Victor Fernando Ribeiro
Orientador: Prof. Nívio Ziviani

A FAMÍLIA MINER DE AGENTES PARA A WORLD-WIDE WEB

Dissertação apresentada ao Departamento de Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Belo Horizonte
1998

Ao Xerife, Quito e Puppy.
Estejam bem, seja onde estiverem.

Agradecimentos

À Alice pelas incansáveis revisões do texto, pela compreensão, amor e carinho e por ser a pessoa maravilhosa que ela é.

À Mãe, Pai, Paula, Renato, Helvécio e Raquel pelo apoio e carinho.

À Companhia Siderúrgica Belgo-Mineira, em especial ao Antônio Eustáquio e ao Jáder, pela abertura do espaço que me permitiu continuar meus estudos.

Aos professores Nívio e Berthier pelo estímulo, serenidade, críticas e auxílio no desenvolvimento deste trabalho.

À Vanessa e ao Emílio pelo ombro e ouvidos amigos nos momentos de alegria e tristeza.

Ao Edleno, Ramurti, Fabiana, Tiago, Pável e Rui pela parceria na construção e testes de alguns dos agentes da *Família Miner*.

Ao Calvin e Haroldo pela diversão e descontração.

E a todos aqueles que, de uma forma ou de outra, contribuíram para a elaboração deste trabalho, em especial, à Dona Rosa, seu Guido, Aline, Rebecca e ao pessoal do Latin: Kita, Drumond, Marcos, Albener e Autran.

Resumo

A *Família Miner de Agentes para a World-Wide Web* é um conjunto de ferramentas que objetiva auxiliar usuários a localizarem informações na Internet. A *Família Miner* é baseada na API Miner e dividida em quatro grupos maiores de agentes: agentes de busca – utilizados para a procura de informações genéricas na *Web*; agentes de compra – utilizados para localizar e comparar preços de produtos vendidos *online*; agentes de notícias – utilizados para coletar e elaborar *clippings* de notícias veiculadas *online* na *Web*; e agentes de monitoramento – utilizados para gerenciar e monitorar toda a *Família Miner*.

Abstract

The Miner Family of Agents for the World-Wide Web is a tool set built to help people to find information in the Internet. The Miner Family uses the Miner API and is divided in four groups: search agents – used to find generic information in the Web; shopping agents – used to find and compare product prices of stores; news agents – used to collect and build news clippings from online available information; and monitor agents – used to manage and trace all the Miner Family agents.

Índice

Lista de Gráficos

Lista de Tabelas

Lista de Figuras

1. Introdução

1.1. MOTIVAÇÃO

O rápido crescimento do uso privado, corporativo e científico da *World-Wide Web* está levando a Internet a ser vítima de seu próprio sucesso. O tráfego, hoje, é tão intenso na rede que a atual administração do governo americano está anunciando a construção da Internet II, apenas para que os cientistas possam prosseguir com seus trabalhos e estudos [Lynch,1997]. O Brasil segue também o mesmo caminho, iniciando movimentos para a separação da rede privada da rede educacional.

Hoje, é possível encontrar quase todo tipo de informação dentro da Internet. Porém, a *Web* não foi planejada para suportar publicações organizadas e recuperação de informações de forma coordenada, o que faz com que até mesmo especialistas gastem uma grande quantidade de tempo para acessar e recuperar um recurso de informação desejado. Simultaneamente, diversas pessoas começaram a organizar páginas na Internet onde seus endereços preferidos eram relacionados, dando início aos atuais diretórios – como o *Yahoo, Cadê?* e tantos outros. No entanto, devido à necessidade do cadastro manual de cada página a ser adicionada ao índice, os diretórios mostram-se incapazes de lidar com o crescimento exponencial da *World-Wide Web*. Surgiram então, os motores de busca – como o *Altavista, Lycos, Excite* – que passaram a utilizar programas de computador, conhecidos como robôs, para automatizar a indexação e classificação da vasta quantidade de informações digitais presentes na *Web*.

Os motores de busca possuem a vantagem de explorar a Internet de maneira rápida e a um custo financeiro baixo, trabalhando de forma democrática através do acesso uniforme a todas as informações disponíveis. Por outro lado, os motores de busca geram enorme tráfego na rede e apresentam problemas na recuperação de documentos relevantes. Já os diretórios, que são mantidos apenas com a intervenção humana, possuem uma melhor organização das informações.

A efetivação de um número cada vez maior de bancos de dados textuais, juntamente com o aumento da interatividade da *Web,* estão aumentando o dinamismo das páginas e documentos disponíveis [Lynch,1997]. Isto impossibilita o trabalho dos atuais robôs que não são capazes de imitar um usuário e não permitem que os diretórios possam indexar todas as informações existentes.

O objetivo deste trabalho é introduzir um novo conjunto de agentes de busca para a Internet, denominado *A Família Miner de Agentes para World-Wide Web*, pretendendo prover o usuário com ferramentas que facilitam o acesso a uma informação procurada, seja ela o preço de um livro, as notícias do dia, um novo lançamento musical ou mesmo um documento qualquer disponível na Internet.

A ferramenta tem como característica principal a capacidade de busca em paralelo a diferentes provedores de conteúdo, tal como o MetaCrawler apresentado em [Selberg & Etzione,1995]. Além disso, expande o conceito de agentes de compras como o *BargainFinder* da Andersen Consulting[1] e aproxima-se do conceito do agente conhecido como *Jango* da Netbot[2]. Além de ser o primeiro conjunto brasileiro de meta ferramenta de busca, a *Família Miner* também apresenta outras novidades como a união das boas características dos diretórios e dos motores de busca, envolvendo aspectos de classificação manual de banco de dados com aspectos de recuperação automática da informação.

A *Família Miner* pretende também servir como ambiente para o desenvolvimento e teste de técnicas de recuperação de informação [Frakes & Baeza-Yates,1992; Salton,1983] no Laboratório para Tratamento da Informação (Latin) do Departamento de Ciência da Computação da UFMG, estando também ligada aos projetos: *AMYRI – Ambiente para Manipulación y Recuperación de Información en WWW* [AMYRI, 1997] e *SIAM – Sistemas de Informação em Ambientes de Computação Móvel* [SIAM, 1997].

1.2. TRABALHOS RELACIONADOS

Os conceitos básicos da Internet e da *Web* são descritos no Capítulo 2. Tais conceitos são fundamentais para o desenvolvimento de qualquer *software* para a Internet. Em geral, podem ser encontrados em documentos conhecidos como RFC (*Request for*

[1] http://www.ac.com/aboutus/tech/cstar/research.html

[2] http://www.jango.com/

Comments) que em sua maioria foram escritos pela IETF (*Internet Engineering Task Force*), *World-Wide Web Consortium* e pelo CERN. As principais RFCs aqui citadas estão relacionadas nas referências bibliográficas e têm de uma forma ou de outra a participação de *Tim Berners-Lee*, pesquisador do CERN, líder do projeto que concebeu a *World-Wide Web*.

As principais referências para a criação de robôs estão em [Koster,1995; Koster,1996; Eichmann,1996] e nas listas *robots mailing list* (robots@mccmedia.com) e *AgentsUMBC* (agents@cs.umbc.edu), onde, além de abordar todos os tópicos dos dois livros relacionados a seguir, é possível também "encontrar" os responsáveis pelos principais robôs e agentes disponíveis na *Web* atualmente. No livro [Cheong,1996], a base para a criação de robôs é relatada juntamente com exemplos de agentes de compra conhecidos como *BookFinder* e *CDFinder*. O livro [Williams,1996], também específico em agentes e robôs, é formado por capítulos escritos por diferentes autores, cobrindo vários tópicos, tais como: a necessidade de robôs e agentes, informações técnicas (arquitetura, linguagens, ética), revisões sobre agentes existentes, análises e previsões sobre o futuro desses programas de computador.

A grande quantidade de informação relacionada sobre motores de busca é derivada em sua maioria de documentos encontrados na Internet e periódicos sobre informática. Em especial, [Sullivan,1997] exibe uma página especializada em motores de busca onde são encontradas as principais características, análises de desempenho e a situação no mercado de cada uma das maiores ferramentas de busca existentes.

O MetaCrawler [Selberg & Etzioni,1995] foi um dos primeiros serviços de busca a utilizar o conceito de **meta ferramenta de busca** – ferramenta de busca que não possui uma base de dados própria e que utiliza simultaneamente os serviços de diferentes motores de busca e diretórios para realizar as pesquisas submetidas pelos seus usuários. Diversos novos trabalhos sobre meta ferramentas de busca estão surgindo desde o MetaCrawler, como a ferramenta apresentada no trabalho de [Chang,1997] que utiliza técnicas de *clustering* ou a interface de visualização desenvolvida em [Smeaton & Crimmins,1996] que ajuda o usuário a melhor compreender a relação entre os resultados retornados pelas diversas ferramentas de busca pesquisadas simultaneamente.

Agentes como o *BargainFinder* da Andersen Consulting e o *Jango* da Netbot trazem conceitos semelhantes aos dos agentes de compra da *Família Miner*.

A principal referência para servidores *proxy* está em [Wessels,1995]. Esse trabalho é parte do projeto Harvest [Manber et al,1995] da Universidade do Colorado, que procurou estudar todo o contexto de busca e recuperação de informação na *World-Wide Web*, estabelecendo padrões que atualmente tem sido utilizados pelo *software* de indexação da Netscape.

Os diversos tópicos sobre recuperação de informação (RI) podem ser encontrados em [Salton,1983] e em [Frakes & Baeza-Yates,1992].

1.3. CONTRIBUIÇÕES DA DISSERTAÇÃO

A *Família Miner* é um conjunto de ferramentas que procura auxiliar o usuário na busca de informação na *World-Wide Web* e traz algumas inovações que abrem um amplo espaço para o estudo de técnicas de recuperação de informação tais como: indexação, classificação, *clustering* e *relevance feedback*. A título de exemplo, citamos o experimento que foi realizado integrando o sistema de classificação Gama [Calado et al,1997] e o *NewsMiner*, dentro do projeto AMYRI [AMYRI, 1997], que será detalhado mais adiante. A seguir detalhamos as contribuições específicas de cada grupo da *Família Miner*.

Agentes de Busca – grupo onde está a primeira meta ferramenta de busca brasileira, o MetaMiner que utiliza ferramentas de busca nacionais e estrangeiras e que pode auxiliar os milhões de brasileiros usuários da Internet a encontrar informações na *Web*. O PSSEMiner, outro agente de busca da *Família Miner*, mostra uma nova forma de coleta automatizada de páginas derivada do uso combinado de robôs com servidores *proxy*, que poderá ajudar na economia de tráfego nos *backbones* da Internet.

Agentes de Compra – grupo de agentes da *Família Miner* que agem como intermediários entre os consumidores e lojas virtuais, beneficiando ambos. Às lojas, através do aumento da facilidade para encontrar os produtos oferecidos. Aos consumidores, através da facilidade para encontrar um produto e para comparar preços em lojas diferentes.

Agentes de Notícias – grupo de agentes da *Família Miner* que apresenta um novo formato para recortes de notícias (*clippings*) a um baixo custo de elaboração e distribuição. Permitindo que qualquer usuário da Internet possa receber, através do

correio eletrônico, um jornal formado pelas as notícias consideradas mais relevantes de acordo com o perfil elaborado para esse usuário previamente. O conceito do NewsMiner, o agente de notícias de jornais da *Família Miner*, pode ser expandido para englobar revistas sobre tópicos específicos ou ainda para focar em cadernos especiais dos jornais – como os de agropecuária e informática.

Agentes de Monitoramento – grupo de agentes da *Família Miner* do qual faz parte o *TraceMiner,* um agente especial que possui inúmeras possibilidades para o estudo de perfil e obtenção de *feedback* automático dos usuários da Internet.

Outra importante contribuição é a API Miner – a base para construção da *Família Miner*. Sua concepção e modelagem orientada a objetos permitem a expansão das atuais classes, que envolvam outros tópicos e serviços, para desenvolver outros agentes para *Família Miner*.

1.4. ORGANIZAÇÃO DA DISSERTAÇÃO

Essa dissertação está dividida da seguinte forma:

- O Capítulo 1 é uma introdução geral onde se encontra a motivação, citação de trabalhos relacionados além das contribuições dessa dissertação;
- No Capítulo 2 são apresentados conceitos e teorias básicas sobre a Internet, ressaltando principalmente os conhecimentos necessários para o desenvolvimento de robôs para a *Web*;
- No Capítulo 3 é descrita a implementação da API Miner conjunto de bibliotecas e pacotes utilizados por todos os membros da *Família Miner*;
- No Capítulo 4 a *Família Miner* é descrita e detalhes de seus principais membros são descritos;
- O Capítulo 5 contém resultados de diversos experimentos realizados com a *Família Miner;*
- O Capítulo 6 contém a conclusão do trabalho bem como potenciais trabalhos e estudos futuros.

2. A Internet e a World-Wide Web

A Internet teve seu início por volta do início da década de 80 quando a *DARPA (Defense Advanced Research Project Agency* – antiga ARPA) começou a converter as máquinas da ARPAnet para o protocolo TCP/IP. Essa transição foi completada em janeiro de 1983, quando todas as máquinas da ARPAnet passaram a rodar o TCP/IP. Nesse mesmo período, a ARPAnet foi dividida em duas redes: a própria ARPAnet que continuaria com trabalhos de pesquisa científica e a MILNET que serviria apenas para operações militares. No início de 1986, a *National Science Foundation* (NSF) forneceu recursos para que fossem estabelecidas conexões entre redes de porte médio de agências governamentais, instituições de ensino e negócios comerciais. Esse foi o primeiro passo para acelerar o rápido crescimento da Internet.

A *World-Wide Web* (WWW ou *Web*) foi iniciada em outubro de 1990 a partir de um projeto do CERN (*European Laboratory for Particle Physics*), liderado por Tim Berners-Lee. O propósito era construir um "sistema de distribuição de hipermídia[3]". Na prática, a *Web* é uma vasta coleção de documentos hipertextos interligados entre si. Uma das vantagens dos hipertextos é que se um usuário deseja obter mais informações sobre um determinado assunto, então, basta clicar sobre o texto do assunto para ler mais detalhes. Assim, documentos podem estar interligados entre si, mesmo que sejam de diferentes autores, ou ainda, que estejam fisicamente separados.

2.1. UNIFORM RESOURCE LOCATIONS – URLS

Uniform Resource Locations (URLs) [Berners-Lee et al, 1994] são a forma sintática e semântica para representar formalmente a localização e a maneira de acessar recursos via Internet. A especificação das URLs é derivada de conceitos de objetos de informações globais introduzidos pela *World-Wide Web*, descritos em [Berners-Lee, 1994].

[3] Hipermídia é um super-conjunto de hipertextos – sendo qualquer tipo de mídia com apontadores para outras mídias. Dessa forma, um documento hipermídia pode conter imagens que referenciam sons ou vídeos, por exemplo.

Em geral, URLs são escritas da seguinte forma:

<forma>:<parte-específica-da-forma>

Uma URL contém o nome da forma que está sendo utilizado, seguido por dois pontos "`:`" e então por uma cadeia de caracteres (`<parte-específica-da-forma>`) cuja interpretação depende da forma. Entretanto, as formas de URL que envolvem o uso de um protocolo baseado em endereços IP para acessar servidores específicos na Internet, utilizam uma sintaxe comum:

//<usuário>:<senha>@<servidor>:<porto>/<caminho>

Algumas, ou todas, as partes de "`<usuário>:<senha>@`", "`:<senha>`", "`:<porto>`", e "`/<caminho>`" podem ser excluídas. Os diferentes componentes da sintaxe acima seguem as seguintes regras:

- `<usuário>` – nome opcional de um usuário;
- `<senha>` – senha opcional, se presente, estará seguindo o nome de um usuário;
- `<servidor>` – nome completo qualificado de um servidor de rede [Mockapetris, 1987; Braden, 1123], ou seu endereço IP;
- `<porto>` – número do porto do servidor para a conexão, quando não especificado na URL, o porto padrão da forma deve ser utilizado;
- `<caminho>` – o resto da URL consiste de informação específica da forma utilizada, contendo detalhes de onde o recurso especificado está localizado no servidor.

As formas abrangidas pelas URLs são descritas na Tabela 1.

Tabela 1: Formas abrangidas por URLs

Forma	Descrição
`ftp`	File Transfer Protocol
`http`	Hypertext Transfer Protocol
`Gopher`	Gopher Protocol
`mailto`	Electronic mail address
`News`	USENET news
`nntp`	USENET news using NNTP access
`telnet`	Reference to interactive sessions
`wais`	Wide Area Information Servers
`File`	Host-specific file names
`prospero`	Prospero Directory Service

As URLs do HTTP, por exemplo, são utilizadas para designar recursos da Internet que são acessíveis através do HTTP [Berners-Lee, 1996; Fielding, 1997]. Uma URL HTTP pode ser escrita da seguinte maneira:

`http://<servidor>:<porto>/<caminho>?<consulta>`

onde:

- `<servidor>` e `<porto>` representam o servidor e o porto do recurso, como especificado acima, sendo 80 o `<porto>` padrão do HTTP;
- `<usuário>` e `<senha>` não são permitidos;
- `<caminho>` é um seletor HTTP;
- `<consulta>` é uma cadeia de caracteres que especifica uma consulta a ser executada no servidor. `<consulta>` deve ser sempre precedida pelo sinal de interrogação "`?`";

Tanto `<caminho>` quanto `<consulta>` são opcionais. Se nenhum dos dois aparecerem, a última barra "`/`" poderá ser omitida. Dentro de `<caminho>` e

`<consulta>` os caracteres: "`/`", "`;`" e "`?`" são reservados. O caracter barra "`/`" é utilizado dentro do HTTP para designar uma estrutura hierárquica.

Um exemplo de uma URL HTTP seria:

```
http://info.webcrawler.com/mak/projects/robot/robots.html
```

Esta é uma referência ao documento `robots.html` localizado dentro da estrutura hierárquica `/mak/projects/robot` de um servidor HTTP que está disponível em `info.webcrawler.com` (nome do servidor), no porto padrão.

2.2. HTTP

O Protocolo de Transferência de Protocolo (*HyperText Transfer Protocol* – HTTP) [Berners-Lee, 1996; Fielding, 1997] é um protocolo da camada de aplicação desenhado para possibilitar velocidade e características necessárias para distribuir e disponibilizar sistemas de informação hipermídia. É um protocolo genérico que pode ser utilizado para diversas tarefas que são executadas através de extensões de seus métodos de requisições (`request-methods`). O HTTP tem sido utilizado pela *Word-Wide Web* desde 1990 sendo a versão mais recente desse protocolo a 1.1 [Fielding, 1997].

O protocolo HTTP é baseado em um paradigma de requisição/resposta descrito a seguir:

- O cliente estabelece uma conexão com um servidor e envia uma solicitação através de um método de requisição formado por uma URI, pela versão do protocolo e por uma mensagem MIME (*Multipurpose Internet Mail Extensions*) [Freed, 1996; Borenstein, 1996; Moore, 1996; Freed et al, 1996; Freed & Borenstein, 1996] que contém modificadores do método de requisição, informações do cliente e possivelmente algum conteúdo;

- O servidor responde com uma linha de *status* que contém a versão do protocolo da mensagem e um código indicando erro ou sucesso. Seguindo essas informações vem uma mensagem MIME contendo dados do servidor, entidades, meta-informação, e possivelmente algum conteúdo.

Uma conexão simples entre um cliente e um servidor é apresentado na Figura 1.

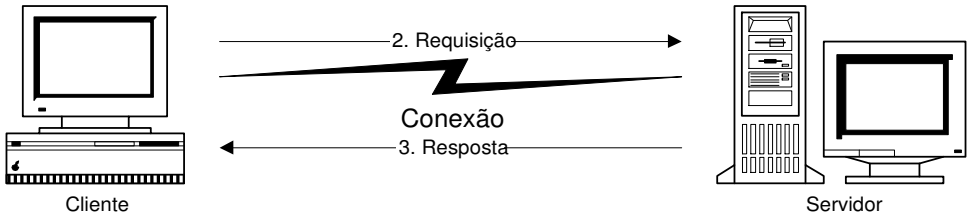


Figura 1: Ilustração de uma conexão HTTP

A conexão HTTP pode ser resumida em quatro estágios:

1. **Abrir conexão**: o cliente (`User-Agent`[4]) contata o servidor em um domínio e porto especificado (que podem estar dentro de uma URL HTTP);

2. **Requisitar informações**: o cliente envia uma mensagem para o servidor requisitando serviços através de um conjunto de modificadores da requisição e possivelmente algum conteúdo que, juntos, definem a transação a ser realizada;

3. **Enviar resposta**: o servidor recebe a requisição, processa e envia uma resposta para o cliente com um campo denominado cabeçalho de resposta (onde o *status* da transação, informações do servidor, o tamanho do conteúdo da resposta e outras informações adicionais são informadas) e possivelmente algum conteúdo (de acordo com a requisição);

4. **Fechar conexão**: a conexão é finalizada.

Situações mais complexas podem ocorrer quando um ou mais servidores são envolvidos como intermediários entre o servidor e o cliente durante o processo de requisição/resposta. Isso pode ocorrer quando é desejável fazer com que computadores de redes particulares tenham acesso à Internet, como no caso do servidor *proxy* que será abordado mais à frente.

[4] `User-Agent` é a denominação técnica a um atributo que identifica o *software* cliente em uma rede.

2.2.1. Métodos de Requisição

Os métodos de requisição (`request-methods`) definem a forma que um pedido deverá ser feito a um servidor por um cliente. Alguns desses métodos são:

- `GET` – requisita um documento a um servidor HTTP. Dessa forma, o cliente receberá um conjunto de modificadores da resposta (Seção 2.2.3) juntamente com algum conteúdo.
  Exemplo: `GET /~victor/exemplo.html HTTP/1.0`

- `HEAD` – o mesmo que o `GET`, porém apenas o conjunto de modificadores da resposta é enviado (o conteúdo não é enviado). Esse método pode ser utilizado para verificar se um documento foi ou não atualizado.
  Exemplo: `HEAD /~victor/exemplo.html HTTP/1.0`

- `POST` – utilizado para enviar uma consulta para um servidor. Apesar do método `GET` também poder ser utilizado para esse tipo de conexão, o `POST` possui a vantagem de incluir um modificador especial da requisição (Seção 2.2.2) que servirá para verificar se a consulta foi recebida corretamente pelo servidor.
  Exemplo: `POST /~victor/query.cgi HTTP/1.0`

2.2.2. Modificadores da Requisição

Os modificadores da requisição (`request-headers`) servem como meio para levar informações do cliente para o servidor, tais como restrições e identidade, entre outros. Alguns dos modificadores da requisição mais utilizados são:

- `Accept:` *`type/subtype`* – contém uma lista dos tipos MIME que são aceitos pelo cliente.
  Exemplo: `Accept: text/html`

- `FROM:` *`mail_address`* – contém um endereço eletrônico para referenciar o cliente que está acessando o servidor.
  Exemplo: `FROM: victor@dcc.ufmg.br`

- `If-Modified-Since:` *`date_time`* – usado junto com o método `GET`. Caso o documento requisitado não tenha sido modificado desde a data indicada, então não é enviado.
  Exemplo: `If-Modified-Since: Tue, 21 May 1996 19:55:44 GMT`

- `Referer:` *`URL`* – contém uma URL que deverá possuir informações sobre o cliente.
  Exemplo: `Referer: http://www.miner.dcc.ufmg.br/miner-info.html`

- `User-Agent:` *`program/version comments`* – contém informações resumidas para descrever e identificar o cliente.
  Exemplo: `User-Agent:Mozilla/2(compatible;miner;victor@ufmg.br)`

Assim, uma mensagem completa enviada a um servidor por um dos agentes da *Família Miner*, seria:

```
GET /psseminerbot.html?q=information+retrieval&t=and HTTP/1.0
Accept: text/html
Referer: http://www.miner.dcc.ufmg.br/miner-info.html
User-Agent: Mozilla/2(compatible;metaminer;victor@dcc.ufmg.br)
From: victor@dcc.ufmg.br
```

2.2.3. Modificadores da Resposta

Os modificadores da resposta (`response-headers`) representam a forma que o servidor utiliza para informar o cliente sobre como sua requisição foi processada. Alguns dos modificadores da resposta mais utilizados são:

- `http_version status_code explanation` – (linha de *status*) a primeira linha de uma resposta de um servidor HTTP contém informações sobre a versão do protocolo utilizado e do *status* da conexão.
  Exemplo: `HTTP/1.0 200 OK`

  Onde `HTTP/1.0` é a versão do protocolo utilizado pelo servidor, `200` é o código do *status* da transação (pode variar de `200` a `599`) e `OK` é uma cadeia de caracteres que contém uma explicação descritiva do *status* da conexão. Códigos

entre `200` e `299` indicam sucesso, entre `300` e `399` indicam redireção, e entre `400` e `599` indicam erro.

- `Date:` *`date_time`* – contém data e hora em que a requisição foi processada no servidor.
  Exemplo: `Date: Tue, 28 May 1996 16:23:28 GMT`

- `Server:` *`name/version`* – contém o nome e a versão do servidor.
  Exemplo: `Server: Apache/1.0.3`

- `Content-length:` *`length`* – contém o tamanho, em *bytes*, da fração da mensagem que contém a informação requisitada.
  Exemplo: `Content-length: 12541`

- `Last-modified:` *`date_time`* – contém a data e hora em que o documento foi modificado pela última vez.
  Exemplo: `Last-modified: Tue, 21 May 1996 19:55:44 GMT`

Dessa forma, uma mensagem recebida por um dos agentes da *Família Miner* seria:

```
HTTP/1.0 200 OK
Date: Tue, 28 May 1996 16:23:28 GMT
Server: Apache/1.0.3
Last-modified: Tue, 21 May 1996 19:55:44 GMT
Content-length: 12541
<linha em branco>
<corpo da mensagem com tamanho de 12541 bytes>
```

2.2.4. Cookies

Geralmente, os servidores HTTP não relacionam uma requisição de um cliente com nenhuma outra anterior ou subseqüente. No entanto, algumas vezes, é necessário que informações sejam trocadas entre o cliente e o servidor HTTP, durante várias conexões, HTTP para que certas tarefas sejam realizadas.

Por exemplo, suponha que uma pessoa abra uma página na *Web* de uma livraria virtual e queira comprar diversos títulos. Essa pessoa terá que pesquisar pelos livros

desejados um a um para depois concluir a compra. Para que a livraria *on-line* saiba quais os livros cada cliente já selecionou, ela terá que manter os livros escolhidos em "cestas de compras" personalizadas. Isso pode ser feito de diversas formas. A mais comum é a geração de uma "chave" única, na conexão inicial à livraria, que será enviada e lida no meio dos documentos HTML que serão transmitidos em cada uma das novas requisições realizadas pelo cliente.

Neste contexto, surgiram os *cookies* [Montulli, 1997; Kristol,1997], pequenos arquivos em **formato texto** que são enviados dentro dos modificadores da requisição e da resposta para estabelecer uma conexão completa entre um cliente e um servidor. Os *cookies* são gravados na máquina do cliente (quando autorizado pelo usuário), possibilitando que um provedor de conteúdo possa "mapear" o que uma pessoa está fazendo em seu *site*. Esse fato tem causado certo receio por parte de diversos usuários da *Web* que temem pela perda do anonimato. Entretanto, um servidor pode ler apenas os *cookies* que ele mesmo enviou, não tendo acesso a nenhuma outra informação do cliente, nem mesmo a outros *cookies* enviados por outros servidores. Ainda assim, o usuário pode sempre configurar seu *navegador* para recusar qualquer *cookie* que esteja sendo enviado ou, ao menos, para ser alertado sempre que um novo *cookie* estiver sendo recebido.

