Universidade Federal de Pernambuco

Centro de Informática

Pós-graduação em Ciências de Computação

Engenhos de Busca Distribuídos

Uma abordagem visando escalabilidade para *Crawling* e Indexação

Marcelo Rômulo Fernandes

Dissertação de Mestrado

Recife,

2001

Universidade Federal de Pernambuco

Centro de Informática

Pós-graduação em Ciências de Computação

Marcelo Rômulo Fernandes

Engenhos de Busca Distribuídos

Uma abordagem visando escalabilidade para *Crawling* e Indexação

Este trabalho foi apresentado à Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Orientador: Silvio Romero Lemos Meira

Recife,

2001

Dedicado carinhosamente a minha avó Umbelina

Agradecimentos

A Deus, pela vida,

Ao pessoal do Centro de Informática da UFPE, que de uma forma ou de outra contribuíram à minha formação e à conclusão deste trabalho,

Ao professor. Silvio Meira, por sua orientação e apoio para a conclusão deste projeto,

Ao professor Pedro Gonçalves Falcão, por sua orientação e sugestões para este trabalho,

Ao Radix, pela oportunidade de praticar as pesquisas desenvolvidas durante o período do mestrado,

A minha família e amigos, pelo apoio constante durante toda a minha vida,

Aos meus novos amigos, Luciano de Andrade Barbosa, Thiago Santos, João Batista, Roberta Coelho, Flávio Renato Couto Oliveira, Ulisses Montenegro e Ana Karla Alves de Medeiros, pelas observações sugeridas à dissertação,

Aos membros da banca examinadora:

Ana Carolina Salgado

Jorge Henrique Cabral Fernandes

Silvio Lemos Meira

Resumo

A Internet é uma das principais fontes de informação utilizadas no apoio à solução de problemas. Paralelamente a este fato, os Engenhos de Busca surgem como um dos meios mais utilizados para pesquisa de informação nesse ambiente. Observa-se que o tamanho extraordinário, o crescimento exponencial e a elevada taxa de modificação da "*World-Wide-Web*" (*www*) requerem novas abordagens aos problemas de indexação e pesquisa de informação na estrutura dos Engenhos de Busca. Neste trabalho, uma solução distribuída para operação de Engenhos de Busca é apresentada, visando escalabilidade e atualidade. São comentadas arquiteturas distribuídas para Engenhos de Busca. Apresenta-se o Radix, um Engenho de Busca distribuído para indexar e pesquisar informação na *www*, baseado em visões *Web*. Um protótipo é desenvolvido, focalizando a implementação de *crawling* e indexação do Radix distribuído, a fim de validar o ambiente proposto. Um estudo de caso comparativo de desempenho entre Engenhos de Busca centralizados e distribuídos é apresentado, encorajando o uso de técnicas de distribuição para elevar os valores de cobertura e atualidade desses sistemas.

Palavras-chave: Engenho de Busca, *crawling* e indexação, sistemas distribuídos, escalabilidade, visão *Web*.

Abstract

The Internet is one of most popular sources for general-purpose problems solving. Due to that fact, Search Engines arises as the one of the most useful tool concerning the information extraction in the Internet environment. It is likely that the huge size, exponential growth and the high modification rate of World Wide Web pages require a new approach to the crawling, indexing and querying structure in the Search Engine. In this work is presented a distributed solution of a Search Engine, concerning the freshness and scalability. Distributed Search Engines architectures are discussed. It is shown a real Search Engine: Radix.com, a distributed multiagent search engine based on Web-views. A prototype is developed to implement the distributed crawling and indexing of the Radix in order to validate the proposed environment. A comparative case of study of a distributed search engines and a centralized one is presented whose results stimulate the use of such distributed techniques to increase the recall and freshness.

Key words: Search Engine, *crawling*, indexing, distributed systems, scalability, web-views, agents.

Conteúdo

Índice de Figuras

Índice de Tabelas

Índice de Equações

Capítulo 1 Introdução

O tamanho extraordinário e o crescimento exponencial da "*World-Wide Web*" (*www*) requerem novas abordagens aos problemas de indexação e pesquisa de informação em sua estrutura.

O ser humano necessita de informação a todo instante. Com o advento dos computadores, grande parte das informações necessária ao dia-a-dia foi armazenada em meio digital. O surgimento, e posterior crescimento das redes de computadores, possibilitou a difusão dessas informações, partindo das Redes Locais (*LANs – Local Area Networks*) que ligam máquinas de um mesmo laboratório, departamento ou centro, passando pelas redes de computadores maiores (*WANs – Wide Area Networks*), e finalmente atingindo a Internet, capaz de interconectar computadores em todo o mundo.

De fato, a Internet tornou-se um das principais fontes de informação utilizadas pelas pessoas em geral [35]. Dada a grande quantidade de informações existentes nesse meio, saber como encontrar as informações desejadas torna-se uma tarefa árdua. Sistemas de computação são desenvolvidos, para auxiliar na busca por informações, recebendo o nome de Sistemas de Recuperação de Informação. A elaboração desses sistemas tornou-se tão relevante a ponto de tornar-se uma área da Computação conhecida como Recuperação de Informação (*Information Retrieval – IR*).

Entre os sistemas de Recuperação de Informação existentes na Internet, destacam-se os Engenhos de Busca [37]. Esses sistemas são responsáveis por coletar documentos na *Web*[1], manter uma base desses documentos, e utilizar essa base a fim de responder às consultas dos usuários.

1.1 Motivação

Atualmente, mais de 80% dos usuários utilizam Engenhos de Busca ao navegar na Internet [37]. Para atender às consultas dos usuários, um Engenho[2] realiza as seguintes tarefas [53] e [1]:

- *Crawling*: consiste em coletar documentos existentes em vários locais da rede. Devido a diferentes disposições e configurações de rede existentes, uma dificuldade reside na distância da execução do Engenho e a fonte de informação

1 - Nessa dissertação, o termo *Web, World Wlde Web e WWW* são sinônimos e correspondem ao repositório de documentos acessíveis na Internet através dos protocolos de aplicação. O termo Internet correponde à rede mundial de computadores.

2 – Os termos Engenho de Busca, Engenho, EB, Motor de Busca, Sistema de Indexação e Busca e Máquina de Busca serão utilizados como sinônimos.

(o conceito de distância digital é discutido na seção 4.2). Além disso, o Engenho deve interagir com milhares de servidores *web;*

- Indexação: o Engenho armazena as informações extraídas dos documentos provenientes de *crawling* em sua estrutura de armazenamento. A concepção dessa estrutura torna-se um problema, pois ela deve facilitar tanto o armazenamento como a recuperação de informações, atividades normalmente conflitantes;

- Consultas aos usuários: dada consiste na procura por informações contidas na estrutura de armazenamento e na seleção dos melhores documentos para sugerir como resposta. O desafio dessa tarefa é encontrar (procurar, elaborar uma lista, ordenar e apresentar) os melhores documentos em um intervalo de tempo muito pequeno, normalmente abaixo de um segundo, pois os usuários desse tipo de sistema não costumam esperar muito tempo pelas respostas [**38**].

Em todas as tarefas do Engenho de Busca existe a necessidade de processamento de um elevado volume de informação, devido a três fatores: o tamanho da *web* [**36**] e [**33**], a taxa de crescimento da *web* [**36**] e a taxa de modificação da *web* [**28**] e [**5**]. Desse fato decorre que o problema de escalabilidade, refletido em todas as tarefas do EB, as quais são:

- *Crawling:* em determinado período de tempo, além de visitar novos documentos, os já coletados devem ser revisitados, de forma a verificar modificação no conteúdo. Por causa da grande quantidade de documentos, o tempo disponível torna-se insuficiente, causando a visita de apenas parte deles;

- Indexação: como o volume de dados cresce com o tempo, deve-se aumentar a quantidade de dispositivos de armazenamento dos documentos. Isto eleva o tempo de processamento a ser consumido durante o armazenamento e recuperação desses documentos nesses dispositivos;

- Consulta dos usuários: na procura por respostas, o Engenho deve escolher o melhores entre todos os documentos associados à consulta do usuário. Devido ao limite de tempo para se responder às requisições, apenas parte dos documentos possíveis como resposta são verificados.

Novamente, o volume de informações processado por um Engenho implica a necessidade de escalabilidade. Essa característica, embora não funcional, torna-se requisito indispensável ao adequado desempenho dos Engenhos de Busca atuais.

A utilização de técnicas de distribuição surge como uma técnica para solucionar para o problema de pesquisa de informação em sua estrutura, pois possibilita a escalabilidade de sistemas. Além disso, outras características surgem como fruto da construção de sistemas distribuídos: compartilhamento de recursos, abertura de informação, concorrência, paralelismo.

A idéia básica da utilização de técnicas de distribuição para atingir escala é dividir para conquistar. Aplicando essa idéia para a Internet, à medida que a quantidade de informações a ser processada aumenta, o número de Engenhos de Busca também aumentaria.

Atualmente, vários fatores encorajam o desenvolvimento de sistemas distribuídos:

- Processamento distribuído: o poder computacional está distribuído em várias máquinas, requerendo um software que coordene a operação entre elas;
- Tecnologias e metodologias de desenvolvimento: existem várias tecnologias e metodologias que auxiliam a construção de sistemas distribuídos;
- Rede: a Internet oferece uma infra-estrutura de rede, favorecendo um ambiente de interconexão de máquinas.

1.2 Objetivos

O objetivo do trabalho descrito foi desenvolver um ambiente distribuído para Engenhos de Busca, a fim de explorar as vantagens provenientes de Sistemas Distribuídos. Através das técnicas de distribuição, pretende-se prover uma solução adequada ao problema de escalabilidade. Uma arquitetura é elaborada, além do estabelecimento de regras de funcionamento.

Especificamente, este estudo interessa-se nas formas e mecanismos de compartilhamento de recursos (banda passante e processamento, entre outros)

pertinentes a *crawling* e indexação entre Engenhos de Busca. Distribuindo as tarefas citadas, pretende-se aumentar a cobertura e a atualidade dos EBs (ver seção 2.3).

1.3 Metodologia de trabalho

Para atingir os objetivos acima, definiu-se a seguinte metodologia de trabalho: Primeiramente, estudaram-se as áreas de Recuperação de Informação e de Sistemas Distribuídos. Posteriormente, discutiram-se as limitações existentes na operação centralizada dos Engenhos de Busca, focalizando a atenção no problema de escalabilidade; em seguida; em seguida comentaram-se recentes arquiteturas distribuídas para Recuperação de Informação; depois se apresentou o Radix, um sistema distribuído para indexar e pesquisar informação na *www*. Para coordenar o funcionamento do sistema distribuído, apresentou-se e utilizou-se o formalismo de visões *Web*. Os esforços na implementação concentraram-se na distribuição das tarefas de *crawler* e indexação do Radix. A partir daí, realizou-se um experimento e instanciou-se um estudo de caso real, para validar as idéias contidas no ambiente de distribuição proposto.

1.4 Estrutura geral da dissertação

Este documento compõe-se de cinco capítulos, incluindo esta introdução.

O capítulo 2 apresenta um estudo sobre Engenhos de Busca. Esse estudo é composto de uma descrição sobre Recuperação de Informação e de medidas para avaliação de Sistemas de Recuperação de Informação. Além disso, o mecanismo de funcionamento de um Engenho de Busca típico é mostrado, salientando as suas limitações.

