilidade de uso por indivíduos não especialistas em computação, facilidade em reproduzir recursos próximos aos disponíveis em salas de aulas presenciais, tais como atividades colaborativas, interações entre grupos de discussão, mecanismos de controle e acompanhamento do aluno e análise da eficácia de um curso.

Vários ambientes para apoiar o desenvolvimento de cursos a distância têm sido desenvolvidos e três deles foram analisados aqui: Aulanet (Aulanet, 2003), Teleduc (TelEduc, 2004) e WebCT (Goldberg, 1997). Todos esses ambientes possuem um conjunto de ferramentas que permitem a criação de cursos, suporte para o professor, controle de tarefas do aluno, recursos de comunicação e algum tipo de controle sobre sua utilização. No entanto as ferramentas são agregadas ao ambiente, inviabilizando sua utilização em cursos desenvolvidos de forma externa à ferramenta ou mesmo outro tipo de aplicação baseada na Web. Muitos destes ambientes de apoio ainda estão em desenvolvimento e melhorias são previstas em quase todos, principalmente em relação ao mecanismo de monitoramento das ações executadas pelo usuário. No Aulanet, a ênfase é dada nas interações entre os grupos, isto é, em um trabalho cooperativo e não individual. No ambiente Teleduc projetos estão em desenvolvimento visando aperfeiçoar as técnicas de registro e análise de dados referentes às interações.

Ferramentas comerciais tais como Webalizer (Barret, 2001) e WebTrends (Webtrends Corporation, 2001) geram apenas dados estatísticos para análise quantitativa e não são apropriadas para avaliar as ações do usuário como desejado em um ambiente de EDMC.

O rastreamento das ações executadas pelo aluno, proveniente de interações com o material instrucional, são informativos importantes para revelar o comportamento do aluno, indicar elementos pedagógicos ou mesmo auxiliar o aluno no processo de aprendizagem pela personalização ou mesmo um simples processo de ajuda ou motivação. A informação

sobre o tempo despendido em cada página pode indicar o grau de dificuldade do aluno na execução das tarefas (Sapiens, 2001).

Em se tratando de aplicações fora do contexto educacional, este rastreamento também é importante, pois pode fornecer indicadores importantes sobre a construção das páginas sobre preferências e interesses do usuário, prover personalização ou mesmo prevenir desorientação do usuário no sítio (Spilioupoulou *et al.,*1999).

Vários aspectos estão relacionados à obtenção destes dados provenientes do uso de aplicações na Web, desde a identificação do usuário, da ação executada, a forma de extração da informação, preparação dos dados para uso e sua visualização.

Portanto, devido à complexidade associada ao processo de avaliação dos ambientes computacionais via Web, em suas diversas dimensões, e visando independência em relação aos ambientes de suporte a EDMC ou mesmo suprir as deficiências das ferramentas comerciais já existentes, identifica-se a necessidade do desenvolvimento de mecanismos mais flexíveis que permitam criar uma base de dados confiável para análise destes ambientes.

Capítulo 3 - Ferramenta para avaliação analítica do curso Read in Web

Procurando obter informações sobre as reais necessidades de um ambiente educacional mediado por computador, assim como encontrar soluções suportadas por recursos tecnológicos que atendessem as necessidades apontadas, foi conduzido um estudo de caso preliminar.

O curso de ensino a distância denominado Read in Web, desenvolvido pelo Instituto de Estudos da Linguagem (IEL) da Unicamp sob a coordenação da Prof[a] Denise Bértoli Braga (Sapiens, 2001), foi utilizado como base para esta fase de estudo. A utilização deste curso como caso de estudo veio ao encontro da proposta do trabalho aqui relatado pois uma das necessidades identificadas no projeto Read in Web (ReadWeb, 2003) foi a de ferramentas que permitissem compilar e analisar os dados sobre a utilização do material que corresponde ao curso. Esta necessidade motivou o início dos trabalhos descritos nas próximas seções.

3.1 O Curso Read in Web

O Read in Web é um material auto-instrucional (Sapiens, 2001) projetado para atender aos alunos de pós-graduação da Unicamp visando ajudá-los a melhorar sua proficiência em língua inglesa para fins acadêmicos. Este curso está disponibilizado desde Fevereiro de 2000 pelo IEL da Unicamp e se encontra em uso atualmente, tendo sido testado e avaliado por vários alunos desde sua implantação.

O material do curso Read in Web utiliza uma orientação reflexiva e construtivista, onde as atividades propostas têm como meta, "conscientizar os alunos das várias

dificuldades que enfrentam durante a leitura de textos acadêmicos em língua inglesa e promover o desenvolvimento de estratégias que auxiliem o aluno a transpor tais dificuldades" (ReadWeb, 2003).

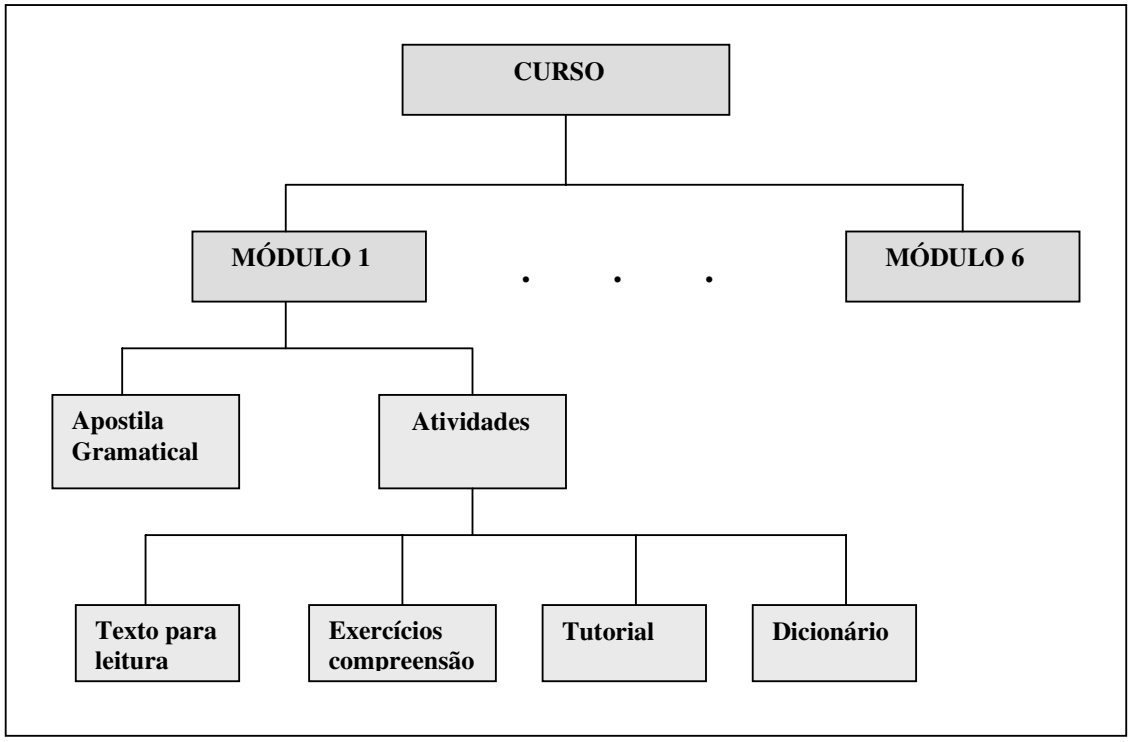

Figura 3.1 - Estrutura parcial do curso Read in Web

O conteúdo programático do curso é segmentado em seis módulos, onde os três primeiros discutem estratégias de leitura e questões lingüísticas e os demais tratam de questões importantes para compreensão no nível textual e discursivo. Cada módulo é composto por uma apostila gramatical e por um conjunto de atividades. A chamada "aula virtual" corresponde a cada uma destas atividades. A organização destas atividades pode ser verificada na Figura 3.1 e consiste em: um texto para leitura, um conjunto de seis exercícios de compreensão com o gabarito correspondente, um tutorial e um dicionário que inclui um glossário.

O material que compõe o curso foi estruturado como um hipertexto. Portanto a navegação pelas páginas, assim como o tempo despendido em cada uma delas, torna-se uma informação importante para a avaliação dos aspectos pedagógicos envolvidos na construção do material.

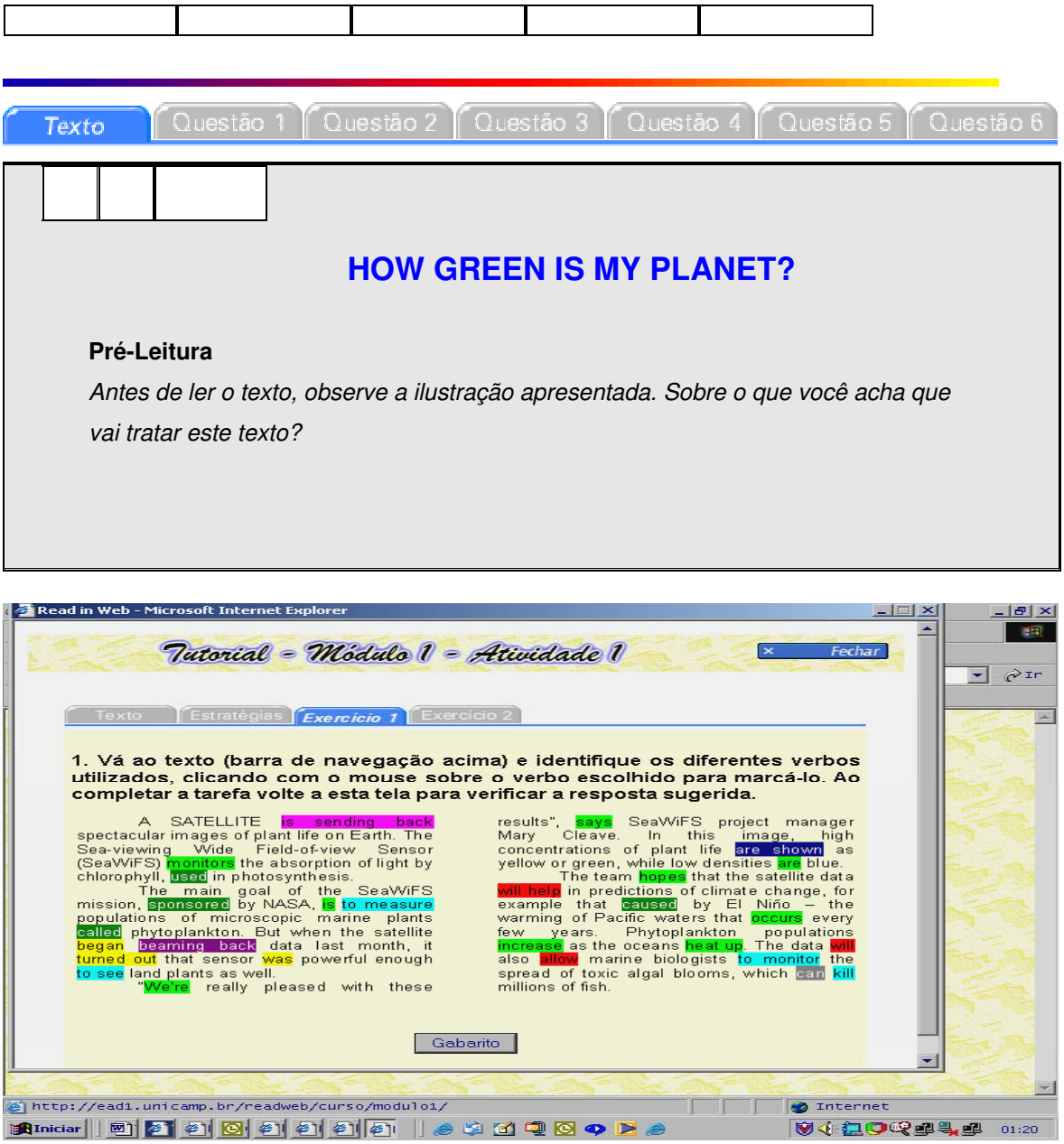

Figura 3.2 - Páginas do curso Read in Web

No aspecto estrutural, podem ser avaliadas as relações entre os segmentos dos módulos ou entre os módulos. É possível também investigar se fatores subjetivos tais como o tipo de motivação, estilo de aprendizagem e nível de conhecimento prévio do aluno, estão de acordo com a concepção do autor.

A Figura 3.2 representa duas páginas disponibilizadas pelo Read in Web onde podemos notar a estrutura hierárquica. Neste caso estão selecionadas as opções: módulo 1, atividade 1 e tutorial.

3.2 Requisitos

Os requisitos que envolveram o mecanismo desenvolvido neste estudo apoiaram-se em elementos tecnológicos e operacionais. Em relação à parte tecnológica, desejava-se que se utilizasse tecnologia voltada a Web para ser utilizado em ambientes de ensino a distância ou a qualquer outra aplicação na Internet, porém que não aumentasse o tempo de processamento de forma significativa. Além disso, algumas restrições foram impostas: a construção do material didático não deveria ser dificultada, o custo de desenvolvimento a ser acrescentado não deveria impossibilitar sua implementação e o mecanismo deveria ser portável. Quanto à parte operacional, buscou-se facilidade de aplicação e que sua utilização fosse transparente ao usuário final.

É importante ressaltar que os autores do curso não desejavam utilizar os ambientes de gerência descritos anteriormente (Seção 2.3), pois estes pressupõem um modelo pedagógico, com ênfase na colaboração entre alunos, diferente do adotado no projeto, de caráter auto-instrucional.