O exemplo abaixo apresenta uma transação completa, com várias conexões HTTP entre um cliente e um servidor, representados por um usuário e uma loja virtual. Alguns detalhes de modificadores da requisição e da resposta foram omitidos. Assume-se que essa seja a primeira conexão entre o servidor e o cliente.

1. *Cliente faz uma requisição para o Servidor (Cliente → Servidor)*
   Usuário identifica-se através de um formulário.

```
POST /acme/login HTTP/1.1
[dados de um formulário identificando o usuário]
```

2. *Servidor responde a requisição do Cliente (Servidor → Cliente)*
   Um *cookie* é enviado para o cliente pelo servidor contendo informações sobre a identidade do usuário. O cliente armazena o *cookie* no disco local do usuário no formato de um arquivo texto.

```
HTTP/1.1 200 OK
Set-Cookie: Customer="WILE_E_COYOTE"; Version="1";
  Path="/acme"
```

3. Cliente faz uma requisição para o Servidor (Cliente → Servidor)

Usuário seleciona um produto que deseja comprar.

```
POST /acme/pickitem HTTP/1.1
Cookie: $Version="1"; Customer="WILE_E_COYOTE";
  $Path="/acme"
[dados de um formulário selecionando um produto]
```

4. Servidor responde a requisição do Cliente (Servidor → Cliente)

Servidor "monta uma cesta de compra" contendo o produto selecionado no item anterior.

```
HTTP/1.1 200 OK
Set-Cookie: Part_Number="Lancador_de_Foquete";
  Version="1"; Path="/acme"
```

5. Cliente faz uma requisição para o Servidor (Cliente → Servidor)

Usuário seleciona a forma de expedição.

```
POST /acme/shipping HTTP/1.1
Cookie: $Version="1"; Customer="WILE_E_COYOTE";
  $Path="/acme"; Part_Number="Lancador_de_Foquete";
  $Path="/acme"
[dados de um formulário selecionando a forma de
  expedição]
```

6. Servidor responde a requisição do Cliente (Servidor → Cliente)

Servidor responde aceitando o método de expedição.

```
HTTP/1.1 200 OK
Set-Cookie: Shipping="FedEx"; Version="1";
  Path="/acme"
```

7. Cliente faz uma requisição para o Servidor (Cliente → Servidor)

Usuário finaliza a ordem.

```
POST /acme/process HTTP/1.1
Cookie: $Version="1"; Customer="WILE_E_COYOTE";
  $Path="/acme"; Part_Number="Lancador_de_Foquete";
  $Path="/acme"; Shipping="FedEx"; $Path="/acme"
[dados de um formulário avisando o fim do pedido]
```

8. Servidor responde a requisição do Cliente (Servidor → Cliente)

A transação está completa. Observe que o cliente fez uma série de requisições para o servidor, recebendo uma atualização do *cookie* após cada uma delas.

```
HTTP/1.1 200 OK
```

Devido ao crescimento do comércio eletrônico, a freqüência da utilização dos *cookies* tem sido cada vez maior na *World-Wide Web*. Seu uso pode ser feito de diferentes formas. Ele pode ser utilizado para manter uma "cesta de compras" em uma loja *online* de forma que, quando um usuário concluir uma compra, não seja necessário selecionar novamente todos os produtos previamente escolhidos. Pode também ser usado para monitorar e montar o perfil do usuário, permitindo assim que o provedor possa escolher qual a melhor propaganda a ser enviada em uma próxima requisição, aumentando, dessa forma, o valor agregado da veiculação dessa propaganda.

2.3. HTML

A *Hypertext Markup Language* (HTML) [Berners-Lee, 1995] é uma linguagem de marcação usada para criar documentos do tipo hipertexto independentes de plataforma. Marcadores HTML podem representar *news*, *mail*, documentação, hipermídia, menus de opções, resultados de pesquisas em bancos de dados ou ainda documentos simples estruturados com gráficos.

O HTML tem sido utilizado na *World-Wide Web* desde 1990, contendo importantes extensões que permitem a criação de ligações (*links*) entre documentos, digitação de dados e interação com os usuários. Documentos em HTML são escritos em modo

texto, contendo apenas caracteres que podem ser normalmente digitados. Conseqüentemente, não é necessário nenhum editor de texto especial para criar documentos em HTML.

Os documentos em HTML são divididos em elementos que usualmente são iniciados e finalizados com *tags*. Por exemplo:

```
<NOME> ... algum texto ... </NOME>,
```

significa que "`... algum texto ...`" é um conteúdo do elemento `NOME`. Alguns elementos não alteram o formato do texto de seu conteúdo e são chamados "elementos vazios". Esse tipo especial de elemento não requer *tags* de finalização. Outros elementos podem possuir atributos, que aparecem dentro do *tag* de inicialização, definindo alguma propriedade. Por exemplo:

```
<H1 ALIGN = "center">,
```

quer dizer que o elemento `H1` (cabeçalho no nível 1) deve estar centralizado.

Elementos e atributos são "`case-insensitives`". Dessa forma,

```
<NOME ATRIBUTO="string"> e <NomE ATrIbuTo="string">
```

são equivalentes. Porém, o valor do atributo (neste caso "`string`") pode ser "`case-sensitive`".

Existem ainda regras básicas que um documento em HTML deve respeitar. Por exemplo, um elemento do tipo cabeçalho não pode conter outros elementos desse mesmo tipo, ou seja,

```
<H1><H2> Este exemplo não </H2> funcionara </H1>,
```

não é uma forma permitida em HTML.

Referências completas sobre HTML podem ser encontradas em [Berners-Lee, 1995] ou ainda em [Graham,1996].

2.4. AGENTES, SPIDERS, WEB CRAWLERS, ROBÔS

Um robô é um *software* que navega automaticamente na estrutura de hipertexto da *World-Wide Web* recuperando documentos e, recursivamente, recuperando todos os outros documentos que são referenciados através de *hiperlinks*. Robôs podem ser utilizados com diferentes objetivos, tais como: indexar *sites*, validar *links*, monitorar atualizações e validar a existência de documentos.

Navegadores não são considerados robôs porque são operados por pessoas e não recuperam automaticamente nenhum documento referenciado (com exceção das imagens inseridas dentro do documento). Entretanto, versões mais atuais do *Internet Explorer* e do *Netscape Navigator*, passaram a oferecer recursos para o usuário que podem ser considerados como robôs, separadamente da navegação manual. Por exemplo, verificar se uma página foi atualizada automaticamente é uma tarefa típica de robôs e que hoje está sendo realizada pelos navegadores citados.

Robôs são também conhecidos como *Web Wanderers, Web Crawlers* ou *Spiders*. Apesar desses nomes darem a impressão de um *software* que se move entre *sites* como um vírus, esse não é o caso. Um robô simplesmente visita *sites* e recupera documentos a partir dele, como se fosse um usuário. A única diferença é que o robô faz isso automaticamente e sem a capacidade de preencher formulários. *Worms* e *WebAnts* também são considerados termos válidos para robôs, apesar do primeiro, tecnicamente, ser um *software* que se multiplica e o segundo ser uma definição para um conjunto de robôs que trabalham com o objetivo de executar uma tarefa em comum.

Tradicionalmente, robôs não são considerados agentes. Mas, devido ao desenvolvimento de mecanismos mais sofisticados para os robôs, esses dois conceitos estão cada vez mais próximos.

Especificamente, o termo “agentes” possui diferentes significados, a saber:

- **Agentes autônomos**: são programas que navegam entre *sites,* decidindo eles mesmos quando mover e o que fazer. Esses programas podem navegar apenas em servidores especiais e atualmente não são muito difundidos na Internet;

- **Agentes inteligentes**: são programas que ajudam usuários a realizar alguma tarefa, tais como: escolher um produto, preencher um formulário ou a localizar alguma coisa. A difusão desse tipo de *software* na Internet é cada vez maior;

- **`User-agent`**: como mencionado anteriormente, é o termo técnico para programas que realizam tarefas em uma rede para um usuário.

2.5. ÉTICA PARA ROBÔS E AGENTES

Juntamente com o crescimento da *World-Wide Web,* o número de robôs e agentes ativos tem aumentado razoavelmente. Para que o bom funcionamento da Internet possa ser mantido, foram desenvolvidos padrões – normas consolidadas em uma ética – que estabelecem como robôs e agentes devem operar na *Web*. Essa ética é válida tanto para robôs quanto para agentes.

2.5.1. Ética para Agentes de Serviços

São considerados agentes de serviços aqueles que trabalham na rede procurando construir bases de informações que estarão disponíveis para a comunidade. Dois agentes/robôs de serviço bastante conhecidos são o *scooter* do *Altavista* e o *webcrawler* do *WebCrawler*.

A ética para agentes de serviço é constituída dos seguintes pontos:

- **Identidade**: uma atividade de um agente de serviços deve ser legível e rastreável até o seu operador;

- **Abertura**: a informação gerada por um agente de serviços deve ser acessível a uma comunidade, cujo tamanho esteja intimamente ligado às diretivas de funcionamento deste agente;

- **Moderação**: uma execução e a freqüência entre as execuções devem ser apropriadas para a capacidade dos servidores origem/destino e para as conexões que o agente irá realizar;

- **Respeito**: um agente de serviços deve respeitar as restrições impostas em cada servidor que ele irá visitar;

- **Robusto**: um agente de serviços deve ser preciso e estar sempre atualizado.

2.5.2. Ética para Agentes de Usuários

São considerados agentes de usuários aqueles que trabalham na rede com um objetivo particular para um determinado usuário. Dois agentes/robôs de usuários bem conhecidos são o WebFerret[5] e o Jango[6].

Os principais aspectos éticos que um agente de usuário deve considerar são:

- **Identidade**: uma atividade de um agente de usuário deve ser legível e rastreável até o seu usuário;

- **Moderação**: uma execução e a freqüência entre execuções devem ser apropriadas para a capacidade dos servidores origem/destino e para as conexões que o agente irá realizar;

- **Apropriado**: um agente de usuário deve propor uma questão específica a servidores específicos, deixando que agentes de serviços sejam responsáveis por gerar bases de informações;

- **Vigilância**: um agente de usuário não deve permitir chamadas que gerem resultados inesperados;

2.5.3. Diretrizes Básicas para Criadores de Robôs

Em [Koster,1996] são descritos critérios e normas básicas que devem ser avaliados antes da construção de um robô. Essas normas e critérios objetivam orientar os criadores de robôs e podem ser resumidas nos seguintes tópicos:

- **Reconsidere.** Não existe nenhum outro robô que já obteve os resultados procurados? O robô realmente ajudará a atingir o objetivo almejado?

[5] http://www.ferretsoft.com/netferret/

[6] http://www.jango.com/

- **Identifique** o robô, o responsável e os objetivos que se pretende alcançar através de documentos disponíveis eletronicamente (preferencialmente via FTP ou via *Word-Wide Web*);

- **Teste** localmente;

- **Não sobrecarregue os recursos** da *World-Wide Web*, moderando a velocidade do robô (tanto durante uma execução simples, quanto entre execuções para atualização);

- **Não busque informações que não serão utilizadas.** Verifique o que está sendo requisitado;

- **Monitore** constantemente a performance e resultados obtidos pelo robô;

- **Compartilhe os resultados** obtidos (erros, conclusões, etc.).

2.5.4. Padrão para Exclusão de Robôs

Resultado de discussões entre criadores de agentes, o padrão de exclusão de robôs [Koster,1995] tem como finalidade permitir que um robô, ao buscar informações em um determinado *site*, conheça suas limitações tais como permissões, áreas para navegação e horários de coleta.

A idéia básica do padrão de exclusão é descrever em um arquivo texto estruturado quais são as partes do servidor que não devem ser acessadas por algum robô específico ou por todos robôs. Como exemplo:

```
# arquivo /robots.txt para http://www.algumlugar.com.br/
# mailto: webmaster@algumlugar.com.br para qualquer duvida
# ou sugestao

User-agent: algumrobo
Disallow:

User-agent: outrorobo
Disallow: /
```

```
User-agent: *
Disallow: /cgi-bin
Disallow: /logs
```

As primeiras linhas, começando com o caracter "#", especificam um comentário. O primeiro parágrafo especifica que o robô chamado `algumrobo` TEM permissão ao *site* inteiro, podendo navegar onde quiser. O segundo parágrafo indica que o robô `outrorobo` não possui acesso a nenhuma URL relativa que comece com "/". Como todas as URLs relativas começam com "/", então `outrorobo` não possui nenhum acesso a esse *site*. O terceiro parágrafo especifica que todos os outros robôs não devem visitar URLs relativas começando com `/cgi-bin` ou `/logs`.

Quando um mantenedor de informações na *World-Wide Web* não possui permissão para criar o arquivo `robots.txt` (não é o administrador do servidor), existe um padrão que utiliza o `tag` META do HTML para avisar quais são as permissões de um robô que funciona da seguinte forma:

- Quando um *tag* do tipo:
  `<META NAME="ROBOTS" CONTENT="NOINDEX">`
  é inserido em um documento HTML, esse documento não deverá ser indexado;

- Quando um *tag* do tipo:
  `<META NAME="ROBOTS" CONTENT="NOFOLLOW">`
  é inserido em um documento HTML, os *links* desse documento não deverão ser seguidos.

2.5.5. As Leis dos SoftBots

As leis dos s*oftbots*[7] foram desenvolvidas por Etzioni e Weld [Eichmann,1996] sendo derivadas das Leis de Asimov para robôs mecânicos. Podem ser resumidas em quatro pontos básicos:

- **Segurança**: um *softbot* não deve alterar destrutivamente o mundo;

[7] Segundo Etzioni: um *softbot* é um agente/robô que interage com um *software* transmitindo comandos e interpretando o retorno de informações do meio-ambiente.

- **Cuidado**: um *softbot* deve deixar o mundo como ele o encontrou;

- **Economia**: um *softbot* deve limitar-se aos recursos que lhe são oferecidos;

- **Vigilância**: um *softbot* não deve permitir ações que possam proporcionar resultados inesperados.

2.6. FERRAMENTAS DE BUSCA

Ferramentas de busca são mecanismos primários utilizados para localizar informações na Internet. As ferramentas de busca presentes na *World-Wide Web,* de acordo com Sullivan [Sullivan,1997] podem ser divididas em três categorias:

1. **Motores de busca (*search engines*)**: ferramentas que possuem robôs visitando, freqüentemente, *sites* na Internet para coletar informações para atualizar seus índices. Usualmente, os motores de busca não requerem trabalho do w*ebmaster* para que novos *sites* sejam incluídos em seu catálogo;

2. **Diretórios**: diferentemente dos motores de busca, os diretórios são criados e atualizados por pessoas de forma manual. Assim, caso um usuário deseje que sua página seja listada por um diretório, ele deverá preencher um formulário e submetê-lo ao w*ebmasters* do diretório em questão, que irá classificar a página em uma ou várias categorias;

3. **Motores de busca híbridos**: são motores de busca que possuem um diretório associado. Em geral, parte dos *sites,* da base de dados desse motor de busca, são revisados e categorizados como em um diretório. Entretanto, quando um usuário realiza uma pesquisa, ela será submetida a uma base de dados que foi elaborada com o auxílio de um robô.

Tanto os motores de busca quanto os diretórios apresentam vantagens e desvantagens. Por um lado, os diretórios precisam de um grande número de horas de trabalho daquelas pessoas que são responsáveis pela leitura e classificação dos documentos submetidos, sendo praticamente impossível cobrir todos documentos disponíveis na *Web*. Entretanto, diretórios freqüentemente apresentam melhores resultados que os motores de busca, devido à intervenção humana no momento do registro da informação. Por outro lado, os motores de busca atingem a maior parte da

Web e não necessitam de grande esforço humano, mas geram um tráfego intenso na rede e não conseguem classificar as páginas tão bem quanto os diretórios.

Ferramentas de busca podem ser localizas através de *sites* na *Web* que são especializados em catalogar diretórios e motores de busca, como no Yahoo Searching Web[8].

2.6.1. Meta Ferramenta de Busca

Meta Ferramenta de Busca [Selberg & Etzioni,1995] é um novo tipo de ferramenta de busca que não possui uma base de dados própria. Seu funcionamento baseia-se em:

- Replicar a pesquisa de um usuário para motores de busca e diretórios que estejam catalogados em seus algoritmos;

- Agrupar as respostas recebidas, muitas vezes realizando análises (como *clustering* em [Chang,1997]) e/ou validando os resultados recebidos.

A vantagem de uma abordagem desse tipo é a maior variedade do resultado obtido para uma pesquisa. Os diferentes algoritmos de classificação e atualização da base de dados das ferramentas de busca geram resultados praticamente sem respostas em comum, ou seja, a interseção dos conjuntos de resposta de cada motor de busca ou diretório é praticamente nula. Além disso, o usuário passa a procurar em diversas ferramentas ao mesmo tempo, gerando uma economia de tempo. A desvantagem está em uma arquitetura centralizada, quando a ferramenta está disponível em um servidor próprio.

Outras meta ferramentas de busca disponíveis na *Web* são: o SavvySearch[9], o ProFusion[10], o MetaFind[11], o Mamma[12], o Dogpile[13] e o AskJeeves[14].

---

[8] http://www.yahoo.com/Computers_and_Internet/Internet/World_Wide_Web/Searching_the_Web/

[9] http://savvy.cs.colostate.edu:2000/

[10] http://www.designlab.ukans.edu/profusion/

[11] http://www.metafind.com/

[12] http://www.mamma.com/

[13] http://www.dogpile.com/

[14] http://www.askjeeves.com/

2.7. PUXAR, EMPURRAR E PUXAR INFORMAÇÃO EM PARALELO

Em março de 1997, a revista *Wired* publicou uma longa matéria [Kelly,1997] sobre o que seria a Internet em um futuro próximo e como novas interfaces para a mídia eletrônica estão surgindo e ocupando parte do espaço dos atuais navegadores. A matéria fala sobre "empurrar informação", uma forma de distribuir informação sem esperar que o usuário venha até ela. A televisão e o rádio são bons exemplos de meios de comunicação que usam empurram informação (Figura 2): basta o usuário ligar o aparelho e canais (emissoras) passam a "empurrar" seus programas e músicas automaticamente, sem que o usuário faça nada.

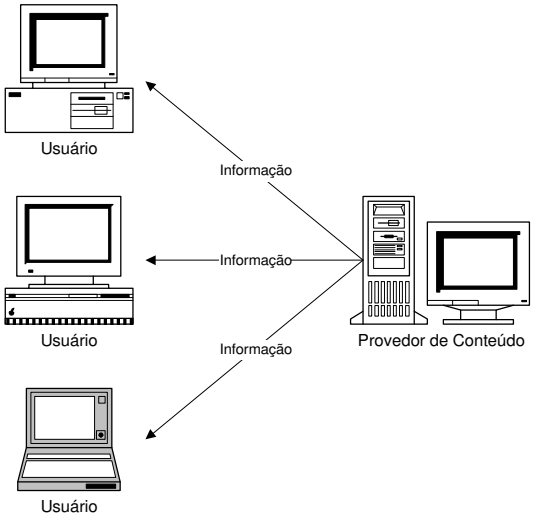


Figura 2: Empurrar informação

A *World-Wide Web* tem sido caracterizada por ser uma nova forma de "puxar informação" (Figura 3). Nela, o usuário abre uma página *Web* digitando um endereço em seu navegador e, a partir daí, ele pode clicar em *links* "puxando" as informações desejadas.

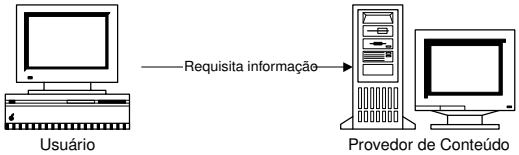


Figura 3: Puxar informação

Para empresas distribuidoras de informação na Internet, **empurrar informação** significa conseguir um número fixo de clientes com perfis conhecidos, garantindo a distribuição de propaganda, sem precisar esperar que o usuário acesse suas páginas na Internet. Esse é o principal motivo pelo qual **empurrar informação** tem capturado tanta atenção dos provedores de conteúdo.

Por trás de tudo isso há boas e más notícias: **empurrar informação** ajuda o usuário a conseguir informação mais relacionada com um determinado assunto. Esta característica pode quebrar parte do caos existente hoje na *Web,* conseqüência da facilidade existente para colocar um documento eletrônico disponível e então relacioná-lo através de *links* com outros documentos (a base para **puxar informação**). Por outro lado, **empurrar informação** deixa mais uma vez o usuário como ouvinte, ou seja, o usuário estará submetendo-se a receber informações selecionadas e formatadas por grandes canais de informação tal como acontece, hoje, com a televisão e o rádio.

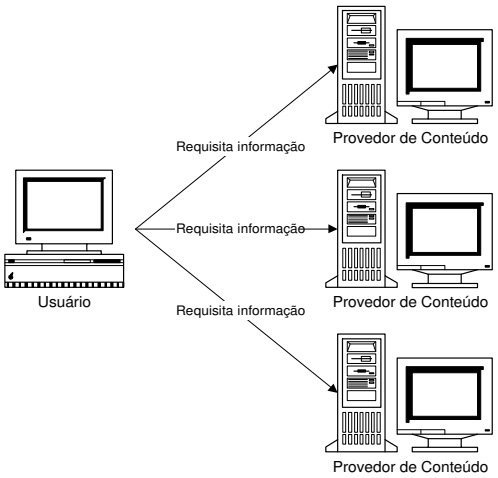


Figura 4: Puxar a informação em paralelo

Outra importante tecnologia de distribuição de informação que está começando na *Web* é **puxar a informação em paralelo** (Figura 4). Essa tecnologia procura combinar as vantagens de **empurrar** e **puxar a informação**, sem absorver as desvantagens de nenhuma delas. **Puxar a informação em paralelo** refere-se a transferência de informação diretamente entre o usuário e múltiplos provedores de conteúdo que são ativados pelo próprio usuário.

A busca em paralelo de informações possibilita que um usuário possa procurar um CD musical em diversas lojas ao mesmo tempo sem necessitar abrir uma conexão com cada uma delas. Mais ainda, as ferramentas de busca em paralelo executam tarefas que facilitam a comparação de preços ou localização de uma determinada informação, como uma notícia sobre algum novo lançamento de um músico ou de uma gravadora. Para isso tudo é preciso que o usuário realize apenas uma única consulta na *Web*.

3. Implementação da API Miner

A *Família Miner* é construída principalmente sobre a linguagem Java e suas APIs (*Application Programming Interfaces*). Os principais motivos dessa escolha estão na própria concepção dessa linguagem de programação.

A linguagem Java é voltada para aplicações que utilizem recursos disponíveis em rede. Sua maior simplicidade perante linguagens como C e C++ possibilitam um ganho de produtividade no desenvolvimento e manutenção de programas, o que é fundamental em aplicações que estejam sendo construídas em um ambiente tão dinâmico como a *World-Wide Web*. A neutralidade de plataforma, as características de segurança e novas extensões, como a API *Java Server* e *servlets* (Seção 3.2), trazem enormes possibilidades para o desenvolvimento de agentes. Além disso, Java é orientada a objetos e oferece capacidade para *multithreading* e aplicações distribuídas.

A história da origem e as principais características da linguagem Java são mostradas na Seção 3.1. Na Seção 3.3, a API Miner – base para a construção dos agentes da *Família Miner* – é introduzida.

3.1. JAVA

A linguagem Java[15] tem sua origem em um projeto de uma das subsidiárias *da Sun Microsystems* conhecida como FirstPerson, Inc. Em 1992, a FirstPerson trabalhava procurando desenvolver *software* para aparelhos de comunicação como telefones celulares e assistentes digitais pessoais (PDA). O objetivo era possibilitar a transferência de informação e aplicações de tempo real através de redes baseadas em pacotes que estariam utilizando raios infra-vermelho. Limitações de memória e de largura de banda impunham restrições que exigiam códigos pequenos e eficientes. A natureza das aplicações também exigia segurança e robustez. Apesar de utilizar no início do desenvolvimento do projeto a linguagem C++, essa logo foi abandonada

[15] http://www.javasoft.com/

devido à sua complexidade natural. A equipe da FirstPerson resolveu então trabalhar em uma nova linguagem que originalmente chamaram de "C++ minus minus".

Com o fracasso comercial dos assistentes digitais pessoais, como ocorreu com o Newton da Apple, a FirstPerson resolveu voltar seus olhos para a TV interativa e apresentou a linguagem chamada "Oak". No entanto, novamente o mercado não se mostrava interessado. A Sun resolveu então que o projeto deveria ser abandonado.

Dois dos principais líderes dos projetos da FirstPerson, Bill Joy e James Gosling, decidiram então traçar uma nova estratégia para a linguagem que eles haviam desenvolvido. Era 1993 e o interesse estava na explosão da Internet e em particular na *World-Wide Web*. Oak era pequena, robusta, independente de arquitetura e orientada a objetos. Tinha todos os requisitos para uma linguagem de rede segura e universal. A Sun Microsystems imediatamente mudou seu foco e, então, Oak tornou-se Java.

No relatório final dos autores da linguagem Java, onze conceitos são destacados como sendo as características principais que definem a linguagem: simples, orientada a objetos, distribuída, robusta, segura, independente de plataforma, portátil, interpretada, alta performance, "*multithreaded*" e dinâmica.

Na linguagem Java, os programas fontes quando compilados produzem um código binário independente da plataforma conhecido como "*byte code*" ou "*j-code*". Ao contrário de programas escritos em C e C++ que quando compilados definem instruções nativas a serem executadas em um modelo particular de processador, o *byte code* da linguagem Java é executado em um interpretador *run-time* que realiza as mesmas tarefas que um processador normal (Figura 5), só que em um ambiente muito mais seguro. Esse interpretador *run-time* é conhecido como a Máquina Virtual Java e é nativa para cada plataforma, sendo sua especificação aberta, possibilitando que qualquer organização possa desenvolvê-la.

Como Java é uma linguagem interpretada, ela não consegue ser tão rápida quanto programas desenvolvidos com linguagens como C. Mas Java é suficientemente rápida, especialmente para aplicações interativas ou baseadas em redes onde a aplicação freqüentemente está aguardando alguma informação do usuário ou esperando dados que deverão chegar através da rede. Além disso, para aplicações que exigem velocidade, implementações de compiladores *Just-in-time,* capazes de

transformar o *byte-code* Java em código nativo da máquina em tempo real, estão sendo agregados às Máquinas Virtuais Java.

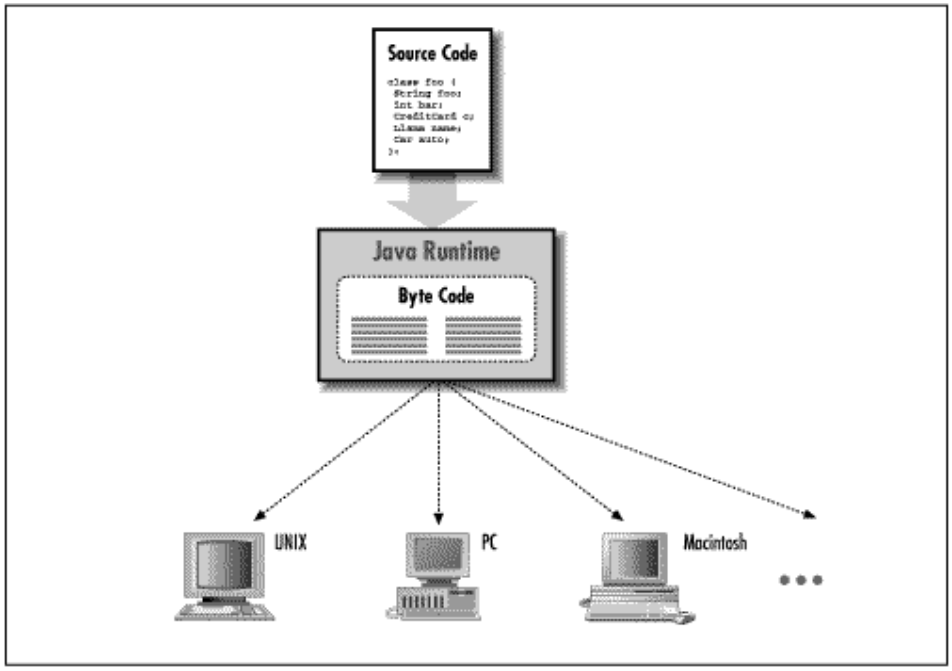


Figura 5: Máquina Virtual Java

3.2. JAVA SERVER API E SERVLETS

Servlets são programas escritos em Java capazes de estender a funcionalidade de um servidor. Como *applets*, os *byte codes* das *servlets* podem ser lidos localmente ou a partir de uma rede e serem executados no cliente.