O capítulo 3 descreve uma solução distribuída para o funcionamento de Engenhos de Busca. Inicialmente, uma descrição sobre Sistemas Distribuídos é realizada. Depois, soluções distribuídas aplicadas a Engenhos de Busca são apresentadas e analisadas. Finalmente, apresenta-se o ambiente de distribuição proposto na dissertação, incluindo o formalismo *visão Web*, os módulos existentes na arquitetura distribuída e o funcionamento do ambiente.

O estudo de caso real é apresentado no capítulo 4. Nele, descreve-se o Agente de Distribuição (AD), a arquitetura de *crawling* e indexação distribuída. Para finalizar esse capítulo, descrevem-se as tecnologias utilizadas no desenvolvimento do protótipo.

O capítulo 5 apresenta as conclusões retiradas da elaboração da pesquisa, incluindo as limitações existentes e trabalhos futuros.

Capítulo 2 Engenhos de Busca

Engenhos de Busca são sistemas de RI para a *Web* cuja base de informação é atualizada automaticamente, sendo utilizados por mais de 80% dos usuários da Internet.

2.1 Introdução

Um EB é um dos sistemas de Recuperação de Informação (SRI ou sistemas de RI) mais utilizados para busca de informação na *World Wide Web* (resumidamente *www* ou simplesmente *Web*) [37]. Os módulos de funcionamento dos Engenhos de Busca são bem determinados: *crawling* (coleta por informações), indexação (armazenamento das informações em uma Base de Índices), e busca (procura por informações nessa Base de Índices). A tarefa dos projetistas e implementadores desses sistemas é otimizar cada um desses módulos.

Neste capítulo, explicam-se os aspectos relacionados a um sistema de indexação e busca na *Web*. O capítulo organiza-se da seguinte maneira: a Seção 2.2 apresenta uma breve introdução sobre Recuperação de Informação; a Seção 2.3 apresenta medidas para se avaliar o desempenho de sistemas IR; a Seção 2.4 descreve um Engenho de Busca típico, mostrando sua arquitetura básica e suas etapas de funcionamento; e a seção 2.5 apresenta conclusões sobre o capítulo.

2.2 Recuperação de Informação

As pessoas dependem de informação nas suas atividades diárias. Existe a necessidade de informação em qualquer atividade, desde atravessar uma rua até diagnosticar complexos problemas de saúde. Algumas são facilmente disponíveis, outras precisam de busca extensiva e a união de diferentes fontes de informação.

A construção de sistemas de computação para ajudar a busca de informação é uma tarefa especializada, necessitando conhecimento sobre como ela está organizada e como as pessoas a procuram [52].

Um SRI trata da representação, armazenamento, organização e acesso a itens de informação. A representação e organização desses itens tem por objetivo permitir ao usuário o fácil acesso à informação na qual ele está interessado [51]. A necessidade do usuário deve ser transformada em um formato que pode ser processado pelo sistema. Na busca por informações essencialmente textuais, normalmente, essa necessidade é

representada por uma expressão contendo um conjunto de palavras-chave. Nesse contexto, a recuperação de informação, em um sistema de Recuperação de Informação, consiste em determinar quais dos documentos presentes em uma coleção contêm as palavras-chave utilizadas na consulta. De modo geral, a finalidade de um sistema IR é recuperar **informação** sobre as requisições do usuário.

Sistemas automáticos para Recuperação de Informação foram utilizados, originalmente, para gerenciar uma grande quantidade de literatura científica desenvolvida desde 1940. Muitas universidades, empresas e bibliotecas públicas utilizam sistemas de Recuperação de Informação a fim de prover acesso a livros, jornais, publicações e outros documentos. Sistemas IR comerciais oferecem bases de dados capazes de armazenar milhões de documentos [6].

Naquela época, os documentos eram indexados utilizando algumas porções representativas dos mesmos como títulos, palavras-chave e resumos, por causa dos escassos recursos computacionais e de armazenamento da época. A maior parte do processo de indexação era feita manualmente, aproveitando a capacidade de análise das pessoas para procurar e selecionar aquelas informações mais representativas [6]. Vale salientar que o índice de um documento significa o conjunto das informações relevantes a esse documento. Por exemplo, o índice de uma página *Web* consiste do título, do endereço, da lista das palavras, a lista de *links,* entre outros.

Os avanços nas tecnologias de processamento e armazenamento de informação, a redução dos custos associados, assim como o surgimento de linguagens, técnicas e ferramentas de programação mais versáteis, permitiram um melhor desenvolvimento de sistemas de indexação: hoje é possível armazenar e localizar mais rapidamente todos os dados relevantes aos documentos; além disso, as interfaces de consulta possuem um nível mais alto de abstração para interagir com os usuários [6].

2.3 Medidas para avaliação de desempenho de SRI

As medidas mais comuns de performance (desempenho) de um sistema são:

- Tempo: um sistema tem um valor de tempo de resposta "t" limitante para suas computações. Quanto maior for a capacidade desse sistema em diminuir esse tempo de resposta "t", melhor será considerado. Por exemplo, segundo Jakob Nielsen [**38**], **0.1 segundo** é o tempo de resposta limite que o usuário percebe um sistema reagindo instantaneamente a suas solicitações.

- Espaço: um sistema deve ocupar uma quantidade de espaço de armazenamento "s" aceitável para suas computações. Quanto maior for a capacidade desse sistema em diminuir o espaço "s" utilizado, melhor será considerado. Normalmente, existe um *tradeoff* entre a complexidade do tempo e a complexidade do espaço [**51**].

Além de tempo e espaço, outras medidas tornaram-se padrão na área de Recuperação de Informação. Antes de serem apresentadas, torna-se necessária a seguinte consideração:

Seja R um conjunto de documentos considerados relevantes em uma coleção; seja $|R|$ o número de documentos em R; seja A o conjunto de documentos da coleção que foram coletados por um sistema de informação especializado em processar documentos desse conjunto; seja $|Ra|$ o número de documentos da interseção dos conjuntos R e A. Esses conjuntos são ilustrados na Figura 1 [**6**].

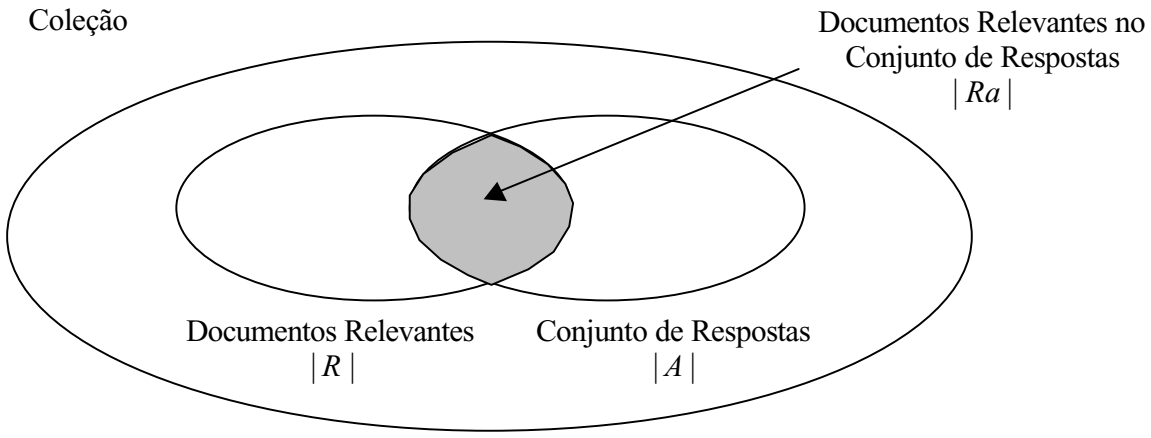

Figura 1 Conjuntos utilizados para definição de precisão e cobertura

Dadas a consideração acima, definem-se as medidas de cobertura e precisão:

- *Recall* (Cobertura): é a fração dos documentos relevantes (o conjunto R) que foram recuperados, ou seja,

$$Cobertura = \frac{|Ra|}{|R|}$$

Equação 1 Fórmula para cálculo de cobertura

O valor de cobertura de um SRI é proporcional à quantidade de documentos relevantes que são coletados e também à quantidade de documentos que ele consegue recuperar.

- *Precision* (Precisão): é a fração dos documentos recuperados (o conjunto A) que é relevante, ou seja,

$$Pr ecisão = \frac{|Ra|}{|A|}$$

Equação 2 Fórmula para cálculo de precisão

O valor da precisão de um SRI na *Web* é proporcional à quantidade de documentos considerados relevantes existentes na base do SI. Esse valor depende de dois fatores: do algoritmo de classificação dos documentos do SI, pois esse algoritmo determina quais são os documentos que serão apresentados como respostas às consultas dos usuários; e da estratégia de armazenamento e da construção dos índices das páginas.

Outras medidas interessantes para se avaliar um SRI são:

- *Freshness* (Atualidade): seja S = $\{e_1, \ldots, e_N\}$ uma base de documentos local com *N* elementos. Idealmente, todos os *N* elementos deveriam estar atualizados, entretanto, somente *M (< N)* elementos estarão atualizados em um determinado intervalo de tempo (atualizado significa que os valores armazenados são iguais aos valores encontrados fora da base de documentos). A atualidade de S é o tempo t dado por F (S; t) = M / N. Em outras palavras, a atualidade é a fração da base de documentos que está atualizada. Por exemplo, F (S; t) será igual a 1 (um) se todos os elementos da base estiverem atualizados, e F (S; t) será igual a 0 (zero) se todos os elementos da base estiverem desatualizados. Define-se a atualidade de uma base de documentos *S* em um tempo *t* como:

$$F(S;t) = \frac{1}{N}\sum_{i=1}^{N} F(e_i;t)$$

Equação 3 Fórmula para cálculo de atualidade

É interessante ressaltar a dificuldade de medição exata da atualidade. Na prática, estima-se valores dado uma amostra de como as informações são alteradas no mundo real [**27**].

- *Age* (Idade): a idade de um elemento de uma base e_i em um tempo t é:

  $A(e_i; t) = 0$, se e_i está atualizado no tempo t ou

  $A(e_i; t) = t - (\text{tempo de modificação de } e_i)$

A idade de uma base de dados S é dada por:

$$A(S;t) = \frac{1}{N}\sum_{i=1}^{N} A(e_i;t)$$

Equação 4 Fórmula para cálculo de idade

A idade de uma base de dados S é definida como a média das idades de cada elemento da base.

2.4 O Engenho de Busca

A Internet é considerada por muitos como um dos mais importantes e revolucionários desenvolvimentos da história da humanidade. Pela primeira vez no mundo um cidadão comum ou uma pequena empresa pode (facilmente e a um custo muito baixo) não só ter acesso a informações localizadas nos mais distantes pontos do globo como também criar, gerenciar e distribuir informações em larga escala, no âmbito mundial, algo que somente uma grande organização poderia fazer usando os meios de comunicação convencionais.

Com a Internet uma pessoa qualquer pode, de sua própria casa, oferecer e utilizar um serviço de informação, a partir de um microcomputador. A Internet transformou-se em

uma das principais fontes de informação para a solução de dúvidas ou problemas em geral [**35**]. Além disso, quando um usuário deseja procurar um serviço ou solucionar algum problema, mas não sabe a sua localização na Internet, ele pode utilizar um Engenho de Busca. De fato, os Engenhos de Busca tornaram-se um dos principais mecanismos de procura por informação nesse ambiente.