Em uma primeira abordagem, esta pesquisa procurou atender as necessidades acima descritas, desenvolvendo-se uma ferramenta que viabilizasse a análise e o acompanhamento do curso de forma imediata uma vez que se fazia premente este tipo de controle.

Conforme relatado pela equipe do Read in Web, era de extrema importância o acompanhamento das ações dos estudantes relativas ao uso do material instrucional, pois o registro eletrônico do percurso de navegação pelas páginas, assim como o tempo despendido permitiria investigar os pressupostos pedagógicos que fundamentaram a elaboração do material, ou mesmo validar algumas questões teóricas sobre ensino e aprendizagem. Verificando-se a seqüência de utilização do material, esperava-se ser possível a caracterização do aluno, o que permitiria avaliar predições teóricas da construção do material. Neste caso, o interesse era verificar se o aluno realizaria a seqüência de ações previstas, isto é: ler o texto, consultar o dicionário ou gramática, responder as questões e, por último, visualizar as respostas. Uma vez que a seqüência de utilização não é controlada pelo Read in Web, o aluno poderia seguir caminhos diversos definindo com isso interesses pedagógicos e estilos de aprendizado diferentes. Na situação em que se encontrava, o curso não continha forma de controle que satisfizesse os requisitos descritos.

3.3 Trabalho Realizado

A primeira parte da pesquisa consistiu em pesquisar ferramentas disponíveis no mercado que fossem capazes de fornecer informações sobre o acesso às páginas Web. Foi analisada a hipótese de se utilizar as ferramentas Webalizer (Barret, 2001) ou WebTrends (WebTrends Corporation, 2001). A maioria destas ferramentas utiliza o arquivo de *log* padrão CLF gerado automaticamente pelo servidor Web (Apache Organization, 2004). Porém, tais ferramentas apresentam informações de maneira sintética e dados estatísticos

que não permitem uma análise mais apurada e muito menos um trabalho mais criterioso de seleção e filtragem de dados.

A solução encontrada foi utilizar os arquivos de *log* padrão. A vantagem desta abordagem está na facilidade de obtenção desses arquivos, uma vez que a maioria dos servidores Web os geram. No entanto, esta opção demandou o desenvolvimento de uma ferramenta própria para limpeza e compilação dos dados (Figura 3.3).

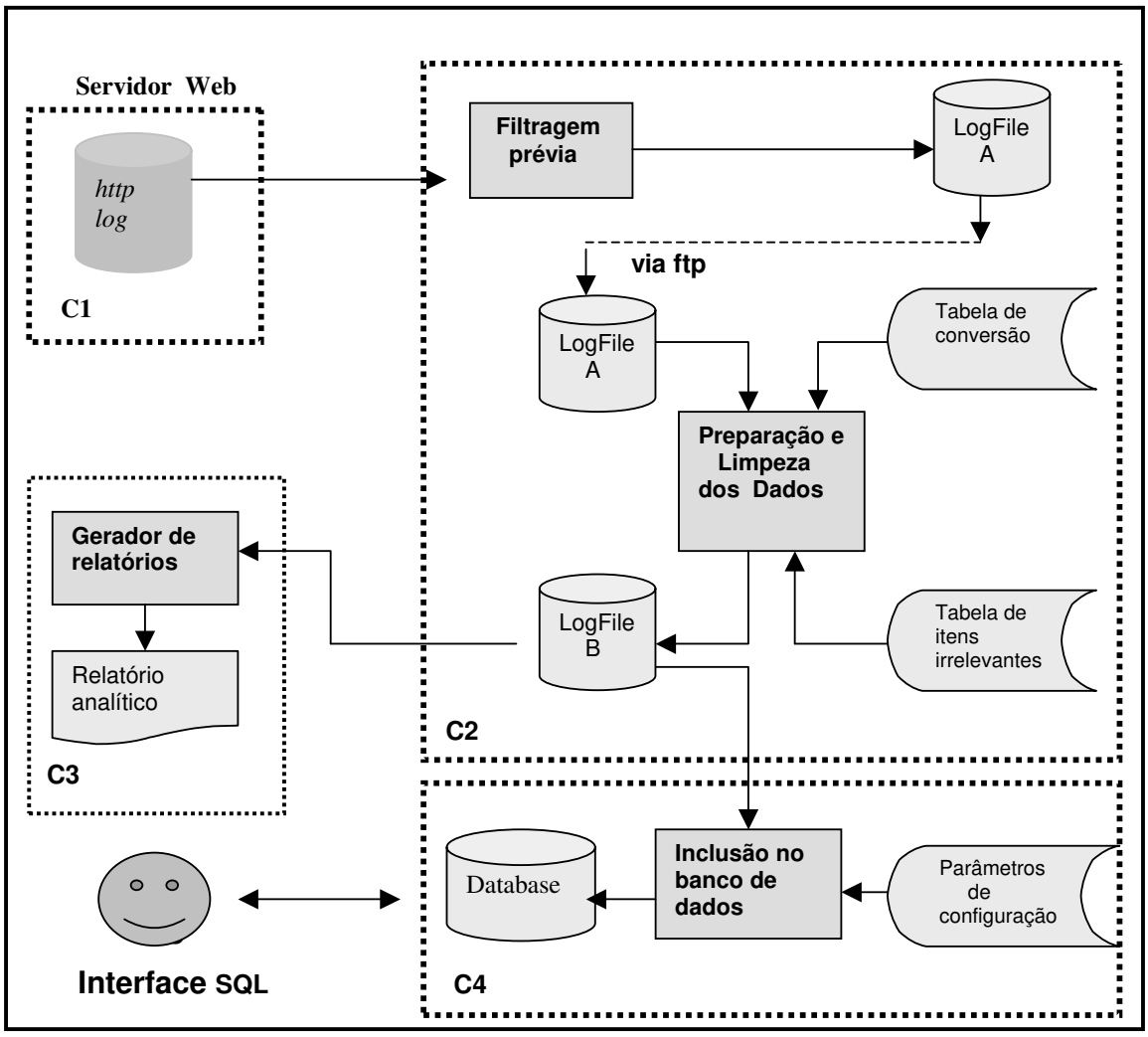

Figura 3.3 - Arquitetura do mecanismo implementado

A ferramenta desenvolvida possui os seguintes componentes:

1- Camada de aquisição dos dados (C1),

2- Camada de preparação e limpeza de dados (C2),

3- Camada de visualização (C3) e

4- Camada de persistência de dados analíticos (C4)

Cada uma destas camadas será detalhada na seqüência.

3.3.1 Camada de aquisição dos dados (C1)

Nesta solução não houve a necessidade de se implementar uma aplicação para realizar o processo de aquisição, uma vez que foi utilizado o arquivo de *log* do servidor Apache (*access.log*) localizado em uma máquina do Centro de Computação da Unicamp.

Como este servidor não era exclusivo do curso Read in Web, existia uma grande quantidade de linhas com chamadas a arquivos os quais não se relacionavam com o curso, sendo portanto irrelevantes para o objetivo final. Para reduzir a quantidade de informações desnecessárias foram implementados os passos da camada de preparação e limpeza de dados descrita a seguir.

3.3.2 Camada de preparação e limpeza de dados (C2)

Normalmente, uma grande variedade de arquivos é acessada como resultado de uma requisição feita por um cliente para visualizar uma determinada página na Web. Estes arquivos podem incluir referências a arquivos de som, imagem, vídeo, executáveis e HTML. A requisição pelo usuário de uma única página pode ocasionar a gravação de

várias entradas nos arquivos de *log* do servidor além da entrada relativa à chamada da página. As entradas consideradas válidas dependem da natureza da aplicação. Na maioria das vezes, as extensões JPEG, JPG, GIF e MAP são consideradas irrelevantes pois normalmente se referem a arquivos que são chamados automaticamente quando uma página é carregada, o que os torna sem utilidade para a descoberta da seqüência de navegação.

A Figura 3.4 apresenta um trecho do *http log file* da forma como foi gerado originalmente pelo servidor Apache, antes do processo de limpeza, onde podemos verificar linhas com informações irrelevantes (por exemplo, arquivos com extensão GIF, sublinhadas no texto apenas para melhor visualização).

O processo de filtragem dos dados (*data cleaning*) é uma tarefa importante em qualquer aplicação que tenha como objetivo uma análise de dados para posterior extração de conhecimento.

```
deq05.ifi.unicamp.br - psmelo [29/Jun/2001:17:53:44 -0300] "GET
/iel/readWeb/curso/modulo4/p2.htm HTTP/1.0" 200 31855
deq05.ifi.unicamp.br - psmelo [29/Jun/2001:17:53:46 -0300] "GET
/iel/readWeb/curso/modulo4/p2.htm HTTP/1.0" 200 31855
deq05.ifi.unicamp.br - psmelo [29/Jun/2001:17:53:46 -0300] "GET
/iel/readWeb/curso/modulo4/p2.htm HTTP/1.0" 200 31855
deq05.ifi.unicamp.br - - [29/Jun/2001:17:53:47 -0300] "GET
/iel/readWeb/curso/modulo4/../../pictures/pic33.gif HTTP/1.0" 200
494
deq05.ifi.unicamp.br - - [29/Jun/2001:17:53:47 -0300] "GET
/iel/readWeb/curso/modulo4/../../pictures/pic34.gif HTTP/1.0" 200
1301
deq05.ifi.unicamp.br - psmelo [29/Jun/2001:17:53:47 -0300] "GET
/iel/readWeb/curso/modulo4/p2.htm HTTP/1.0" 200 31855
deq05.ifi.unicamp.br - psmelo [29/Jun/2001:17:54:46 -0300] "GET
/iel/readWeb/curso/modulo4/p2.htm HTTP/1.0" 200 31855
deq05.ifi.unicamp.br - psmelo [29/Jun/2001:17:58:01 -0300] "GET
/iel/readWeb/curso/modulo4/atividade2/p3.htm HTTP/1.0" 200 11546
```

Figura 3.4 - Parte de um *http log file* original

Na aplicação desenvolvida nessa primeira fase, o objetivo principal foi identificar o comportamento de cada usuário do curso Read in Web de forma isolada. Portanto, a

filtragem dos dados tornou-se imprescindível. Este processo foi realizado segundo os passos listados a seguir:

1. Execução do processo de filtragem prévia, eliminando as linhas que não pertenciam a acessos do curso pois, como já citado, não havia um servidor exclusivo.
2. Aplicação do processo de limpeza de dados por um aplicativo que seleciona apenas informações relevantes excluindo linhas sem interesse. Este aplicativo utiliza um arquivo de parâmetros onde são indicadas as extensões irrelevantes.
3. Conversão do texto de URL para uma linguagem de compreensão mais fácil pelo usuário gerando um arquivo traduzido.
4. Geração de um arquivo formatado.

O primeiro passo foi realizado pela identificação das entradas onde o nome do usuário (*username*) estava preenchido e que possuíam a seqüência de caracteres "readweb" na URL de destino. A restrição relativa à existência de um nome de usuário pôde ser realizada, pois o material instrucional do Read in Web possui a solicitação de nome e senha em sua página inicial, o que permitiu a identificação do usuário, problema este muitas vezes encontrado em ferramentas de aquisição de dados na Web como citado na Seção 2.2.7.

Procurando simplificar o processo, este primeiro passo foi realizado com ajuda de ferramentas do próprio sistema operacional Unix, o comando *grep* que permite filtrar linhas que contenham uma dada seqüência de caracteres. Neste caso, a seqüência foi "readweb", pois a URL de qualquer página de material do curso foi identificada desta forma no processo de autoria. Este passo eliminou um grande volume de informações sem interesse, pois cabe lembrar que os arquivos gerados pelos servidores Web possuem informações da ordem de gigabytes referentes às mais diversas requisições feitas pelos clientes.

Em seqüência ao processo de seleção inicial, foi aplicado um programa implementado em Java que realiza a filtragem mais apurada contemplando os passos 2 e 3 da ferramenta.

Na maioria dos processos de *data cleaning* a maneira mais simples de filtragem é a verificação dos sufixos da URL. No caso do Read in Web, as imagens não definem caminhos de navegação e apenas complementam os textos apresentados. Por este motivo, recursos com sufixos GIF, JPEG, JPG e MAP são identificados como irrelevantes e não são registrados. A relevância ou não de uma chamada a arquivos é dependente da natureza e do objetivo da aplicação. Existem casos em que o acesso às imagens é o ponto principal de investigação, como por exemplo as páginas de uma galeria de arte (Murtagh & Tao, 2000). Neste caso estas extensões devem ser registradas.

Procurando flexibilizar o software desenvolvido, foram utilizados arquivos de configuração. Tais arquivos possibilitam realizar alterações na ferramenta sem ser necessário alterar os fontes dos programas, pois estes arquivos são lidos em tempo de execução.