Servlets possibilitam a geração de páginas dinâmicas como *CGI (Common Gateway Interface*), apresentando algumas características particulares interessantes, tais como:

1. Uma *servlet* pode continuar rodando em *background* após o término do processamento de uma requisição de um cliente, estando pronta para processar uma nova requisição sem onerar custos de inicialização. Isso é muito melhor do que a visão tradicional de CGIs que inicializam um novo processo a cada requisição. Mais ainda, *servlets* podem usar *threads* para processar eficientemente requisições simultâneas, além de possibilitar a transferência de dados entre múltiplas conexões, podendo agir, por exemplo, como um servidor de jogos;

2. Uma *servlet* pode comunicar interativamente com um *applet* em um cliente. Um programa CGI recebe uma requisição de um cliente e envia uma resposta, sendo a conexão fechada nesse momento. Dessa forma, uma resposta do cliente aos dados enviados pelo programa CGI irá inicializar um novo programa CGI que deverá mapear o atual estado da transação. Em contraste, *applets-servlets* podem manter um canal aberto de comunicação, realizando inúmeras transferências de informação entre o cliente e o servidor;

3. *Servlets* podem ser lidas por clientes. Como *applets*, *servlets* rodam em um ambiente seguro de forma que não é necessário preocupação com *servlets* hostis. Por exemplo, um cliente pode carregar uma *servlet* que indexa as páginas de um *site* da *Web*. Como o acesso será local, a coleta será muito mais rápida e o processamento de indexação distribuído.

Servlets são escritas utilizando a API *Java Server* que será incorporada à versão 1.2 do JDK (*Java Development Kit*).

Muitos servidores *Web* já apresentam funcionalidade para executar *servlets*, entre eles: o *Microsoft Internet Information Server*, o *Apache*, o *Netscape Enterprise* e o *Java Web Server* da *Sun Microsystems*, o primeiro a oferecer tal recurso.

Em sua maioria, os agentes da *Família Miner* são *servlets*. O servidor *Web* escolhido foi o *Java Web Server*, tendo sido utilizado desde sua primeira versão *alpha*, quando ainda era conhecido como *Jeeves*. Características das *servlets* como as apresentadas nos itens acima – especialmente no item 3 – são os principais razões da construção da *Família Miner* como *servlets* e conseqüentemente em Java.

3.3. A API MINER

A API Miner contém uma série de *packages*[16] que faz todo o acesso necessário a um determinado recurso da Internet. A API Miner é a base de todos os membros da *Família Miner*. A API Miner controla todos os recursos a serem acessados, todas as

[16] *Package* é a forma como classes específicas são agrupadas em um conjunto dentro da linguagem Java, como por exemplo, o *package java.awt* que faz parte da API Java e oferece recursos para criação e desenvolvimento de interfaces gráficas para usuários.

conexões além de processar informação (como a interpretação de documentos HTML).

As classes são elaboradas agrupando recursos similares existentes na Internet e definindo uma estrutura básica da informação de interesse existente nesses recursos. Por exemplo, livrarias são classes, dentro do `package miner.book`, que produzem objetos do tipo `Livro` que possuem atributos como: título da obra, autor, preço e editora.

As classes que representam os recursos de interesse da Internet são extensões da classe `java.lang.thread`, o que permite que uma consulta possa ser realizada "simultaneamente" em diversos recursos. Em geral, a criação de um objeto da API Miner, como uma livraria ou uma ferramenta de busca, gera o processamento ilustrado na Figura 6.

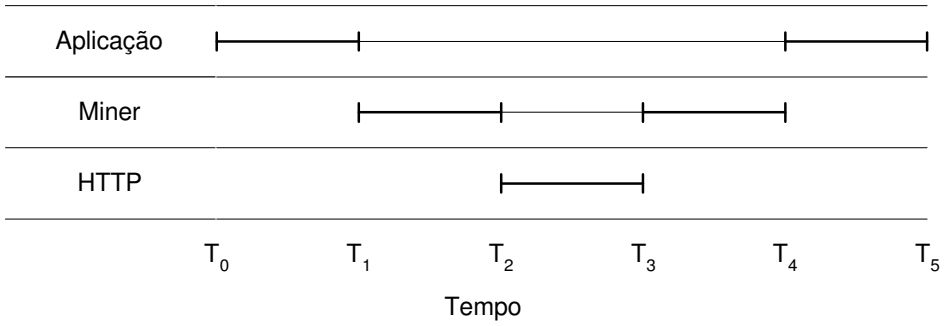


Figura 6: Funcionamento de um objeto da API Miner

Essa figura mostra uma aplicação iniciada em T_0 e que realiza tarefas até T_1, quando ela então cria um novo objeto *Miner* que representa um recurso na Internet a ser consultado (como uma livraria, por exemplo). A *thread* do objeto *Miner* realiza algumas inicializações e abre uma conexão HTTP com o recurso ao qual está associado em T_2, sendo bloqueada logo após a abertura dessa conexão. A conexão HTTP é feita em outra *thread* que fica sobre controle do objeto *Miner*. Em T_3, a *thread* da conexão HTTP é finalizada. A *thread* do objeto *Miner* entra novamente em execução e realiza o processamento da resposta recebida pela conexão HTTP elaborando uma lista de objetos (livros, software, CDs, *links*) que é o principal resultado da aplicação. Ao final desse processamento em T_4, a aplicação realiza as tarefas finais buscando no objeto *Miner* as informações procuradas e apresentando os resultados finais para o usuário, sendo encerrada em T_5.

É claro que durante a execução de um dos agentes da *Família Miner* muitos objetos são criados simultaneamente, enquanto a aplicação permanece executando tarefas entre T_1 e T_4.

Cada classe responsável por uma conexão HTTP oferece informações que identificam a API Miner, seguindo a ética e diretivas para construções de robôs.

3.3.1. O *package miner.book*

O *package miner.book* (Figura 7) contém as classes necessárias para realizar uma consulta em uma livraria presente na *World-Wide Web*.

A classe `Livro` possui atributos que identificam um livro (título, autor, código ISBN, editora, URL e preço do livro). A classe `ListaLivro` possibilita a criação de uma lista de livros onde cada novo livro que for adicionado, será inserido em ordem alfabética pelo Título dentro da lista já existente e agrupado, quando possível, a outro membro através do código ISBN. A classe `PrecoLivraria` estabelece uma forma para armazenar, dentro de uma `ListaLivro`, os preços e as URLs de um livro nas diversas livrarias. A classe `HTTPConnectBook` é responsável por realizar uma conexão com as livrarias enviando as consultas e recebendo as respostas. Os modificadores abaixo são adicionados na requisição HTTP por essa classe para garantir a identificação da aplicação de acordo com a ética e diretivas para o desenvolvimento de robôs:

```
Referer: http://www.miner.dcc.ufmg.br/
From: victor@dcc.ufmg.br
User-Agent: Mozilla 2 (compatible,bookminer,victor@dcc.ufmg.br)
```

O processamento das respostas das conexões HTTP é feito dentro das classes específicas das livrarias que devem ser derivadas da classe `Bookstore`.

Algumas das livrarias consultadas atualmente são as brasileiras Booknet, Livraria Cultura, Editora Campus e a Livraria Siciliano, as americanas Bookpool, BookStacks, Amazon.com, McGraw-Hill Bookstore, O'Reilly, Prentice-Hall, a inglesa iBS e a francesa Side.

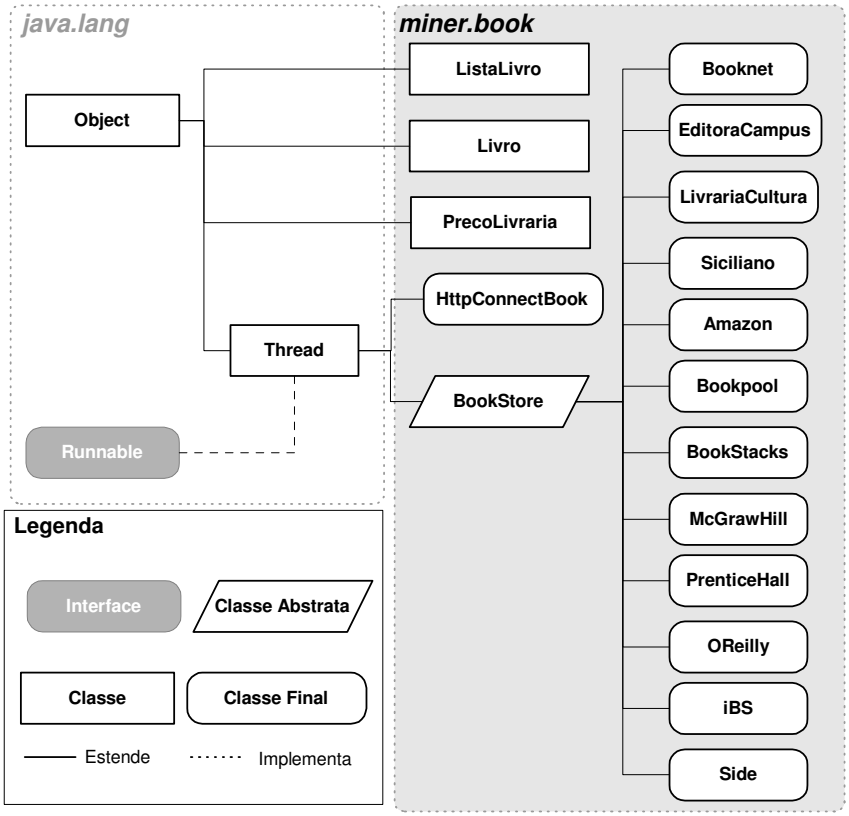


Figura 7: o package miner.book

3.3.2. O *package miner.cd*

O *package miner.cd* (Figura 8) contém as classes necessárias para realizar uma consulta em uma loja de CDs presente na *World-Wide Web*.

A classe `CD` possui atributos que identificam um CD (título do álbum, artista ou grupo musical, gravadora, URL, preço do CD e preço da fita K7 (*tape*) – quando disponível). A classe `ListaCD` possibilita a criação de uma lista de CDs onde cada novo CD a ser adicionado, é inserido em ordem alfabética pelo título do álbum dentro da lista já existente. A classe `PrecoCD` estabelece uma forma para armazenar, dentro de uma `ListaCD`, os preços e as URLs de um CD nas diversas lojas. A classe `HTTPConnectCD` é responsável por realizar uma conexão com as lojas de CD, enviando as consultas e recebendo as respostas. Os seguintes modificadores são adicionados na requisição HTTP por essa classe para garantir a identificação da aplicação de acordo com a ética e diretivas para o desenvolvimento de robôs:

```
Referer: http://www.miner.dcc.ufmg.br/
From: victor@dcc.ufmg.br
```

```
User-Agent: Mozilla 2 (compatible,cdminer,victor@dcc.ufmg.br)
```

O processamento das respostas das conexões HTTP é feito dentro das classes específicas das lojas que devem ser derivadas da classe `CDstore`.

Algumas lojas nacionais e internacionais já inseridas dentro do `package miner.cd` são: Planet Music, Boulevard Music, CDNow.

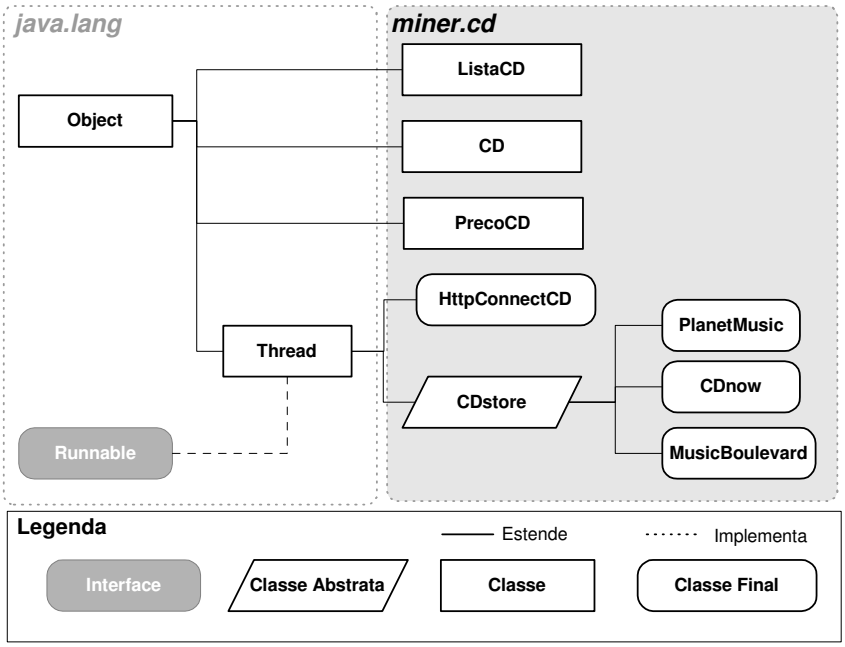


Figura 8: o package miner.cd

3.3.3. O *package miner.engine*

O *package miner.engine* (Figura 9) contém as classes necessárias para realizar uma consulta através de uma ferramenta de busca presente na Internet, seja ela um motor de busca ou um diretório.

A classe `LinkResultado` tem atributos que identificam um *link* obtido como resultado na consulta do usuário (título da página, URL, resumo da página e tamanho da página). A classe `ListaLink` possibilita a criação de uma lista cujos elementos são formados por objetos do tipo `LinkResultado` agrupados pela URL. Melhorias podem ser feitas nessas classes, desenvolvendo, por exemplo, uma lista que seja capaz de organizar *clusterings* com os objetos recebidos. A classe `HTTPConnectThread` é responsável por realizar uma conexão com as ferramentas de busca, enviando as

consultas e recebendo as respostas. Os modificadores listados abaixo são adicionados na requisição HTTP por essa classe para garantir a identificação da aplicação de acordo com a ética e diretivas para o desenvolvimento de robôs:

```
Referer: http://www.miner.dcc.ufmg.br/
From: victor@dcc.ufmg.br
User-Agent: Mozilla 2 (compatible,metaminer,victor@dcc.ufmg.br)
```

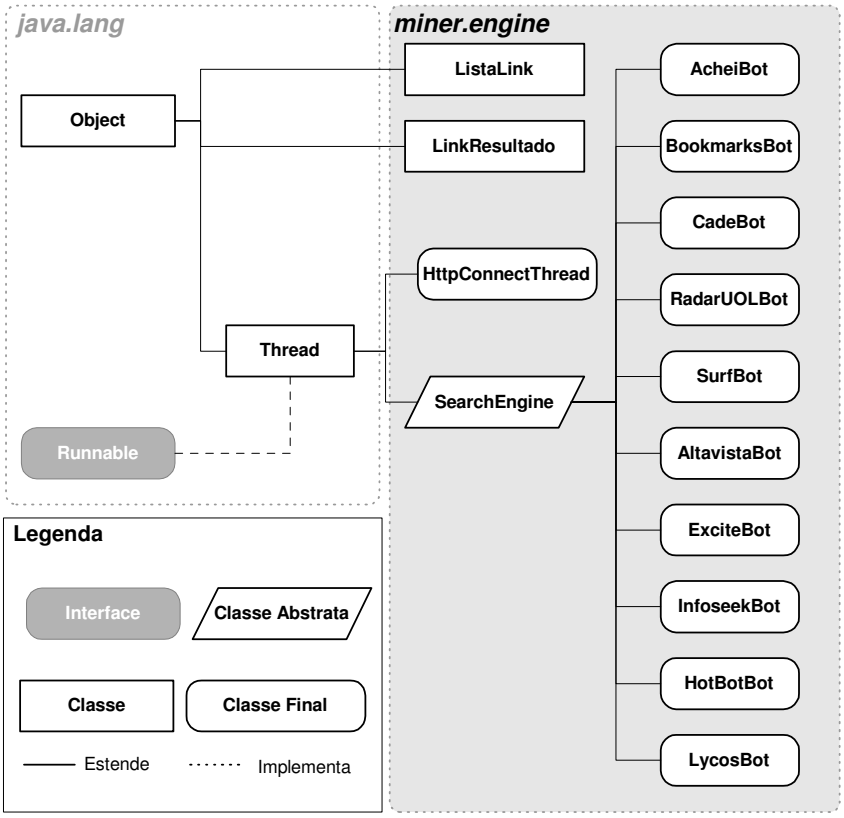


Figura 9: o package miner.engine

O processamento das respostas das conexões HTTP é feito dentro das classes específicas das ferramentas de busca que devem ser derivadas da classe `SearchEngine`.

A seguir são relacionadas algumas características das ferramentas de busca brasileiras e internacionais e uma listagem daquelas que já foram inseridas dentro do `package miner.engine`.

Ferramentas de Busca Brasileiras

As ferramentas de busca brasileiras caracterizam-se principalmente pelo uso da tecnologia desenvolvida pelos maiores motores de busca da *Web*. Poucas, como a *Surf*, possuem uma tecnologia própria.

Em geral, os diretórios e motores de busca brasileiros estão associados a algum serviço de provedor de Internet, mas é possível encontrar ferramentas de busca independentes, como o Achei!!. Isso caracteriza o mercado de motores de busca no Brasil como um serviço complementar oferecido pelos provedores de Internet, ao contrário dos principais motores de busca mundiais que funcionam como negócios independentes.

As ferramentas de busca brasileiras já implementadas dentro do `package miner.engine` são:

- **Achei!!** (http://www.achei.com.br/) – no ar desde junho de 1996, o Achei!! é um motor de busca que utiliza o *software* do Altavista. O Achei!! expandiu a pouco tempo seu *site* na Internet, passando a fornecer, além da maneira tradicional de buscas, uma categorização de *sites* semelhante à existente nos diretórios. O Achei!! conta também com serviços como o *Acheivocê*!, especialista em procura de pessoas;

- **Bookmark** (http://www.bookmarks.com.br/) – no ar desde 1996, o Bookmarks é um motor de busca mantido pela Alternex, um dos primeiros provedores de Internet no Brasil. Atualmente, o Alternex está conectado às três principais redes brasileiras: RNP, Embratel e Global One e está expandindo seus negócios para ser um provedor nacional (incluindo Belo Horizonte, Curitiba, São Paulo e Rio de janeiro, entre outras cidades). O Bookmarks utiliza a tecnologia da *OpenText Corporation*, tendo mais de 600.000 páginas catalogadas;

- **Cadê?** (http://www.cade.com.br/) – um dos primeiros diretórios da *Web* brasileira (no ar desde 1995), o Cadê? teve seu início como a maioria dos diretórios existentes: um *site* com os *links* considerados mais importantes pelo seu fundador. Hoje, o Cadê? é tido como uma das principais ferramentas de busca brasileiras, tendo sido premiado diversas vezes. O Cadê? está, atualmente, expandindo seus negócios para atingir o mercado de Intranets através da venda de versões de seu

software que tem como função ajudar a organizar informações em redes empresariais;

- **Surf** (http://www.surf.com.br/) – descendente do antigo Yaih? da RNP, o Surf é um motor de busca que trabalha juntamente com o serviço de um diretório, originalmente conhecido como Webra. O Surf faz parte da Dialdata, também uma das primeiras provedoras de Internet no Brasil;

- **RadarUOL** (http://www.radaruol.com.br/) – marcando a entrada do Universo Online (UOL) no mercado de ferramentas de busca, o RadarUOL é a versão brasileira do HotBot que utiliza a tecnologia da Inktomi. O RadarUOL acrescentou à sua base de dados os textos da Folha de S.Paulo (o jornal brasileiro de maior circulação) e das revistas Exame e Veja (duas das principais revistas brasileiras), tendo isso como diferencial competitivo em relação as demais ferramentas de busca;

Ferramentas de Busca Estrangeiras

Hoje é possível encontrar diversas ferramentas de busca espalhadas pelo mundo inteiro. O EuroSeek considerado a maior ferramenta de busca européia, o Sapo em Portugal, a Matilda na Austrália ou o diretório inglês conhecido como GOD. Entretanto, as principais ferramentas de busca da *World-Wide Web* são de empresas americanas que, em geral, iniciaram seus negócios no segmento de busca na Internet baseando a tecnologia de seus produtos em dissertações de mestrado e/ou teses de doutorado.

As ferramentas de busca internacionais já implementadas dentro do `package miner.engine` são:

- **Altavista** (http://www.altavista.com) – iniciou suas operações em dezembro de 1995. O Altavista foi desenvolvido pela *Digital Equipment Corporation*, tendo obtido grande reconhecimento e sucesso na *World-Wide Web* devido a sua enorme base de dados e velocidade para responder as pesquisas dos usuários. A Digital assinou um contrato de parceria com o Yahoo! em junho de 1996, onde o Altavista passou a ser considerado como o motor de busca preferencial do Yahoo!;

- **Excite** (http://www.excite.com) – lançado no final de 1995, o Excite cresceu rapidamente e adotou uma estratégia de aquisição de seus concorrentes. Em julho

de 1996, comprou o Magellan e, em novembro do mesmo ano, adquiriu o WebCrawler. Pesquisas no Excite podem ser feitas de três maneiras distintas:

1. *Excite Search* (maneira tradicional que utiliza a base de dados gerada por robôs);
2. Canais (*sites* são listados por tópicos, sendo que a categorização de assunto é feita manualmente pelo editores do Excite); e
3. *Excite NewsTracker* (pesquisa apenas em uma base de dados especial gerada pela captura de *sites* especializados em notícias);

- **Infoseek** (http://www.infoseek.com) – no ar desde o início de 1995, o Infoseek é, segundo [Sullivan, 1997], um *site* muito bem conhecido e revisado. Enquanto o antigo "Infoseek Guide" tinha cerca de 1 a 2 milhões de URLs catalogadas, um novo serviço, conhecido agora apenas como Infoseek, foi apresentado ao público, em 1996, com uma base de dados com 50 milhões de URL catalogadas. O Infoseek possui um diretório que funciona separadamente de seu motor de busca. Nesse diretório, as páginas são listadas por tópicos que são gerados automaticamente através de um *software* de categorização. Algumas das páginas são, também, listadas com uma marca vermelha, representando uma URL que foi selecionada manualmente, sendo recomendada pelos editores do Infoseek. Em maio de 1997, o Infoseek lançou o Ultramatch: uma ferramenta de direcionamento de anúncios. Uma tecnologia semelhante é utilizada pelo Lifestyle Finder[17], da Andersen Consulting, que detecta e organiza campos de interesse para o usuário através de rastreamento (usando *cookies*) e da análise das palavras-chave das buscas realizadas por esse usuário. O Ultramatch permite que propagandas sejam selecionadas e enviadas de acordo com o perfil que é desenhado para cada usuário;

- **HotBot** (http://www.hotbot.com/) – inaugurado em maio de 1996, o HotBot marca a entrada da Wired no mercado de motores de busca. O HotBot utiliza o motor de busca da *Inktomi*. Entretanto, isso não significa que o catálogo localizado em http://www.inktomi.com/ é o mesmo do HotBot, apenas que os dois usam a mesma tecnologia. A mesma tecnologia é utilizada também por vários outros motores de busca, como o RadarUOL do Universo OnLine;

[17] http://lifestyle.cstar.ac.com/lifestyle/

- **Lycos** (http://www.lycos.com/) – no ar desde maio de 1994, o Lycos é um dos serviços de busca mais antigos da Internet tendo nascido de um projeto na *Carnegie Mellon University*. O Lycos lista páginas de duas diferentes formas: através das listas de seu motor de busca (coletadas por um robô) e, também, através de um diretório associado conhecido como *"Web Guides*", onde existem "guias" (índices) orientando para uma área com um determinado tipo de assunto, como por exemplo notícias ou esportes. Adicionado aos serviços do Lycos em fevereiro de 1996, o *Web Guides* era conhecido como A2Z. O Lycos possui um serviço conhecido como *Top* 5% que apresenta o que é considerado como o melhor da *World-Wide Web* em termos de conteúdo e *design*. O *Top* 5% era mantido inicialmente pela *Point*, uma das empresas independentes mais antigas na *Web,* que prestava serviço de *rating*, e que foi adquirida pelo Lycos.

Outras ferramentas de busca internacionais são: Yahoo!, WebCrawler, OpenText, Magellan, Northern Light, LookSmart, AOLNetFind, World-Wide Web Worm, Galaxy e DejaNews especializado em *newsgroups* (USENET) apenas.

3.3.4. O *package miner.news*

O *package miner.news* (Figura 10) contém as classes necessárias para a coleta das notícias do dia em um jornal presente na Internet.

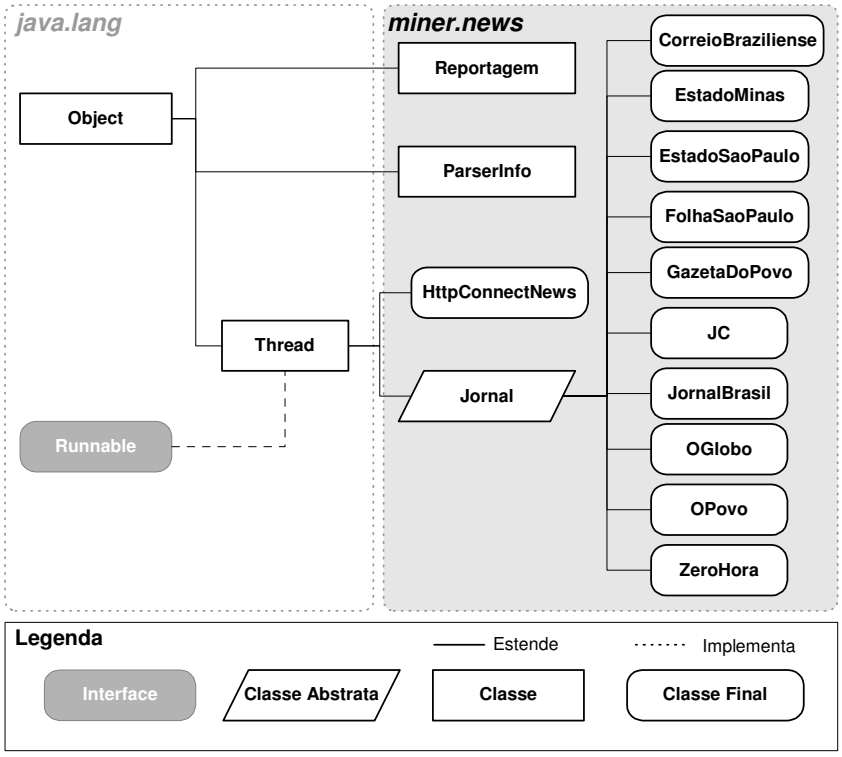


Figura 10: o package miner.news

A classe `Reportagem` possui atributos que identificam uma reportagem obtida durante a coleta das notícias (título da reportagem, repórter, texto, URL, caderno – seção do jornal). A classe `ParserInfo` oferece recursos para realizar o *parsing* dos documentos HTML recebidos durante uma coleta, removendo tags e sintaxes próprias dessa linguagem de marcação. A classe `HTTPConnectNews` é responsável por realizar uma conexão com os jornais, buscando os cadernos e reportagens que são solicitados pelas classes específicas de cada Jornal. Os seguintes modificadores são adicionados na requisição HTTP por essa classe para garantir a identificação da aplicação de acordo com a ética e diretivas para o desenvolvimento de robôs:

```
Referer: http://www.miner.dcc.ufmg.br/
From: victor@dcc.ufmg.br
User-Agent: Mozilla 2 (compatible,newsminer,victor@dcc.ufmg.br)
```

O processamento das respostas das conexões HTTP é feito dentro das classes específicas das ferramentas de busca que devem ser derivadas da classe `Jornal`.