A maioria dos sistemas de indexação e procura da *Web* é baseada em uma das seguintes estratégias:

- Diretórios: hierarquias de índices de assuntos construídas e mantidas manualmente. Essa estratégia requer aprovação humana para inserir índices, assim como para editar a hierarquia de índices. Embora existam bastante sistemas que adotem essa estratégia, o tamanho [33] , o crescimento [36] e taxa de modificação [28] e [5] da *Web* tendem a aumentar a demanda por trabalho humano para manter um índice completo e atualizado. Este tipo de índice é difícil de manter atualmente, e a situação tende a se tornar inviável. Exemplos de diretórios de busca que utilizam essa estratégia são *Yahoo*! [59] e *Cade*? [56];

- Engenhos de Busca: bases de índices automaticamente atualizadas por programas que percorrem e colecionam páginas da *Web*, os conhecidos Robôs *Web*. *Altavista* [58] utiliza essa estratégia. Não há intervenção humana durante todo o processo de funcionamento do sistema, exceto na etapa de configuração. Para resolver o problema de escalabilidade de crescimento da *Web*, esta maneira é mais atrativa que atualização manual de listas.

Alguns Engenhos de Busca mesclam as duas estratégias, como o Radix [**17**] e o *Google* [**57**], visando um melhor serviço de busca na *Web*, utilizam as vantagens de cada uma: no primeiro caso, a qualidade dos *links* contidos no índice, já que são selecionados por humanos; no segundo caso, a abrangência atingida pela busca em profundidade dos Robôs.

2.4.1 Processos e componentes do Engenho de Busca

De forma geral, os Engenhos de Busca possuem etapas e componentes básicos de funcionamento. Um Engenho de Busca realiza três tarefas básicas:

- Coleta de Documentos (*crawling*): nessa tarefa, surgem o Robô *Web*, também chamado de *spider* ou *crawler*. Os Robôs são programas que operam da seguinte forma: recebem referências (*links*) para documentos; realizam o "*download*" desses documentos; processam-nos localmente, a fim de extrair índices. Eles recursivamente seguem *links* de hipertexto encontrados no processamento, para executar uma exaustiva coleta da *Web* (ver Figura 2).

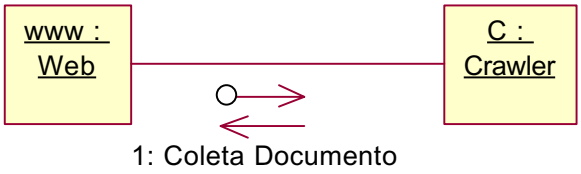

Figura 2 Diagrama de Colaboração demonstrando a coleta de documentos um Engenho de Busca

- Armazenamento de Índices (indexação): aqui surge o componente Indexador, capaz de armazenar os índices dos documentos coletados pelos Robôs. Normalmente, os índices são armazenados em um SGBD utilizando um esquema de banco de dados. O Indexador é o componente responsável pela manutenção da Base de Índice (ver Figura 3).

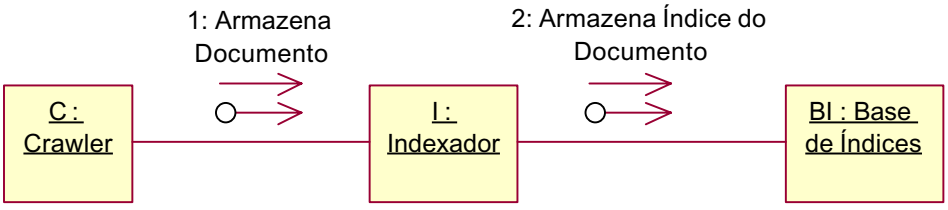

Figura 3 Diagramas de Colaboração demonstrando a indexação.

- Procura na Base de Índices (busca): nesse processo, surgem os usuários do sistema a realizar consultas. A interação do usuário é realizada através de uma interface, que usualmente possibilita consultas baseadas em palavras ou expressões. As consultas são processadas pelo Sistema de Busca, que utiliza a

Base de Índices do sistema, a fim de encontrar as respostas para as consultas (ver Figura 4).

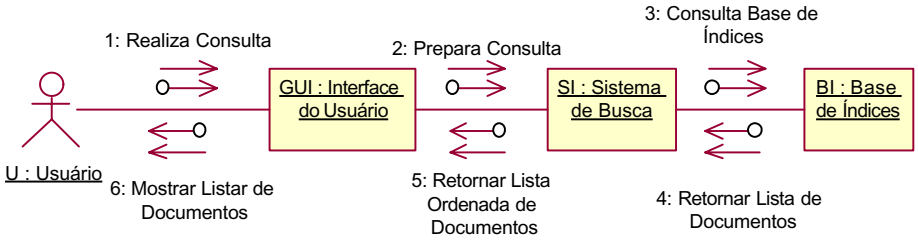

Figura 4 Diagrama de Colaboração demonstrando uma busca por informações.

2.4.2 Limitações dos Engenhos de Busca

Em 1994, um dos primeiros Engenhos de Busca, o *World Wide Web Worm (WWWW),* possuía uma base com 110 mil documentos. Em novembro de 1997, o maior Engenho de Busca existente atingiu 2 milhões quando existiam 100 milhões de documentos [36]. A partir do ano 2000, alguns Engenhos de Busca atingiram a marca de mais de 1 bilhão de documentos [57]. Paralelamente a esse fato, a quantidade de consultas realizadas nos Engenhos de Busca também cresceu. Em março e abril de 1997, o *WWWW* recebia em média 1.500 consultas ao dia. Em novembro de 1997, *Altavista* processava 20 milhões de consultas ao dia [36]. Dados esses números, surge a necessidade de **escalabilidade** na tecnologia dos Engenhos de Busca.

Atualmente, poucos Engenhos de Busca afirmam que cobrem a maioria do conteúdo da *Web* [34], sugerindo que a **cobertura** dos Engenhos individuais é cada vez mais reduzida.

Manter uma base de dados em um Engenho de Busca centralizado para indexar a toda *Web* requer elevado poder de processamento e espaço de armazenamento. Tratando de mais de um bilhão de documentos [33] e supondo que esses documentos tenham, em média, 20 Kb [ver tamanho médio das páginas utilizadas no Apêndice B], necessitaria de cerca de 20 terabytes de espaço de armazenamento.

A freqüência de atualização dos documentos na *Web* varia bastante. Alguns são imutáveis, outros possuem atualizações mensais ou com período ainda maior. Neste

caso, a **atualidade** de um Engenho de Busca não é um problema. Entretanto, a atualidade das páginas coletadas em um Engenho de Busca torna-se importante na medida em que ele indexa páginas cuja freqüência de atualização é extremamente elevada. Isso ocorre principalmente em *sites* de conteúdo, tipicamente jornais [60]. Na *Web*, existem vários desses exemplos, tornando a tarefa de manter atualizada a Base de Índices de um SI algo extremamente difícil [36] , [28] e [5].

Outro fator que limita a operação de um Engenho é a topologia da rede onde a *Web* está inserida. Desse fato decorre que os Robôs podem estar situados muito distantes das informações consideradas mais importantes. (A noção de distância é descrita com mais detalhes na seção 3.2.1)

2.5 Conclusões do capítulo

Neste capítulo foram apresentados os conceitos relacionados a sistemas de Recuperação de Informação e, mais especificamente, sobre um dos sistemas de Recuperação de Informação mais utilizados na *Web*, o Engenho de Busca.

IR trata da representação, armazenamento, organização e acesso a itens de informação. Os avanços nas tecnologias de processamento e armazenamento de informação, a redução dos custos associados, assim como o surgimento de linguagens e técnicas de programação mais versáteis, permitem um melhor desenvolvimento de sistemas IR.

Assim como a *Internet* ganha importância no dia-a-dia da vida das pessoas, os Engenhos de Busca surgem como um dos principais sistemas de acesso de informação. O tamanho extraordinário e a alta taxa de crescimento e modificação da *Web* comprometem o funcionamento adequado desses sistemas. Devido ao problema de escalabilidade, a cobertura e a atualidade do EB fica comprometida, desafiando os projetistas desses sistemas.

Capítulo 3 Engenhos de Busca Distribuídos

Um único sistema de indexação e busca não resolve a contento o problema da indexação e procura da *Web* com relação a sua taxa de crescimento e ao seu tamanho atual. A fim de atingir níveis aceitáveis de escalabilidade, soluções distribuídas surgem no contexto da *Web*, um ambiente naturalmente distribuído.

3.1 Introdução

A aplicação de técnicas de distribuição a sistemas computacionais vem se tornando uma requisição não-funcional essencial. Sistemas Distribuídos mostram algumas vantagens sobre sistemas centralizados: escalabilidade, tolerância a falhas e compartilhamento de recursos.

Neste capítulo, explica-se a solução de distribuição adotada nos Engenhos de Busca. O capítulo está disposto da seguinte forma: a seção 3.2 apresenta um estudo sobre Sistemas Distribuídos; a seção 3.3 narra soluções distribuídas para o problema de indexação e busca; a seção 3.4 apresenta o Radix, um EB capaz de participar de um ambiente de distribuição; e a seção 3.5 apresenta conclusões sobre o capítulo.

3.2 Sistemas Distribuídos

Segundo Coulouris [21], define-se Sistemas Distribuídos (SD) como uma coleção de computadores autônomos integrados em rede, utilizando um software desenvolvido para produzir facilidades de integração computacional.

Seguindo essa definição, existem três exigências para se definir um sistema distribuído:

- Um SD é composto por uma coleção de computadores autônomos, ou seja, um computador existente em um SD continuaria sua operação caso fosse excluído do ambiente distribuído;

- Os computadores em um ambiente distribuído são integrados em rede, ou seja, a rede é o meio que permite a comunicação entre os computadores. A ausência de uma estrutura de rede de computadores inviabiliza um sistema distribuído;

- Os computadores utilizam um software desenvolvido para produzir facilidades de integração computacional. Esse software de distribuição define como a rede será utilizada para prover comunicação, ou seja, qual o protocolo a ser utilizado pelos computadores integrantes do ambiente, e quais recursos computacionais podem ser compartilhados.

O uso de sistemas distribuídos surgiu na década de 1970. Isso ocorreu devido a vários fatores que influenciaram o surgimento de um ambiente cooperativo: o uso de microcomputadores como estações gráficas de trabalho, o crescimento no desenvolvimento de software, o surgimento de redes locais de alta velocidade. Um dos primeiros sistemas distribuídos desenvolvidos foi o servidor de arquivos distribuídos da Xerox em 1977, desenvolvido no *Palo Alto Research Center* [21].

3.2.1 Características dos sistemas distribuídos

Embora um sistema distribuído seja composto por uma coleção de computadores autônomos, a interligação deles se justifica pelas várias características atingidas em um ambiente distribuído, explicando, assim, o desenvolvimento desses sistemas:

- Compartilhamento de recursos (*Resource sharing)*: define o conjunto de objetos que podem ser compartilhados em um ambiente colaborativo. Esses objetos variam de componentes de *hardware* tais como discos rígidos, impressoras; e estruturas de *software* como arquivos, janelas, bases de dados ou outros objetos de informação;

- Abertura (*Openness):* essa característica determina a forma como o sistema pode ser estendido. A abertura em sistemas distribuídos é atingida pela especificação e documentação das *interfaces* mais importantes do sistema, tornando-as públicas para os desenvolvedores. Um sistema pode ser considerado aberto ou fechado com relação à extensão de *hardware*, por exemplo, com a adição de periféricos: memória ou interfaces de comunicação; ou com relação a extensões de software, por exemplo, com a adição de ferramentas nos sistemas operacionais e de protocolos de comunicação.