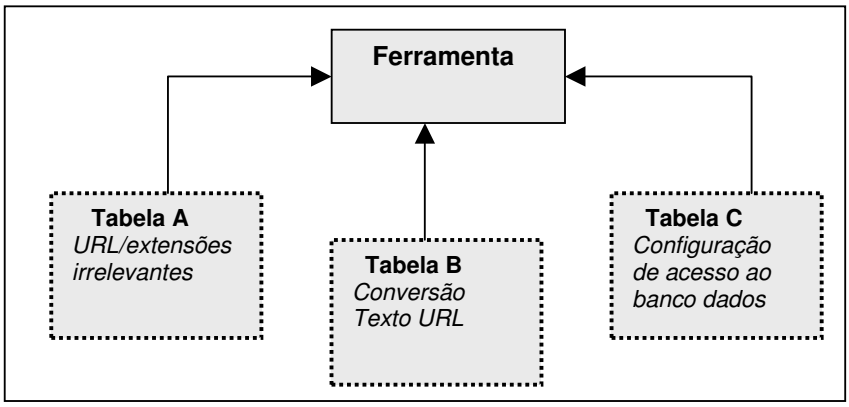

Figura 3.5 - Arquivos de configuração

A Figura 3.5 mostra um detalhe da arquitetura desta ferramenta onde podemos notar os arquivos:

A- Tabela de extensões irrelevantes

B- Tabela de conversão URL-Texto,

C- Tabela de configuração de acesso ao banco de dados.

A Tabela A tem como função indicar as extensões dos arquivos chamados pelo navegador para os quais não há interesse em registrar o acesso. A manutenção desta tabela pode ser realizada utilizando-se um editor de textos.

A Tabela B é utilizada para a conversão das *strings* de URL para o formato de um texto escolhido pelo usuário. Este processo foi incluído visando facilitar a visualização do caminho percorrido, uma vez que o arquivo original de *logs* indica a URL tal como foi requisitada ao servidor Web. Notamos que esta informação é confusa para pessoas que não participaram do desenvolvimento da parte computacional. A edição desta tabela pode ser feita por qualquer editor de texto. A coluna da esquerda mostra a URL real e a coluna da direita a conversão a ser feita. A separação entre as colunas é indicada por dois sinais de igualdade (Figura 3.6).

```
curso/modulo1/p2.htm==modulo1 apostila gramatical
curso/modulo2/p2.htm==modulo2 apostila gramatical
curso/modulo3/p2.htm==modulo3 apostila gramatical
curso/modulo4/p2.htm==modulo4 apostila gramatical
curso/modulo5/p2.htm==modulo5 apostila gramatical
curso/modulo6/p2.htm==modulo6 apostila gramatical
curso/p2.htm==teste do material
curso/p1.htm==lista de palavras de função
p1.htm==texto de leitura da atividade
p2.htm==texto do tutorial da atividade
p3.htm==dicionário da atividade
p4.htm==questão 1
p5.htm==questão 2
p6.htm==questão 3
p7.htm==questão 4
p8.htm==questão 5
p9.htm==questão 6
p10.htm==estratégias de leitura e compreensão
p11.htm==exercício 1 do tutorial
p12.htm==exercício 2 do tutorial
p13.htm==exercício 3 do tutorial
p14.htm==exercício 4 do tutorial
p15.htm==exercício 5 do tutorial
```

Figura 3.6 - Tabela B - conversão de URL

A tabela C configura o acesso ao banco de dados, sendo utilizada na camada de inserção de dados (C4) e será definida com mais detalhes mais adiante.

Após o Passo 3 o arquivo de *log* é reduzido para a estrutura mostrada na Figura 3.7. Estas informações incluem a identificação do usuário, data e hora da utilização e a URL acessada já convertida, conforme indicado na tabela de conversão. Se compararmos com o formato apresentado na Figura 3.4 e com a estrutura do curso Read in Web (Figura 3.1) fica clara a facilidade de interpretação das ações pelo professor ou autor/administrador do curso. Os asteriscos (*1, *2 e *3) são usados como separadores entre os campos.

Um exemplo de informações que podem ser extraídas deste arquivo seria a análise do comportamento do aluno. Notamos que o usuário "duarcide" acessa o texto de leitura, a seguir tenta resolver a questão 1 e volta ao texto de leitura antes de resolver novas questões. Alguns alunos poderiam tentar resolver as questões sem rever o texto lido ou sem consultar o dicionário, por exemplo. Estes dados também podem ser confrontados com a seqüência de navegação prevista pelo autor do material possibilitando a análise do material instrucional.

```
duarcide *102-06-2001:06:19:08*2modulo3/atividade1/texto de leitura da atividade*3
duarcide *102-06-2001:06:19:15*2modulo3/atividade1/questão 1*3
duarcide *102-06-2001:06:19:33*2modulo3/atividade1/texto de leitura da atividade*3
duarcide *102-06-2001:06:21:19*2modulo3/atividade1/questão 2*3
duarcide *102-06-2001:06:21:37*2modulo3/atividade1/questão 3*3
duarcide *102-06-2001:06:24:25*2modulo3/atividade1/questão 4*3
duarcide *102-06-2001:06:25:55*2modulo3/atividade1/questão 5*3
duarcide *102-06-2001:06:31:33*2modulo3/atividade1/questão 6*3
duarcide *102-06-2001:06:32:05*2modulo3/atividade1/texto do tutorial da  atividade*3
duarcide *102-06-2001:06:32:12*2modulo3/atividade1/estratégias de leitura e
compreensão*3
duarcide *102-06-2001:06:34:39*2modulo3/atividade1/exercício 1 do tutorial*3
duarcide *102-06-2001:06:38:30*2modulo3/atividade1/exercício 2 do tutorial*3
duarcide *102-06-2001:06:42:16*2modulo3/atividade1/exercício 3 do tutorial*3
duarcide *102-06-2001:06:47:21*2modulo3/atividade1/exercício 4 do tutorial*3
alnunes *102-06-2001:18:44:54*2modulo2/atividade1/texto de leitura da atividade*3
alnunes *103-06-2001:00:49:43*2modulo3/atividade1/texto de leitura da atividade*3
alnunes *103-06-2001:01:01:28*2modulo2/atividade2/texto de leitura da atividade*3
alnunes *103-06-2001:01:08:20*2modulo2/atividade2/questão 1*3
alnunes *103-06-2001:01:08:39*2modulo2/atividade2/texto de leitura da atividade*3
alnunes *103-06-2001:01:09:51*2modulo2/atividade2/questão 2*3
```

Figura 3.7 - Arquivo após conversão de URLs

3.3.3 **Relatórios para visualização** (C3)

Para facilitar o acesso aos dados gerados nas camadas anteriores, foi desenvolvido um relatório analítico ordenado pelo nome do usuário. Este relatório foi utilizado pelos responsáveis pela autoria do material instrucional do Read in Web para identificar o comportamento do aluno.

```
Unicamp - Faculdade de Engenharia Eletrica e Computacao

              Relatorio de Utilizacao das Paginas ReadWeb
              -------------------------------------------

User: turchete
----

      11-06-2001  07:49:46   modulo1/atividade2/texto de leitura da atividade
      11-06-2001  07:49:54   modulo1/atividade2/questão 1
      11-06-2001  07:49:56   modulo1/atividade2/questão 2
      11-06-2001  07:49:58   modulo1/atividade2/questão 3
      11-06-2001  07:50:01   modulo1/atividade2/questão 4
      11-06-2001  07:50:03   modulo1/atividade2/questão 5
      11-06-2001  07:58:29   modulo1/atividade2/texto de leitura da atividade
      11-06-2001  07:59:06   modulo1/atividade2/questão 1
      11-06-2001  07:59:09   modulo1/atividade2/questão 2
      11-06-2001  07:59:11   modulo1/atividade2/questão 3
      11-06-2001  07:59:13   modulo1/atividade2/questão 4
      11-06-2001  07:59:17   modulo1/atividade2/questão 5
      11-06-2001  07:59:19   modulo1/atividade2/questão 6
      11-06-2001  07:59:28   modulo1/atividade2/texto de leitura da atividade
      11-06-2001  08:05:28   modulo1/atividade2/texto de leitura da atividade
      11-06-2001  08:05:31   modulo1/atividade2/questão 1
      11-06-2001  08:05:34   modulo1/atividade2/questão 2
      11-06-2001  08:05:36   modulo1/atividade2/questão 3
      11-06-2001  08:05:37   modulo1/atividade2/questão 4
      11-06-2001  08:05:38   modulo1/atividade2/questão 5
      11-06-2001  08:05:39   modulo1/atividade2/questão 6
      11-06-2001  08:06:23   modulo1/atividade2/texto do tutorial da  atividade
      11-06-2001  08:06:27   modulo1/atividade2/estratégias de leitura e compreensão
      11-06-2001  08:14:29   modulo1/atividade2/exercício 1 do tutorial
      11-06-2001  08:14:34   modulo1/atividade2/exercício 2 do tutorial
      18-06-2001  15:11:06   modulo1/atividade2/texto de leitura da atividade
```

Figura 3.8 - Trecho de um relatório analítico

Desenvolvido em Java, o aplicativo que gera este relatório lê os dados de uso já preparados direto do arquivo formatado pelos passos anteriores da ferramenta e lista as chamadas às páginas de forma organizada. O relatório é ordenado por nome do usuário, data e hora de utilização. Embora seja uma boa fonte de informações, este relatório não

apresentou facilidade de análise devido ao grande volume de dados. A Figura 3.8 apresenta um trecho de um relatório.

Procurando melhorar os aspectos relacionados às consultas, os dados do arquivo de *log*, após limpeza e conversão, foram inseridos em uma base de dados persistente.

3.3.4 Camada de persistência dos dados analíticos (C4)

A base de dados, composta de transações efetuadas pelo usuário, foi importada para um SGBD (Sistema Gerenciador de Banco de Dados) relacional. Com este processo, a análise dos dados foi facilitada pois tornou-se possível a utilização da linguagem SQL para a realização de consultas *ad hoc*, o que não era possível através dos relatórios inicialmente disponibilizados. Desta forma, algumas consultas puderam ser efetuadas de maneira mais simples tais como: verificar a seqüência de utilização das páginas, isto é, o caminho percorrido, calcular o tempo despendido em cada página ou atividade, avaliar se a seqüência prevista quando da construção do material é realmente a utilizada pelos alunos e outras consultas voltadas à análise do ambiente e do aluno.

O banco de dados utilizado foi o SGBD Oracle, devido a sua disponibilidade no ambiente de desenvolvimento. Outros testes foram realizados utilizando-se os sistemas gerenciadores Microsoft Access e o MySQL e não foram detectados problemas.

A conexão com o banco de dados a partir de uma aplicação Java utiliza a API JDBC desenvolvida pela (Sun Corporation, 2003), cujo objetivo é atender às funcionalidades básicas requeridas pelas aplicações que utilizam banco de dados. A conexão com banco de dados é feita por um procedimento pelo qual é necessário registrar um *driver* do banco para obter a conexão por uma URL JDBC. Esses *drivers* JDBC podem ser programas escritos

com rotinas em linguagens diferentes de Java e podem ser classificados em quatro grupos (Ramon, 2002):

1. Ponte JDBC-ODBC com *driver* ODBC;
2. Driver com API parcialmente nativa;
3. Driver puro Java JDBC-rede;
4. Driver puro Java com protocolo nativo.

Os dois primeiros tipos de *drivers* exigem um controle maior sobre a distribuição do programa pois são necessárias configurações adicionais na máquina cliente e portanto são mais voltados para ambientes do tipo cliente servidor em duas camadas. Os *drivers* do tipo 3 e 4 não exigem instalação do lado do cliente, sendo recomendados para aplicações desenvolvidas em três camadas.

Neste trabalho foi utilizado um *driver* do tipo 4, que acessa diretamente o gerenciador de banco de dados usando o protocolo nativo do mesmo. Este tipo é fornecido pelos fabricantes de banco de dados e é normalmente conhecido como *thin driver* devido ao seu tamanho reduzido.

A seguir são apresentados os passos necessários para realizar a conexão com um banco de dados relacional.

1. Importar as classes que referenciam o driver JDBC apropriado. Neste caso, as classes estão definidas no pacote java.sql. Este pacote é parte da API padrão da plataforma Java J2SE:

```
import java.sql.*;
```

2. Carregar o driver. No exemplo, o driver é um objeto da classe OracleDriver que faz parte do pacote oracle.jdbc.driver, disponibilizado pelo fabricante.

```
// Database Access / connecting  (registrando o Driver)
   DriverManager.registerDriver
                    (new oracle.jdbc.driver.OracleDriver());
```

3. Especificar um banco de dados indicando uma URL de conexão. Neste exemplo, o servidor de banco de dados está localizado na máquina "copacabana" da rede "dca.fee.unicamp.br" e aceita requisições na porta 1521. A base de dados identificada pelo nome WG73 será utilizada. A variável pUrl é do tipo *string*.

```
pUrl =
jdbc:oracle:thin:@copacabana.dca.fee.unicamp.br:1521:WG73
```

4. Abrir a conexão. No exemplo, as variáveis pUser e pPassword são objetos *String* que contêm a informação sobre a identificação do usuário e a senha, respectivamente, para acesso ao sistema gerenciador da base de dados.

```
// Abrindo a conexão
  Connection conn =
             DriverManager.getConnection(pUrl,pUser,pPassword) ;
```

5. Criar um Statement. O objeto dessa classe é utilizado para enviar comandos ao SGBD.

```
//  Criando um statement
    Statement stmt = conn.createStatement;
```

6. Submeter um comando SQL. O método 'executeUpdate' recebe como argumento um objeto *String* com o comando SQL que se deseja executar no SGBD. Neste exemplo, a *string* especifica a inclusão na tabela `Tab_Web_Log` de informações sobre o acesso do usuário indicado pela *string* 'newUser'.

```
// Submetendo um comando SQL
   String insertString = new String();
   insertString = "insert into Tab_Web_Log " +
      "values (' " + newUser +"',to_date('"+ dataUser +
              "','dd-mm-yyyy:hh24:mi:ss'),'"+acaoUser
                                          +"',sysdate ) ";
            stmt.executeUpdate(insertString );
```

O Passo 3 realiza a especificação do banco de dados indicando uma URL de conexão cujo formato depende do tipo de *driver* utilizado. No caso desta aplicação o formato é o seguinte:

jdbc:oracle:<driver>:@<URL>:<porta>:<database>

Para a especificação dos dados referentes a esta conexão foi utilizada uma tabela externa de configuração (Tabela C) . Esta tabela contém os parâmetros necessários para a conexão e é carregada em tempo de execução. O usuário pode editar esta tabela alterando as configurações caso haja troca do banco de dados. Abaixo, na Figura 3.9, encontra-se o arquivo de parâmetros. Ele é composto do nome do usuário, senha e *string* de conexão. A *string* à esquerda do sinal de igualdade identifica o parâmetro e não deve ser alterada. A *string* à direita deve ser substituída conforme especificação relativa ao gerenciador de banco de dados adotado.

```
db.user=scott
db.password=tiger1
db.url=jdbc:oracle:thin:@copacabana.dca.fee.unicamp.br:1521:WG73
```

Figura 3.9 - Arquivo de parâmetros para JDBC

Os dados importados para o SGBD obedeceram à mesma estrutura do modelo de dados dos arquivos gerados pelos aplicativos em Java. A descrição dos campos da tabela segue abaixo:

NumSeq - Identificação seqüencial do registro gerada pelo aplicativo.