Ao contrário dos outros *packages* da API Miner, as classes do `package miner.news` não devem ser utilizadas sempre que um usuário realizar uma pesquisa. O funcionamento dessas classes é diferente das classes dos outros *packages* da API Miner. As classes do `miner.news` são construídas para coletar todas as reportagens disponíveis em um jornal *on-line* em um determinado instante. Como essas notícias não irão variar até o dia seguinte, não há necessidade de coletá-las mais do que uma vez por dia. Além disso, esse *package* foi construído para oferecer recursos para a criação de *clippings* eletrônicos. Assim, todas as reportagens, de todos os jornais, precisam ser coletadas para que os processamentos que irão construir os *clippings* personalizados de cada usuário sejam realizados.

O fato das classes do `package miner.news` estarem trabalhando de forma semelhante a um robô de indexação faz com que o padrão de exclusão de robôs para essas classes seja também implementado. As reportagens coletadas são armazenadas localmente através de métodos da classe `Jornal`.

Alguns dos jornais já inseridos dentro do `package miner.news` são: o Correio Braziliense (DF), o Estado de Minas (MG), o Estado de S.Paulo (SP), o Gazeta do Povo (PR), o Globo (RJ), o Jornal do Brasil (RJ), o Jornal do Commercio (PE), o Povo (CE) e mais o jornal Zero Hora (RS).

A Tabela 2 mostra uma comparação entre esses jornais com base em dados do agente Alexa[18].

Tabela 2: Jornais comparados segundo o Alexa

Jornal	*Classificação*	*Visitas*	*Links*	*Velocidade*	*Taxa de*	*Páginas*	*URL*

[18] Alexa é um serviço que procura auxiliar as pessoas a navegarem na Internet através da disponibilização de informações sobre a *Web*. Algumas das informações geradas pelo Alexa são:

- **Classificação** – Quantidade de tráfego no *site* baseado em informações de vários *caches* da *Web*;
- **Visitas (Alexa)** – Número de visitas realizadas por usuários do Alexa;
- ***Links*** – Número de páginas diferentes que apontam para esse site;
- **Veloc** – tempo médio para recuperar uma página (Rápida:50kB/s, Média:28kB/s, Baixa:Menos que:14kB/s);
- **Taxa de Atualização** – Freqüência de atualização das páginas (Alta: atualizada em menos do que 6 meses, Média: aproximadamente 1 ano, Baixa: mais do que 2 anos);
- **Páginas** – Número de páginas do site, baseado em arquivos do Alexa.

		(Alexa)		*Atualização*			
Correio Braziliense	Baixo tráfego	-	18	-	-	-	http://www.correiobraziliense.com.br/
Estado de Minas	Baixo tráfego	2.123	1	Média	Média	798	http://www.em.com.br/
Estado de São Paulo	Top 10.000	5.027	298	Baixa	Média	5.059	http://www.estado.com.br/
Gazeta do Povo	Baixo tráfego	319	132	Média	Alta	14,879	http://www.dopovo.com/
Jornal do Brasil	Top 10.000	4.053	701	Baixa	Média	3.996	http://www.jb.com.br/
O Globo	Top 10.000	7.910	481	Baixa	Alta	106	http://www.oglobo.com.br/
O Povo	Baixo tráfego	325	36	Média	Alta	2.004	http://www.opovo.com.br/
Zero Hora	Top 10.000	6.141	39	Baixa	Alta	574	http://www.zh.com.br/

3.3.5. O *package miner.soft*

O *package miner.soft* (Figura 11) contém as classes necessárias para a procura de *software* em repositórios disponíveis na *Web*.

A classe `Software` possui atributos que identificam um *software* (nome do *software*, plataforma, tamanho, URL e descrição). A classe `HTTPConnectSoft` é responsável por realizar uma conexão com os repositórios de *software*, enviando as consultas e recebendo as respostas. Os seguintes modificadores são adicionados na requisição HTTP por essa classe para garantir a identificação da aplicação de acordo com a ética e diretivas para o desenvolvimento de robôs:

```
Referer: http://www.miner.dcc.ufmg.br/
From: victor@dcc.ufmg.br
User-Agent: Mozilla 2 (compatible,softminer,victor@dcc.ufmg.br)
```

O processamento das respostas das conexões HTTP é feito dentro das classes específicas de cada repositório que devem ser derivadas da classe `SoftLibrary`.

Alguns dos repositórios de *software* já inseridos dentro do `package miner.soft` são: o FileZ, o TucowsUOL, o HotFiles, o SoftSeek e o Download.com.

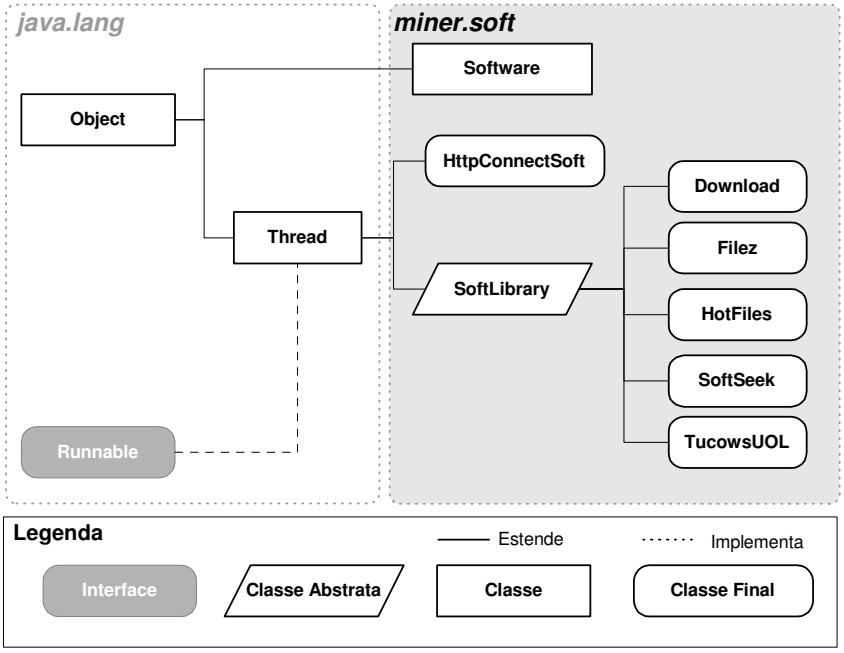


Figura 11: o package miner.soft

3.3.6. O *package miner.util*

O *package miner.util* (Figura 12) possui uma série de classes que são utilizadas pelas outras classes da API Miner.

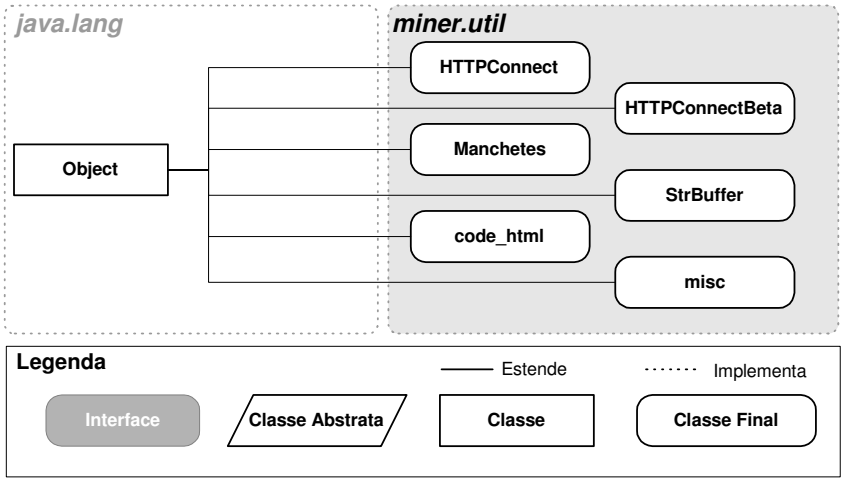


Figura 12: o package miner.util

A classe `HTTPConnect` é a responsável pelas conexões HTTP realizadas por todas as classes da API Miner. A classe `HTTPConnectBeta` é utilizada durante o processo de inserção de um novo recurso na API Miner. Ao contrário da classe `HTTPConnect`,

a `HTTPConnectBeta` oferece alguns recursos a mais que possibilitam um melhor rastreamento da conexão, sendo bem menos otimizada do que a `HTTPConnect`.

A classe `Manchetes` elabora o código HTML necessário para que o `applet Light` (que apresenta notícias aleatórias do dia durante a pesquisa de um usuário) possa ser inserido dentro de uma página HTML que será enviada como resposta a uma consulta realizada a um dos membros da *Família Miner*.

A classe `StrBuffer` é uma versão da classe `java.lang.StringBuffer` mas com algumas otimizações especiais que permitem um ganho de performance no processamento de *Strings*.

A classe `code_html` possui métodos específicos para inserir formulários, cabeçalhos e rodapés nas páginas que são enviadas como respostas para os usuários da *Família Miner*.

A classe `misc` possui uma variedade de métodos estáticos que são utilizados para o processamento de *strings* e armazenagem de informações consideradas relevantes (como erros e exceções) durante uma execução de um membro da *Família Miner*.

4. A Família Miner

A *Família Miner de Agentes para a World-Wide Web* é um conjunto de ferramentas desenvolvida sobre a API Miner, que objetiva auxiliar usuários a localizarem informações na Internet. A *Família Miner* é a primeira meta ferramenta de busca brasileira e pode ser é dividida em quatro grupos maiores de agentes: agentes de busca – utilizados para a procura de informações genéricas na *Web* (semelhantes aos motores de busca como *Altavista* e meta ferramentas de busca como o *MetaCrawler*); agentes de compra – utilizados para localizar e comparar preços de produtos vendidos *online*; agentes de notícias – utilizados para coletar e elaborar *clippings* de notícias veiculadas *online* na *Web*; e agentes de monitoramento – utilizados para gerenciar e monitorar toda a *Família Miner*.

4.1. AGENTES DE BUSCA

Os agentes da busca da *Família Miner* oferecem serviços semelhantes aos oferecidos pelas atuais máquinas de busca da Internet, tais como *Altavista, MetaCrawler, Yahoo, Excite*, entre outros. Contudo, algumas inovações são introduzidas como por exemplo, a eliminação de robôs transitando nos *backbones* através do uso de servidores *proxy*, conforme será visto mais adiante.

4.1.1. WebMiner

O WebMiner é um motor de busca tradicional semelhante ao Altavista e ao WebCrawler. Tendo sido inteiramente desenvolvido em Java, o WebMiner foi concebido para coletar páginas na Web (ou em uma Intranet) e elaborar uma base de dados com esses documentos.

O WebMiner é dividido em diversos módulos com fins específicos. Em cada execução do WebMiner, os módulos são executados em cadeia ativados um pelo outro. O início da execução acontece com uma lista de URLs que servem como "semente" para o WebController. A seguir são descritos cada um dos módulos do WebMiner.

O **WebControler** é a parte responsável por todo o controle ético e verificação de padrão de exclusão. Cada URL fornecida como parâmetro de entrada é verificada de diferentes formas. Perguntas como: esse servidor já foi visitado? Se sim, quando foi realizada a última visita a esse servidor? Se não, o arquivo robots.txt já foi coletado? O arquivo robots.txt permite a coleta dessa URL? Essa URL já foi coletada? Se sim, ela precisa ser atualizada? Já foi passou o período de tempo suficiente entre uma coleta e outra para não sobrecarregar esse servidor? Com base nas respostas dessas perguntas, ordens de coleta de páginas são enviadas para os módulos conhecidos como WebMinerBots.

Os **WebMinerBots** são responsáveis pela coleta de uma página na Web. Uma vez que tenham recebido a ordem de coleta, eles dão procedimento a essa coleta imediatamente. Todo o controle de ética, como já foi mencionado, cabe apenas ao WebController. As páginas coletadas pelos WebMinerBots são enviadas diretamente para o WebTesaurer.

O **WebTesaurer** é responsável pelo processamento e armazenagem de todas as páginas coletadas pelos WebMinerBots. De cada página recebida, os textos são extraídos para indexação enquanto os links existentes na página são identificados e adicionados a uma lista de URLs que será enviada para o WebController fechando, dessa forma, o ciclo de execução.

Graficamente, o funcionamento básico do WebMiner está representado na Figura 13.

O WebMiner é particularmente útil para construir pequenas bases de dados para estudos de recuperação de informação.

A interface de consulta do WebMiner com o usuário é feita através de um formulário HTML e de uma *servlet* que acessa a base de dados gerada durante a execução do WebMiner.

Apesar de não oferecer nenhuma nova característica à atual concepção de motores de busca, o WebMiner foi particularmente útil para a *Família Miner* por ser o primeiro membro da Família a possuir toda a ética, diretivas e padrão de exclusão implementado.

A evolução planejada para o WebMiner é baseada no conceito de *servlets*. A possibilidade de criação de uma ferramenta de busca que poderá ser carregada remotamente por um servidor *Web* e no próprio cliente realizar as tarefas de coleta e indexação, quebram uma série de paradigmas existentes nas atuais concepções de motores de busca.

O Projeto Harvest [Manber et al, 1995] pregava um modelo semelhante, mas que exigia que cada pequena rede possuísse um servidor de indexação funcionando e que também esse servidor de indexação estivesse posicionado dentro de uma estrutura hierárquica de servidores. A vantagem das *servlets* seria que os WebMasters não precisariam se preocupar em manter um software em funcionamento e muito menos em cuidar do posicionamento de seus servidores dentro de uma hierarquia complexa. Entretanto, ainda assim seria necessário que alguém responsável pelo servidor a ser indexado iniciasse a execução da *servlet*.

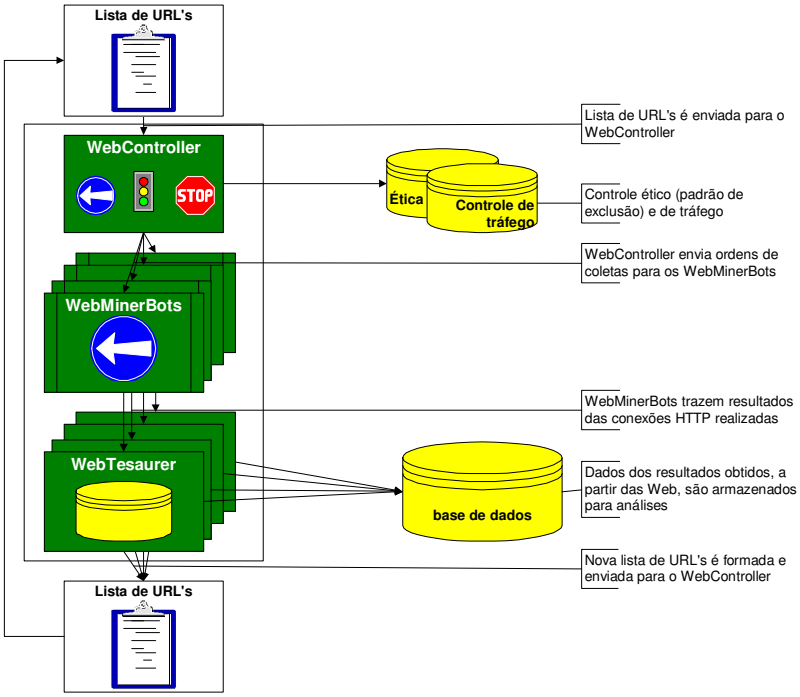


Figura 13: Funcionamento do WebMiner

4.1.2. MetaMiner

O MetaMiner é uma meta ferramenta de busca semelhante ao MetaCrawler. Contudo, sua interface com o usuário permite que sejam selecionados quais mecanismos de busca deseja-se utilizar ao realizar uma consulta. O MetaMiner, além disso, é a

primeira meta ferramenta de busca que utiliza ferramentas de busca brasileiras tais como o Surf, Bookmarks, Achei!!, RadarUOL, entre outras.

O MetaMiner pretende servir como uma meta ferramenta de busca para o usuário da Internet. Em especial, para o usuário brasileiro que pode realizar pesquisas utilizando apenas ferramentas de busca nacionais.

O MetaMiner foi desenvolvido para ser, também, um laboratório para estudar técnicas de recuperação de informação como:

- *clustering* [Chang,1997]: os resultados retornados pelo MetaMiner podem ser agrupados por assunto para auxiliar o usuário em sua pesquisa;

- classificação [Selberg & Etzioni,1995]: as referências retornadas podem ser recuperadas automaticamente e então classificadas usando métodos como TF/IDF [Salton,1983];

- *relevance feedback* [Salton,1983]: uma interface pode ser criada para que o usuário possa fornecer *feedback* sobre a importância dos documentos recuperados com intuito de melhorar futuras consultas.

Outro papel fundamental da meta ferramenta de busca da *Família Miner* é servir como ambiente para testes de outros agentes da Família e comparar as diversas ferramentas de busca disponíveis. O MetaMiner foi utilizado, juntamente com o TraceMiner, para realizar comparações entre diferentes ferramentas de busca e os agentes da família (exemplo: PSSEMiner x Ferramentas de busca tradicionais).

O MetaMiner é uma *servlet* desenvolvida inteiramente em Java que utiliza a API Miner e em especial o `package miner.engine`, originalmente concebido para atender esse membro da *Família Miner*. Como o MetaCrawler, o MetaMiner não segue o padrão de exclusão de robôs e também não verifica o tempo existente entre intervalos de acesso aos recursos utilizados. O motivo para tanto é bastante simples, o MetaMiner é ativado diretamente por um pessoa que seleciona **manualmente** quais ferramentas de busca devem ser ativadas. Dessa forma, o MetaMiner age simplesmente como um catalisador para realizar uma consulta de um usuário.

A questão se meta ferramentas de busca devem obedecer ou não o padrão de exclusão de robôs foi amplamente debatida dentro da *robots mailing list* e permanece ainda em aberto. Porém, a conclusão mais aceita é que uma meta ferramenta de busca não precisa obedecer ao padrão de exclusão de robôs, mas deve seguir outras éticas como identidade. Além disso, caso a meta ferramenta de busca realize buscas adicionais em servidores que não estejam preparados para atender um grande número de requisições, então ela deverá obedecer ao padrão de exclusão. Esse último caso ocorre quando a resposta retornada pelas ferramentas de busca é também coletada pela meta ferramenta de busca para realizar análises para o usuário.

Assim, apesar de inicialmente o MetaMiner possuir capacidade para coleta dos *links* fornecidos como resultado pelas diversas ferramentas de busca consultadas, essa etapa não mais existe, conforme é possível ver na Figura 14 que mostra o processo completo de funcionamento do MetaMiner. O motivo para tanto é que o respeito ao padrão de exclusão poderia tornar uma consulta muito lenta para o usuário.

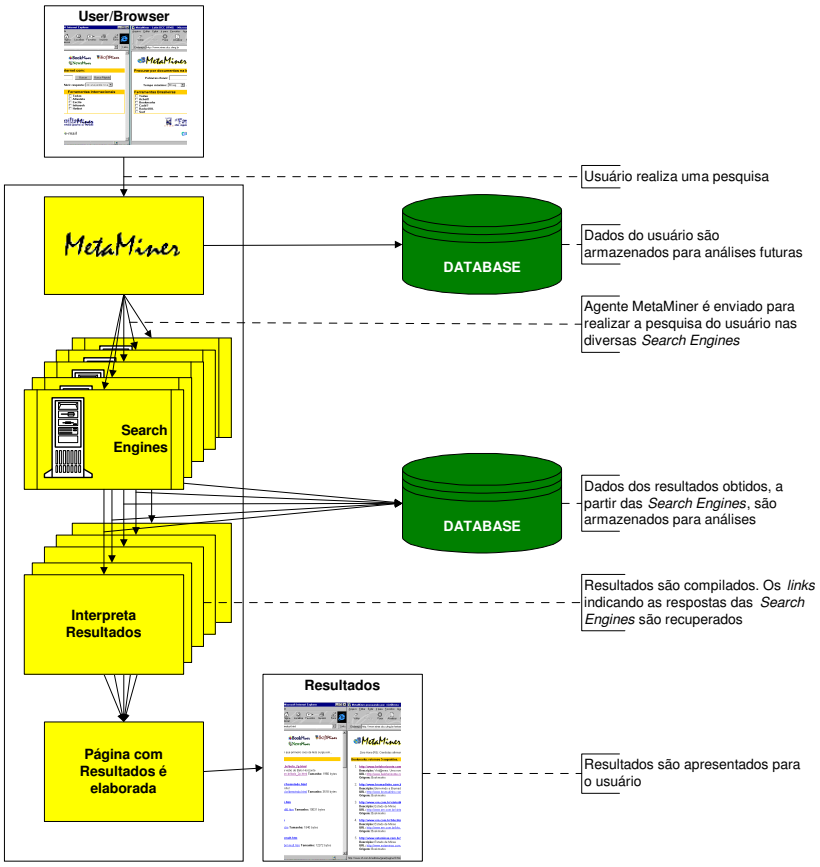


Figura 14: Funcionamento do MetaMiner

O MetaMiner possui uma opção de busca rápida de forma que ao receber uma resposta de uma ferramenta de busca essa é enviada diretamente para o usuário, não existindo portanto o agrupamento de *links* repetidos. Essa opção foi inserida devido aos resultados obtidos em [Selberg & Etzioni,1995] que mostram que o número de *links* iguais enviados pelas diferentes ferramentas de busca é ínfimo. Dessa forma, a espera de todas as respostas objetivando realizar um agrupamento não faz sentido a menos que processamentos mais sofisticados sejam realizados.

Informações Armazenadas pelo MetaMiner

O MetaMiner armazena diversos *logs* obtidos em cada pesquisa realizada. Indo da palavra-chave utilizada até o tempo de resposta de cada uma das ferramentas de busca utilizadas em uma pesquisa. Esses resultados são armazenados em dois arquivos texto, onde cada linha é um registro, e que são estruturados da seguinte forma:

- Arquivo `LogMetaMiner.txt` – data e hora da pesquisa, endereço IP do cliente, `user-agent` do cliente, palavras-chave da pesquisa, opção de janela destino, opção de tempo limite para pesquisa, tempo total da pesquisa em milisegundos, número de resultados obtidos (total).

- Arquivo `LogEngines.txt` – data e hora da pesquisa, endereço IP do cliente, nome da ferramenta de busca, tempo de busca nessa ferramenta, total de resultados retornados por essa ferramenta.

4.1.3. PSSEMiner

O *Proxy Server Search Engine Miner* (PSSEMiner) é um motor de busca cujos robôs buscam informações em servidores *proxy*, ao contrário das outras ferramentas de busca que possuem robôs que vão até a Internet para coletar dados. Esse novo tipo de abordagem permite uma grande economia de tráfego na rede quando o servidor *proxy* estiver conectado através de uma rede local ao servidor do motor de busca.

O PSSEMiner foi desenvolvido juntamente com Tiago Garcia, como projeto do curso “Arquitetura de Servidores WWW” ministrado pelo professor Virgílio Almeida durante o 1º semestre de 1997 no DCC UFMG.

O principal objetivo do PSSEMiner é apresentar uma nova forma para coletar documentos da Internet com a finalidade de elaborar uma base de dados para uma ferramenta de busca. O aproveitamento de recursos, como o *cache* dos servidores *proxy*, possibilita que os robôs do PSSEMiner sejam os próprios usuários do *proxy*.

O PSSEMiner pode ser utilizado também como ambiente para o estudo de *clustering*, classificação e *relevance feedback*.

Servidores Proxy

O servidor *proxy* é um meio para armazenar objetos requisitados na Internet em um *site* mais próximo ao requisitante do que a fonte original dos dados [Wessels,1995], reduzindo, assim, o tempo de acesso a esses objetos bem como largura de banda consumida. Um servidor *proxy* pode ser visto como um *cache* de rede.

Squid: O Servidor Proxy utilizado pelo POP-MG

Squid é um servidor *proxy* para clientes *Web* derivado do projeto HARVEST [Manber et al,1995] e resultado do esforço de inúmeras pessoas, em especial de *Duane Wessels* [Wessels,1995] do "*National Laboratory for Applied Network Research*", que o implementou.

O *Squid* tem capacidade para suportar objetos de dados de diferentes protocolos (como HTTP, FTP e *Gopher*); armazenar buscas de DNS (*domain name server*); manter, na memória principal, os objetos mais importantes; suportar SSL (*secure server layer*); controlar o acesso de forma extensiva e também para criar um registro completo de requisições. O *Squid* pode ser organizado hierarquicamente para economizar mais largura de banda, conforme exemplificado na Figura 15.

Buscando informações no Servidor Proxy

O funcionamento da coleta de objetos da *World-Wide Web* em um servidor *proxy* pelo PSSEMiner pode ser dividido em duas etapas principais:

1. Elaboração de uma lista de objetos presentes no servidor *proxy*;

2. Coleta dos objetos desejados a partir do servidor *proxy*;

A primeira etapa é feita através de uma página HTML especial que o módulo de administração do *Squid* gera automaticamente, listando todos os objetos presentes no *cache*, e que estaria disponível diariamente em um horário pré-determinado.

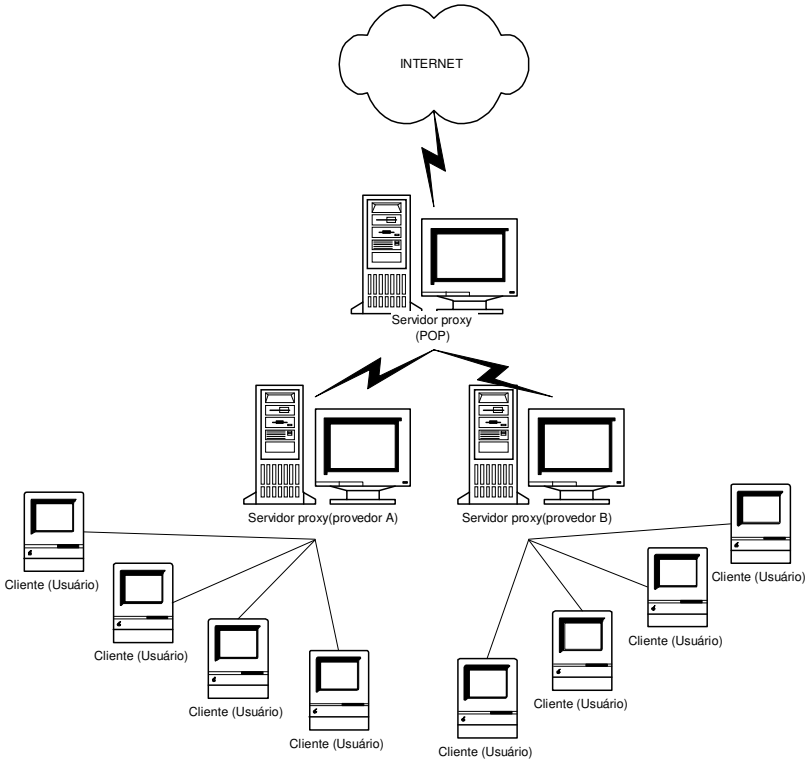


Figura 15: Exemplo de hierarquia com o *Squid*

A segunda etapa consiste da análise da lista obtida na etapa anterior, verificando quais documentos devem ser inseridos (páginas novas) ou atualizados. A requisição desses documentos é feita através do protocolo HTTP conectando-se diretamente ao servidor *proxy,* para então, requisitar as páginas de interesse.