- Concorrência (C*oncurrency):* em sistemas distribuídos, existem vários computadores, cada um possuindo um ou mais processadores. Dessa forma, mais de um processo pode requisitar a utilização de um mesmo recurso, causando o surgimento de concorrência. Nesse contexto, a solução para esse problema se dá através da sincronização do acesso aos recursos compartilhados.

- Escalabilidade (*Scalability):* a necessidade por escalabilidade, ou seja, a capacidade de crescimento, não é simplesmente um problema de performance de *hardware* ou rede. Em sistemas de computação centralizados, alguns recursos compartilhados, como memória e processador, são limitados e não podem ser replicados indefinidamente. Em sistemas distribuídos, essa limitação é removida, pois, potencialmente, um sistema distribuído pode possuir um número ilimitado de computadores, cada um deles possuindo memória e processador próprios. Entretanto, a elaboração de um SD deve possibilitar uma utilização eficiente de escalabilidade, pois, à medida que o tamanho e a complexidade de um SD aumenta, torna-se difícil mantê-lo eficiente.

- Tolerância a falhas: é a capacidade de um sistema continuar operando quando ocorre alguma falha. As falhas em computadores podem ser de: *hardware* e de *software*. Para se projetar um sistema tolerante a falhas, o projetista deve inserir redundância de *hardware*, substituindo componentes à medida que falhas ocorrem, e projetando *software* com capacidade de recuperação, pois usualmente um sistema encontra-se em situações diferentes do seu funcionamento normal, podendo causar falhas em sua operação.

- Transparência: é definida como a capacidade de abstrair do usuário ou de uma aplicação a localização dos componentes de um sistema distribuído. Esta característica permite que um sistema distribuído não seja observado como uma coleção de componentes independentes, mas como um sistema único.

No projeto de sistemas distribuídos, outras características são necessárias:

- Nomeação: a localização de recursos em um sistema distribuído é normalmente realizada através dos nomes atribuídos a esses recursos. Essa atribuição deve possuir significado independente da localização do objeto. Nesse contexto, encontra-se um sistema interpretador de nomes, que permite programas acessarem recursos através de seus nomes.

- Comunicação: a utilização adequada de técnicas de comunicação entre computadores é essencial para uma performance adequada de um sistema

distribuído. A utilização exagerada de camadas de software, por exemplo, pode aumentar o tráfego de dados na rede.

- Modularização do software: a abertura de um sistema é atingida através da construção de componentes de software com *interfaces* bem definidas. Os serviços de um sistema distribuído são os gerenciadores de objetos de um dado tipo, enquanto que a *interface* para o serviço é o conjunto de suas operações. Seguindo essa técnica, novos serviços podem ser introduzidos em um SD sem haver interferência nem duplicação de serviços existentes.

- Balanceamento de processamento das máquinas: todos os sistemas devem funcionar com um nível aceitável de performance. Em um SD, esse objetivo é atingido com uma adequada alocação do recurso de processamento de cada máquina integrante de um sistema distribuída.

3.3 Arquiteturas Distribuídas para Engenhos de Busca

Vários sistemas e protocolos têm sido propostos como extensão ao funcionamento de um Engenho de Busca. Abaixo, analisam-se algumas dessas propostas, focalizando a análise nas principais contribuições à distribuição de cada uma.

3.3.1 Harvest

Este sistema consiste em um conjunto de ferramentas configuráveis para extrair, organizar, procurar e replicar informações relevantes disponíveis na Internet. Observa-se a Figura 5 descrevendo a arquitetura Harvest: [87]

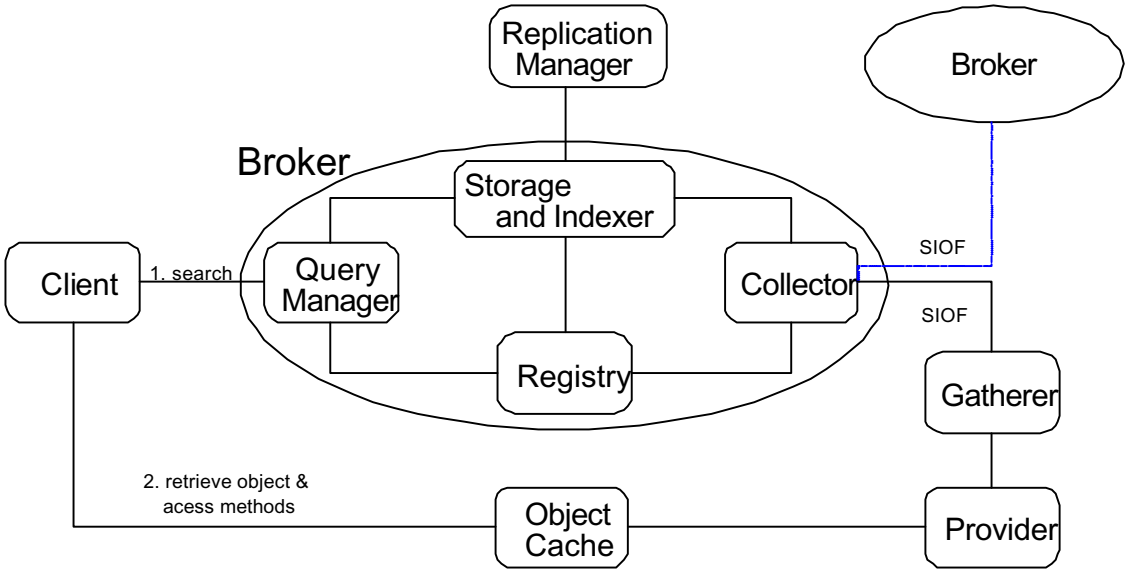

Figura 5 Arquitetura Harvest

- ***Gatherer***: provê uma maneira eficiente e configurável para coletar índices de informação;

- ***Broker***: provê uma interface de consulta sobre a informação indexada pelos *gatherers*. Ele recolhe informação de um ou mais *gatherers* ou outros *brokers*, e atualiza seus índices incrementalmente. O *broker* é o componente de distribuição da arquitetura, responsável pela distribuição da informação contida no *gatherer* e também é capaz de se comunicar com outros componentes *broker*, possibilitando a construção de uma complexa árvore hierárquica de sistemas de indexação. É composto pelos componentes: *Storage and Indexer*, *Query Manager*, *Collector* e *Registry*.

- ***Provider***: uma interface genérica de indexação para acomodar uma variedade de Engenhos de Busca;

- ***Replication Manager***: um sistema de arquivos nos quais os *brokers* são replicados;

- ***Object Cache***: uma hierarquia de cache de objetos, para atingir as demandas de acesso de rede e informação dos servidores.

Harvest utiliza uma arquitetura distribuída para recolher e distribuir dados, que é considerada mais eficiente que a arquitetura centralizada [51]. *Harvest* resolve os seguintes problemas enfrentados pelos Engenhos de Busca: (1) servidores *Web* recebem requisições de diferentes Robôs, aumentando sua carga; (2) o tráfego *Web* aumenta porque os Robôs coletam objetos (páginas) inteiras, mas a maioria do seu conteúdo é descartada; e (3) a informação é coletada independentemente do Robô, sem coordenação entre os Engenhos de Busca. O principal problema dessa arquitetura reside no fato que Harvest necessita da coordenação entre vários servidores *Web* [51].

3.3.2 CHIC-Pilot

CHIC-Pilot [48], abreviação da expressão *Cooperative Hierarchical Indexing Coordination*, é um projeto piloto fundado pela instituição TERENA (*Trans-European Research and Education Networking Association*) com a intenção de investigar a distribuição de indexação e recursos de rede.

Uma arquitetura foi desenvolvida para particionar a tarefa de indexação entre vários componentes, utilizando um conjunto de padrões (ver ilustração na Figura 6). Dentre os padrões utilizados, citam-se *whois++* [65], *CIP* (*Common Indexing Protocol*) [50] e *SOIF (Harvest Summary Object Interchange Format)* [22].

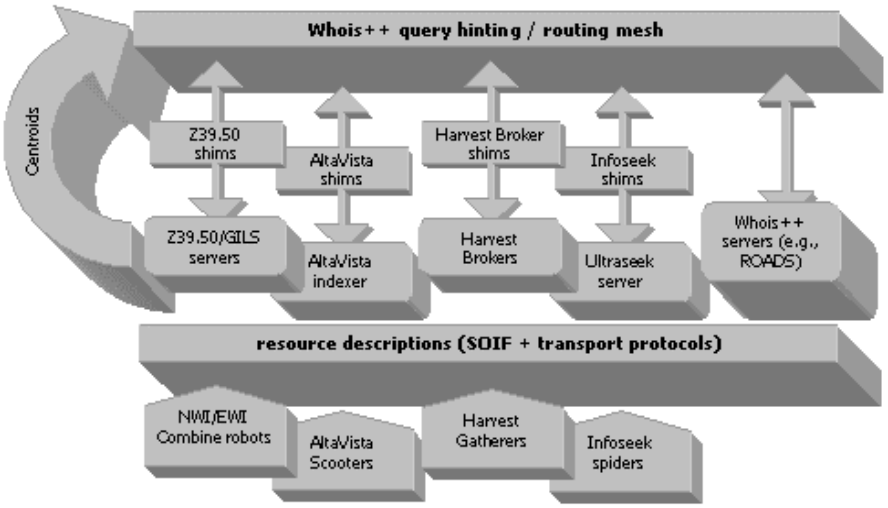

Figura 6 Tecnologias utilizadas em CHIC-PILOT

Existem diferentes Engenhos de Busca operando com técnicas de busca monolíticas. Esta situação proporciona problemas de escalabilidade. Por exemplo, um determinado *site* pode ser acessado por diferentes Robôs de EBs distintos.

A intenção do projeto CHIC-Pilot é prover novas soluções para o problema de busca e indexação na *Web,* mas principalmente utilizar protolocos existentes a fim de possibilitar a interoperabilidade das tecnologias. A dificuldade dessa solução consiste na adoção dos EBs existentes a essa arquitetura.

3.4 Conclusões do capítulo

Foi apresentado um estudo sobre Sistemas Distribuídos. Soluções existentes utilizando técnicas de distribuição foram apresentadas, extraindo-se algumas observações com relação ao *design* dos Engenhos de Busca:

- Modularidade: *Harvest* [41] não é uma solução modular, pois o *Broker,* módulo de distribuição existente na arquitetura, é um módulo essencial desse sistema. Em outras palavras, o sistema *Harvest* não funciona sem o componente *Broker*.

- Abertura: o projeto *CHIC-Pilot* [48] preocupa-se com a definição de *interfaces* públicas e a utilização de tecnologias padrões, a fim de facilitar a extensão do sistema com a inserção de novos componentes.

Capítulo 4 Radix: Uma Arquitetura Distribuída baseada em Visões Web

4.1 Introdução

Esse trabalho propõe a elaboração de um sistema distribuído de Engenhos de Busca. A finalidade do sistema é, através do compartilhamento de recursos pertinentes a esses sistemas de RI (banda passante, processamento, entre outros), prover uma solução adequada para o problema de escalabilidade existente.

4.2 Visões web (Web-views)

Há duas motivações básicas para o desenvolvimento do conceito de visões da *Web* para o problema de indexação e busca: (1) atingir eficiência computacional, através de uma arquitetura que permite uma solução plausível para o problema, e (2) atingir eficiência na maneira de prover localização de documentos que são relevantes a requisições dos usuários. A fim de atingir esses objetivos, a definição de visões *Web* é realizada em termos de quatro conceitos de distância digital (DD), descritos abaixo [43].