UserLog - Identificação do usuário obtida na certificação exigida para entrar na primeira página do curso.

UrlConv - Texto que representa a página requisitada. Obtido pela conversão da URL da página em um texto mais facilmente identificável pelo usuário.

DatTime - Representa o dia/mês/ano hora , minuto e segundo da requisição da página. Este formato pode variar de acordo com o banco de dados. O formato utilizado para o SGBD ORACLE é *dd/mm/yyyy:hh:mi:ss.*

Utilizando o SGBD relacional foi possível a realização de algumas consultas para descobrir, por exemplo:

- quantas vezes um aluno acessou uma determinada página;
- qual o aluno que mais acessou uma determinada página;
- qual a página mais acessada;
- o tempo gasto em cada página.

A seguir são apresentadas algumas consultas realizadas que, embora não tenham a pretensão de atuar como uma ferramenta de *data mining,* fornecem informações

significativas para uma análise inicial. A composição dos comandos utiliza sintaxe referente à linguagem SQL-Plus do SGBD Oracle 8i.

Exemplo 1*:* Listar quantas vezes cada usuário acessou a página referente ao "modulo1/atividade2/estratégias de leitura e compreensão"

```
SQL > select  userid "Identificação do usuário",  count(*)  "Numero de acessos" , url
2    from tab_web_log
3    where    url    like    '%modulo1/atividade2/estratégias de leitura e compreensão%'
4    group  by    userid, url;
```

Resultado exibido:

Id do usuário	N.acessos	URL
alnunes	1	modulo1/atividade2/estratégias de leitura e compreensão
guest	6	modulo1/atividade2/estratégias de leitura e compreensão
psmelo	1	modulo1/atividade2/estratégias de leitura e compreensão
turchete	2	modulo1/atividade2/estratégias de leitura e compreensão

Este comando utiliza a função *count(*)* que retorna o número de linhas da tabela que obedecem à condição da cláusula *where*, que neste caso é: o conteúdo do campo caminho deve conter a *string* "exercício 1 do tutorial" em qualquer posição. O caractere % representa o coringa. Como resultado é exibido o nome do usuário e o número de acessos agrupados por usuário. A cláusula *group by* implementa o agrupamento.

Exemplo 2*:* Listar o aluno que mais acessou a página de texto de leitura da atividade.

```
 SQL> select  userid,  count (*)
2     from    tab_web_log
3     where   url like '%texto de leitura da atividade%'
4     group by  userid
5     having count(*)  =  (select  max(count(*)) from  tab_web_log
6                          where url like '%texto de leitura da atividade%'
7                          group by userid  );
```

Resultado exibido:

USERID	COUNT(*)
psmelo	110

Neste exemplo, a consulta interna (sub-consulta dentro dos parênteses) retorna um valor numérico que corresponde ao número máximo de vezes que um usuário qualquer acessou a URL correspondente ao "texto de leitura da atividade" sem identificar qual é o usuário. A consulta externa lista a identificação do(s) usuário(s) onde o número de vezes de acesso a URL especificada foi igual ao número máximo obtido.

Exemplo 3: Listar a página mais acessada

```
SQL> Select  url "Página mais acessada"
 2    from tab_web_log
 3    group by url
 4    having count(*) = ( Select  max(count(*))
 5                        from logrw
 6                        group by url );
```

Resultado exibido:

Página mais acessada
---
modulo4 apostila gramatical

Neste exemplo, a consulta interna devolve um valor numérico que corresponde ao número máximo de vezes que uma página foi acessada sem identificar qual é esta página. A consulta mais externa exibe o caminho (URL acessada) que corresponde à página cujo número de acessos é igual ao máximo encontrado pela consulta mais interna.

Exemplo 4: Cálculo do tempo gasto em cada página

O tempo gasto é obtido pela técnica duração de referência (Cooley *et al*., 1997) citada na Seção 2.2.6, onde o tempo despendido em uma página é calculado pela diferença entre o tempo da página corrente e a chamada da próxima. Esta técnica apresenta um problema em relação ao cálculo do tempo para a última página pelo fato desta não possuir próxima. Portanto, o tempo gasto na última página é desprezado.

Para o cálculo do tempo despendido o algoritmo utilizado foi:

1- Ordenar os registros da tabela tab_web_log por nome de usuário, data e hora.
2- Ler registro da tabela.
3- Enquanto houver registro:
    3.1 Calcular :
        diferenca_horas = data registro atual – data registro anterior.
        diferença_horas = diferenca_horas * 24 * 60 (ajuste para resultado em minutos).
    3.2 Gravar registro na tabela resumo (tab_web_log_resumo), contendo:

nome do usuário, URL e a diferença em segundos.

3.3 Ler novo registro.

3.4 Voltar ao passo 3.

4- Encerrar.

A diferença entre o tempo de entrada e saída de uma página é calculado e armazenado em uma tabela temporária. A seguir é apresentado o exemplo de uma consulta a esta tabela onde o tempo de permanência é apresentado. Apenas as entradas com número de seqüência entre 550 e 560 foram selecionadas. A medida utilizada foi quantidade em segundos.

```
SQL> select userid,difhoras “Tempo Gasto”,url, nseq
  2  from logtothora
  3  where userid like '%charlene%' and nseq between 550 and 560;
```

Resultado exibido:

> USERID	Tempo Gasto	URL
charlene	4	modulo2/atividade1/questão 2
charlene	5	modulo2/atividade1/texto de leitura da atividade
charlene	194	modulo2/atividade1/questão 2
charlene	19	modulo2/atividade1/questão 3
charlene	30	modulo2/atividade1/texto de leitura da atividade
charlene	8	modulo2/atividade1/questão 2
charlene	34	modulo2/atividade1/questão 3
charlene	56	modulo2/atividade1/questão 4
charlene	33	modulo2/atividade1/texto de leitura da atividade
charlene	85	modulo2/atividade1/questão 4
charlene	140	modulo2/atividade1/questão 5

11 linhas selecionadas.

3.4 Vantagens do Mecanismo Desenvolvido

A arquitetura utilizada apresentou vantagens em relação às ferramentas comerciais disponíveis no mercado, principalmente devido ao processo de filtragem de dados realizado e à adequação do texto de URL.

Com o processo de filtragem dos dados, foram mantidas apenas as entradas com informações relevantes para o monitoramento das atividades do aluno. Este processo originou uma diminuição no volume de dados trazendo benefícios em termos de tempo e necessidade de armazenamento físico.

A conversão do texto de URL facilitou a análise dos dados pois permitiu uma associação mais precisa entre a estrutura do curso e o percurso realizado pelo aluno, além de facilitar a interpretação das URLs.

O fato de apresentar relatórios analíticos pode auxiliar a avaliação dos aspectos pedagógicos, embora esta análise seja subjetiva e envolva ainda um grande volume de dados.

A ferramenta gera como produto um arquivo em formato texto, o qual pode ser importado para qualquer banco de dados ou mesmo acessado por aplicativos desenvolvidos em qualquer linguagem de programação. Outra possibilidade é a incorporação do arquivo em ferramentas de análise de dados ou *data mining*, permitindo consultas *ad hoc* ou mesmo para inferir regras que permitam associar comportamentos a fatores subjetivos tais como estilos cognitivos e tipo de motivação.

A ferramenta é configurável, tanto em relação às informações que devem ser extraídas do arquivo de *log*, como no texto de conversão de URL e no acesso ao banco de

dados, o que agrega uma flexibilidade não contemplada pela maioria das ferramentas comerciais de análise de *log* da Web.

Em relação aos requisitos iniciais apontados pela equipe de coordenação do Read in Web foi possível distinguir o caminho percorrido e o tempo dedicado a cada página como solicitado. A única ressalva é em relação ao tempo gasto na última página que, pela estrutura da ferramenta, não pôde ser calculado.

3.5 Considerações Finais

Nesta primeira parte da pesquisa, procurou-se desenvolver uma ferramenta para solucionar o problema relacionado ao acompanhamento da utilização de um curso de ensino a distância, no caso o Read in Web desenvolvido pelo IEL da Unicamp.

Embora a ferramenta desenvolvida neste estudo de caso tenha alcançado bons resultados, foram identificadas deficiências que motivaram a proposta do sistema descrito a seguir no Capítulo 4.

Dentre as limitações identificadas podemos citar: o grande volume inicial proveniente do arquivo de *log* do servidor antes do processo de filtragem torna difícil a manipulação e o próprio armazenamento dos dados. Isto torna o processo lento. A falta de confiabilidade na origem dos dados, provenientes dos arquivos de *log* dos servidores web, também é um ponto importante que não deve ser deixado de lado. Esta limitação quanto à confiabilidade dos dados armazenados é devida a problemas do próprio HTTP tal como a existência de servidores *proxies* e a *cache* do *browser* que não forçam o registro de acesso a páginas anteriormente visitadas, uma vez que estas já se encontram na máquina do cliente.

Outro aspecto é a dificuldade de análise dos dados em tempo-real. A disponibilidade destas informações depende da execução de todo o processo descrito que, embora possa ser executado com freqüência, demanda um certo tempo – principalmente porque a quantidade de dados a ser manipulada pode exceder alguns gigabytes por dia (Shahabi *et al*., 2001).

Esta solução também introduz um erro em relação ao cálculo do tempo despendido. Como o método utilizado para este cálculo considera a diferença entre o tempo da página corrente e a chamada da próxima, temos um problema em relação à última página, pois como esta não possui próximo, não é possível registrar o tempo gasto. Isto compromete a confiabilidade dos dados principalmente se o conteúdo da última página acessada for o tópico alvo de estudo.

Analisando os problemas identificados neste estudo de caso, foi desenvolvida a proposta de um novo sistema a fim de possibilitar uma forma mais eficiente e confiável de aquisição de dados, visto que este era um dos aspectos mais preocupantes. Também foram estudadas soluções mais eficientes para os demais processos envolvidos.

Capítulo 4 - Sistema de análise de uso de recursos Web para educação

O objetivo deste trabalho, desenvolver uma ferramenta de aquisição de dados para monitorar a utilização dos recursos na Web, foi parcialmente alcançado pelo estudo de caso relatado anteriormente. No entanto, a ferramenta apresentada atende a um ambiente em particular – o Read in Web. Buscando uma solução genérica que pudesse ser implementada na mais diferentes aplicações na Web, foi idealizada uma nova arquitetura conforme relatado nas próximas seções.

4.1 Requisitos do Sistema

O requisito fundamental imposto ao novo modelo foi a desvinculação dos arquivos de *log* dos servidores Web como fonte de obtenção dos dados relativos à interação do aluno com as páginas. Tal requisito se justifica por vários motivos dentre os quais a falta de exatidão dos dados provenientes dos arquivos de *log*, a incerteza em relação ao tempo despendido em cada página e as dificuldades impostas pelo pré-processamento de grandes quantidades de dados. Permitir análise de dados em tempo real, transparência para o usuário final, portabilidade e confiabilidade também foram considerados requisitos importantes.

4.2 Modelo Conceitual

A arquitetura utilizada foi baseada na implementação clássica de três camadas

(Larman, 2000), onde a característica principal é a separação da lógica da aplicação em uma camada intermediária separada de software. Uma descrição clássica das camadas é a seguinte:

1- Apresentação – janelas, relatórios, formulários, etc.
2- Lógica da aplicação – tarefas e regras que governam o processo.
3- Armazenamento – mecanismo de armazenamento persistente.

A representação da lógica da aplicação em componentes separados permite a reutilização do software e a sua implementação em computadores distintos, como é o caso do sistema aqui proposto. Também acrescenta segurança em relação ao acesso à base de dados e flexibilidade para alterações, uma vez que os componentes podem ser substituídos sem necessidade de reconstruir todo o sistema.