Por exemplo, suponha que um documento fictício representado pela URL `http://www.algumlugar.com.br/` não estivesse presente na base de dados do PSSEMiner mas estivesse na lista de objetos do *Squid*. Uma conexão HTTP é estabelecida com o servidor *proxy* em cache.pop-mg.rnp.br no porto 3128 e então, a seguinte mensagem é enviada:

```
GET http://www.algumlugar.com.br/ HTTP/1.0
Accept: text/html
Referer: http://www.miner.dcc.ufmg.br/info.html
```

```
User-Agent:Mozilla 2.0(compatible;psseminer;victor@dcc.ufmg.br)
From: victor@dcc.ufmg.br
```

O resultado desta transação é a cópia do documento dessa URL, que está armazenado no servidor *proxy*. A diferença entre esse tipo de conexão (via *proxy*) e uma conexão normal HTTP é que o *proxy* atua como um intermediário. A conexão que irá solicitar o documento requisitado, passa a ser feita com o servidor *proxy* (que está localizado em `cache.pop-mg.rnp.br:3128`), e não com o servidor final – localizado em `www.algumlugar.com.br` (Figura 16).

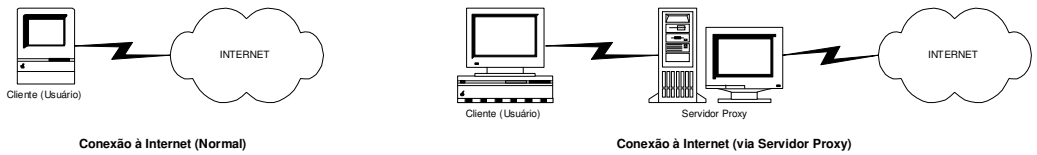


Figura 16: Conexão à Internet via Servidor *Proxy*

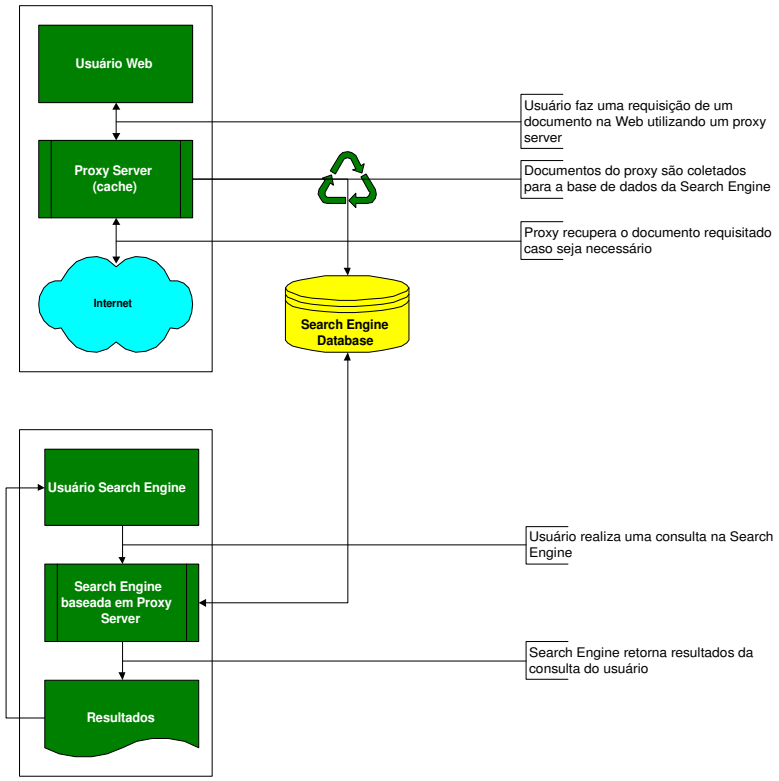


Figura 17: Funcionamento do PSSEMiner

O critério para atualização de uma página na base de dados do PSSEMiner é baseado no tamanho do documento. Sempre que o tamanho mencionado na lista fornecida pelo *Squid* for diferente do tamanho previamente armazenado, uma atualização ocorre.

Armazenamento das Informações e Estrutura da Base de Dados

A base de dados do PSSEMiner foi estruturada para permitir consultas de usuários que buscam documentos na *Web* e para que análises pudessem ser realizadas (como tipo de servidores *Web* presentes na Internet, tamanhos de documentos, tempos de recuperação, versões de protocolo HTTP mais utilizadas).

Tabela 3: Tabela tbl_index da base de dados do PSSEMiner

tbl_index: tabela contendo índice dos documentos armazenados		
Atributo	**Tipo**	**Descrição**
url	Texto	url do documento
page_size	Texto	Tamanho da página (em bytes – fornecido pelo Squid)

Tabela 4: Tabela tbl_paginas da base de dados do PSSEMiner

tbl_paginas: tabela contendo Informações dos documentos recuperados		
Atributo	**Tipo**	**Descrição**
url	Texto	url do documento
Date	Texto	Data em que o documento foi recuperado
Title	Texto	Título do documento
webserver	Texto	Servidor *Web* provedor do documento
Httpcode	Texto	Código da transação HTTP realizada
page_size	Inteiro	Tamanho da página (em bytes)
retrieve_time	Inteiro	Tempo de recuperação do documento em (miliseg.)
Content	Texto	Conteúdo da página

4.2. AGENTES DE COMPRA

Os agentes de compra da *Família Miner* procuram explorar o crescimento do comércio eletrônico na Internet. O modelo de classes utilizado é muito semelhante para todos os

agentes de compra o que possibilita uma fácil expansão para agentes ligados a outros produtos além dos aqui citados.

4.2.1. BookMiner

O BookMiner é um agente especializado em localizar livros em livrarias brasileiras e estrangeiras que estão presentes na *World-Wide Web*. Como o BookFinder [Willians,1996], o BookMiner replica a procura por um livro para diversas livrarias diferentes e, ao contrário do BookFinder, agrupa os diversos títulos retornados apresentando uma formatação única que permite não só uma melhor visualização dos resultados como também facilita a comparação de preços pelo o usuário.

O comércio de livros é um dos negócios pioneiros na Internet. A facilidade para manipulação de livros (produto não perecível, sem problemas com temperaturas elevadas ou baixas, fácil expedição – correio comum) explica o sucesso desse negócio na Internet. Para o Brasil em especial, o custo da compra de um livro no exterior possui apenas o acréscimo do frete que geralmente não cobre o preço de produto nacional semelhante. Para a comunidade científica, principalmente de áreas que estão em constante evolução – como a computação – o comércio de livros pela Internet facilitou o acesso às obras atualizadas.

Uma tendência que vem surgindo nos serviços de venda de livros pela *World-Wide Web* é a criação do representante virtual: qualquer pessoa que sugerir livros em sua *home-page* e que, devido a essa sugestão, acabe por realizar uma venda na livraria indicada, terá uma comissão sobre essa venda.

Nos registros de visitantes do BookMiner constam nomes de máquinas da Amazon, Bookpool, Livraria Cultura e Booknet. Em especial, a gerência da Livraria Cultura manifestou seu interesse no BookMiner e agradeceu a inclusão da Livraria Cultura como uma das opções para os usuários.

O BookMiner é uma *servlet* desenvolvida inteiramente em Java que utiliza a API Miner e em especial o `package miner.book`. Pode ser definida como uma ferramenta destinada ao consumidor de livros nacionais e internacionais que procura não só facilitar a localização de títulos como também permitir que o usuário possa escolher o melhor preço. O BookMiner, como o MetaMiner e os outros agentes de compra

também não obedece o padrão de exclusão de robôs mas segue outras éticas e diretivas básicas para o desenvolvimento de agentes.

O funcionamento do BookMiner é ilustrado na Figura 18.

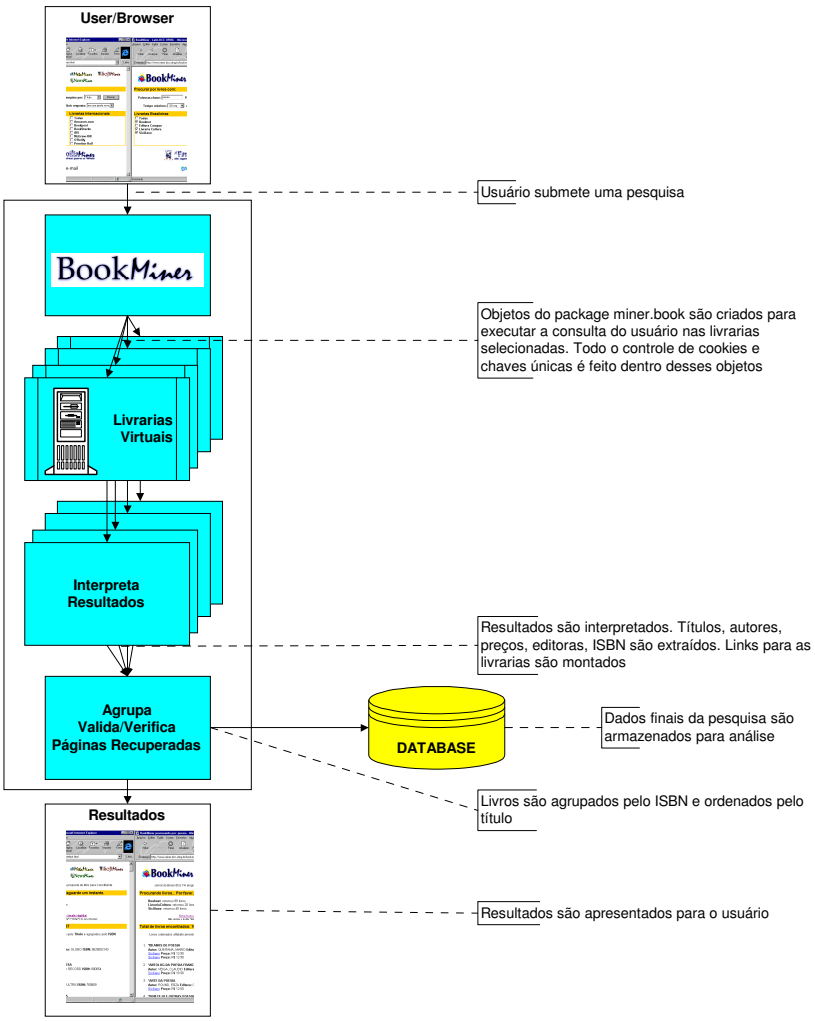


Figura 18: Funcionamento do BookMiner

Informações Armazenadas pelo BookMiner

Como o MetaMiner, o BookMiner armazena diversos *logs* obtidos em cada pesquisa realizada, indo da palavra-chave utilizada até o tempo de resposta de cada uma das livrarias utilizadas em uma consulta. Esses resultados são armazenados em dois arquivos texto, onde cada linha é um registro, e que são estruturados da seguinte forma:

- Arquivo `LogBookMiner.txt` – data e hora da pesquisa, endereço IP do cliente, `user-agent` do cliente, palavras-chave da pesquisa, tipo da pesquisa (autor ou título), opção de janela destino, opção de tempo limite para pesquisa, tempo total da pesquisa em milisegundos, número de resultados obtidos (total).

- Arquivo `LogLivrarias.txt` – data e hora da pesquisa, endereço IP do cliente, nome da livraria, tempo de busca nessa livraria, total de resultados retornados por essa livraria.

4.2.2. SoftMiner

O SoftMiner é um agente especial da *Família Miner* especializado em consultar repositórios de *software* dos tipos *shareware* ou *freeware*. A idéia básica para o desenvolvimento desse agente é derivada do aparecimento de diversas palavras-chave nos *logs* do MetaMiner sugerindo uma procura por *software*.

O SoftMiner é uma ferramenta destinada ao usuário da Internet que esteja procurando por *software* dos tipos *shareware* ou *freeware*, procurando facilitar a localização e escolha do melhor produto que atenda a suas expectativas.

Como livros, software é um outro objeto de fácil venda e distribuição na Internet. Programas de computador são atrativos básicos para usuários da Internet que logicamente possuem computadores em casa ou no trabalho.

Pequenos desenvolvedores podem se beneficiar da facilidade de distribuição existente na Internet para venderem seus produtos. Os regimes de venda do tipo *shareware*, onde uma versão limitada ou mesmo completa é oferecida por um período de tempo limitado ao usuário, também são bastante utilizados na Internet.

O funcionamento do SoftMiner é bastante semelhante ao da busca rápida do MetaMiner (Figura 19). Devido a não existência de uma chave que sirva como meio para agrupar uma ou mais resposta, não há motivos para que todos os resultados sejam recebidos para depois serem enviados para o usuário. Uma possibilidade, para o agrupamento, seria o nome do arquivo, entretanto esse nem sempre é apresentado diretamente nas respostas enviadas pelos repositórios de *software* durante uma consulta.

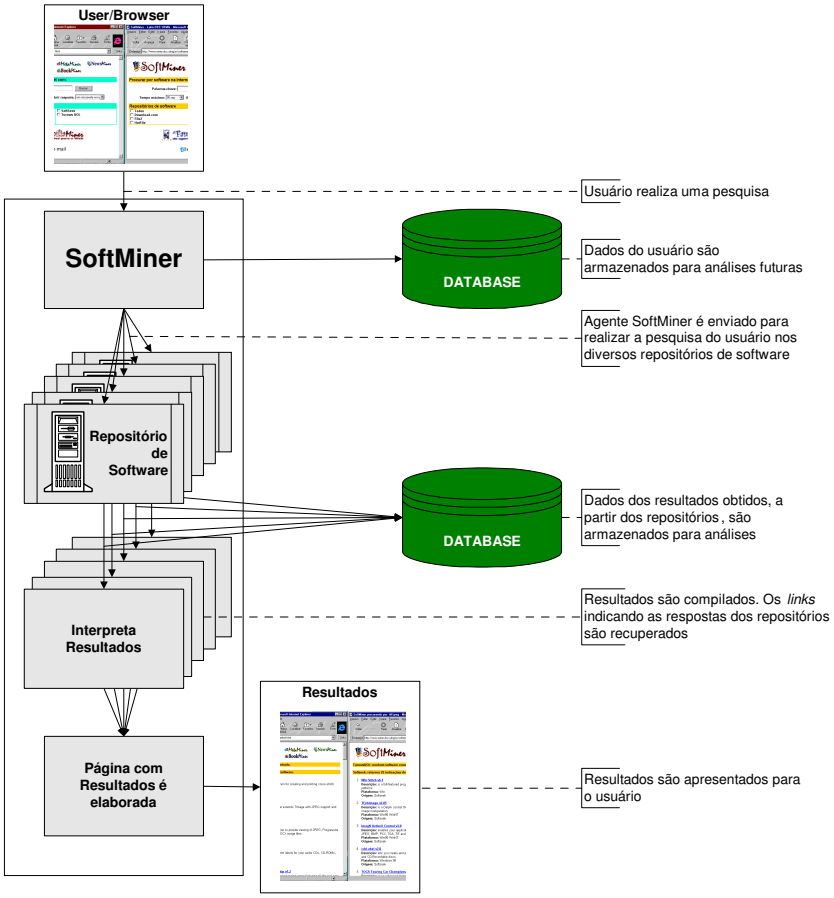


Figura 19: Funcionamento do SoftMiner

Informações Armazenadas pelo SoftMiner

O SoftMiner armazena diversos *logs* obtidos em cada pesquisa realizada, como os outros agentes da *Família Miner*. Os dados são armazenados em dois arquivos texto, onde cada linha é um registro, e que são estruturados da seguinte forma:

- Arquivo `LogSoftMiner.txt` – data e hora da pesquisa, endereço IP do cliente, `user-agent` do cliente, palavras-chave da pesquisa, opção de janela destino, opção de tempo limite para pesquisa, tempo total da pesquisa em milisegundos, número de resultados obtidos (total).

- Arquivo `LogRepositorios.txt` – data e hora da pesquisa, endereço IP do cliente, nome do repositório de *software*, tempo de busca nesse repositório, total de resultados retornados por esse repositório.

4.2.3. CDMiner

CDs, como livros, são produtos de fácil comércio na Internet. A facilidade de manuseio, distribuição entre outras características aliadas com o perfil dos usuários da Internet, tornam o CD um produto comercialmente forte na Internet.

O CDMiner é um agente da *Família Miner* especializado na procura de CDs em lojas brasileiras e estrangeiras presentes na *World-Wide Web*. Funcionando como o CDFinder [Willians,1996], o CDMiner replica a procura por um CD para diversas lojas diferentes. Entretanto, como o BookMiner, o CDMiner ordena os diversos títulos retornados, apresentando uma formatação única que permite não só uma melhor visualização dos resultados como também facilita a comparação de preços pelos usuários.

Seguindo a mesma tendência apresentada pelas livrarias, as lojas de CD estão criando os representantes virtuais que passam a receber comissões quando ajudam a loja a realizar uma venda através da indicação de links em *home-pages.*

O CDMiner é uma *servlet* que utiliza a API Miner e em especial o `package miner.cd`. Pode ser definida como um serviço destinado ao consumidor de CDs nacionais e internacionais que procura não só facilitar a localização de CDs como também permite ao usuário escolher a loja que oferece o melhor preço.

O funcionamento do CDMiner é ilustrado na Figura 20 e é bastante semelhante ao funcionamento do BookMiner.

Informações Armazenadas pelo CDMiner

O CDMiner armazena diversos *logs* obtidos em cada pesquisa realizada, como os outros agentes da *Família Miner*. Os dados são armazenados em dois arquivos texto, onde cada linha é um registro, e que são estruturados da seguinte forma:

- Arquivo `LogCDMiner.txt` – data e hora da pesquisa, endereço IP do cliente, `user-agent` do cliente, palavras-chave da pesquisa, tipo da pesquisa (artista ou título da obra musical), opção de janela destino, opção de tempo limite para pesquisa, tempo total da pesquisa em milisegundos, número de resultados obtidos (total).

- Arquivo `LogLojasCD.txt` – data e hora da pesquisa, endereço IP do cliente, nome da loja, tempo de busca nessa loja, total de resultados retornados por essa loja.

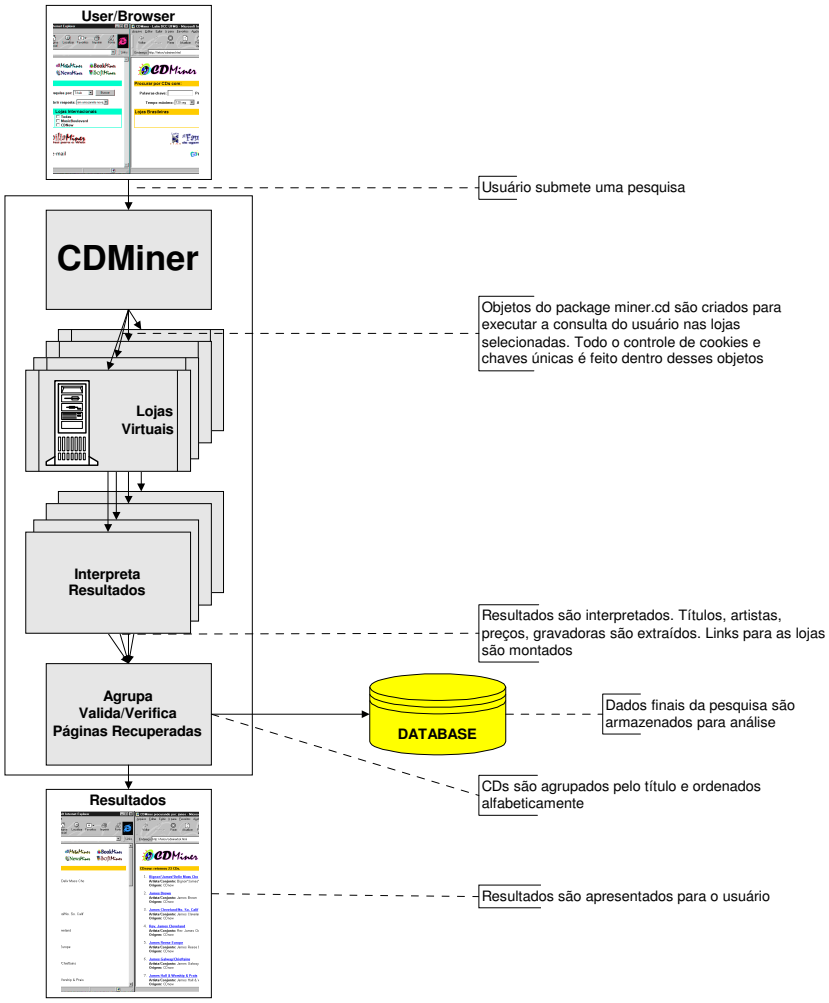


Figura 20: Funcionamento do CDMiner

4.3. AGENTES DE NOTÍCIAS

A informação certa no momento certo sempre é peça fundamental para que empresas e pessoas possam tomar decisões corretas. O grande número de meios de comunicação e a grande variedade de empresas existentes na área jornalística, criaram um novo mercado de especialistas que passam literalmente o dia "recortando" notícias de diversas revistas e jornais, para compor o que é chamado de *clipping*: um boletim contendo notícias de diversas fontes focadas em um assunto específico.

A elaboração de *clippings* é tradicionalmente manual. Pessoas lêem diversas revistas e jornais e com base no interesse de seus clientes, as notícias consideradas importantes são selecionadas e adicionadas ao *clipping*.

Com a chegada da era digital, os próprios jornais, revistas e agências de notícias passaram a disponibilizar notícias na *World-Wide Web*. Ao mesmo tempo, as empresas de *clipping* passaram a utilizar o correio eletrônico como uma nova forma para distribuir os boletins que são editados manualmente.

Na próxima seção é apresentada uma nova concepção para a elaboração de *clippings* de notícias. A idéia básica é substituir o serviço manual de seleção das notícias por algoritmos de classificação e *relevance feedback* e pela coleta e distribuição automática das notícias, utilizando para tanto as informações disponibilizadas na Internet.

4.3.1. NewsMiner

Em abril de 1997, um novo serviço de informações surgiu na *Web*: o NewsTracker[19] – um motor de busca americano especializado em jornais presentes na *Web*. Nessa mesma época, o NewsMiner – agente de notícias da *Família Miner* – era desenvolvido como parte da disciplina "Arquitetura de Servidores WWW" ministrado pelo professor Virgílio Almeida do Departamento de Ciência da Computação da UFMG.

A primeira versão do NewsMiner entrou em funcionamento em maio de 1997, coletando notícias de diversos jornais brasileiros. Sua função era e ainda é buscar todas as reportagens disponibilizadas por jornais presentes na Internet previamente escolhidos. Observando a estrutura das páginas HTML coletadas, uma base de dados estruturada era elaborada para, então, ser disponibilizada para consultas por usuários da *Web* e, também, para compor *clippings* eletrônicos a serem distribuídos por meios digitais. Os programas de indexação e classificação, responsáveis pela elaboração desses *clippings*, foram desenvolvidos em C++ por Edleno Silva de Moura e Ramurti de Alencar Barbosa.

A coleta e a estruturação de informações do NewsMiner são baseadas no `package miner.news` da API Miner. Dentro dos objetos de cada jornal é inserido o conhecimento suficiente para que as informações buscadas sejam coletadas com

[19] http://www.newstracker.com/

sucesso, o que é feito através de uma análise prévia do *site* de cada jornal na *Web*. A estruturação das páginas no *site*, o formato em que é publicada uma reportagem, as URLs das principais páginas juntamente com qualquer característica particular do jornal são estudados para elaboração dos algoritmos de busca.

Todas as páginas coletadas são analisadas para estruturar as informações nelas contidas. Dados como o título da reportagem, repórter e a matéria publicada são localizados dentro do documento HTML e armazenados de forma estruturada. Isso é, particularmente, útil para o desenvolvimento de interfaces de consulta e servindo também como informação para o estudo de técnicas de classificação.

A concepção inicial do NewsMiner possibilitava o cadastramento de usuários que desejem receber diariamente um *clipping* de notícias através de correio eletrônico. Para tanto, os usuários tinham que informar seu perfil de interesse. Isso era feito usando uma interface que foi desenvolvida para as pessoas cadastrarem palavras-chave de interesse juntamente com um número que indicaria a relevância de cada uma dessas palavras-chave.

O funcionamento do NewsMiner podia ser resumido da seguinte forma (Figura 21): a coleta das reportagens é feita uma vez ao dia (diariamente às 4:00 am). Ao final dessa fase, um programa desenvolvido em C++ é chamado para gerar uma lista invertida das reportagens coletadas, sendo que utilizado apenas o texto de cada reportagem para gerar o índice. Com a lista invertida gerada, um programa de classificação baseado no modelo vetorial é executado utilizando o perfil de cada usuário cadastrado para classificar e filtrar as notícias do dia que serão enviadas para esse usuário. Finalizando uma execução diária, o NewsMiner era encarregado de compor e enviar um *e-mail* contendo as notícias classificadas para cada usuário cadastrado.

Entretanto, o NewsMiner foi planejado também para funcionar como um ambiente para análise e estudo de técnicas de recuperação de informação como indexação, classificação, *clustering* e *relevance feedback*. Essa capacidade, de servir como laboratório, foi comprovada no experimento em que o NewsMiner foi integrado ao Sistema Gama [Calado et al,1997], que é detalhado a seguir.

NewsMiner e o Sistema GAMA

O Gama é um sistema de inferência automática de perfis genéricos que pode ser utilizado por diferentes tipos de aplicações. O Gama é capaz de fazer uma

classificação automática de documentos baseado em um perfil que é gerado interativamente com o usuário que periodicamente deve submeter documentos com notas que variam de 1 a 5 que irão indicar o grau de importância desse documento. Com esse perfil montado, o Gama passa a oferecer a possibilidade de classificar qualquer documento que o usuário queira.