Definição 1: DD por Localização Geográfica (DDGL)

A DDGL entre dois documentos é a distância física, em quilômetros (Km), entre dois servidores *Web* onde os documentos são armazenados.

Definição 2: DD por Localização de Rede (DDNL)

Dados dois documentos, o DDNL é calculado de acordo com os endereços IP (Protocolo Internet) dos servidores *Web* onde são armazenados, segundo a Tabela 1.

Servidores *Web*	**DDNL**
Mesmo Endereço IP	0
Mesmo Endereço Classe C	1
Mesmo Endereço Classe B	2
Mesmo Endereço Classe A	3
Redes Classe A diferentes	4

Tabela 1Calculando DDNL's

Definição 3: DD por Localização de Hipertexto (DDHTL)

O DDHTL entre dois documentos é o comprimento do menor caminho de hipertexto entre dois documentos na *Web*. O DDHTL de qualquer documento para ele mesmo é zero.

Definição 4: DD por Contexto (DDC)

O DDC entre dois documentos é calculado de acordo com a função F(ddc): (Doc x Doc) → N, onde Doc é o conjunto de todos os documentos da *Web*, e N é o elemento do conjunto de todos os números naturais. O DDC entre dois documentos com contextos idênticos é zero, e existem documentos com contextos diferentes cujos DDC são zero, dependendo da função F(ddc) escolhida.

Dadas as definições de distância digital, uma maneira simples de caracterizar distância digital pode ser definida:

Definição 5: Distância Digital (DD) do tipo X

A DD do tipo X e raio R sobre a referência de um documento *Web* é o espaço onde são incluídos todos os documentos cuja DD do tipo X para a referência do documento é menor ou igual a R, onde X denota um tipo simples de DD (por exemplo, DDGL, DDNL, DDHTL ou DDC).

As localizações dos servidores, geográfica e topológica, são utilizadas nas definições de DDGLs e DDNLs, respectivamente. Isto permite a identificação de grupos de servidores *Web* que estão fisicamente localizados em uma região geográfica, ou conectados a uma subrede IP. Módulos de indexação que focalizam na área geográfica ou vizinhança de rede evitam a necessidade de utilizar longas distâncias entre *backbones* para indexação, e têm um menor número de páginas para indexar.

O contexto dos documentos é utilizado nas definições de DDHTLs e DDCs. Isto conta para indexação efetiva por introduzir medidas de similaridade entre documentos baseados em seus contextos e também na conexão entre os *link*s de hipertexto. Módulos de indexação podem então ser especializados nessas áreas. Técnicas na área de Recuperação de Informação podem ser utilizadas para escolher a função F(ddc) da definição 4. Por exemplo, uma simples função para calcular a distância entre dois

documentos pode se basear na comparação das listas de termos indexados desses documentos. A relevância da distância de hipertexto (DDHTL) surge do uso da estrutura dada pelos autores dos documentos *Web*. Esta estrutura constitui uma importante maneira para identificar informações correlacionadas.

Resumidamente, documentos *Web* podem ser caracterizados por: (a) localização do servidor; (b) contexto do documento; (c) o objeto onde o documento é armazenado (por exemplo, em arquivo, um registro em um banco de dados, ou um documento hipertexto); e (d) metainformação extraída do contexto do documento (por exemplo, título e idioma). Os atributos (c) e (d) acima compõem o esquema de representação do documento.

Agora, dadas as definições de DD e representação do esquema, uma forma de caracterizar escopos poder ser definida:

Definição 6: Visão *Web* (Escopo)

Uma visão *Web* é uma composição de DDs opcionalmente filtrados por um conjunto de restrições de páginas *Web*. Normalmente denomina-se visão da *Web* como escopo. Estas restrições aplicam-se aos atributos utilizados no esquema de representação dos documentos.

Em outras palavras, uma visão *Web* pode ser vista como uma expressão *booleana* utilizando os operadores E, OU e NÃO e tendo as distâncias digitais (DD) como operandos.

Um exemplo de visão *Web* de procura que pode ser modelado com visões *Web* é: "Considere todos os documentos que são similares ao contexto do documento em `http://algumhost/x.hml`, e que são armazenados em servidores a, no máximo, 200 km longe de Recife, PE, Brasil."

DD_1 = { tipo: contexto; conceito: similar a http://algumhost/x.html}

DD_2 = { tipo: geográfico; local: Recife, PE, Brasil; raio: 200 Km}

$VW_1 = (DD_1 \cap DD_2)$

A visão *Web* VW_1 do exemplo acima pode ser utilizada para descrever uma área de coleta de documentos de um EB.

De fato, nessa dissertação, o formalismo *Web-views* é utilizado para descrever a área de coleta de documentos dos EBs participantes do ambiente de distribuição proposto.

4.3 Módulos da Arquitetura

Seja proposta uma arquitetura distribuída para o ambiente de indexação e busca da *Web*, baseada em dois conjuntos de módulos autônomos (Engenho de Busca e Facilitador). A interação entre eles é ilustrada na Figura 7 e na Figura 8 abaixo:

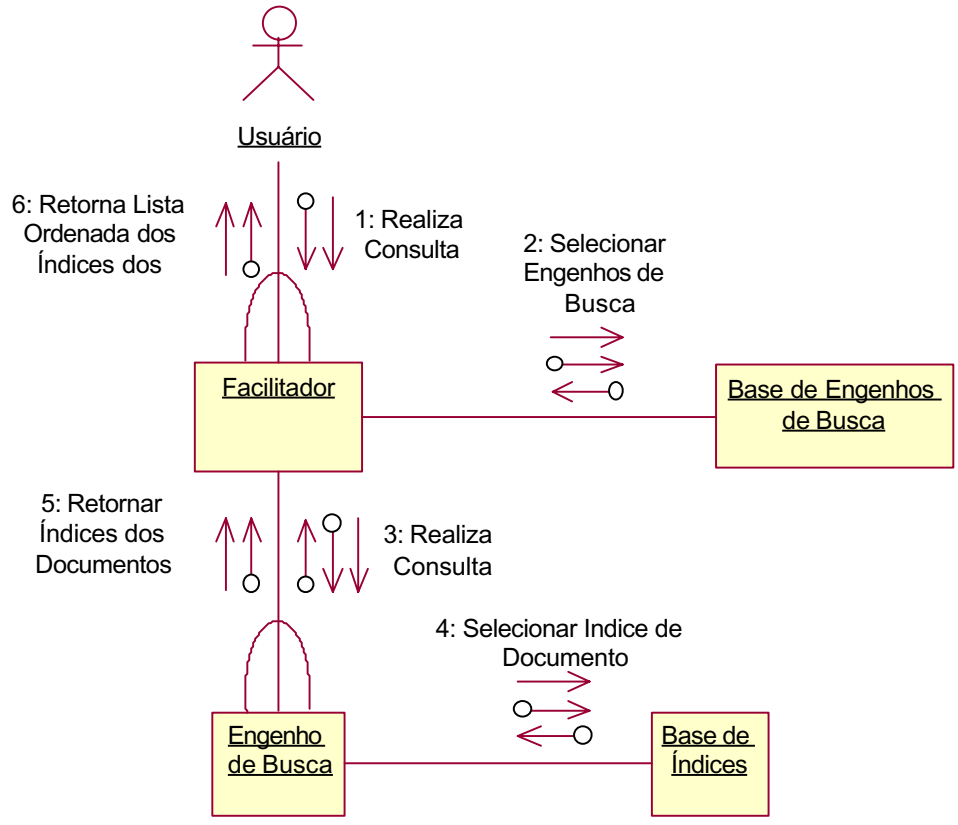

Figura 7 Diagrama de colaboração i lustrando uma busca na arquitetura distribuída

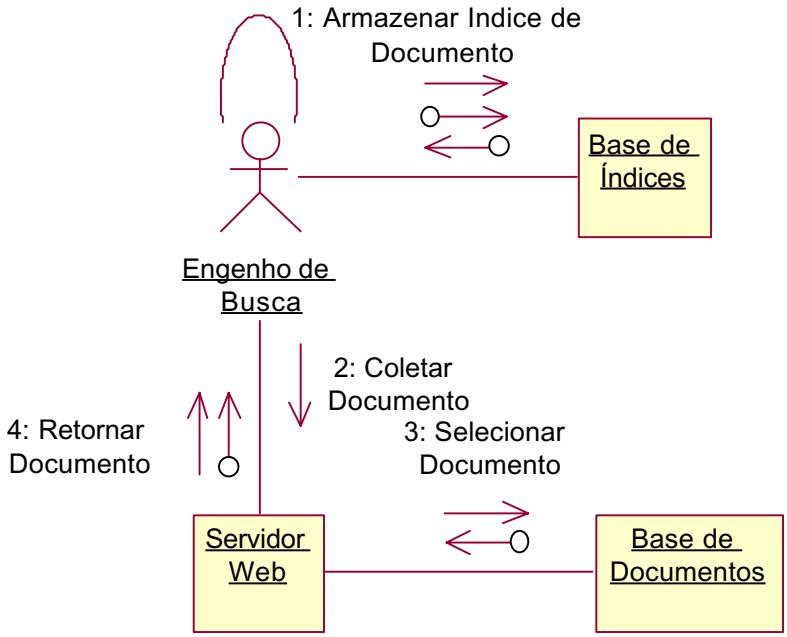

Figura 8 Diagrama de colaboração ilustrando a coleta e indexação de documentos na arquitetura distribuída

- Os Engenhos de Busca (EBs): cada EB tem sua própria Base de Índices (BI) e focaliza uma particular visão *Web*; cada Engenho também contém um Robô que continuamente percorre a *Web* procurando por páginas que estejam dentro de seu escopo, e, portanto, deveriam ser indexadas e armazenadas em seu BI. A busca é realizada em sua BI (essa situação está descrita na seção 2.4.1). Adiciona-se ao EB um componente chamado Agente de Distribuição ou simplesmente AD, a fim de inserir o Engenho em um ambiente distribuído. Esse componente é responsável pela comunicação existente entre os Engenhos.

- Os Facilitadores (Fs): uma camada de mais alto nível de abstração é construída acima do EB. Os Facilitadores selecionam os Engenhos apropriados para responder a requisições do usuário, gerenciando sessões de procura, possivelmente envolvendo múltiplos Engenhos. Os Facilitadores também são responsáveis por manterem um registro de todos os Engenhos existentes no sistema.