No modelo concebido neste trabalho, a primeira camada é uma página HTML onde reside um mecanismo de aquisição de dados, executado no lado cliente (Mecanismo de coleta de dados), o qual dispara o segundo componente executado no servidor Web (Mecanismo de manipulação). Este segundo componente é o responsável pela implementação da lógica da aplicação, incorporando algumas funcionalidades de transformação de dados, além de realizar o processo de comunicação com a terceira camada (Mecanismo de armazenamento), que realiza o armazenamento dos dados em meio físico permanente.

O objetivo deste modelo em três camadas é separar as funcionalidades, aproveitando as facilidades disponibilizadas em cada plataforma. A obtenção dos dados no lado cliente garante principalmente a confiabilidade, pois a aquisição de dados no lado servidor pode resultar em imprecisões. Por exemplo: uma página pode ficar guardada na *cache* do navegador fazendo com que um retorno a esta página não gere nova solicitação ao servidor e portanto não seja registrado este novo acesso. Outro problema é o aspecto temporal uma

vez que o registro de tempo, quando realizado pelo servidor, inclui o tempo de transferência na rede gerando distorções (Shahabi *et al*., 2001).

A Figura 4.1 ilustra a arquitetura do modelo proposto, onde podemos verificar os mecanismos citados. O primeiro mecanismo é processado do lado do cliente, neste caso pelo navegador, o segundo e o terceiro têm seu processamento executado pelo servidor Web. A arquitetura permite também que a terceira camada seja implementada em uma plataforma distinta atuando como servidor de banco de dados.

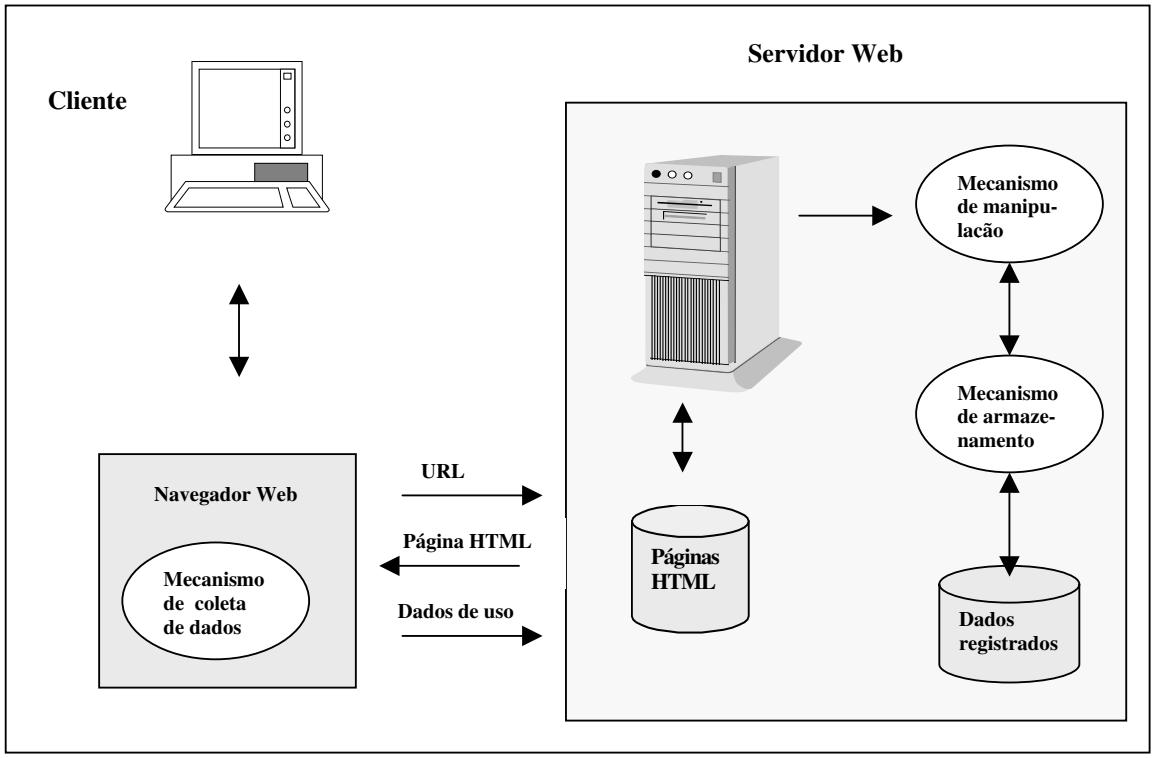

Figura 4.1 - Arquitetura do modelo proposto

As ações executadas pelo mecanismo de coleta de dados, conforme mostradas na Figura 4.1, obedecem à seqüência:

1- O cliente solicita a página inicial do curso ao servidor Web.

2- O servidor envia a página solicitada ao cliente.

3- O mecanismo de coleta de dados é iniciado.

4- O cliente solicita a próxima página.

5- O mecanismo de coleta envia as informações recolhidas a respeito da página corrente para o servidor Web, isto é, para o mecanismo de manipulação de dados, antes de enviar a requisição da próxima página.

6- O servidor Web envia a próxima página.

O mecanismo de manipulação de dados ao ser ativado no servidor executa:

1- Obtém os dados enviados referentes aos acessos;

2- Converte a URL e analisa extensões irrelevantes;

3- Ativa o mecanismo de armazenamento de dados.

O mecanismo de armazenamento grava os dados em um meio persistente.

4.3 Implementação do Modelo

A seguir serão descritos os detalhes de implementação dos mecanismos que compõem a aplicação desenvolvida.

4.3.1 Mecanismo de coleta

Várias técnicas têm sido utilizadas para aquisição de dados do lado cliente, dentre elas, *cookies*, CGI e agentes, como descrito na Seção 2.2. Como a privacidade do usuário é algo a ser considerada, a utilização de *cookies* foi descartada. Também o uso de CGI foi

desconsiderado, devido aos inconvenientes associados à sua utilização como a geração de um novo processo no servidor a cada solicitação efetuada.

Apesar da tecnologia de agentes também ser apropriada para implementação do mecanismo de coleta, esse não foi o caminho escolhido devido a complexidade de sua utilização. A opção foi por uma solução mais simples que atendesse aos requisitos definidos.

```
<p>
<applet code=AppletTime width=0 height=0>
<param name = "location" value =
     "http://localhost/CursoSqL.java">
</applet>
```

Figura 4.2 - Referência ao *applet* dentro de uma página HTML

A implementação do mecanismo de coleta de dados foi realizada pela inclusão na página HTML da referência a um *applet* (Figura 4.2), que consiste em um pequeno programa escrito em Java. Os *applets* Java são embutidos diretamente em páginas HTML e, quando os usuários acessam essas páginas, eles são descarregados para o computador cliente. Sua execução ocorre no lado cliente, desde que o navegador seja compatível com a linguagem Java. Deste modo, os *applets* não dependem de largura de banda ou velocidade do modem quando estão sendo executados.

Os estados pelos quais o *applet* passa desde o momento em que é carregado até quando não for mais acessível compreendem o ciclo de vida do *applet*, como mostra a Figura 4.3. Cada transição de estado no ciclo de vida é marcada pela invocação de um método (Java Sun, 2004):

void init() - o navegador envia a mensagem *init*() uma única vez para o *applet* logo depois de criá-lo.

void start() - o navegador envia a mensagem *star*t() toda vez que a página do *applet* é visitada. Em outras palavras, cada vez que uma página é revisitada todos os *applets* hospedados nela recebem a mensagem *start().*

void stop() - o *applet* sempre recebe a mensagem *stop()* quando a página onde está hospedado é sobreposta por outra página.

void destroy() - este método é invocado imediatamente antes da destruição do *applet*, quando todos os recursos do *applet* necessitarem ser liberados.

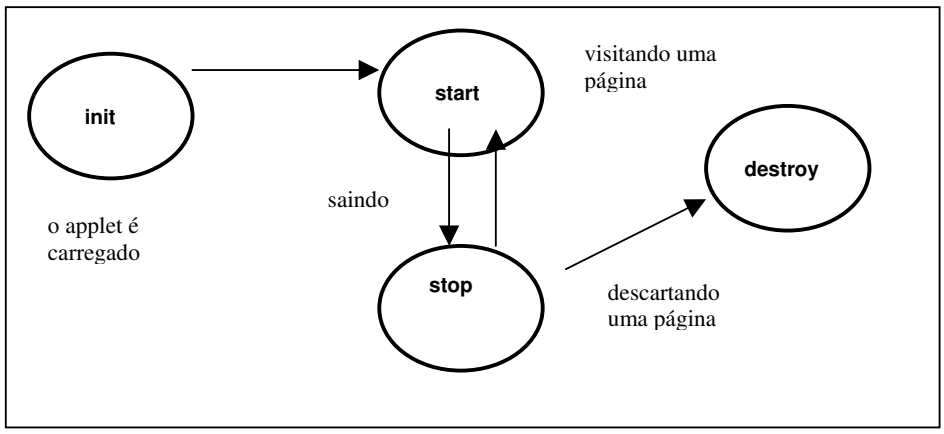

Figura 4.3 - Ciclo de vida de um *applet*

O *applet* desenvolvido para a implementação deste modelo, denominado AppletTime, extrai as informações necessárias tais como identificação do usuário, tempo de permanência na página e URL da página acessada. Na seqüência, ativa o mecanismo de manipulação de dados instalado no servidor, o qual recebe esta informação, realiza um processo de transformação e registra o resultado em um meio físico permanente. O mecanismo de manipulação de dados foi implementado por meio de *servlets* e será descrito na próxima seção.

O *applet* não exibe informações na página chamada pelo navegador. Seu objetivo é apenas permitir o registro da entrada e da saída de cada página. Para isto, em seus métodos *start*() e *stop()* foi embutida uma rotina de extração de tempo e esta informação, associada à identificação do usuário e URL da página, é passada como parâmetro para um *servlet* invocado dentro do *applet* (Figura 4.4). Parte da rotina de extração de tempo está implementada dentro do método *start*(); portanto, toda vez que o *applet* é executado, o instante referente ao início da execução é anotado. Este tempo é expresso em hora, minuto e segundo e é extraído da máquina do cliente, não havendo assim problemas em relação à disparidade de tempo entre máquina cliente e servidor, além de resolver o problema de distorção causado quando se considera o tempo de transferência na rede. Quando o usuário sai da página, automaticamente o *applet* executa seu método *stop*(), que anota o momento da saída.

```
// invoca o servlet
   try {

      URLservlet = new URL
     ("http://localhost:80/servlets/ServletLogDBac?linha=
       " + result);
```

Figura 4.4 - Trecho do código do *applet* onde é chamado o *servlet*

4.3.2 Mecanismo de Manipulação

Este componente foi implementado utilizando-se *servlets,* aplicativos escritos em Java que podem ser acoplados em diversos tipos de servidores Web. A motivação principal para o uso de *servlets* é a capacidade de adicionar funcionalidade dinâmica a servidores da Web, sem a sobrecarga da iniciação de um novo processo associada à execução de *scripts* CGI.

Um *servlet* é executado em um servidor de aplicação que tenha a máquina virtual Java instalada. Um servidor de aplicação é um tipo especial de servidor Web (IBM-Developers, 2004); ele estende a capacidade do servidor Web para lidar com requisições para *servlets* e aplicações Web. O servidor carrega, executa e gerencia os *servlets*.

O fluxo básico de execução de um *servlet* é apresentado abaixo.

1. O cliente envia uma requisição ao servidor.
2. O servidor carrega o *servlet* e cria uma linha de execução (*thread*) para atender a solicitação do *servlet*. O *servlet* é carregado desde a primeira requisição e permanece carregado até o servidor ser desligado.
3. O servidor envia a informação requisitada para o *servlet.*
4. O *servlet* constrói uma resposta e envia para o servidor.
5. O servidor envia a resposta de volta para o cliente.

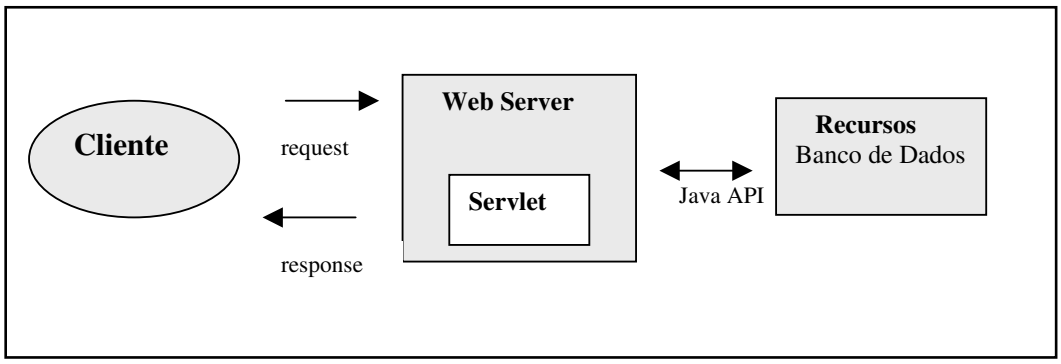

Figura 4.5 - Ações do *servlet*

As vantagens do uso de *servlets* são muitas, principalmente se comparadas ao uso de CGI, pois todos os *servlets* associados com um servidor Web são executados dentro de um único processo. Este processo reside em uma máquina virtual Java (JVM), um programa dependente da plataforma que executa programas Java compilados, o que traz portabilidade

do lado do servidor. O *servlet* não precisa ser carregado a toda hora e as conexões não precisam ser feitas a cada requisição. Outra vantagem é poder manipular várias conexões simultâneas.