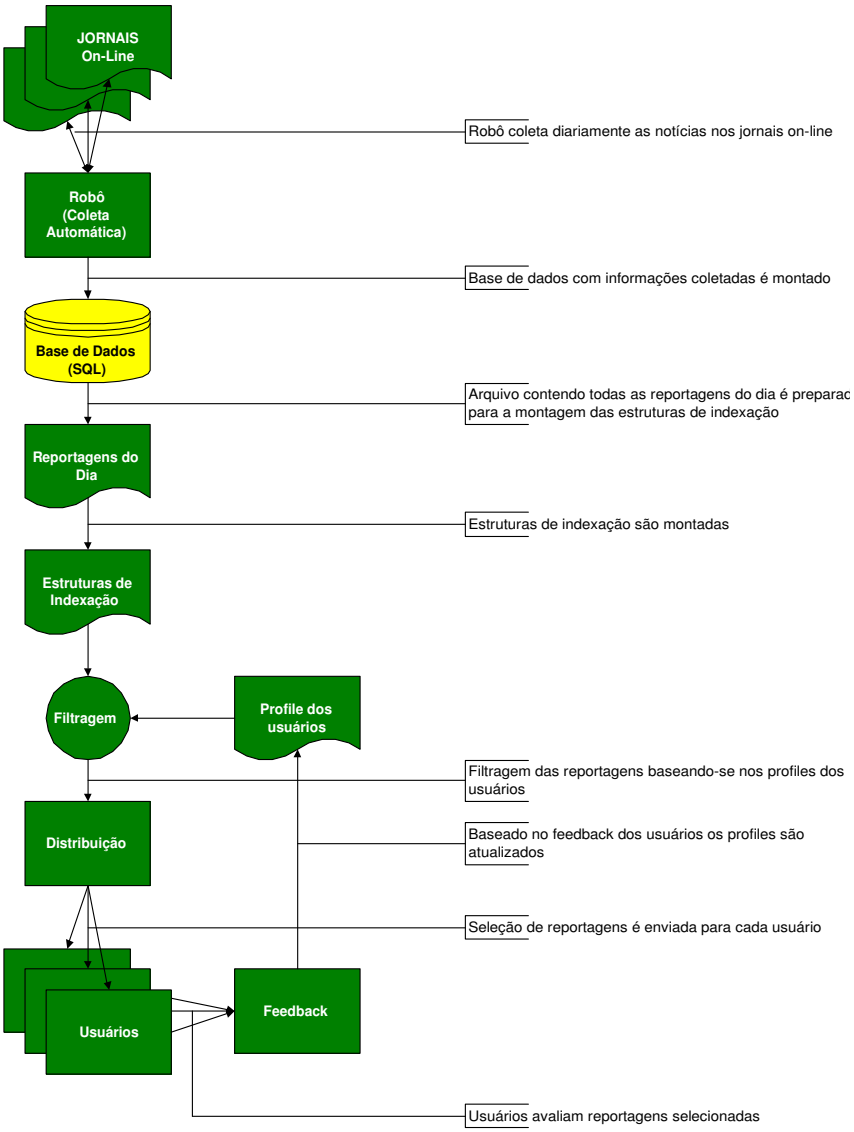


Figura 21: Funcionamento do NewsMiner

Com o NewsMiner, o Gama substituiu os módulos de geração de lista invertida, classificação e eliminou a interface necessária para o cadastramento de palavras-chave que formariam o perfil do usuário.

Usando o Gama, o usuário do NewsMiner pode ao ler uma notícia de um jornal, avaliá-la e submetê-la para que o sistema atualize o seu perfil automaticamente, o que de certa forma parece ser melhor do que exigir que o usuário tenha que cadastrar e lembrar de todas as palavras-chave que são importantes para ele. O perfil que estaria sendo montado seria utilizado diariamente pelo NewsMiner que solicitaria ao Gama para classificar as notícias do dia com base nos perfis dos usuários cadastrados.

4.4. AGENTES DE MONITORAMENTO DA FAMÍLIA

Os agentes de monitoramento da *Família Miner* foram desenvolvidos para observar, testar e facilitar o gerenciamento de todos os outros agentes da Família.

Esses agentes estão mais próximos de conceitos – a serem inseridos dentro dos algoritmos e estruturas dos outros membros da *Família Miner* – do que propriamente de programas que funcionam isoladamente.

4.4.1. WatcherMiner

O WatcherMiner é um *software* de monitoração capaz de enviar um *e-mail* para o responsável pela *Família Miner,* alertando sobre alguma anormalidade que tenha acontecido durante a execução de um dos membros da *Família Miner*. Isso é particularmente útil para aplicações que estejam disponíveis para consulta 24 horas por dia na *World-Wide Web*, permitindo que medidas sejam tomadas, o mais rápido possível, sempre que necessárias.

No MetaMiner, por exemplo, sempre que uma das ferramentas de busca não retornar resultados 10 vezes seguidas, o WatcherMiner é ativado para alertar sobre a possível mudança de formato nas páginas dessa ferramenta. De maneira semelhante, esse agente de monitoramento é utilizado no BookMiner, CDMiner e SoftMiner.

No NewsMiner, o WatcherMiner envia um relatório – ao final de cada sessão de coleta – contendo o número de reportagens recuperadas no dia em cada jornal. A presença de um jornal que não tenha nenhuma reportagem coletada pode significar uma mudança de formato de suas páginas ou da organização do *site* desse jornal na *Web*.

A evolução natural do WatcherMiner é a inserção de mecanismos que verifiquem automaticamente se o formato dos recursos consultados foi realmente alterado ou não, reduzindo assim a necessidade de uma verificação manual.

Chamadas do WatcherMiner também são inseridas dentro dos agentes da *Família Miner* para reportar mensagens de erros.

4.4.2. TestMiner

Conceitualmente, o TestMiner é um *software* que foi desenvolvido utilizando a API Miner, onde cada *package* dessa API possui um TestMiner correspondente. A idéia é criar a possibilidade para realizar testes com os diversos recursos da Internet já cadastrados e a serem cadastrados na API Miner. O TestMiner é também utilizado para realizar testes de performance com os próprios agentes da *Família Miner*.

Na prática, o TestMiner é uma "máscara" que é constantemente alterada para realizar testes que sejam necessários com os agentes da *Família Miner* ou com os recursos por ela utilizados. Em resumo, o TestMiner "imita" uma página HTML contendo um formulário.

O TestMiner foi utilizado, por exemplo, para realizar uma série de testes com o PSSEMiner. Uma lista contendo 567 consultas realizadas no MetaMiner, contendo uma ou mais palavras-chave, foi submetida ao TestMiner que realizou cada uma das consultas no PSSEMiner observando se eram ou não retornados resultados.

Porém, como é possível observar, o TestMiner possibilita apenas análise quantitativas como performance, tempos de execução e quantidade de resultados obtidos. Nada relacionado a qualidade dos resultados retornados pode ser concluído com base em respostas fornecidas pelo TestMiner.

4.4.3. TraceMiner

O TraceMiner foi criado procurando criar uma forma para medir a qualidade dos resultados retornados pelos agentes da *Família Miner*, através de observações feitas a partir do comportamento do usuário com a resposta obtida de uma consulta a um dos agentes da Família.

Dessa forma, o TraceMiner pode ser utilizado para responder perguntas do tipo: entre os resultados gerados pelo MetaMiner, qual é a ferramenta de busca que possui mais URLs seguidas pelos usuários? Entre os livros retornados pelas livrarias do BookMiner qual tem sido selecionado? Existe alguma tendência em selecionar os resultados listados primeiramente? E assim por diante.

O TraceMiner pode funcionar de duas formas distintas:

- Primeiramente, como um agente especial desenvolvido em Java/JavaScript [Flanagan,1997; Kramer,1997] enviado juntamente com as respostas para o usuário para observar qual item entre os fornecidos como resposta está sendo seguido. Dessa forma, quando o usuário seleciona um dos itens da resposta, antes do navegador abrir o documento apontado pela URL selecionada, o TraceMiner dispara uma mensagem para o servidor da *Família Miner* indicando qual foi essa URL e qual foi o recurso original consultado pelo agente Miner que enviou essa resposta;

- A outra forma é feita através da alteração da URL que é enviada como resposta a uma consulta de um usuário a um agente da *Família Miner*. Ao invés de enviar diretamente uma URL que aponta para o documento desejado, uma nova URL é inserida. Essa nova URL contém uma consulta com a URL original e outras informações e que aponta para uma *servlet* no servidor da *Família Miner*. Assim, quando o usuário selecionar uma das respostas fornecidas, a *servlet* apontada no servidor da Família é ativada, registrando a resposta escolhida e redirecionando a requisição do usuário para a URL original, fornecida como parâmetro.

A primeira forma apresenta a vantagem de manter as URLs fornecidas nas respostas intactas – escritas tais como elas realmente foram indicadas. Por outro lado, JavaScript ainda não está suficientemente padronizado e adotado, o que impede que amostras razoáveis sejam coletadas. A segunda forma, apesar de alterar o formato da URL, torna possível o rastreamento para qualquer um dos usuários da *Família Miner*, independentemente do navegador que ele esteja utilizando.

5. Resultados

O principal resultado alcançado nesse trabalho foi a implementação da API Miner que possibilitou o desenvolvimento dos agentes da *Família Miner*. Originalmente apenas com os *packages* `miner.engine` e `miner.book`, a API Miner conta atualmente com mais 7 *packages* específicos para suportar a *Família.* O desenvolvimento de novas aplicações bem como a criação de novos *packages* podem ser feitos facilmente

Neste Capítulo, são apresentados resultados específicos do MetaMiner, BookMiner, PSSEMiner e NewsMiner. São mostrados dados contendo características de seus usuários e dos serviços utilizados.

5.1. AGENTES DE BUSCA

5.1.1. MetaMiner

Nesta seção, são mostradas análises sobre as consultas realizadas no MetaMiner. O período de amostragem refere-se, principalmente, ao intervalo entre 27 de Abril de 1997 e 4 de Agosto de 1997. No entanto, o armazenamento de alguns dados, como o número de URLs retornadas pelos serviços de busca, teve início após o dia 27 de Abril de 1997.

Nem todas ferramentas de busca utilizadas, atualmente, pelo MetaMiner serão abordadas. O motivo para tanto é o pouco tempo e o pequeno número de amostras referentes a serviços que foram inseridos mais tardiamente na API Miner, como, por exemplo, RadarUOL e Infoseek.

As análises que seguem estão assim divididas: as estatísticas gerais de acesso detalham dados sobre sistemas operacionais e navegadores utilizados pelos usuários do MetaMiner. Em seguida, é exposta uma análise do número de palavras-chave utilizadas nas consultas. Finalizando, são mostrados dados sobre a velocidade de resposta e sobre o número de URLs retornadas por cada serviço de busca.

Estatísticas Gerais de Acesso

O principal navegador utilizado pelo usuário do MetaMiner, no período de análise, foi o Netscape Navigator (77% dos acessos – Gráfico 1). Além do Netscape e do Microsoft Internet Explorer, nenhum outro navegador foi detectado. Um total de 76 versões diferentes desses navegadores, variando entre sistemas operacionais e *releases*, foram observadas.

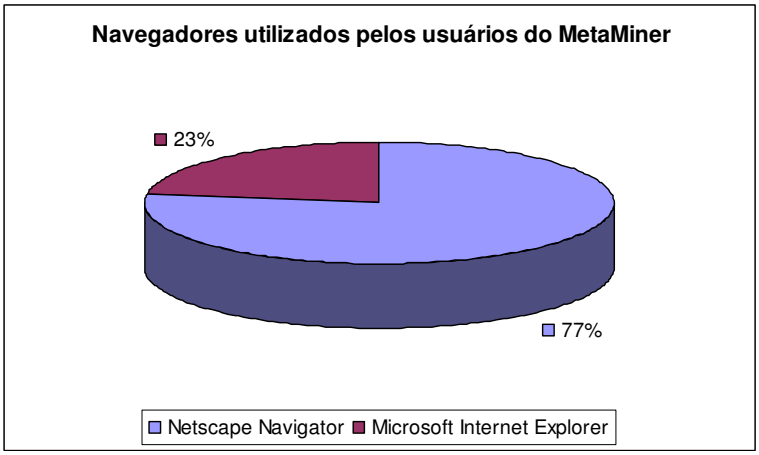


Gráfico 1: Navegadores mais utilizados pelos usuários do MetaMiner

Amostra refetrente ao período de 27/Abr/1997 a 04/Ago/1997

A plataforma Microsoft Windows é a predominante nos navegadores dos usuários do MetaMiner (94% - Gráfico 2). Apenas 6% utilizam o SunOS da Sun Microsystems, enquanto a quantidade de usuários com Machintosch e Linux é irrisória.

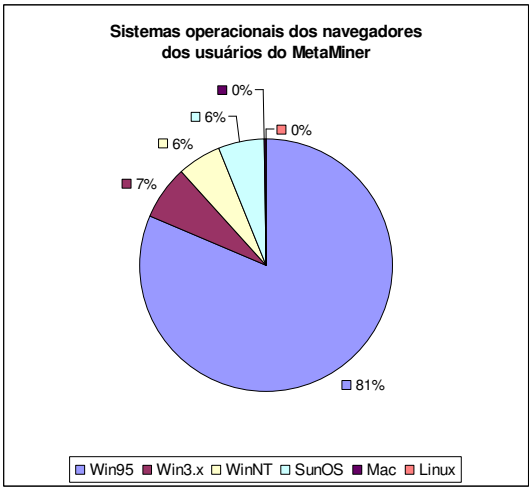


Gráfico 2: Sistemas operacionais dos navegadores dos usuários do MetaMiner

Amostra referente ao período de 27/Abr/1997 a 04/Ago/1997

Número de Palavras-Chave Utilizadas nas Consultas

52% das consultas realizadas, no MetaMiner, são formadas por apenas 1 (uma) palavra-chave. 6% das consultas foram realizadas com mais de 4 palavras (Gráfico 3).

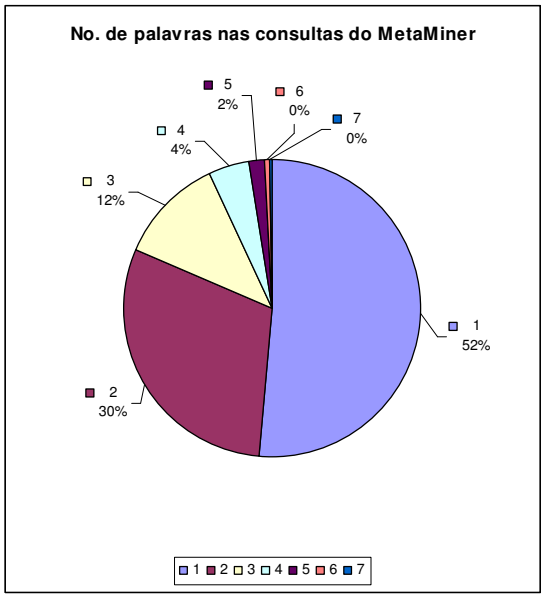


Gráfico 3: Número de palavras-chave utilizadas nas consultas ao MetaMiner

Amostra referente ao período de 27/Abr/1997 a 04/Ago/1997

Tempo de Reposta das Ferramentas de Busca

Em média, os serviços de busca nacionais retornaram resultados às consultas realizadas mais rapidamente do que os serviços estrangeiros, conforme é possível observar no Gráfico 5. É importante salientar que aqui não está sendo medida a qualidade das URLs retornadas apenas a velocidade em que as consultas foram respondidas.

Entre os serviços brasileiros, o Yaih? foi o mais rápido. Ressalta-se que o Yaih? foi um projeto pioneiro da RNP e também que a ferramenta estava diretamente ligada à rede do DCC da UFMG, ao contrário do Cadê? que está ligado ao *backbone* da Embratel.

Em geral, 98% dos resultados de todos os serviços foram retornados em menos do que 60 segundos, estando a média situada na casa dos 30 segundos (Gráfico 4).

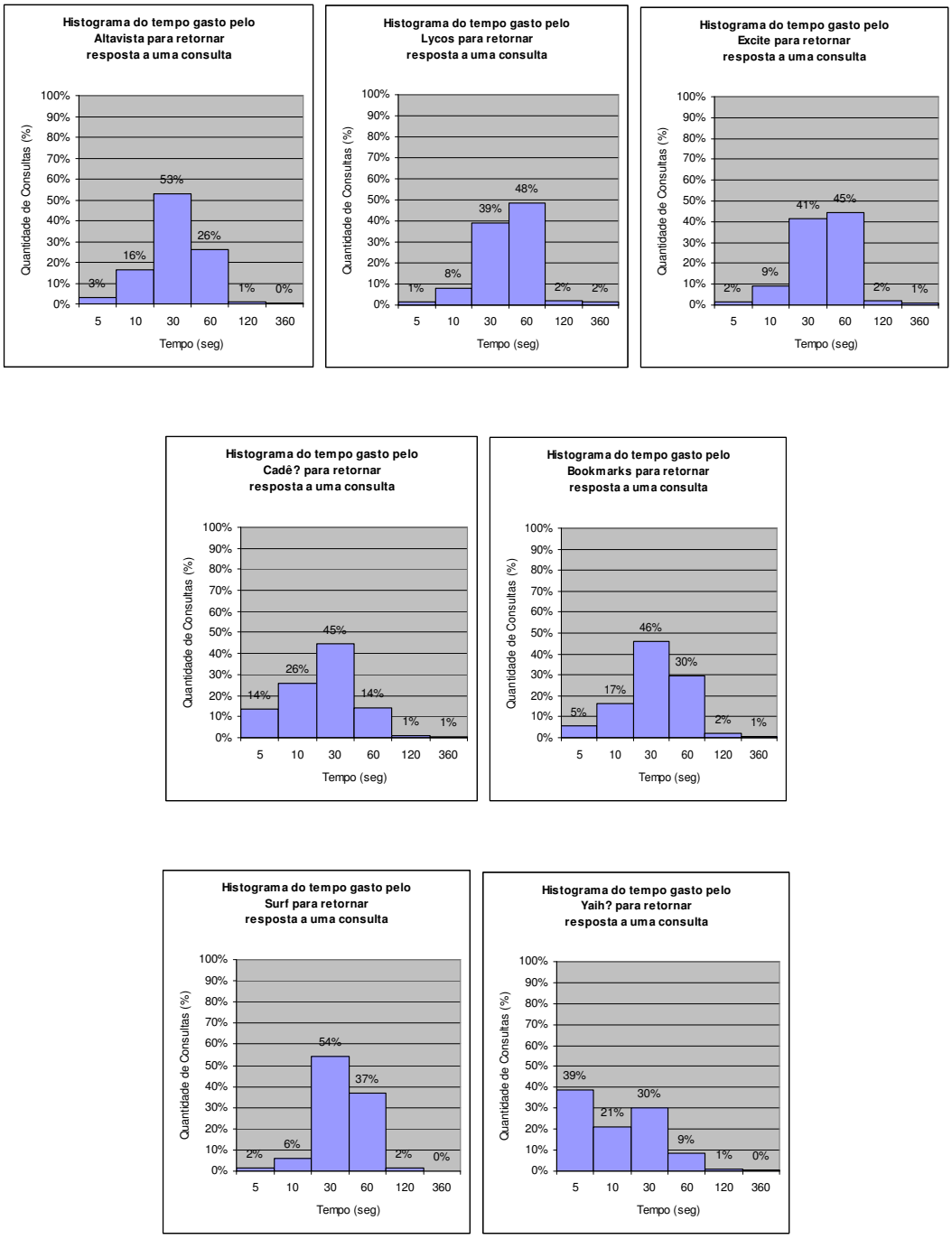


Gráfico 4: Histogramas do tempo gasto pelos serviços de busca para realizar uma consulta do MetaMiner

Amostra referente ao período de 27/Abr/1997 a 04/Ago/1997

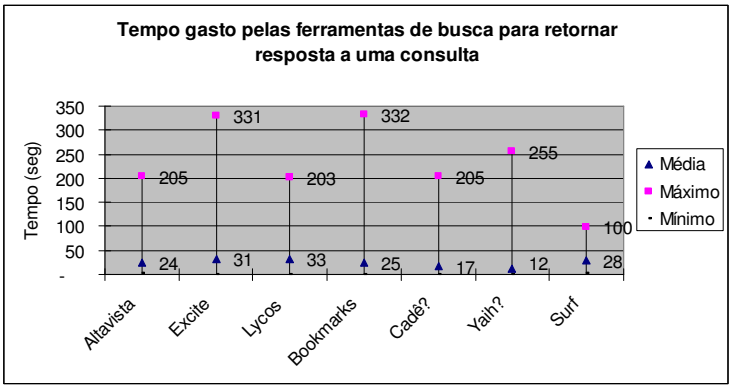


Gráfico 5: Tempo gasto pelos serviços de busca para realizar uma consulta do MetaMiner

Amostra referente ao período de 27/Abr/1997 a 04/Ago/1997

Quantidade de URLs Retornadas por Consulta

O MetaMiner retornou em média 27 URLs por consulta (mínimo de zero e máximo de 133 URLs – Gráfico 6).

No Gráfico 7 o número de URLs recuperadas é detalhada para cada serviço de busca. Observa-se que os motores de busca dificilmente retornam poucos resultados, enviando quase sempre o número máximo de URLs que o MetaMiner coleta[20], ao contrário dos diretórios que retornam menos resultados às consultas realizadas.

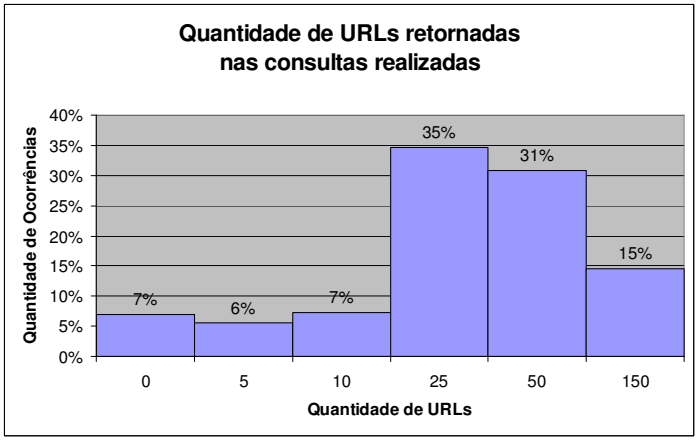


Gráfico 6: Quantidade de URLs retornadas nas consultas realizadas no MetaMiner

[20] O MetaMiner recupera apenas a primeira página de resposta enviada pelas ferramentas de busca. Em geral, a primeira página contém um máximo de 10 resultados.

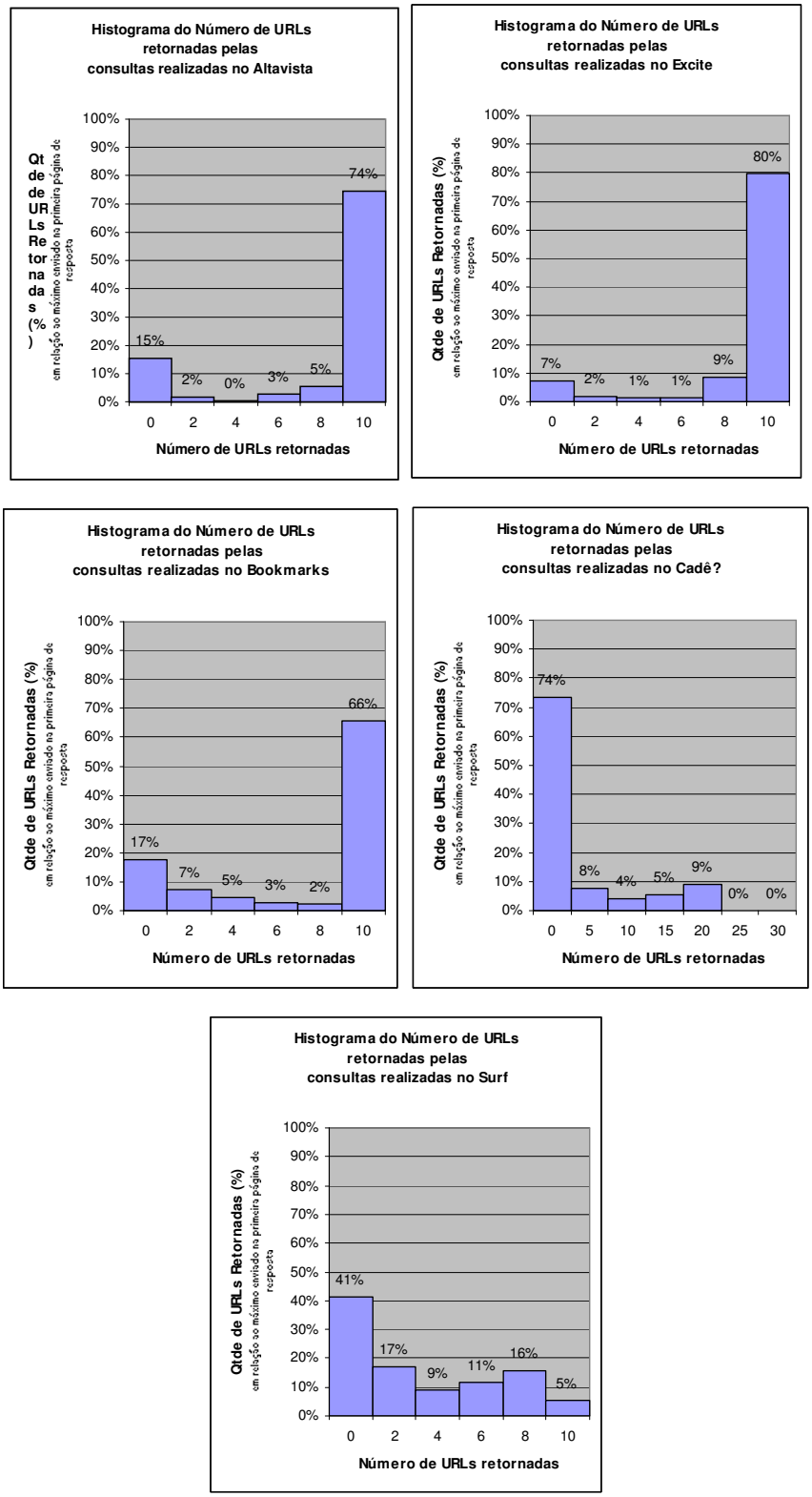


Gráfico 7: Histogramas do número de URLs retornadas em consultas realizadas

Amostra referente ao período de 16/Jul/1997 a 04/Ago/1997

5.1.2. PSSEMiner

O PSSEMiner trabalhou coletando informações a partir do servidor *proxy* do POP-MG (disponível em `cache.pop-mg.rnp.br:3128`) ativo para toda a comunidade do POP-MG.

Afim de não sobrecarregar recursos, foi fixado um horário para que o PSSEMiner fizesse a coleta de páginas no servidor. Isso impediu que todas as páginas novas que passavam pelo *proxy* fossem catalogadas. Entretanto, se fosse comprovada a eficácia e eficiência dessa nova ferramenta de busca usando apenas um horário limitado para colher informações, então a qualidade dessa ferramenta seria muito maior, caso todos os documentos que passassem pelo *proxy* fossem coletados. O horário estabelecido para a execução do PSSEMiner foi entre às 5:30 a.m. e 9:00 a.m., período de baixa utilização local da *Web*, segundo as estatísticas de tráfego do POP-MG.

Tipos de Servidores Web na Internet

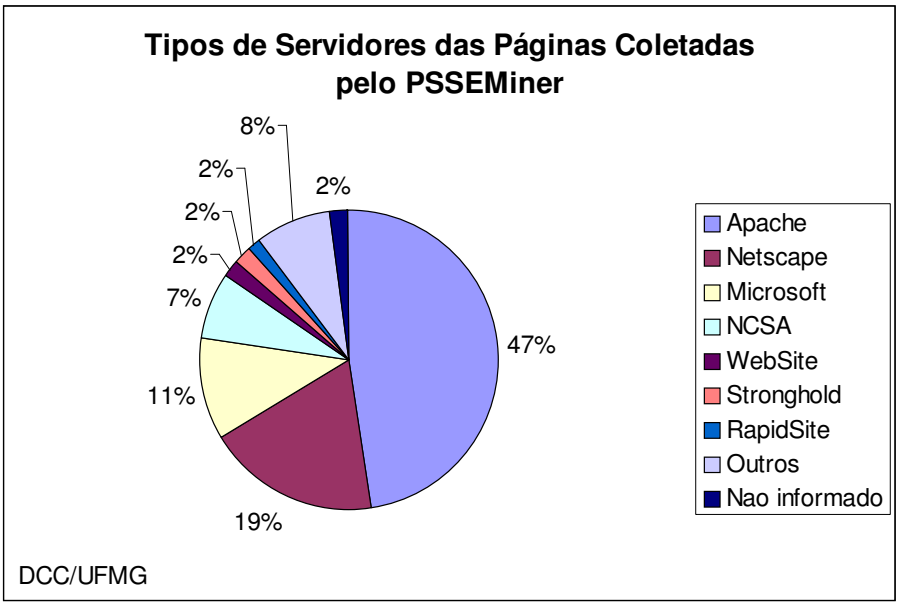


Gráfico 8: Servidores das páginas coletadas pelo PSSEMiner

Apesar da guerra comercial entre a Microsoft e a Netscape, o domínio do mercado de servidores *Web* na Internet continua sendo dos servidores Apache que possui quase 50% do mercado. Além disso, observou-se uma pulverização nesse mercado, sendo encontrados 85 tipos diferentes de servidores *Web,* o que aponta uma tendência para um mercado de *commodities* nesse segmento.