Uma instanciação da arquitetura proposta é realizada nas figuras abaixo, contemplando as tarefas de busca, coleta e indexação de documentos:

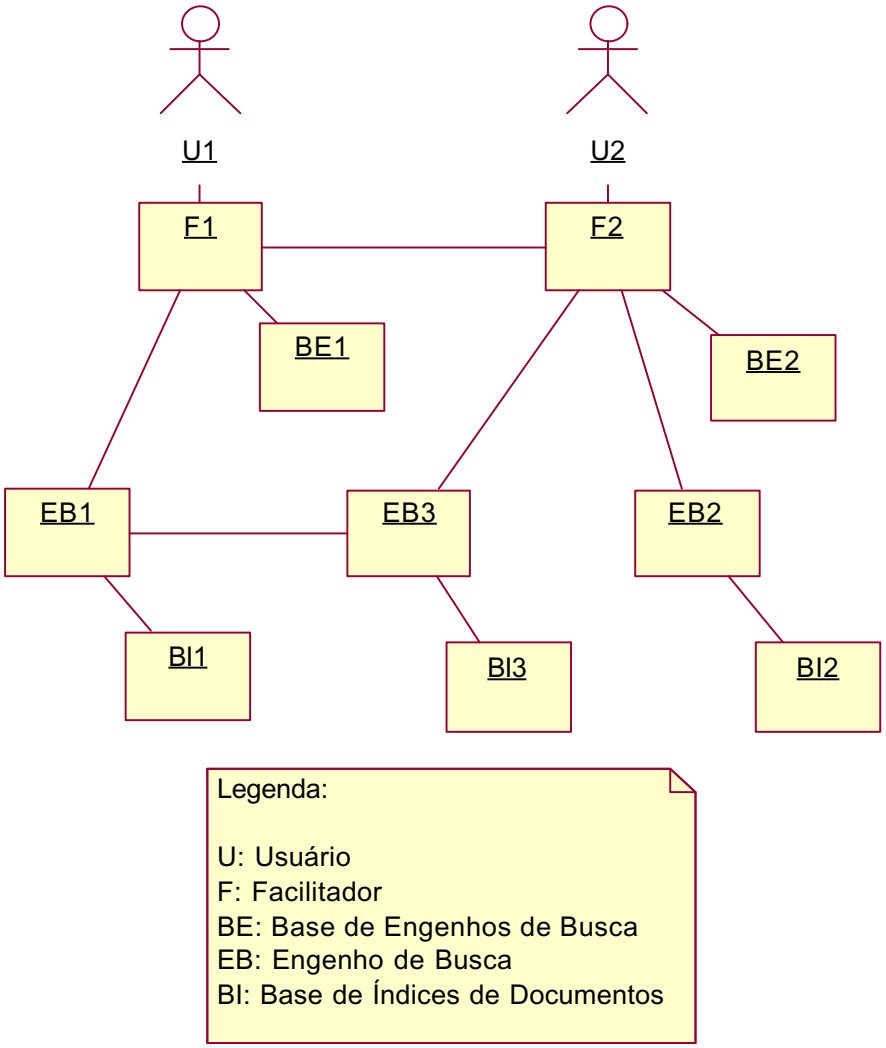

Figura 9 Instanciação da arquitetura distribuída ilustrando a tarefa de consulta

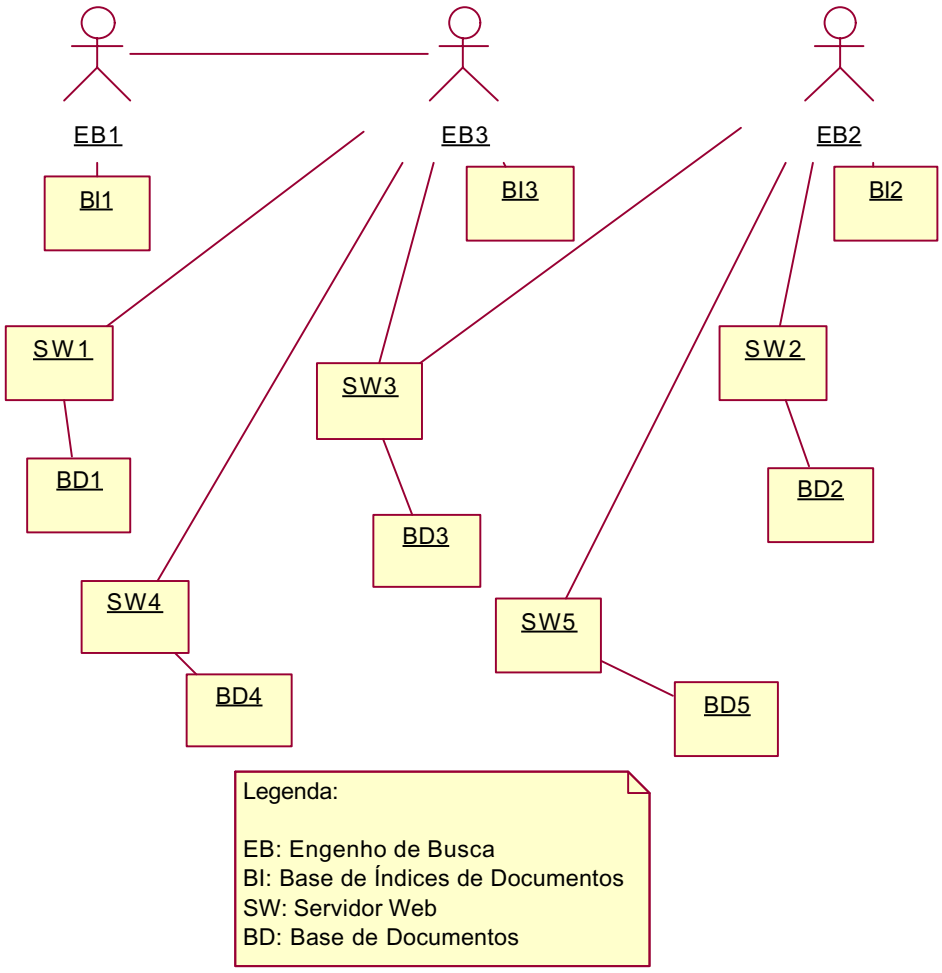

Figura 10 Instanciação da arquitetura distribuída ilustrando a tarefa de coleta e indexação de documentos

Como mostrado na Figura 8, cada Servidor *Web* SW_i possui uma base de dados (BD_i) onde suas páginas *Web* são armazenadas; cada Engenho de Busca EB_i possui sua Base de Índices (BI_i); e de acordo com a Figura 7, cada Facilitador tem sua Base de Engenho de Busca (BE_i), onde mantém informação de escopo sobre EBs conhecidos e referências para outros Facilitadores. Engenhos de Busca requerem páginas de servidores *Web* para indexação, e Facilitadores requerem índices dos Engenhos para responder a requisições dos usuários. EBs cooperam trocando índices de páginas (por exemplo, EB_1 e EB_3 da Figura 10), e Facilitadores cooperam trocando metainformação a respeito dos Engenhos (por exemplo, F_1 e F_2 na Figura 9).

4.4 Compartilhamento de recursos entre Engenhos de Busca

Um único sistema de indexação e busca não resolve a contento o problema da indexação e procura na *Web*, levando em conta a sua taxa de crescimento e o seu tamanho atual. A fim de atingir níveis de escalabilidade aceitáveis, soluções distribuídas surgem no contexto da *Web*, um ambiente naturalmente distribuído. Atualmente, existem inúmeros Engenhos de Busca em operação. Normalmente, esses sistemas funcionam isoladamente, não considerando a existência de outros sistemas. Se houvesse comunicação entre eles, haveria a possibilidade de colaboração de recursos nas diferentes tarefas realizadas pelo Engenho:

- C*rawling*: supõe-se dois Engenhos de Busca interessados em um mesmo conjunto de páginas. Se o Robô de um Indexador A processa essas páginas, não haveria necessidade do Robô do outro Indexador B processá-las, caso houvesse colaboração entre os indexadores. Na tarefa de *crawling,* outro fator considerável é a topologia da rede: caso os Robôs do indexador A estejam mais próximos das páginas a serem indexadas (A noção de distância é descrita detalhadamente na seção 4.2), haveria um ganho de tempo para se realizar o "*download*" das páginas.

- Indexação: supondo a colaboração proposta no tópico acima, os índices das páginas poderiam ser enviados de um EB para outro de forma "inteligente" (por exemplo, compactadas, para consumir menos banda passante e em horário de baixa carga da rede).

- Busca: dois fatores comprometem a qualidade de atendimento de um sistema de busca: o número de usuários e o número de documentos a serem processados. Na medida que aumenta o número de páginas existente no Engenho, exige-se um maior poder de processamento a fim de não comprometer o tempo de resposta do sistema, pois há mais informações a serem processadas. Na medida que aumenta o número de usuários do sistema, aumenta-se a quantidade de requisições a serem processadas. Tanto a quantidade de consultas dos usuários quanto a quantidade de páginas processadas por um sistema de busca poderiam ser distribuídos entre os integrantes de um sistema distribuído de Engenhos de Busca. Além disso, na resposta dada a um usuário, um Engenho de Busca poderia sugerir que as

consultas realizadas em seu índice fossem também realizadas em outro sistema. Desse fato decorre uma maior variedade de respostas para uma mesma pergunta, aumentando a possibilidade de satisfazer o usuário.

O motivo básico da elaboração de um ambiente distribuído para sistemas de indexação e busca é a possibilidade de compartilhamento adequado de recursos computacionais, possibilitando um funcionamento mais eficiente para cada sistema do ambiente distribuído. No ambiente de distribuição proposto, o compartilhamento de recursos é realizado através da negociação entre os Engenhos. Na negociação entre os Engenhos, um dos principais elementos utilizados é a área de coleta, descrita por uma visão *Web* de cada EB. Vejam as formas de compartilhamento de recursos possíveis:

- Compartilhando esforços de *crawling* e indexação: um módulo EB_1 pode deixar a tarefa de indexar a interseção entre os escopos para o módulo EB_2. (Isso ocorre no seguinte caso: sendo EB_1 um novo módulo com escopo E_1, EB_2 um módulo existente com escopo E_2, e I_{12} uma interseção não vazia entre E_1 e E_2, ou seja, $I_{12} = (E_1 \cap E_2)$ ô). Assim, EB_1 precisa receber os índices da interseção dos escopos periodicamente. O envio dos índices pode ser realizado de duas formas: EB_1 requisita os índices da atualização da interseção com EB_2, ou EB_2 envia os índices da atualização da interseção periodicamente para EB_1. Esta situação economiza recursos computacionais (processamento, rede) de EB_1, pois não é necessário que seus Robôs processam as páginas existentes na interseção; e também salva recursos de servidores *Web* acessados por EB_2, pois as páginas da interseção somente serão processadas por apenas um EB.

- Compartilhando serviços de busca: supondo que não houve compartilhamento de esforços de indexação, EB_1 pode não manter dados de índices duplicados sobre a interseção com EB_2. Nesse caso, EB_1 pode compartilhar serviços de consulta com EB_2. EB_1 não mantém índices sobre a interseção dos escopos, e consulta o módulo cooperante todo instante que necessite de realizar procuras envolvendo a interseção. Isto salva tanto recursos computacionais como espaço de armazenamento de EB_1, mas o torna dependente de EB_2 para realizar consultas, implicando menor tolerância a falhas e possivelmente maior tempo de resposta.

4.5 Definição do *Framework*

Na definição do *framework*, houve o interesse em utilizar um método formal para descrevê-lo. Mais especificamente, foi utilizado o formalismo de Redes de Petri Coloridas (CPN) [**68**], [**31**], e [**29**], uma ferramenta formal especialmente apropriada para análise automática de sistemas concorrentes, distribuídos, estocásticos e/ou não-determinísticos. De acordo com Kurt Jensen [31], existem três razões para se modelar sistemas:

- Um modelo é uma descrição da modelagem de um sistema, e pode ser utilizado como uma especificação (do sistema que se deseja construir) ou como uma apresentação (do sistema que se deseja explicar a outras pessoas). Através da criação do modelo, um sistema pode ser investigado antes de ser construído. Esta é uma vantagem óbvia, em particular para sistemas onde erros de "*design*" geram risco de segurança ou implicam elevados gastos para correção das falhas detectadas nas fases posteriores da construção;

- O processo de descrição e análise aumenta o entendimento do sistema modelado;

- O comportamento de um modelo pode ser analisado, através de simulação (equivalente à execução ou à depuração de um programa) ou por meio de métodos de análise formal (equivalente à verificação do programa).

O intuito de utilizar um formalismo para descrever o *framework* pretende atingir as duas primeiras das razões citadas. A última razão está fora do escopo dessa dissertação.