A utilização de *servlets* também torna mais fácil a implementação da aplicação em camadas separando a lógica de apresentação da lógica de negócio o que traz as vantagens inerentes da arquitetura de aplicações em três camadas.

Para a utilização de *servlets* é necessário instalar um *Servlet Engine* que direciona a solicitação ao *servlet*, que é executado. Existem vários *Servlet Engines* disponíveis para serem adicionados aos servidores Web. Alguns dos mais comumente usados são o Jserv e Jrun. Neste trabalho foi utilizado o Apache Jserv (Apache Organization, 2004).

O servlet desenvolvido, denominado ServletLog, é um aplicativo invocado pelos métodos do *applet (*denominado AppletTime) descrito anteriormente, que está embutido nas páginas HTML da aplicação. Sua funcionalidade envolve a conversão de URLs, semelhante ao processo implementado no estudo de caso relatado no Capítulo 3, o cálculo da diferença entre a hora de entrada e a hora de saída na página HTML e a chamada ao Mecanismo de armazenamento.

A filtragem de extensões relevantes e irrelevantes, existente no estudo de caso anterior, não se fez necessária nesta nova implementação pois apenas a URL da próxima página é anotada pela *applet* do mecanismo de coleta de dados. Logo, ao contrário dos arquivos de *log* gerados pelo servidores Web, os arquivos de imagens, sons, animações não são registrados. Isto economiza o tempo dedicado à tarefa de limpeza de dados (*data cleaning),* normalmente realizada quando se utiliza o *log* dos servidores Web como fonte de aquisição de dados.

Assim como realizado na primeira solução, procurou-se ao máximo parametrizar o software desenvolvido, de forma que a especificação do banco de dados também pode ser

configurável por meio de tabelas. A arquitetura permite também que o gerenciador de banco de dados esteja em outro servidor que não o servidor Web, o que garante escalabilidade de plataformas. Um trecho do *servlet* com a chamada ao método de inclusão no SGBD pode ser visualizado na Figura 4.6.

```
// monta informações para registro no log do BD e arquivo txt.

        String sURI = request.getRequestURI();
        String sDate = request.getParameterValues("linha")[0];

         IncludeDBMS(out,sURI, sDate);
```

Figura 4.6 - Trecho do *servlet* implementado

A inclusão dos dados de uso em tempo real no banco de dados também agrega eficiência ao sistema, pois permite a realização de análises simultaneamente à utilização das páginas. Isto pode ser útil para, por exemplo, adicionar personalização à aplicação tal como sugerir caminhos opcionais ou incluir preferências pessoais.

Identificação do usuário e da transação.

A identificação do usuário é apresentada em muitos trabalhos como um dos pontos principais para a análise do uso. Algumas ferramentas como a apresentada em Shahabi *et al*. (2001) utilizam a técnica parecida com a que foi utilizada neste trabalho, servindo-se de *applets* para passagem de parâmetros para o servidor. No entanto, a identificação do usuário é realizada por uma ID única fornecida pelo software que é executado no servidor. Essa identificação é enviada junto com o *applet* para o cliente na primeira vez que ela é executada. Como já descrito no Capítulo 2, algumas outras técnicas incluem *cookies* ou agentes.

Neste trabalho, a identificação do usuário não precisou de implementação adicional pois, assim como na primeira solução, foi considerada a existência de uma página inicial que solicita a identificação do usuário composta de nome e senha de acesso. Desta forma o próprio HTTP inclui na URL da página requisitada o nome do usuário solicitante (*user)*.

A identificação de uma transação poderá ser feita utilizando-se a técnica descrita em Cooley *et al.* (1997), onde se considera a diferença de tempo entre um acesso e outro do mesmo usuário. Se esta diferença representar um tempo muito grande, significa que outra transação foi iniciada. Neste caso estamos considerando transação como uma sessão, isto é, o conjunto de acessos consecutivos de um mesmo usuário considerando o momento em que ele entra no sistema por *login* e sai do sistema fechando a janela do navegador ou acessando páginas que não fazem parte do texto referenciado. A página que não faz parte do material do curso não possui embutido o mecanismo de registro. Portanto, interações com tais páginas não são armazenadas no banco de dados.

Cálculo do tempo despendido em cada página

O cálculo do tempo despendido em cada página é calculado pela diferença entre o momento de entrada e o momento de saída da página. Este processo é realizado pelo *servlet*. Como a coleta desta informação é realizada no cliente (pelo *applet*), a defasagem de tempo entre o cliente e o servidor, caso exista, não trará problemas pois o que é registrado é o tempo entre o início e o fim da exibição de uma página, portanto o tempo relativo.

O motivo pelo qual preferiu-se implementar este algoritmo de cálculo de tempo no servidor foi primeiramente deixar o *applet* o menor possível, livrando-o da execução de códigos referentes à lógica de negócio já que sua execução se processa na máquina do cliente, cuja capacidade de processamento não podemos avaliar.

4.3.3 Mecanismo de armazenamento

Esta camada da aplicação utiliza um sistema gerenciador de banco de dados relacional como repositório de dados e finaliza o processo de registro de informações sobre os recursos utilizados pelo aluno e sobre o tempo gasto em cada página.

O SGBD utilizado foi o Oracle, mas é possível parametrizar os dados referentes ao banco de dados a ser utilizado. Isto traz flexibilidade à ferramenta, visto que alterações podem ser realizadas sem modificar o código. O processo realizado é semelhante ao implementado no estudo de caso relatado no Capítulo 3. Foi utilizada a API JDBC desenvolvida pela Sun (Sun Corporation, 2003), com *driver* tipo 4, e a montagem da URL de conexão é feita por meio dos parâmetros indicados em um arquivo de configuração externo ao programa.

O modelo de dados implementado no banco de dados é simples e pode ser consultado por meio da linguagem SQL, por usuário especializados, ou mesmo por aplicativos específicos para emissão de relatórios ou consultas diversas. Este modelo encontra-se especificado na Figura 4.7. As consultas enumeradas na Seção 3.3.3 para o estudo de caso anteriormente relatado podem também ser aplicadas aos dados coletados neste novo modelo.

No modelo apresentado acima, podemos notar a existência dos campos dataEntrada, dataSaida e horasDif. Isso, aparentemente, pode indicar redundância de informações. A diferença entre as horas poderia ser calculada sempre que necessário pelo próprio SGBD, dispensando a existência do campo horasDif. Porém, deixar esta tarefa para ser executada na camada de gerência de dados traz um acoplamento não desejado com o banco de dados, o que pode restringir a flexibilidade do modelo. Implementando o cálculo da diferença entre as horas no *servlet*, estamos atribuindo esta tarefa à camada da lógica de negócio, liberando o SGBD e a camada de interface com o usuário (neste caso o navegador) deste

processamento. Porém, continuamos precisando dos campos dataEntrada e dataSaida armazenados no banco de dados para realização de relatórios mais detalhados, uma vez que a diferença é expressa em segundos.

Nome do campo	**Tipo e Tamanho**	**Descrição**
NumSeq	Numérico	Identificação do registro
UserLog	String	Identificação do usuário. Representa a autenticação exigida na entrada do *sítio*.
DataEntrada	Date	Dia, mês, ano, hora, min., seg. de acesso a página.
DataSaida	Date	Dia, mês, ano, hora, min., seg. de saída da página.
HoraDif	Numérico	Tempo em segundos calculado a partir da diferença entre a hora de saída e a hora de entrada
URL	String	String com o endereço da página

Figura 4.7 - Modelo de dados

4.4 Vantagens da Aplicação da Ferramenta

A ferramenta desenvolvida, baseada na nova arquitetura proposta, apresenta muitas vantagens em relação ao mecanismo utilizado no estudo de caso inicialmente conduzido ou mesmo em relação às ferramentas que o mercado oferece.

A desvinculação dos arquivos de *log* dos servidores Web representa uma das grandes vantagens desta arquitetura pois, como mencionado anteriormente, esses arquivos trazem uma quantidade enorme de informações, muitas das quais são irrelevantes para o

processo de análise. Há também os já citados problemas com a c*ache* embutida nos navegadores e a *cache* gerada pelos servidores *proxy*.

Outra vantagem é o registro do tempo gasto em cada página incluindo a última, o que não foi alcançado na primeira solução. O tempo gasto em cada página é um parâmetro importante pois, além de servir de base para análise do material e do aluno, também permite descartar páginas consideradas apenas navegacionais, isto é, páginas de passagem que, segundo critério usado em Cooley *et al*. (1997b), podem ser identificadas pelo tempo médio de duração como citado na Seção 2.2.6.

O tempo médio de permanência em uma página também pode ser utilizado para identificar alunos com dificuldades específicas ou páginas mal escritas, onde o tempo médio despendido supera o estimado pelo autor.

A inclusão dos dados em tempo real no SGBD consiste em um aspecto significativo, pois possibilita a incorporação das informações presentes nesta base de dados em ferramentas de tutoria ou de personalização de páginas da Web, assim como em sistemas de *data mining*.

A característica dinâmica do modelo implementado também permite utilizar os dados relativos às atividades realizadas pelos alunos como base para outras ferramentas, tais como o "Sistema de Alertas Inteligentes" apresentado em Musa e Oliveira (2000). Nesse trabalho, Musa apresenta um sistema projetado para auxiliar o processo de avaliação do comportamento do aluno na educação a distância, pelo monitoramento de um banco de dados (que guarda informações sobre os acessos) e a decorrente emissão de avisos de alertas quando detectadas situações anormais. Um exemplo de situação anormal seria: usuário não solicitou uma determinada página importante, que deveria ter sido requisitada, em um certo período de tempo.

A implementação do modelo em uma arquitetura de três camadas trouxe várias vantagens: entre elas, o fato da camada de aquisição ser executada no lado cliente, o que reduz a chance de erros decorrentes da disparidade entre o tempo do servidor e o cliente. A camada de gerência de dados isolada da aquisição dos dados permite flexibilidade na escolha do gerenciador de dados. A introdução de uma terceira camada com a lógica da aplicação, no caso o *servlet,* também agrega vantagens pois podemos adicionar regras de negócio facilmente sem interferir nas páginas já construídas ou ter que reescrever o material instrucional.

Esta modularização facilita a utilização da ferramenta não somente em ambientes de EDMC, mas também em aplicações comerciais na Web, onde regras de negócio podem ser disparadas a partir de chamadas às páginas ou pela avaliação do caminho percorrido pelo usuário das páginas.

Por último podemos também citar as vantagens associadas à linguagem de implementação. Sendo este mecanismo desenvolvido em Java, torna-se portátil e fácil de ser agregado a ambientes de apoio a curso a distância de maneira simples. A maioria dos ambientes de apoio a EDMC existentes não implementa esta funcionalidade ou, se a possui, esta contém deficiências ou tem que ser utilizada em conjunto com os outros módulos da aplicação, não sendo possível utilizá-la separadamente.

4.5 Mecanismo de visualização

Embora a ferramenta disponibilize uma rica base de dados, a extração de informações para o acompanhamento das atividades é um processo que envolve tempo. A visualização do caminho percorrido em um relatório analítico de acessos é útil para análises mais detalhadas, porém exige muito tempo por parte do professor.

Realizar as consultas por meio da linguagem SQL, sem dúvida, permite flexibilidade e rapidez de resposta, principalmente quando se trata de grandes volumes de dados. Porém, essa tarefa exige um conhecimento especializado por parte do usuário, o que restringe sua utilização por usuários leigos. Portanto, como uma extensão ao trabalho realizado, foi desenvolvido o protótipo de um mecanismo que permite a visualização gráfica do caminho navegacional percorrido pelo aluno.

4.5.1 Preparação dos dados para o mecanismo de visualização.

A representação da seqüência das páginas acessadas pelo aluno durante sua interação com o material do curso pode ser visto como uma série de nós e arestas que se interligam semelhantemente a um grafo[3] simples. A proposta é utilizar esta estrutura para facilitar a visualização das ações executadas pelo aluno. Neste caso, os nós correspondem às páginas acessadas e as arestas apontam para o nó seguinte identificando a ordem de chamada das páginas.

A Figura 4.8 apresenta um grafo gerado pelo protótipo desenvolvido. Esta simulação utilizou uma base de dados previamente gerada, que corresponde aos acessos feitos ao curso Read in Web. Estas informações, indicadas na Figura 4.9, foram registradas no estudo de caso apresentado no Capítulo 3. O grafo apresentado corresponde às interações efetuadas pelo aluno "André" durante um intervalo de tempo selecionado.

Durante uma sessão, o aluno pode acessar o nó inicial (página de entrada para o curso) e seguir para o nó seguinte (página seguinte) ou retornar para o nó anterior. Portanto, não há uma seqüência fixa. O aluno pode realizar o mesmo caminho várias vezes.

[3] Um grafo é um conjunto de pontos, chamados vértices (ou nodos), conectados por linhas, chamadas arestas (ou arcos).