Tamanhos dos Arquivos HTML na Internet

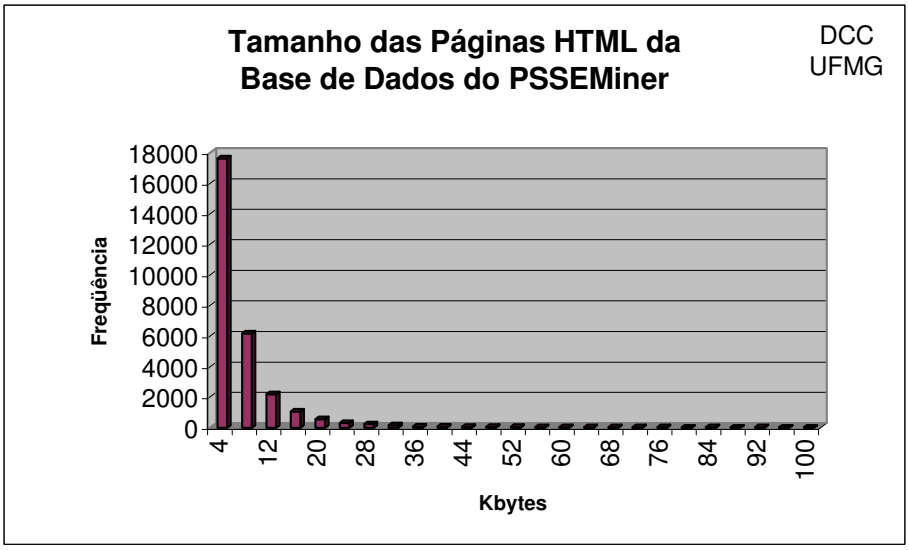


Gráfico 9: Tamanho das páginas HTML coletadas pelo PSSEMiner

O *PSSEMiner* coletou 28667 páginas durante as duas semanas analisadas. Isso representa uma média aproximada de 2050 documentos por dia, ou 10 páginas por minuto (se considerarmos apenas o tempo permitido para a coleta diária – 210 minutos). Deste total de documentos, 61% possuem menos do que 4Kbytes enquanto 96% possuem menos do que 20Kbytes. Menos do que 1% possui mais do que 100Kbytes, isso mostra uma grande concentração de informação escrita em poucos documentos.

Tipos de Arquivo no Proxy POP-MG

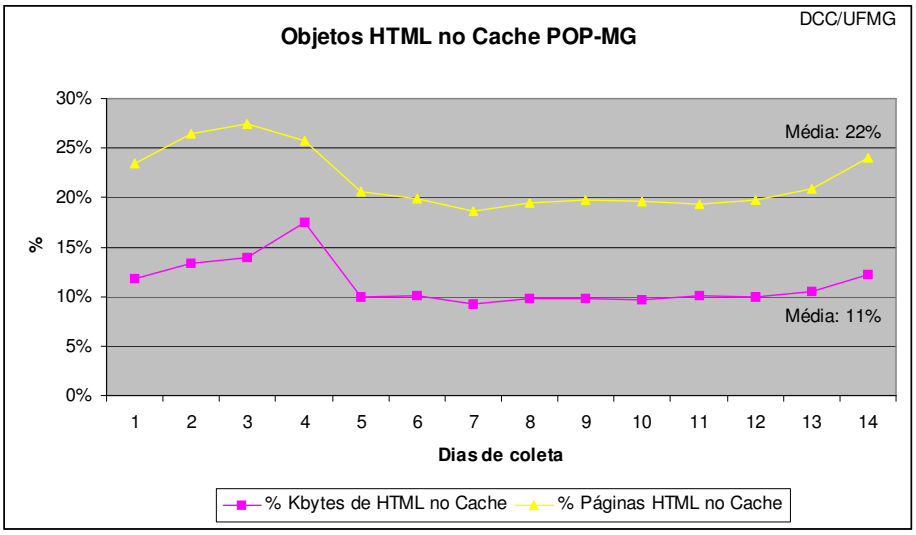


Gráfico 10: Quantidade de objetos HTML no *Proxy* do POP-MG

Em média, apenas 22% das páginas armazenadas no cache do POP-MG são HTML, sendo os outros 78% compostos de objetos multimídia (imagens, sons, vídeo). As páginas HTML representam apenas 11% do total de Kbytes armazenados pelo cache.

Tempos de Coleta (*Time-out*)

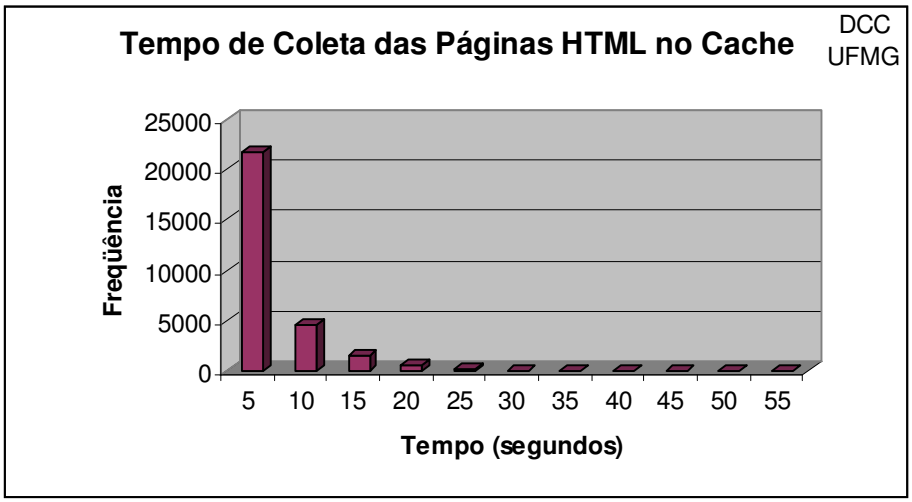


Gráfico 11: Tempo de coleta de páginas pelo PSSEMiner

Do número total de páginas coletadas, 76% levaram menos do que 5 segundos para serem recuperadas, além disso 99% foram coletadas em menos do que 20 segundos. Nas duas semanas de teste, não houve nenhuma busca com tempo esgotado durante a coleta de um documento.

Evolução da Base de Dados

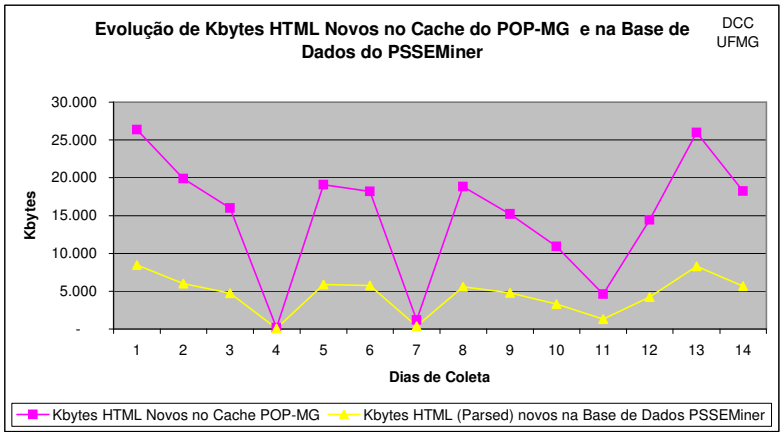


Gráfico 12: Quantidade de dados novos na base de dados do PSSEMiner

Nos dias 4, 7 e 11 o tempo consumido na primeira fase no PSSEMiner foi alto o que impossibilitou a coleta dos objetos presentes no *cache* do POP-MG, isso explica a baixa quantidade de dados recuperada nesses dias. Apenas nos dias 13 e 14 houve tempo suficiente para que todos os documentos novos no *proxy* pudessem ser coletados.

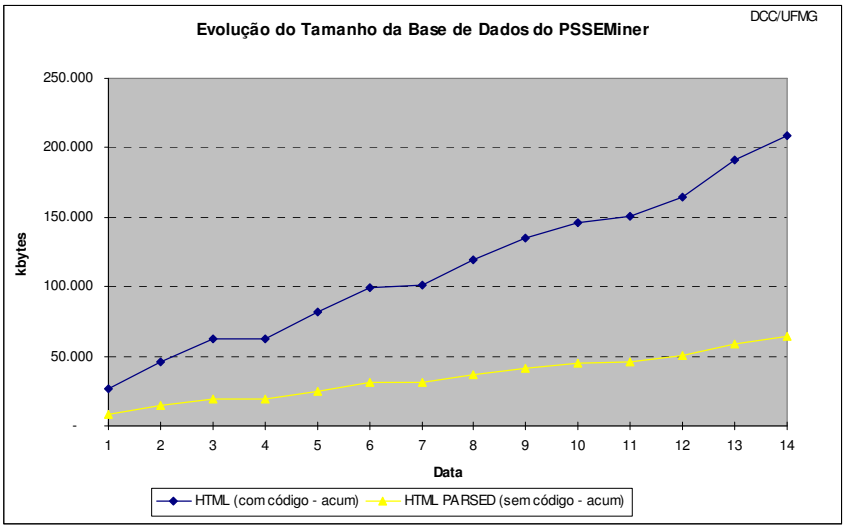


Gráfico 13: Evolução da base de dados do PSSEMiner

O banco de dados do PSSEMiner cresceu a uma taxa média de 5.7 Mbytes por dia enquanto a quantidade de objetos HTML novos no cache do POP-MG manteve-se em uma média diária de 18.4 Mbytes[21]. Essa diferença, 13 Mbytes, é explicada pela remoção do código HTML das páginas antes da inserção na base de dados. Isso mostra que, atualmente, a maior parte do texto de uma página é formada por código HTML e não pela informação nela contida.

[21] Os cálculos dessas médias não levaram em consideração os dias 4, 7 e 11.

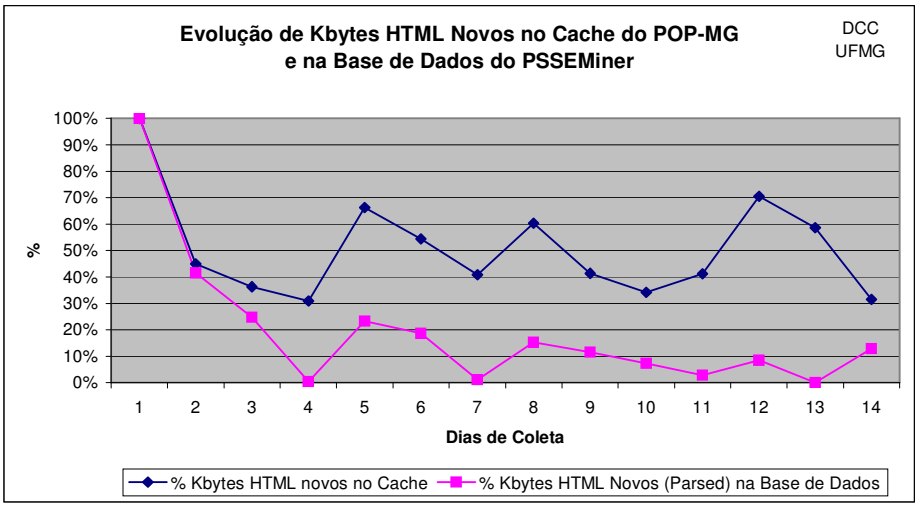


Gráfico 14: Quantidade de kbytes de páginas HTML novas no *Proxy* do POP-MG

A taxa de crescimento da Base de Dados do PSSEMiner mostrou-se proporcional à quantidade de bytes HTML novos no *Cache* do POP-MG, conforme o Gráfico 14. Entretanto, nos dois últimos dias apresentados, nota-se uma possível inversão nessa tendência. Nestes dias, ao contrário dos outros, não houve esgotamento do tempo disponível para a coleta de páginas no *cache* (ou seja, a coleta total ocorreu antes das 9:00 a.m.).

Comparando o PSSEMiner com outras Ferramentas de Busca

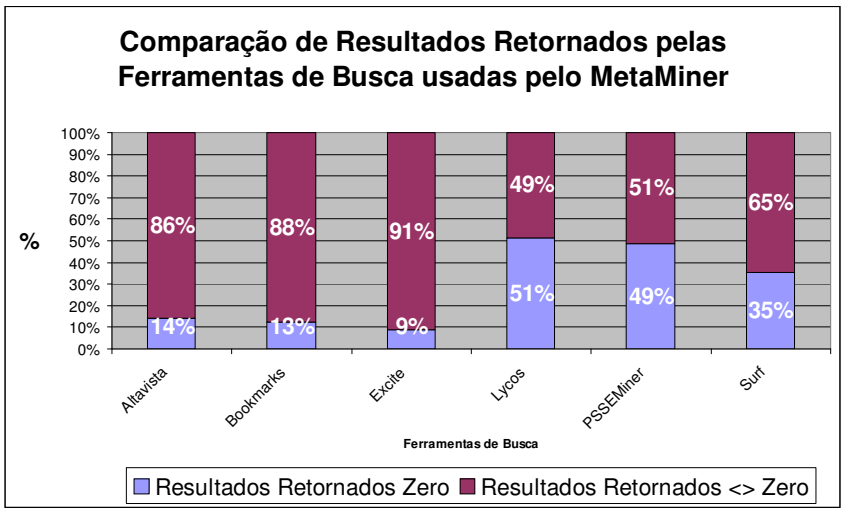


Gráfico 15: Comparação de resultados retornados pelos Serviços de Busca

O Gráfico 15 apresenta o percentual do número de vezes em que as diversas ferramentas de busca utilizadas pelo MetaMiner não retornaram nenhum *link* como resultado (ou seja, não encontraram nenhuma página que satisfizesse a pesquisa do

usuário) juntamente com o percentual de vezes em que essas ferramentas retornaram resultados que atendiam a busca do usuário. Observa-se que os motores de busca (Altavista, Bookmarks e Excite), com exceção do Lycos, atendem aproximadamente 90% das pesquisas dos usuários. Já o diretório Surf, atende apenas 65% das buscas realizadas. O PSSEMiner teve uma taxa de retorno de aproximadamente 50%. Porém, esses dados referem-se ao período total de execução (14 dias), intervalo de tempo em que a base de dados ainda estava sendo elaborada. Quando foram analisadas apenas as pesquisas realizadas nos 7 últimos dias, a taxa dos motores de busca manteve-se em 90% enquanto a taxa do PSSEMiner aumentou para 69%, conforme é apresentado no Gráfico 16.

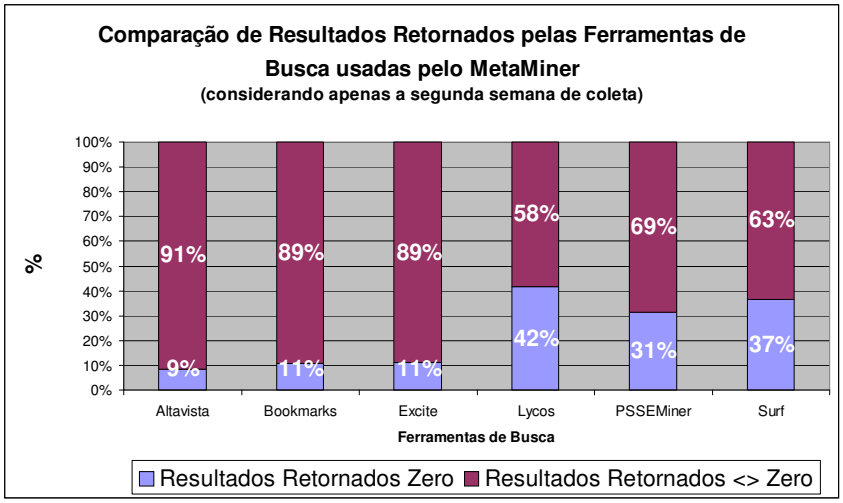


Gráfico 16: Comparação de resultados retornados pelos Serviços de Busca

Considerando apenas a segunda semana de coleta do PSSEMiner

Analisando a variedade da Base de Dados Coletada – TestMiner

O TestMiner, aplicado ao PSSEMiner, procurou verificar a variedade da base de dados coletada, calculando o percentual do número de pesquisas automáticas que obteve resultados não nulos. Dessa forma, estamos avaliando se o PSSEMiner está encontrando resultados para pesquisas de usuários da *Web*. No entanto, através desse tipo de análise não é possível verificar a qualidade dos resultados retornados, isso é, se esses resultados são ou não relevantes.

O TestMiner foi executado após o 14º dia de funcionamento do PSSEMiner. De 567 pesquisas realizadas pelo TestMiner, 466 (82%) retornaram ao menos um *link* como resultado. Isso demonstra uma maior aproximação dos resultados alcançados pelos motores de busca (seção anterior). Além disso, 370 (65%) pesquisas obtiveram 15

links como resultado (esse é o maior número de *links* retornados na primeira página de resposta pelo PSSEMiner, sendo essa página a única recuperada pelo TestMiner).

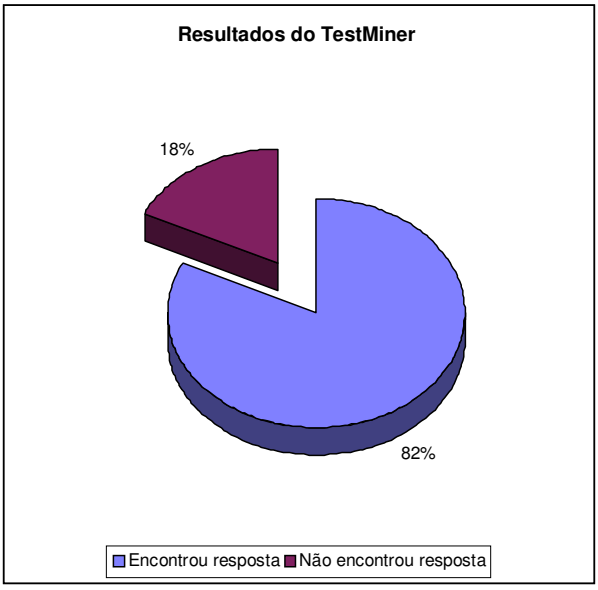


Gráfico 17: Resultados do TestMiner aplicado ao PSSEMiner

5.2. AGENTES DE COMPRA

5.2.1. BookMiner

O período de amostragem dos dados de acessos e consultas no BookMiner, corresponde ao intervalo entre 29 de Março de 1997 a 4 de Agosto de 1997.

Como no MetaMiner, nem todas as livrarias utilizadas, atualmente, pelo BookMiner serão abordadas. Livrarias como Siciliano, iBS e Side, foram inseridas mais tardiamente na API Miner sendo pequeno o número de amostras referentes a esses serviços.

As análises que seguem estão assim divididas: as estatísticas gerais de acesso detalham dados sobre sistemas operacionais e navegadores utilizados pelos usuários do BookMiner. Em seguida, são mostrados dados sobre o número de livros retornados. Finalizando, são apresentadas informações sobre a velocidade de resposta de cada uma das livrarias.

Estatísticas Gerais de Acesso

Como no MetaMiner, o principal navegador utilizado pelo usuário do MetaMiner, no período de análise foi o Netscape Navigator (72% dos acessos – Gráfico 18). Além do Netscape e do Microsoft Internet Explorer foi detectado um número reduzido de usuários com outros navegadores. 105 tipos/versões diferentes de navegadores foram observados.

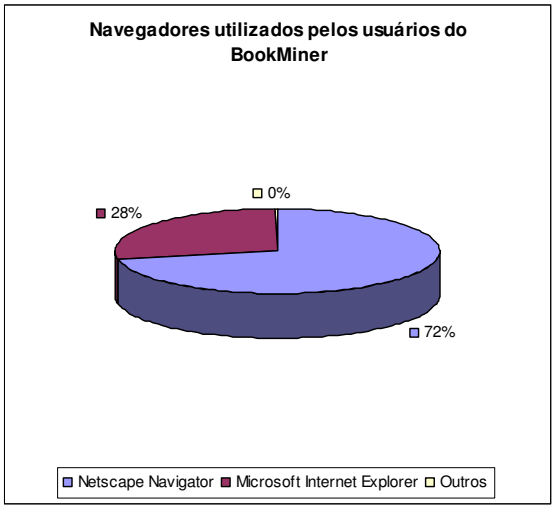


Gráfico 18: Navegadores utilizados pelos usuários do BookMiner

A plataforma Microsoft Windows também foi predominante no BookMiner (Gráfico 19). No entanto, é possível notar um aumento substancial de pessoas que utilizam o SunOS, o que demonstra uma possível diferença de perfis entre os usuários do MetaMiner e BookMiner.

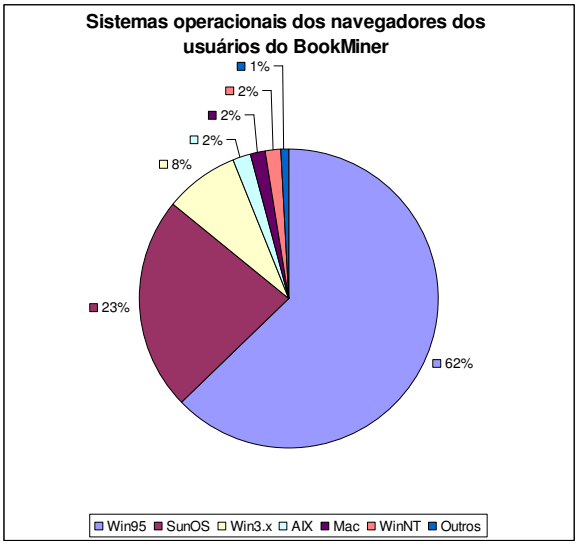


Gráfico 19: Sistemas operacionais dos navegadores dos usuários do BookMiner

Quantidades de Livros Encontrados

33% das consultas realizadas no BookMiner não retornaram resultados Gráfico 20. Dessas, 6.5% são devido a erros de caligrafia ou de digitação. O número médio de livros encontrados foi 25, com um mínimo de 0 e um máximo de 232.

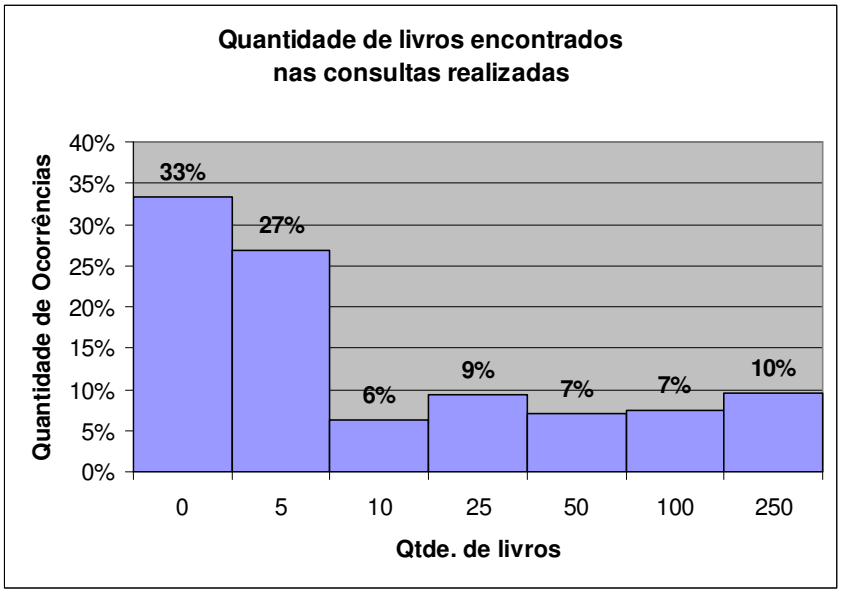


Gráfico 20: Quantidade de livros encontrados nas consultas realizadas no BookMiner

Tempo de Resposta das Livrarias

Ao contrário do MetaMiner, as livrarias estrangeiras retornaram resultados às consultas mais rapidamente do que as livrarias nacionais. No Gráfico 21, é possível observar, também, uma maior proximidade nos tempos de resposta médio de todas as livrarias.

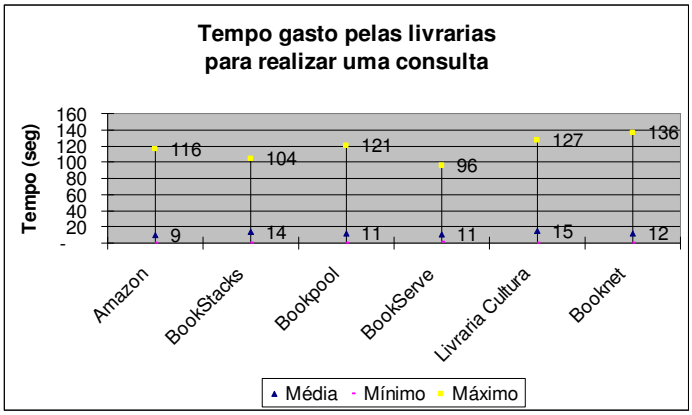
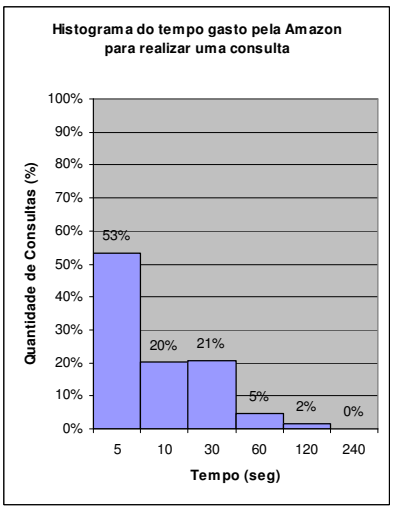
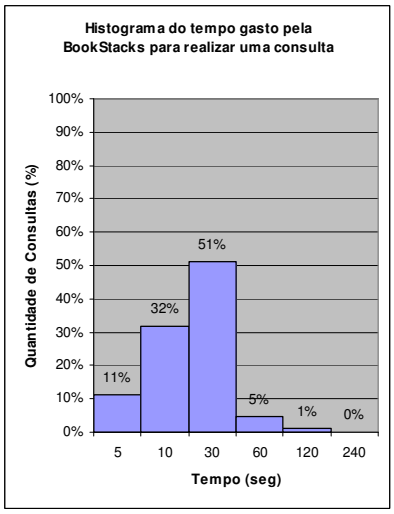


Gráfico 21: Tempo gasto pelas livrarias para realizar uma consulta





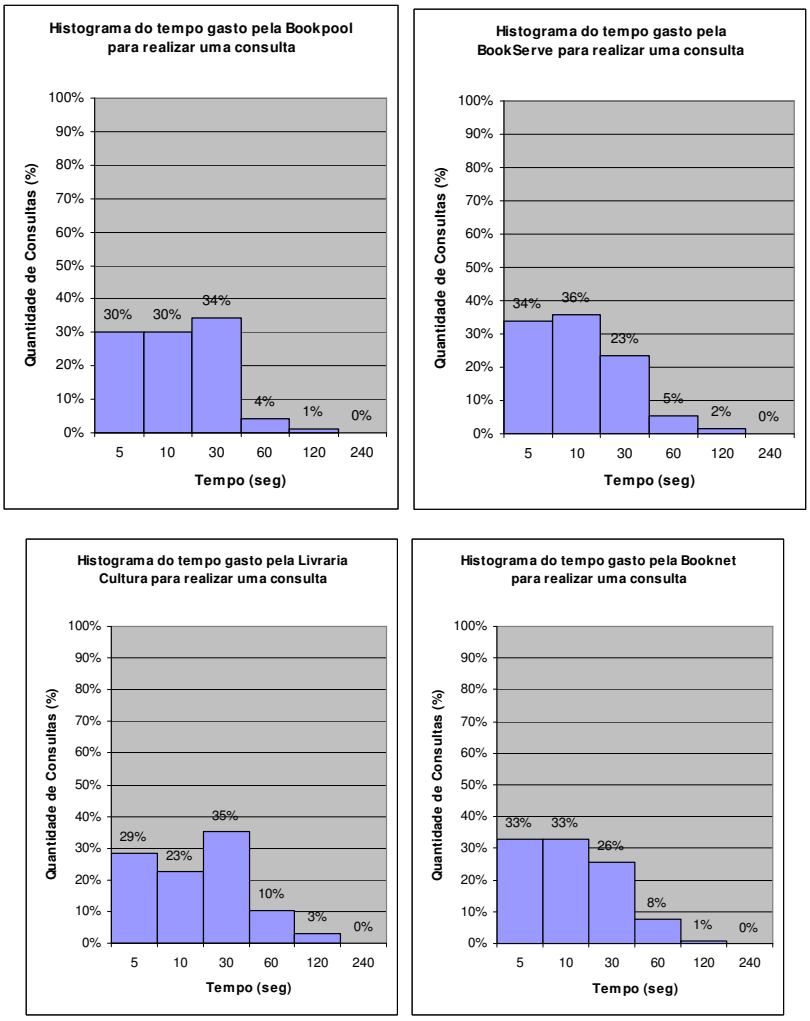


Gráfico 22: Histogramas do tempo gasto por uma livraria para realizar uma consulta

Também, como no MetaMiner, é importante salientar que aqui não está sendo medida a qualidade dos resultados retornados. Apenas a velocidade em que as consultas foram respondidas.