Além disso, existem mais três razões para se modelar sistemas com Redes de Petri (PN):

- Redes de Petri possuem uma representação gráfica. Essa representação é fácil de entender até mesmo para pessoas não familiares com os detalhes das PN;

- PN possui semântica para tratamento de concorrência. Isto significa que as noções de conflito, concorrência e dependência podem ser definidas de uma forma

natural. PNs mostrou-se como uma das linguagens de modelagens mais utilizadas em sistemas que contém processos concorrentes;

- PN possui uma semântica bem definida, permitindo análise formal.

Nessa dissertação, utilizou-se o formalismo Redes de Petri Coloridas (CPN), uma extensão de alto nível de Redes de Petri. Realizando uma analogia com linguagens de programação, utilizar CPN ao invés de CP significa preferir uma linguagem da 3ª. geração à linguagem de máquina.

Apresentadas as razões para a adoção do formalismo, segue a descrição do *framework*, ilustrado através dos eventos e estados que nele ocorrem no ambiente de Engenhos de Busca Distribuídos. A Figura 11 apresenta a declaração das cores comuns a todas as CPNs

Declarações de Cores:

```
color EBId: int;
color Escopo: ESCOPO;
color EB: product EBId * Escopo;
color GrupoEB: list EB;
color EBIdGrupoEB: product EBId * GrupoEB;
color EBDistribuido: product EB * GrupoEB;
color FId: int;
color F: product FId * Escopo;
color GrupoF: list F;
color FIdGEB: product FId * GrupoEB;
color FIdGF: product FId * GrupoF;
color FDistribuido: product F *GrupoEB*GrupoF
```

Figura 11 Declarações de cores utilizadas em todas as CPNs

Eventos que ocorrem no ambiente distribuído

Instanciação de um Engenho de Busca

```
Declarações:

var f: F;
var gf: GrupoF;
var eb: EB;
var idEB: EBId;
var geb: GrupoEB;
var escopo: Escopo;
fun FAdiEB(eb, geb) =
    if (ExisteIntersecao(eb, head(geb)) then
        head(geb)::FAdiEB(eb,tail(geb))
    else FAdiEB(eb,tail(geb))
```

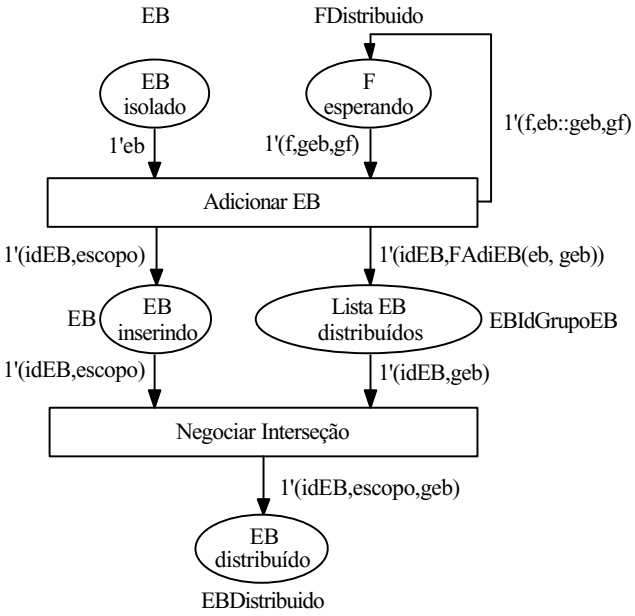

Figura 12 CPN ilustrando a inserção de um EB em um ambiente distribuído

Quando um Engenho EB é instanciado, determina-se seu escopo de cobertura. O módulo EB é registrado em um Facilitador F. Este atualiza o seu registro de módulos EBs e retorna a EB uma lista contendo informações sobre os módulos EBs com os quais existe a possibilidade de compartilhamento de recursos (transição "Adicionar EB" da Figura 12.

Cabe ao Facilitador a função de verificar a possibilidade de compartilhamento de recursos entre os EBs, pois ele é o módulo do ambiente distribuído que conhece a localização de todos os EBs. O módulo Facilitador verifica a possibilidade de compartilhamento de recursos entre dois EBs através do escopo de ambos. Como o escopo define a área de cobertura de um EB, ele verifica se há interseção entre os escopos dos EBs. Caso positivo, existe a possibilidade de cooperação entre os sistemas.

Quando o EB recebe a lista de módulos EBs no qual pode comunicar-se, inicia-se uma negociação com cada EB existente na lista recebida. Em cada negociação, seus

integrantes informam quais recursos desejam compartilhar (transição "Negociar Interseção da Figura 12).

Remoção de um Engenho de Busca

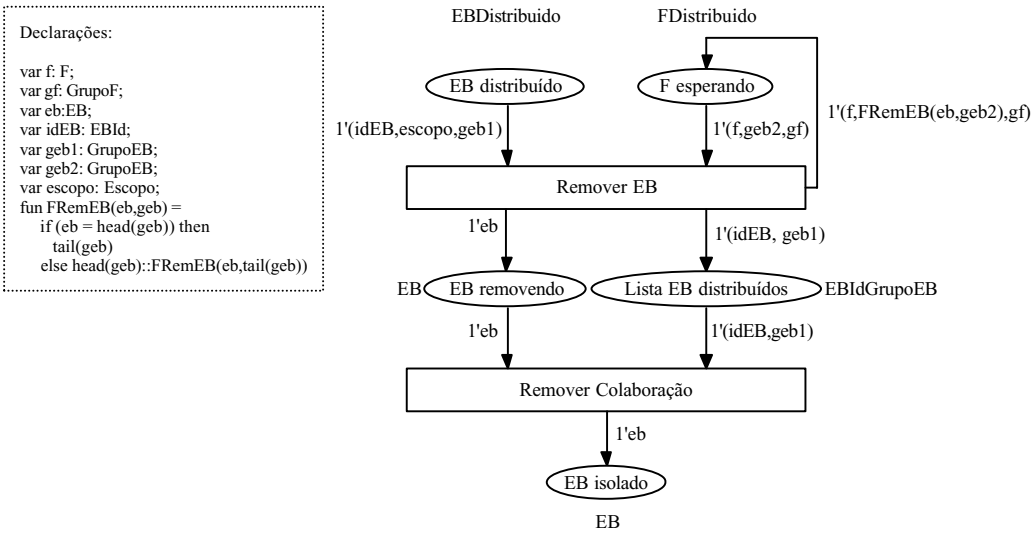

Figura 13 CPN ilustrando a remoção de um EB em um ambiente distribuído

Um EB deixa de participar de um ambiente distribuído quando ele gera um evento solicitando a sua exclusão ao Facilitador. Ao tratar esse evento, o Facilitador remove a referência do EB da sua base de EBs (transição "Remover EB" da Figura 13) e informa a todos os EBs cooperantes com EB, para providenciarem a remoção da colaboração (transição "Remover Cooperação" da Figura 13).

A remoção da colaboração entre EBs ocorre da seguinte forma:

- Na existência de compartilhamento de esforço de indexação (seção Compartilhamento de recursos entre Engenhos de Busca), não haverá mais o envio de índices do EB a ser removido do ambiente de distribuição (EB_1) para o EB que permanece (EB_2) se EB_1 indexava a interseção dos escopos. Assim, EB_2 passa a indexar a área da interseção dos escopos. Caso EB_2 indexasse a interseção dos escopos, ele deixaria de enviar índices para EB_1;

- Na existência de compartilhamento de esforço de busca, EB_2 não servirá mais consultas para EB_1 se EB_2 indexava a interseção dos escopos. Caso contrário, EB_2

não mais consultará EB_1, passando a indexar a extinta área de interseção dos escopos.

Alteração do escopo de um Engenho de Busca

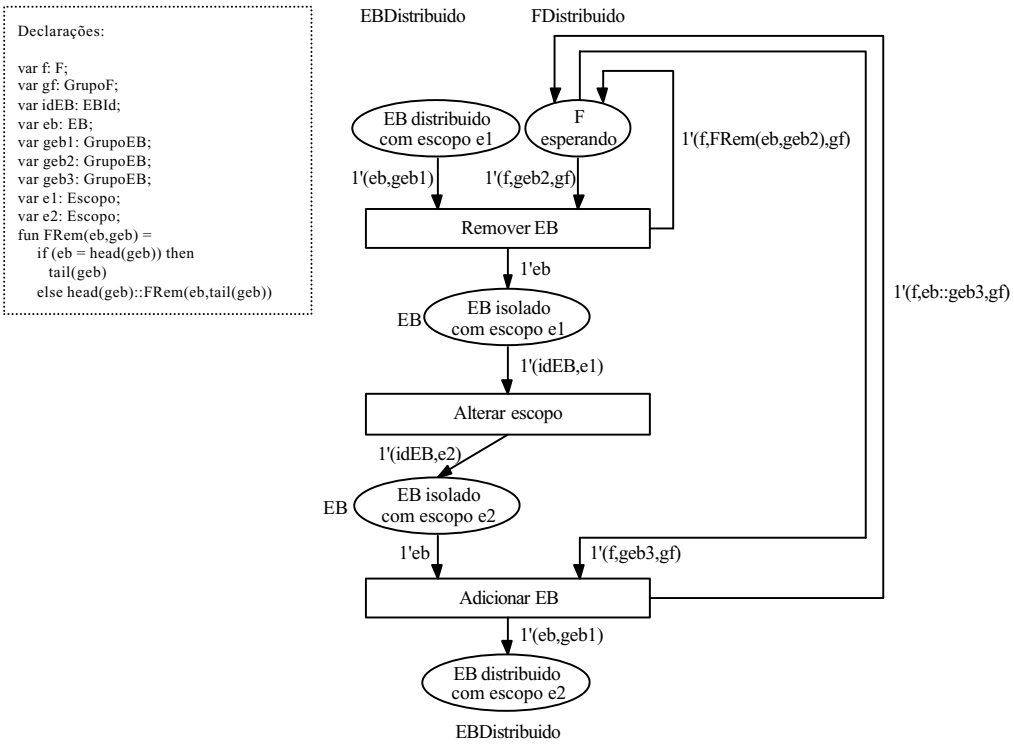

Figura 14 CPN ilustrando a alteração do escopo de um EB em um ambiente distribuído

A alteração de uma visão *Web* e_1 de um EB para e_2 equivale à remoção de um EB cujo escopo é e_1 e a instanciação de outro EB cujo escopo é e_2. De fato, quando um Engenho tem seu escopo alterado, ele reflete essa alteração através do tratamento de dois eventos no Facilitador: o primeiro é para removê-lo do sistema (transição "Remover EB" da Figura 14), alterando o seu escopo (transição "Alterar Escopo" da Figura 14); o segundo é para inseri-lo com o novo escopo (transição "Adicionar EB" da Figura 14). Os dois eventos são os descritos nos subtópicos anteriores.