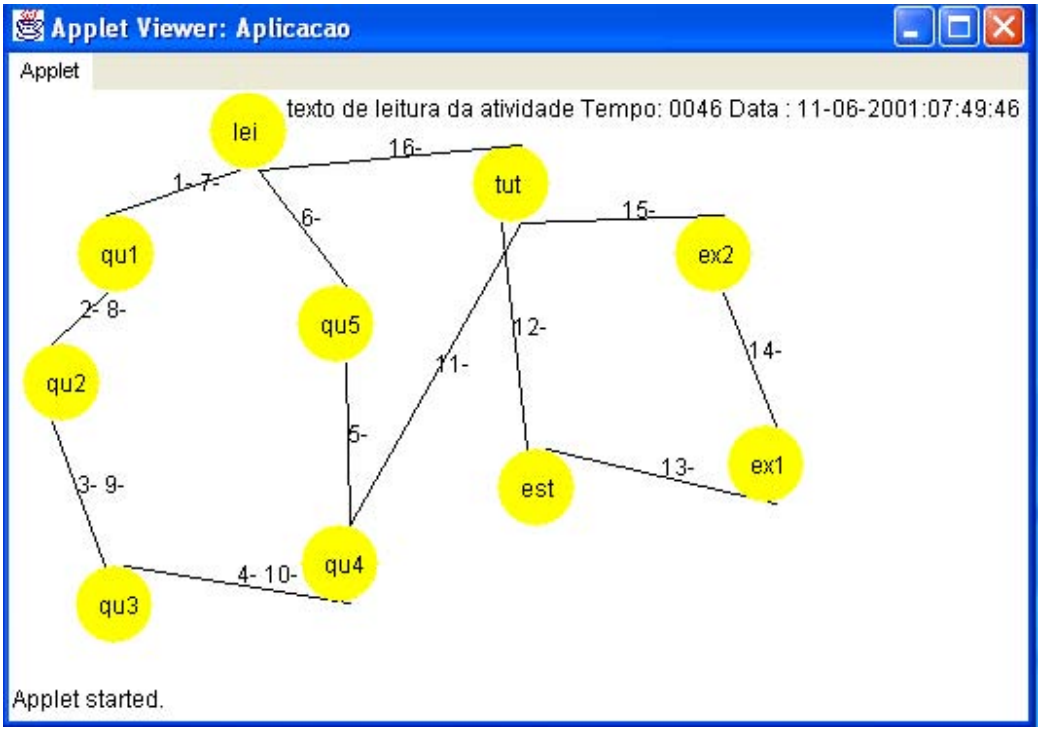

Figura 4.8 - Visualização das interações de um aluno

Para o desenvolvimento do mecanismo de visualização foram consideradas algumas restrições. Como o número de interações entre um nó e outro poderiam ser vários (movimento de ir e voltar de uma página a outra), optou-se por não representar uma aresta para cada interação, pois isso causaria um excesso de informação visual. A solução encontrada foi indicar acima das arestas a seqüência de números que correspondem a cada interação. Por exemplo, a indicação "1-7-" na aresta que liga os nós "lei" e "qu1" na Figura 4.8, significa que o aluno passou por este caminho na primeira e na sétima interação.

Dentro de cada nó foi inserido um acrônimo indicando a página visitada. Ao passar com o cursor sobre o nó, o nome completo da URL é apresentado, facilitando a compreensão.

```
André 11-06-2001  07:49:46   modulo1/atividade2/texto de leitura da atividade
André 11-06-2001  07:49:54   modulo1/atividade2/questão 1
André 11-06-2001  07:49:56   modulo1/atividade2/questão 2
André 11-06-2001  07:49:58   modulo1/atividade2/questão 3
André 11-06-2001  07:50:01   modulo1/atividade2/questão 4
André 11-06-2001  07:50:03   modulo1/atividade2/questão 5
André 11-06-2001  07:58:29   modulo1/atividade2/texto de leitura da atividade
André 11-06-2001  07:59:06   modulo1/atividade2/questão 1
André 11-06-2001  07:59:09   modulo1/atividade2/questão 2
André 11-06-2001  07:59:11   modulo1/atividade2/questão 3
André 11-06-2001  07:59:13   modulo1/atividade2/questão 4
André 11-06-2001  08:06:23   modulo1/atividade2/texto do tutorial da  atividade
André 11-06-2001  08:06:27   modulo1/atividade2/estratégias de leitura e compreensão
André 11-06-2001  08:14:29   modulo1/atividade2/exercício 1 do tutorial
André 11-06-2001  08:14:34   modulo1/atividade2/exercício 2 do tutorial
André 11-06-2001  08:15:23   modulo1/atividade2/texto do tutorial da  atividade
André 11-06-2001  08:16:29   modulo1/atividade2/texto de leitura da atividade
```

Figura 4.9 - Fonte de dados utilizada para geração do grafo da Figura 4.8

A Figura 4.10 mostra o arquivo de conversão de URL utilizado para a geração do grafo. Este arquivo é o mesmo utilizado para o estudo de caso descrito no Capítulo 3, acrescido de um novo campo que é um acrônimo de três letras associado à URL.

```
curso/modulo1/p2.htm==modulo1 apostila gramatical==m1g
curso/modulo2/p2.htm==modulo2 apostila gramatical==m2g
curso/modulo3/p2.htm==modulo3 apostila gramatical==m3g
curso/modulo4/p2.htm==modulo4 apostila gramatical==m4g
curso/modulo5/p2.htm==modulo5 apostila gramatical==m5g
curso/modulo6/p2.htm==modulo6 apostila gramatical==m6g
curso/p2.htm==teste do material==tes
curso/p1.htm==lista de palavras de função==lip
p1.htm==texto de leitura da atividade==lei
p2.htm==texto do tutorial da  atividade==tut
p3.htm==dicionário da atividade==dic
p4.htm==questão 1==qu1
p5.htm==questão 2==qu2
p6.htm==questão 3==qu3
p7.htm==questão 4==qu4
p8.htm==questão 5==qu5
p9.htm==questão 6==qu6
p10.htm==estratégias de leitura e compreensão==est
p11.htm==exercício 1 do tutorial==ex1
p12.htm==exercício 2 do tutorial==ex2
p13.htm==exercício 3 do tutorial==ex3
p14.htm==exercício 4 do tutorial==ex4
p15.htm==exercício 5 do tutorial==ex5
```

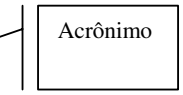

Figura 4.10 - Nova versão da tabela de conversão de URL

Os passos para a geração do grafo de visualização são os seguintes: a partir do arquivo de dados de acesso (indicado na figura 4.9) é realizado uma filtragem para seleção das interações a serem consideradas. Neste protótipo o conjunto selecionado considerou apenas as primeiras interações do dia "11/06/2001" do aluno "André" e foi realizado por meio de uma consulta em SQL. A partir do conjunto selecionado é gerada uma tabela intermediária que formata os dados para facilitar o desenho do grafo. Esta tabela contém a identificação do usuário, data de acesso, tempo despendido na página, URL associada à página, acrônimo da URL, identificação do nó de origem e identificação do nó de destino. Com base nesta tabela a aplicação em Java, que é um *applet*, gera o desenho do grafo utilizando as indicações do nó de origem e de destino.

4.5.2 Vantagens e restrições do mecanismo de visualização

O mecanismo de visualização foi desenvolvido com o propósito de reduzir o volume de dados a ser manipulado. Pelo grafo de interações é possível verificarmos a seqüência de uso das páginas de forma mais fácil do que em longos relatórios analíticos. A análise da seqüência percorrida pode auxiliar a avaliação do ambiente e do material instrucional, ou seja, se o uso do material corresponde às expectativas do seu autor. Pode também ser uma ferramenta complementar para a análise do comportamento do aluno.

Este mecanismo encontra-se em fase de desenvolvimento. Algumas rotinas precisam ser desenvolvidas como por exemplo, a leitura a uma base de dados. Atualmente, as informações correspondentes às seqüências de acesso e a tabela de conversão de URLs estão definidas como constantes dentro do próprio código fonte, uma vez que a ênfase foi a idealização gráfica das interações.

Outro procedimento a ser implementado é a filtragem dos dados e geração do arquivo resumo por um aplicativo em linguagem Java. Para este experimento os dados

foram filtrados utilizando-se a linguagem SQL, assim como a geração do arquivo resumo. A base de dados utilizada para o grafo poderá ser também a originada pela ferramenta de aquisição definida no Capítulo 4. Como esta ferramenta utiliza um *servlet* como mecanismo de aquisição de dados, será interessante alterar este *servlet* para que este já inclua o acrônimo da URL acessada no conjunto de dados persistidos.

4.6 Considerações Finais

Neste capítulo foi apresentado um mecanismo que permite a coleta e o registro dos dados representativos das interações dos usuários com as páginas em um ambiente de ensino a distância. Embora este mecanismo ainda não tenha sido incorporado a uma aplicação em uso na Web, os testes realizados indicam resultados positivos em relação a sua eficiência e adequação aos requisitos propostos inicialmente, isto é, um método simples e eficiente de registro de informações.

Muitas pesquisas têm sido feitas em relação a soluções para apoiar a análise e a avaliação dos ambientes educacionais na Web. Espera-se que este mecanismo possa ser utilizado como ferramenta de apoio à avaliação qualitativa e quantitativa destes ambientes e dos seus usuários, podendo ser integrado a aplicações já existentes. A tecnologia orientada a objetos utilizada na implementação deste mecanismo facilita o reuso de código e, conseqüentemente, a portabilidade de seus módulos.

A proposta de um mecanismo para visualizar os dados coletados surgiu como complemento à ferramenta de extração de informações sobre interações que foi desenvolvido. O objetivo é facilitar a tarefa de análise das seqüências das interações, pela apresentação de conjuntos de dados selecionados, em uma interface mais fácil de usar do que relatórios analíticos de acessos. É claro que este mecanismo de visualização não substitui o uso de técnicas de mineração de dados, onde informações mais complexas

podem ser extraídas ou mesmo associadas a outras bases de dados para a descoberta de conhecimento, mas já representa um avanço em relação ao simples registro de que houve um acesso a um determinado recurso.

Capítulo 5 - Conclusão

Neste trabalho, foi apresentado um mecanismo de monitoramento das ações realizadas pelo aluno em um ambiente de ensino na Web. Neste capítulo, serão apresentadas algumas considerações a respeito do trabalho realizado, tendo em vista as limitações, contribuições e trabalhos futuros.

5.1 Limitações

A ferramenta desenvolvida utiliza como base na implementação do mecanismo de coleta de dados um *applet* que deve ser incluído em cada página HTML passível de registro. Embora a incorporação do *applet* à página possa ser facilmente automatizada pelo uso de um script no servidor HTTP, alguns navegadores não conseguem executar *applets* sendo necessário instalar um *plug-in* específico que faz parte do Java 2 Runtime Environment, Standard Edition (JRE). Este *plug-in* estabelece uma conexão entre os navegadores e a plataforma Java permitindo executar *applets* nas páginas Web. Porém, considerando-se que os *applets* são largamente utilizados em páginas da Web, os navegadores mais comumente utilizados já implementam esta funcionalidade.

Em relação o mecanismo para a visualização gráfica do caminho navegacional percorrido pelo aluno há a necessidade do desenvolvimento de módulos que permitam sua integração com a base de dados dinâmica gerada pela ferramenta de aquisição.

5.2 Trabalhos Futuros

Se considerarmos a possibilidade de associar a base de dados gerada pela ferramenta com outras informações coletadas, provenientes de cadastros previamente elaborados, teremos uma grande fonte de dados para a extração de informações. Um exemplo seria a implementação de uma base de dados do perfil do aluno. Estes dados seriam previamente cadastrados e forneceriam informações tais como: idade, sexo, formação acadêmica e objetivos a serem atingidos com o curso. Isso forneceria um suporte mais eficiente para a elaboração de processos para descoberta de conhecimento (*data mining*), como por exemplo a extração de regras, que correlacionem um conjunto de variáveis com o conjunto dos dados de uso extraídos da Web, permitindo avaliar hipóteses tais como: "alunos de cursos de humanas tendem a explorar o caminho x enquanto os de exatas exploram o caminho y".

5.3 Considerações Finais

Com a disseminação da educação a distância baseada na Web uma série de mudanças têm sido apontadas como necessárias nos processos tradicionais de avaliação.

Como especificado no Capítulo 2, os ambientes de EAD têm deficiências no processo de avaliação seja do aluno, no que se refere a sua efetiva participação, ou do curso, no que se refere ao material instrucional, procedimentos de ensino, ritmo de aulas, dificuldades, etc. (Musa e Oliveira, 2000).

A característica não presencial desses ambientes dificulta verificar a qualidade e a eficácia do aprendizado da maneira como é realizada em ambientes tradicionais, onde há uma variedade maior de mecanismos tais como testes, trabalhos, exercícios, exposição oral e até mesmo a análise da expressão facial dos alunos.

Em ambientes de ensino a distância mediados por computador (EDMC) o acompanhamento efetivo dos alunos envolve, entre vários fatores, a análise das interações entre o aluno e o material, aluno e aluno ou aluno e professor.

No trabalho aqui relatado, foi procurado solucionar o processo de obtenção dos dados referentes à interação dos alunos com as páginas do curso, visto que as soluções disponíveis nem sempre são adequadas às aplicações na Web, podendo fornecer dados incompletos ou pouco confiáveis.

O mecanismo desenvolvido na primeira fase atendeu aos requisitos especificados, em relação a ser possível verificar a seqüência de utilização. Isto permitiu caracterizar o aluno, definindo interesses pedagógicos e estilos cognitivos diferentes, como solicitado pela equipe do curso de educação a distância Read in Web, utilizado como estudo de caso.