No Gráfico 22, observa-se que, em geral, 98% dos resultados foram retornados em menos do que 60 segundos, estando a média situada na casa dos 12 segundos.

5.3. AGENTES DE NOTÍCIAS

5.3.1. NewsMiner

O período de amostragem dos dados de coleta de reportagens pelo NewsMiner, corresponde ao intervalo entre 7 de Junho de 1997 a 8 de Julho de 1997.

Abaixo são exibidos dados sobre o número e tamanho das reportagens em cada caderno, a divisão do total de matérias por cadernos em cada jornal e também sobre a quantidade de informações recuperadas de cada jornal.

Reportagens por Caderno

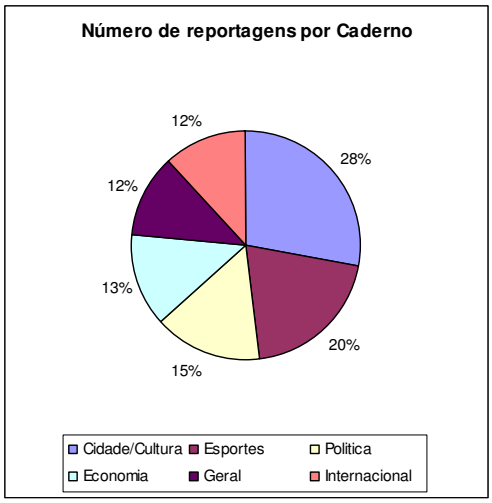


Gráfico 23: Número de reportagens em cada caderno

O Gráfico 23 mostra o número total de reportagens, de todos os jornais, por caderno. É possível observar um número ligeiramente maior destinado a notícias sobre o cotidiano (Cidade/Cultura) e esportes sobre os demais assuntos.

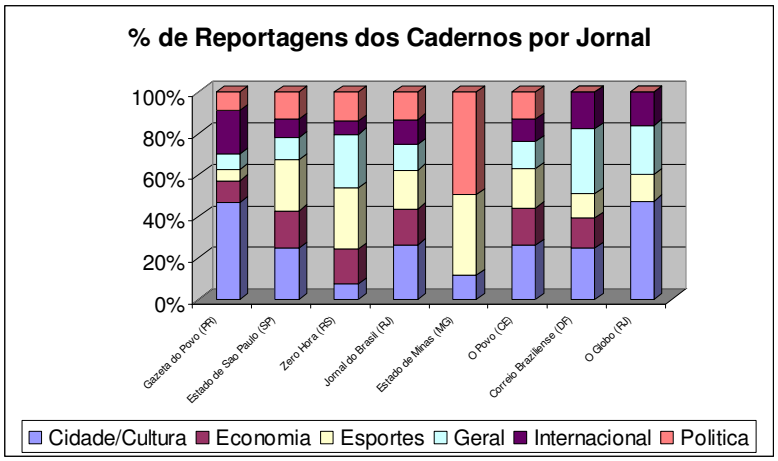


Gráfico 24: Quantidade de reportagens/cadernos por jornal (%)

No Gráfico 24, é possível observar que não existe um padrão definido da quantidade de reportagens por caderno nos diversos jornais analisados. O Estado de Minas, por exemplo, publica na Internet apenas notícias sobre política, esportes e cotidiano (Cidade/Cultura), enquanto, jornais como o Estado de S.Paulo, O Povo, Gazeta do Povo e Jornal do Brasil publicam na Internet notícias sobre assuntos mais variados.

Quantidade e Tamanho das Reportagens Coletadas por Jornal

Em média, foram coletados 759 Kbytes por dia (mínimo de 539 Kbytes e máximo de 1075 Kbytes) de reportagens em todos os jornais escolhidos. O número médio de reportagens foi de 344, sendo o número mínimo 259 e o número máximo 427 (Gráfico 26).

O jornal que mais publicou informações foi o Gazeta do Povo (PR) – média de 202 Kbytes/dia, seguido pelo Estado de S.Paulo (SP) – média de 189 Kbytes/dia (Gráfico 25).

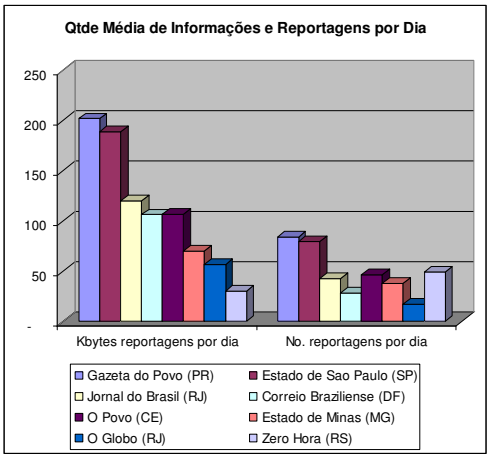


Gráfico 25: Quantidade média diária de kbytes e reportagens por caderno

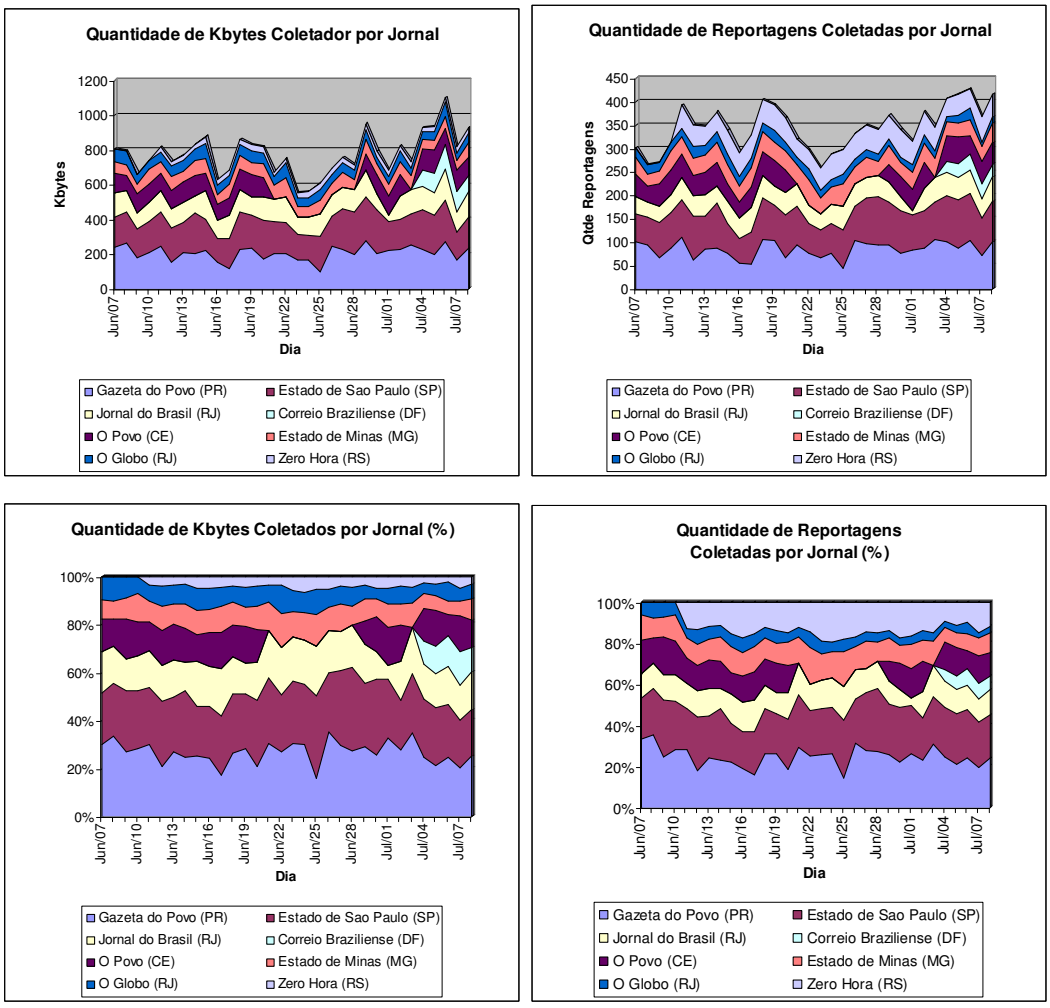


Gráfico 26: Quantidade de Kbytes e reportagens por jornal

6. Conclusão

Neste trabalho foi apresentada *A Família Miner de Agentes para a World-Wide Web* – um conjunto de ferramentas que procuram auxiliar o usuário da Internet a localizar livros, CDs, notícias, software e documentos presentes na *Web*.

A *Família Miner* foi criada a partir da API Miner – uma biblioteca de classes baseada no conceito de busca de informação em paralelo, cuja concepção permite o desenvolvimento de novos agentes que explorem vários tópicos específicos presentes na Internet.

Entre os agentes desenvolvidos, destacam-se: o MetaMiner – a primeira meta ferramenta de busca brasileira; o NewsMiner – meta ferramenta de busca especializada em notícias veiculadas em jornais *online*; o BookMiner – serviço de busca especializado em livros; e o PSSEMiner – uma nova ferramenta de busca baseada em servidores *Proxy*, que em apenas 14 dias de coleta de informações, em horário limitado (210 minutos diários), conseguiu compor uma base de dados variada com resultados quantitativos próximos àqueles obtidos por serviços de busca conhecidos.

Os ganhos observados pela abordagem do PSSEMiner foram:

- Economia de conexão com a Internet. O tráfego de rede externo para elaborar uma base de dados de mais de 50 Mbytes em 14 dias foi nulo;

- Possibilidade de indexar documentos indisponíveis para robôs e de páginas dinâmicas, devido ao fato que os documentos foram coletados a partir de um repositório contendo páginas e documentos gerados através de intervenção humana;

Por outro lado, o PSSEMiner tem uma desvantagem devido a impossibilidade de recuperar documentos que não estejam sendo acessados pelos usuários do *proxy*.

Os agentes da *Família Miner*, além de poderem auxiliar as pessoas a encontrarem informações na Internet, servem para estudar e analisar tópicos diversos sobre recuperação de informação, entre eles: indexação, classificação, filtragem, *relevance feedback* e *clustering*.

Várias tarefas ainda devem ser estudadas e analisadas em trabalhos futuros, tais como:

- A API Miner pode ainda sofrer alterações para melhorar a abstração de suas diversas classes, permitindo a criação de uma super-classe a partir da qual todos os agentes seriam derivados;

- No PSSEMiner, deve-se realizar experimentos para saber qual é a qualidade da base de dados obtida e, também, para verificar a evolução dessa base de dados caso todas as páginas que estivessem passando pelo *proxy* fossem catalogadas. Outro ponto é o estudo da união do conceito de robôs com o de motor de busca baseado em servidores *proxy*, o que poderá gerar um *proxy* "ativo" e uma ferramenta de busca de maior valor;

- A busca das reportagens, no NewsMiner, pode ser refinada agregando a coleta de notícias de última hora, principais manchetes de cada jornal e cadernos específicos, como por exemplo, informática e agropecuária. A interface que permite a criação de perfis para gerar *clippings* eletrônicos deve ser reavaliada para permitir a escolha entre as duas opções já existentes: a primeira onde o usuário digita as palavras-chave de sua preferência, e a segunda onde o usuário irá utilizar o sistema Gama de geração automática de perfis de interesse;

- Para o MetaMiner, algoritmos eficientes de *clustering* devem ser avaliados para possibilitar o agrupamento das URLs obtidas como resposta a partir dos diversos serviços de busca;

- Por fim, diversos novos agentes podem ser desenvolvidos baseados nas classes já criadas e em outras novas que utilizem os conceitos aqui apresentados. Alguns em estudo são: DoctorMiner – especializado na recuperação de artigos técnicos da área médica, CatalogMiner – serviço de para registrar automaticamente uma home-page nas diversas ferramentas de busca da Web, AgroMiner – agente de notícia especializado em agropecuária.

7. Referências Bibliográficas

[AMYRI, 1997] *AMYRI – Ambiente para Manipulación y Recuperación de Información en WWW* in http://www.dcc.ufmg.br/latin/amyri/, 1997

[Berners-Lee, 1994] Berners-Lee, T., "*Request for Comments 1630: Universal Resources Identifiers in WWW*", Network Working Group, June 1994 in ftp://ds.internic.net/rfc/rfc1630.txt

[Berners-Lee et al, 1994] Berners-Lee, T., et al, "*Request for Comments 1738: Uniform Resource Locators (URL)*", Network Working Group, December 1994 in ftp://ds.internic.net/rfc/rfc1738.txt

[Berners-Lee, 1995] Berners-Lee, T., "*Request for Comments 1866: Hypertext Markup Language – 2.0*", Network Working Group, November 1995 in ftp://ds.internic.net/rfc/rfc1866.txt

[Berners-Lee, 1996] Berners-Lee, T., et al, "*Request for Comments 1945: Hypertext Transfer Protocol -- HTTP/1.0*", Network Working Group, May 1996 in ftp://ds.internic.net/rfc/rfc1945.txt

[Borenstein, 1996] Borenstein, N., Freed, N., "*Request for Comments 2046: Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types*", Network Working Group, November 1996

[Braden, 1123] Braden, R., (editor), "*Request for Comments 1123: Requirements for Internet Hosts -- Application and Support*", Network Working Group, October 1989 in ftp://ds.internic.net/rfc/rfc1123.txt

[Calado et al,1997] Calado,P., Pereira, R.M.B., Oliveira, A.L., "*Implementação de Algoritmos para Inferência Automática de Perfis de Interesse*", Relatório Técnico, INESC, Lisboa, 1997

[Chang,1997] Chang, C., and Hsu, C., "*Customizable Multi-Engine Search Tool with Clustering*" in Sixth International World-Wide Web Conference, 1997

[Cheong,1996] Cheong, F., "*Internet Agents – Spiders, Wanderers, Brokers, and Bots*", New Riders, 1996

[Eichmann,1996] Eichmann, D., "*Ethical Web Agents*", Repository Based *Software* Engineering Program, Research Institute for Computer and Information Science – University of Houston, 1996

[Fielding, 1997] Fielding, R., et al, "*Request for Comments 2068: Hypertext Transfer Protocol -- HTTP/1.1*", Network Working Group, January 1997 in ftp://ds.internic.net/rfc/rfc2068.txt

[Flanagan,1997] Flanagan, D., "*JavaScript – The Definitive Guide*", O'Reilly, 1997

[Frakes & Baeza-Yates,1992] Frakes,W., Baeza-Yates, R. (Ed.), "*Information Retrieval – Data Structures and Algorithms*", Prentice-Hall, 1992

[Freed, 1996] Freed, N., Borenstein, N., "*Request for Comments 2045: Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies*", Network Working Group, November 1996

[Freed & Borenstein, 1996] Freed, N., Borenstein, N., *"Request for Comments 2049: Multipurpose Internet Mail Extensions (MIME) Part Five: Conformance Criteria and Examples*", Network Working Group, November 1996.

[Freed et al, 1996] Freed N., et al, "*Request for Comments 2048: Multipurpose Internet Mail Extension (MIME) Part Four: Registration Procedures*", Network Working Group, November 1996

[Graham,1996] Graham, I.S., "*HTML Sourcebook – A Complete Guide to HTML*", Second Edition, John Wiley & Sons, 1996

[InternetWorld,1997] Cartas à Redação, *Internet World Magazine*, Novembro de 1997

[Kelly,1997] Kelly, K., Wolf, G., "*PUSH! Kiss your browser goodbye: The radical future of media beyond the Web*", Wired Magazine, March 1997

[Koster,1995] Koster, M., "*A Standard for Robot Exclusion*", in http://info.webrawler.com/mak/projects/robots/robots.html, 1995

[Koster,1996] Koster, M., "*Guidelines for Robot Writers*", in http://web.nexor.co.uk/mak/doc/robots/guidelines.html, 1996

[Kramer,1997] Kramer, D., "*The Java Platform, a White Paper*" in http://www.javasoft.com/, 1997

[Kristol,1997] Kristol, D.M., Montulli, L., "*HTTP State Management Mechanism (Rev1)*", HTTP Working Group, INTERNET DRAFT, 1997 in http://www.ics.uci.edu/pub/ietf/http/draft-ietf-http-state-man-mec-04.txt

[Lynch,1997] Lynch, C., "*Searching the Internet*" in Scientific American, March 1997

[Manber et al,1995] Manber, U. et al*, "Harvest: A Scalable, Customizable Discovery and Access System*", Technical Report, Department of Computer Science, University of Colorado, 1995

[Mockapetris, 1987] Mockapetris, P., "*Request for Comments 1034: 1987 DOMAIN NAMES – CONCEPTS AND FACILITIES*", Network Working Group, November 1987 in ftp://ds.internic.edu/rfc/rfc1034.txt

[Montulli, 1997] Montulli, L., Kristol, D., "*Request for Comments 2109: HTTP State Management Mechanism*", Network Working Group, February 1997 in ftp://ds.internic.net/rfc/rfc2109.txt

[Moore, 1996] Moore. K., "*Request for Comments 2047: MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text*", Network Working Group, November 1996

[Salton,1983] Salton, G., "*An Introduction to Modern Information Retrieval*" , McGraw-Hill, 1983

[Selberg & Etzioni,1995] Selberg, E., and Etzioni, O., *"Multi-Service Search and Comparison Using the MetaCrawler*" in Proceedings of the Fourth International World Wide *Web* Conference, 1995

[SIAM, 1997] *"Sistemas de Informação em Ambientes de Computação Móvel",* Departamento da Ciência da Computação, ICEx, UFMG, 1997

[Smeaton & Crimmins,1996] Smeaton, A., and Crimmins, F*., "Using a Data Fusion Agent for Searching the WWW*", Dublin City University, Ireland, 1996

[Sullivan,1997] Sullivan, D., "*A Webmaster's Guide to Search Engines*", 1997 in http://searchenginewatch.com/

[Wessels,1995] Wessels, D., "*Intelligent Caching for World-Wide Web*", MSc. Thesis, University of Colorado, 1995

[Williams,1996] Williams, J., "*Bots and Other Internet Beasties*", Sams Net, 1996

8. URLs Relacionadas

[Achei] *search engine* brasileira http://www.achei.net/

[Adim] diretório brasileiro http://adim.goldnet.com.br/search/adimsrch.htm

[Alexa] http://www.alexa.com/

[Altavista] *search engine* http://www.altavista.digital.com/

[Alternex] provedor http://www.alternex.com/

[Amazon] livraria virtual http://www.amazon.com/

[AndConsult] http://www.ac.com/aboutus/tech/cstar/research.html

[AOLNetFind] *search engine* http://www.aol.com/netfind/

[Aonde] diretório brasileiro http://www.aonde.com.br/

[Apple] Apple http://www.apple.com/

[AskJeeves] *meta search engine* http://www.askjeeves.com/

[Bookmarks] *search engine* http://www.bookmarks.com.br/

[BookMiner] meta busca de livros http://www.miner.dcc.ufmg.br/bookminer.html

[Booknet] livraria virtual http://www.booknet.com.br/

[Bookpool] livraria virtual http://www.bookpool.com/

[BookStacks] livraria virtual http://www.bookstacks.com/

[BookServe] livraria virtual http://www.bookserve.com/

[BuscaWeb] diretório brasileiro http://www.busca*Web*.com.br/

[Cade] diretório brasileiro http://www.cade.com.br/

[Campus] editora de livros brasileira http://www.campus.com.br/

[CDMiner] meta busca de cds http://www.miner.dcc.ufmg.br/cdminer.html

[CDNow] loja americana de CDs http://www.cdnow.com.br/

[C|NET] http://www.news.com/Sends search requests

[CorreioBsb] jornal brasileiro http://www.CorreioBsb.com.br/

[DejaNews] *search engine* de *news* (USENET) http://www.dejanews.com/

[Dialdata] provedor http://www.dialdata.com.br/

[Digital] empresa americana http://www.digital.com/

[DirGuide] diretório de *search engines* http://www.directoryguide.com/

[Dogpile] *meta search engine* http://www.dogpile.com/

[DomBrasil] diretório brasileiro http://www.websul.com/db/

[Download] busca de *software* http://www.download.com/

[Ediouro] editora de livros brasileira http://www.ediouro.com.br/
[EstadoMG] jornal brasileiro http://www.estaminas.com.br/
[EstadoSP] jornal brasileiro http://www.estado.com.br/
[EuroSeek] search engine européia http://euroseek.net/
[Editoras] editoras *on-line* http://www.editoras.com.br/
[Exame] revista semanal brasileira com notícias para negócios
http://www.uol.com.br/exame/
[Excite] *search engine* http://www.excite.com/
[ExLibris] livraria brasileira http://www.exlibris.com.br/
[FamiliaMiner] conjunto de agentes de meta busca na *Web*
http://www.miner.dcc.ufmg.br/
[Ferrs] loja brasileira de CDs http://www2.uol.com.br/ferrs/
[Filez] busca de *software* http://www.filez.com/
[FSP] jornal brasileiro http://www.uol.com.br/fsp/
[Galaxy] *search engine* http://lmc.einet.net/
[GztMercantil] jornal brasileiro http://www.GztMercantil.com.br/
[GOD] Global Online Directory – diretório inglês http://www.god.co.uk/
[GuiaWeb] diretório brasileiro http://www.guiaweb.com/
[Hi-Fi] loja brasileira de CDs http://www.hifilaser.com.br/
[HotBot] search engine http://www.hotbot.com/
[Hotfiles] busca de *software* http://www.hotfiles.com/
[iBS] livraria virtual http://www.bookshop.co.uk/
[InfFind] *meta search engine* com *clustering* http://www.inference.com/infind/
[Infoseek] *search engine* híbrida http://www.infoseek.com/
[Inktomi] desenvolvedora de tecnologia para search engine http://www.inktomi.com/
[Intertain] livraria virtual http://www.intertain.com/
[JornalBrasil] jornal brasileiro http://www.jb.com.br/
[JCPE] jornal brasileiro http://www.uol.com.br/jc/
[Jumbo] busca de *software* http://www.jumbo.com/
[L&PM] editora de livros http://www.via-rs.com.br/lpm/
[LfstyFinder] agente para criar perfil de usuários http://lifestyle.cstar.ac.com/lifestyle/
[LivCultura] livraria virtual http://www.livcultura.com.br/
[LookSmart] *search engine* http://www.looksmart.com/
[Lycos] *search engine* híbrida http://www.lycos.com/
[L&PM] editora de livros brasileira http://www.lpm.com.br/
[Magellan] diretório http://images.mckinley.com/
[Mamma] *meta search engine* http://www.mamma.com/

[Matilda] *search engine* australiana http://www.aaa.com.au/

[McGraw-Hill] editora e livraria virtual http://www.mcgrawhill.com/

[MetaCrawler] *meta search engine* http://www.metacrawler.com/

[MetaFind] *meta search engine* http://www.metafind.com/

[MetaMiner] *meta search engine* brasileira
http://www.miner.dcc.ufmg.br/metaminer.html

[Microsoft] http://www.microsoft.com/

[MusicBoul] loja internacional de CDs http://www.musicblvd.com/

[MyShareware] busca de *software* http://www.myshareware.com/

[Netbot] agente de meta busca na *Web* http://www.jango.com/

[Netscape] http://www.netscape.com/

[NewsTracker] *search engine* especializada em notícias http://www.newstracker.com/

[NorthernLight] *search engine* http://www.northernlight.com/ ou
http://www.nlsearch.com/

[OGlobo] jornal brasileiro http://www.oglobo.com.br/

[Ondeir] diretório brasileiro http://www.ondir.com.br/

[OpenText] OpenText Corporation http://www.opentext.com/

[OPopular] jornal brasileiro http://www.opopular.com.br/

[OPovo] jornal brasileiro http://www.opovo.com.br/

[OReilly] editora e livraria virtual http://www.ora.com/

[POP-MG] provedor de backbone brasileiro http://www.pop-mg.rnp.br/

[PlanetMusic] loja de CDs brasileira http://www2.uol.com.br/planet/

[PrenHall] editora e livraria virtual http://www.prenhall.com/

[ProFusion] *meta search engine* http://www.designlab.ukans.edu/profusion/

[PSSEMiner] *search engine* baseada em servidores proxy
http://www.miner.dcc.ufmg.br/psseminer.html

[Quem] busca de pessoas http://www.quem.com.br/

[RadarUOL] *search engine* brasileira http://www.radaruol.com.br/

[Sapo] *search engine* de Portugal http://sapo.ua.pt/

[SavvySearch] *meta search engine* http://savvy.cs.colostate.edu:2000/

[Search.com] *search engine* http://www.search.com/

[SempreUmPapo] cultura http://www.sempreumpapo.com.br/

[Shareware] busca de *software* http://www.shareware.com/

[Siciliano] livraria virtual http://www.siciliano.com.br/

[Side] livraria virtual http://www.side.fr/

[SoftMiner] meta busca de *software* http://www.miner.dcc.ufmg.br/softminer.html

[Softseek] busca de *software* http://www.softseek.com/

[Sun] Sun Microsystems http://www.sun.com/

[Supermail] busca de pessoas http://www.supermail.com.br/

[Surf] *search engine* híbrida http://www.surf.com.br/

[Tucows] busca de *software* http://tucows.soteris.com.br/

[UOL] provedor http://www.uol.com.br/

[Veja] revista semanal brasileira de notícias gerais http://www.uol.com.br/veja/

[Yahoo] diretório http://www.yahoo.com/

[YahooSW] diretório de *search engines* http://www.yahoo.com/Computers_and_Internet/Internet/World_Wide_Web/Searching_the_Web/

[WebCrawler] *search engine* http://www.webcrawler.com/

[WebFerret] *software* para meta procura http://www.ferretsoft.com/netferret/

[WebTaxi] diretório de *search engines* http://www. webtaxi.com/

[Windows95] busca de *software* http://www.windows95.com/

[Wired] revista sobre tendências e tecnologia http://www.wired.com/

[WWWW] *search engine* http://www.cs.colorado.edu/wwww

[ZeroHora] jornal brasileiro http://www.zh.com.br/