Como apresentado no Capítulo 3, algumas vantagens foram detectadas nesta ferramenta em relação às ferramentas comerciais disponíveis que também utilizam os arquivos de *log* dos servidores Web. O processo de tratamento e conversão aplicado aos dados extraídos destes *logs* permitiu um volume menor de dados a ser manipulado em comparação com o arquivo original, além de uma melhor compreensão do caminho percorrido devido à conversão do formato de URL para texto em linguagem natural.

Embora o trabalho tenha atingido bons resultados, algumas limitações foram identificadas nesta primeira fase. As desvantagens associadas a esta implementação motivaram a proposta de desenvolvimento de um mecanismo mais eficiente, especificado e implementado na segunda fase da pesquisa.

Esta nova implementação, gerada na segunda fase do trabalho, mostrou-se mais eficiente e dinâmica apresentando desvinculação com os arquivos de *log* dos servidores

Web, fornecendo independência e flexibilidade, além de solucionar os problemas encontrados no protótipo desenvolvido na primeira fase.

Apesar do trabalho ter sido direcionado para aplicação em um ambiente de ensino a distância mediado por computador, este mecanismo pode ser inserido em qualquer aplicação na Web onde se deseje monitorar o comportamento do usuário.

Podemos concluir, por este estudo, que o rastreamento das atividades realizadas pelo aluno em um ambiente de educação a distância mediada por computador pode ser útil para auxiliar a análise da qualidade e efetividade de um curso, assim como o nível de aprendizado do aluno.

A implementação do mecanismo apresentado neste trabalho poderá suprir a carência de ferramentas mais flexíveis que auxiliem a monitorar as interações na Web seja em ambientes educacionais como em qualquer outra aplicação na Web.

Referências

APACHE. Apache Organization

The common log file format

disponível em www.apache.org/docs/logs.html

acessado em Julho/2004

ADRIANO, C. M.; DELGADO, A.L.N.; DA SILVEIRA JR, L.; BOSNARD, R.; RICARTE, I. L. M.; MAGALHÃES, L.P. (1999).

Inquiring the course paradigm with CALM.

In: Proceedings of the First International Conference on Engineering and Computer Education (Hugo Hernández Figueroa, editor), pgs. 265-269. IEEE/ABENGE, Rio de Janeiro, RJ.

AULANET

Home page do Aulanet

disponível em http://asgard.les.inf.puc-rio/aulanet

acessado em Julho/2003

BARCELLOS, M. V. (1998)

Uma Proposta de Educação a Distância Mediada por Computador (EDMC) para Cursos de Graduação.

Dissertação de Mestrado – Instituto de Informática/PUC Campinas 1998

BARRET, B. L. (2001).

The Webalizer.

URL http://www.webalizer.com/.

acessado em Julho/2003

BLOIS, M. ; CHOREN, R.; LAUFER, C.; FERRA, F.; FUKS, H. (1999).
Desenvolvendo Aplicativos para a Web com o Scriba
Anais do XXVI SEMISH – Seminário Integrado de Hardware e Software,
Sociedade Brasileira de Computação (SBC), Rio de Janeiro, Julho 1999,
pgs 119-133.

BRODER, A. (2000)
Data Mining, the Internet, and Privacy
WEBKDD99 - Lecture Notes in Artificial Intelligence V1836 – pgs 56-73,
Springer-Verlag Berlin Heilderberg 2000.

CERI, S.; FRATENALI, P.; PARABOSHI, S. (1999).
Design Principles for Data Intensive Web Sites
Sigmod Record Vol 28, no. 1, March 1999

COOLEY, R.; MOBASHER, B. ; SRIVASTAVA, J. (1997).
Grouping Web Page References into Transactions for Mining World Wide Web
Browsing Patterns.
Proceedings of the 1997 IEEE Knowledge and Data Engineering Exchange
Workshop, Nov. CA, USA

COOLEY, R.; MOBASHER, B. ; SRIVASTAVA, J. (1997b).
WebMining: Information and Pattern Discovery in Worl Wide Web
Technical Report – TR 97-027 University of Minnesota , 1997

COOLEY, R.; TAN, P.; SRIVASTAVA, J. (1999)
Discovery of Interesting Usage Patterns from Web Data
WEBKDD99 - Lecture Notes in Artificial Intelligence V1836 – pgs 163-182,
Springer-Verlag Berlin Heilderberg 2000.

COOLEY, R. ; MOBASHER, B. ; SRIVASTAVA , J. (1999b).
Data preparation for mining World Wide Web browsing patterns
Journal of Knowledge and Information Systems 1 – Springer Verlag 1999

CUNHA, L.; FUKS, H.; LUCENA, C.J.P. (2002).
Sistema Multi-Agente e Instrução Baseada na Web
Electronic Proceedings of the 7th International Conference on Engineering and Technology Education - INTERTECH – Santos-SP Brasil Março 17-20

FRANCO, M.; BARBETTI, D. (1998).
Desenvolvimento de cursos on-line usando WebCT
Disponível em www.ccuec.unicamp.br/treinamentos/webct
Acessado em Julho/2003

FUKS, H.; GEROSA, M.; LUCENA, C.J.P. (2001).
Sobre o Desenvolvimento e Aplicação de Cursos Totalmente a Distância na Internet.
Revista Brasileira de Informática na Educação, No. 9, Setembro 2001
Sociedade Brasileira de Computação, pgs 61-75

FUKS, H.; GEROSA, M.; LUCENA, C.J.P. (2002).
Usando a Categorização e Estruturação de Mensagens Textuais em Cursos pelo Ambiente AulaNet
Revista Brasileira de Informática na Educação, No.10, Abril 2002
Sociedade Brasileira de Computação, pgs 33-44

FUKS, H.; GEROSA, M.; LUCENA, C.J.P.; PIMENTEL, M.;RAPOSO A. B.; MITCHELL, L.H.R.G. (2003).
Evoluindo para uma Arquitetura de Groupware Baseada em Componentes: O estudo de Caso do Learningware AulaNet

III Workshop de Desenvolvimento Baseado em Componentes – WDBC 2003
São Carlos - SP, 10 a 12 setembro 2003

GOLDBERG, M.; SALARI, S. (1997).
An Update on WebCT – A tool for the creation of Sophisticated Web-Based Lerning Environments.
Disponível em www.cs.ubc.ca/wccce/program97/murray/murray.html
Acessado em Julho 2003

HACK, L. (2000).
Mecanismos Complementares para a avaliação do Aluno na Educação a Distância.
Tese de Mestrado – Universidade Federal do Rio Grande do Sul

HENTEA, M.; SHEA, M.; PENNINGTON, L. (2003).
A Perspective on Fullfilling the Expectations of Distance Education
Conference on Information Technology Curriculum 4
October 16-18

HORSTMAN, C.; CORNER, G. (2000).
Core Java2, V1 Fundamentos
Editora Makron Books – 1ª. Edição 2000

IBM – Developers (2004).
Building servlets with session tracking – Tutorial
disponível em www.ibm.com/developersWorks
acessado em Julho/2004

JAVA SUN – 2004

The Life Cycle of an Applet

Disponível em Java.sun.com/docs/books/tutorial/applet/overview/lifeCycle.html

Acessado em outubro/2004

LACHI, R.; OTSUKA, J.; ROCHA, H. (2002).

Uso de agentes de interface para auxiliar a avaliação formativa no ambiente TelEduc.

Anais do XIII Simpósio Brasileiro de Informática na Educação (SBIE 2002)

São Leopoldo 12-14 Novembro.

LARMAN, G. (2000).

Utilizando UML e Padrões

Editora Bookman, 2000 – Porto Alegre

pgs 269-275

LOCATELLI, A. R. (2004).

E-learning e Hipermídia

Disponível em

www.elearningbrasil.com.br/comunidade/seu_espaco/Files/angela_rosa.doc

LOYOLLA, W.; PRATES, M. (1998).

Educação à distância mediada por computador (EDMC): Diretrizes de projetos para Pós-Graduação.

IV Congresso RIBIE, Brasília ,1998.

MAES, P. (1994).

Agents that Reduce Work and Infomation Overload

Communications of the ACM July, 1994/ Vol 37, No. 7, pgs 31-40

MASAND, B.; SPILIOUPOULOU, M. (2000).
Web Usage Analysis and user Profile
Lecture Notes in Computer Science V.1836 pgs 1-6 , 2000
Springer-Verlag

MENEZES, R.; FUKS, H.; GARCIA, A. C. (1998).
Utilizando Agentes no Suporte à avaliação Informal no Ambiente de Instrução Baseada na Web – AulaNet
IX Simpósio Brasileiro de Informática na Educação, Fortaleza, Novembro

MOBASHER, B.; DAI, H.; LUO, T. (2000).
Integrating Web Usage and Content Mine for More Effetive Personalization
In: Proceedings of the International Conference on E-Commerce and Web Technologies , LNCS – Lecture Notes in Computer Science, v1875, Springer-Verlag, pgs. 165-176, London, Sep.

MORREALE, P. (1998).
Agents on the Move
MIT Media Lab – Autonomous Agent Group
IEEE Spectrum, Abril 1998

MURTAGH, F.; TAO, F.(2000).
Towards Knowledge Discovery from WWW Log Data
Proceedings of the International Conference on Information Technology: Coding and Computing

MUSA, D.; OLIVEIRA, J. (2000).
Alertas Inteligentes na Educação à Distância
Disponível em
www.inf.ufgrs/pos/SemanaAcademica/Semana2000/DanielaMusa/

acessado em Nov/2003.

MUSA, D.; BICA, F.; OLIVEIRA, J.; VICARI, R. (2001).
Agentes para a avaliação de aprendizagem em ambientes de ensino na Web.
Universidade Federal do Rio Grande do Sul.
Disponível em www.inf.ufes.br/~sbie2001/figuras/artigos
acessado em Outubro/2003

OTSUKA, J.; LACHI, R.; FERREIRA, T.; ROCHA, H. (2002).
Suporte à Avaliação Formativa no Ambiente de Educação a Distância TelEduc
Anais do VI Congresso Ibero Americano de Informática Educativa
20-22 Novembro, 2002

PEREIRA, A.; GEYER, C. (2000).
Um agente para a seleção de estratégias de ensino em ambientes
educacionais na Internet.
SBIA – Iberamia Simpósio Brasileiro de I.A. - Atibaia - SP , Nov 2000

PEROSA, G.; SANTOS, M. (2003).
Interatividade e aprendizagem colaborativa em um grupo de estudo online
Educação Online
Editora Loyolla – São Paulo Brasil – 2003

RAMON, F. (2002).
Acesso a banco de dados usando a linguagem Java – Guia de Consulta Rápida
Editora Novatec - 2002

READWEB (2003).
Sítio do curso Read in Web
Disponível em www.ead.unicamp.br/readweb

Acessado em Maio/2003

ROMANI, L. (2000).
Intermap: Ferramenta para Visualização da Interação em ambientes de educação à distância na Web
Dissertação de Mestrado – Instituto de Computação/Unicamp

RUSSEL, S. ; NORVIG, P. (1995).
Artificial Intelligence : a Modern Approach
Editora Prentice-Hall do Brasil pg 31

SAPIENS (2001)
Relatório final
Disponível em
www.dca.fee.unicamp.br/projects/sapiens/Reports/rf2000/node40.html
Acessado em Janeiro/2003

SHAHABI, C.; BNAEI-KASHANI, F.; FARUQUE, J. (2001).
A Reliable, Efficient, ans Scalable System for Web Usage Data Acquisition
In Procedings of the WebKDD2001 Workshop in conjunction with the
ACM- SIGKDD 2001, San Francisco, CA, Aug.

SHERRY, L. (1996).
Issues in Distance Learning
International Journal of Education Telecommunications
Vol 4, pgs 337-365

SOUTO, M.A.; BICA, F.; WARPECHOWSKIM, M.; VICARI, R.; OLIVEIRA, J.P.; ZANELLA, R. e SONNTAG, A. (2000).

Ferramentas de Suporte a Monitoração do Aluno em um Ambiente Inteligente de Ensino na Web.
Disponível em www.inf.ufrgs.br/~palazzo/docs/artigos
Acessado em Abril/2003

SOUZA, T. R.P. (2002).
Avaliação como Prática Pedagógica
CEAD – Universidade de Brasília
Disponível em http://www.abed.org.br/publique/cgi/cgilua.exe/sys/start.html

SPILIOPOULOU, M.; FAULSTICH, L. (1998).
WUM: A Web Utilization Miner
in: Proceedings of the EDBT Workshop, WebDB98,
LNCS – Lecture Notes in Computer Science
v.1590 pg 109-115 Springer-Verlag, Valencia, Spain

SPILIOPOULOU M.; FAULSTICH, L.; KARSTEN, W. (1999).
A Data Miner Analyzing the Navigational Behaviour of Web Users
in Proceedings of the Workshop on Machine Learning in User Modeling of the ACAI'99 Int. Conf., Creta,Greece, July 1999

SUN Corporation (2003).
Sun MicroSystems do Brasil - Políticas de Privacidade
Disponível em http://br.sun.com/privacy
Acessado em Julho/2003

TAROUCO, L. R. (2000).
O processo de avaliação na educação a distância.
Webfólio Educação à Distância. UFRGS, Out. 2000
Disponível em www.pgie.ufrgs/webfolioead/biblioteca/artigo6

TELEDUC (2004).

Teleduc – Página do projeto

Disponível em www.teleduc.nied.unicamp.br/teleduc/sobre30.php

Acessado em Outubro/2004

WEBTRENDS Corporation, (2001).

Webtrends homepage.

URL http://www.webtrends.com/.

Acessado em Novembro/2001