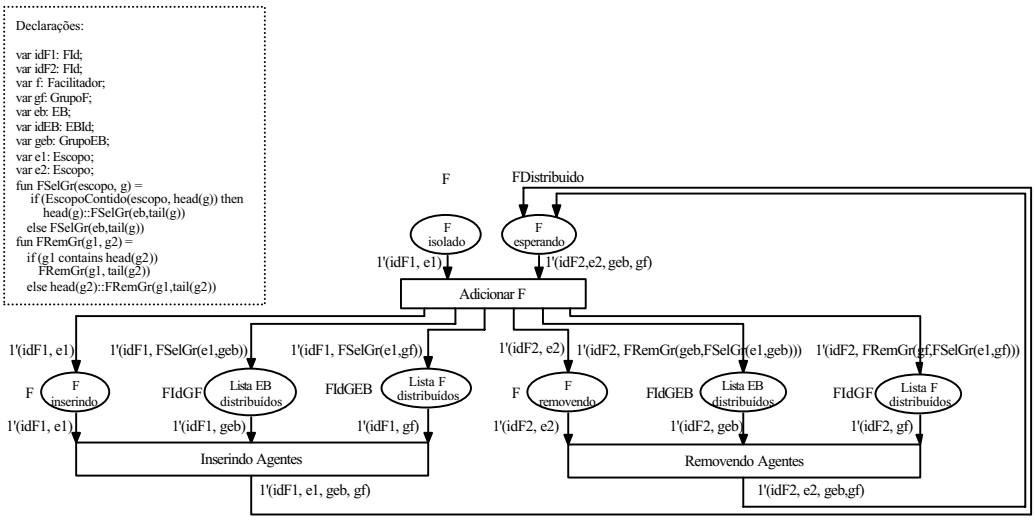

Figura 15 CPN ilustrando a inserção de um Facilitador em um ambiente distribuído

Quando um Facilitador F (F_1) é instanciado no sistema, define-se uma visão *Web* para esse módulo. Para ser inserido no sistema distribuído, o Facilitador envia uma mensagem de cadastramento para outro Facilitador (F_2) existente no sistema (transição "AdicionarF" da Figura 15).

F_1 recebe de F_2 uma lista de EBs e uma lista de Facilitadores, inserindo-os em seu registro de EBs e Facilitadores, respectivamente (transição "Inserindo Agentes" da Figura 15). Além disso, F_2 remove de seu registro os elementos das listas enviados a F_1 (transição "Removendo Agentes" da Figura 15).

Os elementos de cada uma dessas listas são determinados através da comparação da visão *Web* de cada módulo. Um EB ou um Facilitador somente podem ser registrados em um outro Facilitador quando a visão *Web* deste contém a visão *Web* do primeiro (sendo F_I um novo módulo com escopo e_I, F_{II} um módulo existente com escopo e_{II}, e I uma interseção entre e_I e e_{II}, então o escopo e_I contém e_{II} quando $I = (e_1 \cap e_2) = e_2$). Essa exigência facilita a operação e comunicação entre os Facilitadores, criando uma árvore de dependência entre eles.

Remoção de um Facilitador

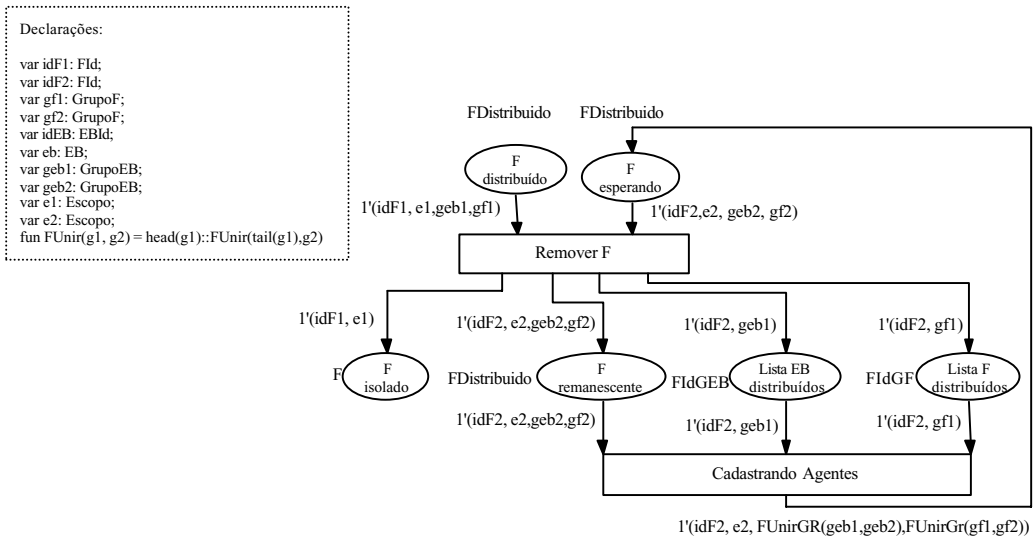

Figura 16 CPN ilustrando a remoção de um Facilitador em um ambiente distribuído

Quando um Facilitador é removido do sistema (F_1), ele envia uma mensagem de remoção ao Facilitador no qual está cadastrado (F_2), correspondendo à transição "Remover F" da Figura 16. Nesta mensagem, F_1 envia também as listas de EBs e Facilitadores que mantinha, a fim de que sejam cadastrados no F_2 (transição "Cadastrando Agentes" da Figura 16).

Alteração do escopo de um Facilitador

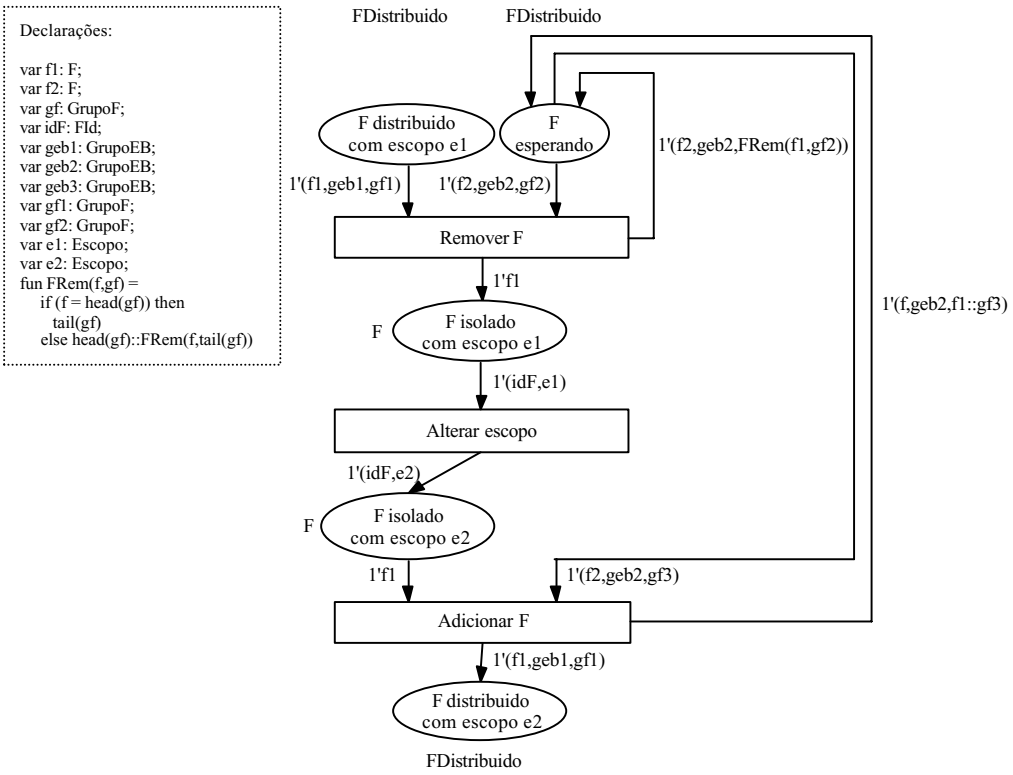

Figura 17 CPN ilustrando a alteração do escopo de um Facilitador em um ambiente distribuído

De forma análoga à alteração do escopo de um SI, a alteração do escopo de um Facilitador corresponde aos eventos de remoção (transição "RemoverF" da Figura 17) e inserção (transição "AdicionarF" da Figura 17) do Facilitador no sistema, intercalados pela alteração da escopo do Facilitador (transição "Alterar Escopo" da Figura 17).

Estados do Sistema Distribuído de Engenhos de Busca

O ambiente distribuído para indexação e busca possui um regime normal de operação. Esse funcionamento possui alguns estados similares a um EB centralizado, embora adaptados ao ambiente distribuído. Entretanto, existem novos estados exclusivos do sistema distribuído.

Consulta de um usuário

```
Declarações:

color Pergunta: String;
color Consulta: product Pergunta * Escopo;
color ConsultaFDistribuido: product Consulta * FDistribuido;
color CFDistribuidoGEB: product ConsultaFDistribuido * GrupoEB;
color Resposta: String;
color GrupoR: list Resposta;
color CFDistribuidoGrupoR: product ConsultaFDistribuido *GrupoR;
var f: F;
var gf: GrupoF;
var eb: EB;
var geb: GrupoEB;
var e: Escopo;
var c: Consulta;
var p: Pergunta;
var gr: GrupoR;
fun FSelEB(e, geb) =
    if (ExisteIntersecao(e, head(geb)) then
        head(geb)::FSelEB(e,tail(geb))
   else FSelEB(e,tail(geb))
fun FPegaRes(p, geb) =
    return Respostas(p, head(geb)::FPegaRes(p,tail(geb))
```

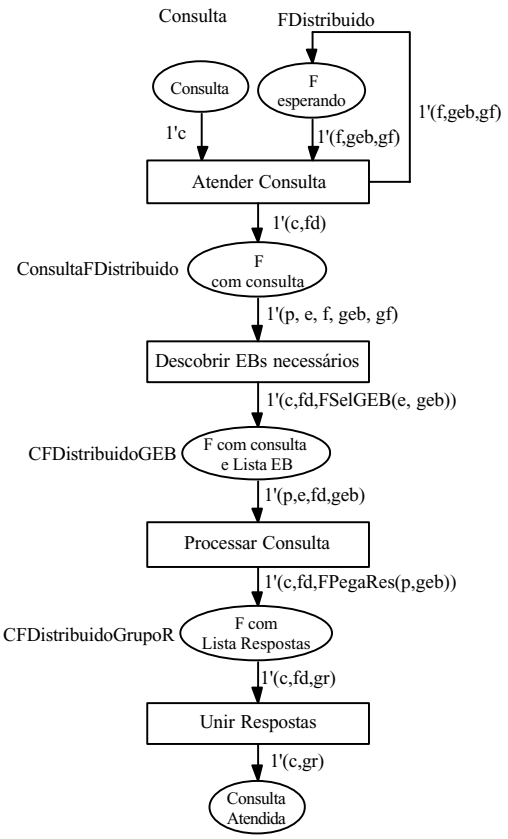

Figura 18 CPN ilustrando uma consulta realizada em um ambiente distribuído

Assume-se que o estilo típico de interação do usuário com EBs consiste em uma série de requisições. O sistema adota um estilo de interação orientado a sessão. No início de uma sessão de procura, o usuário interage com o módulo Facilitador e define, interativamente, uma *Web-view* onde a sessão ocorrerá (transição "Atender Consulta" da Figura 18).

Neste ponto, o módulo Facilitador em uso checa se os módulos EBs que ele conhece são suficientes para cobrir o escopo requisitado, ou seja, se existe uma combinação de Engenhos conhecidos (transição "Descobrir EBs necessários" da Figura 18). A fim de realizar operações em visões *Web* em escopos de procura de vários módulos, tais como encontrar composições de escopo de indexação que confere o escopo de procura selecionado, a álgebra de visões *Web* é utilizada.

Neste estágio, o Facilitador está pronto para processar a requisição de seção. Ele envia a consulta aos EBs selecionados, coletando as respostas processadas pelos EBs (transição "Processar Consulta" Figura 18), unindo e retornando-as ao usuário (transição "Unir Respostas" da Figura 18).

Coleta de páginas (*crawling)*

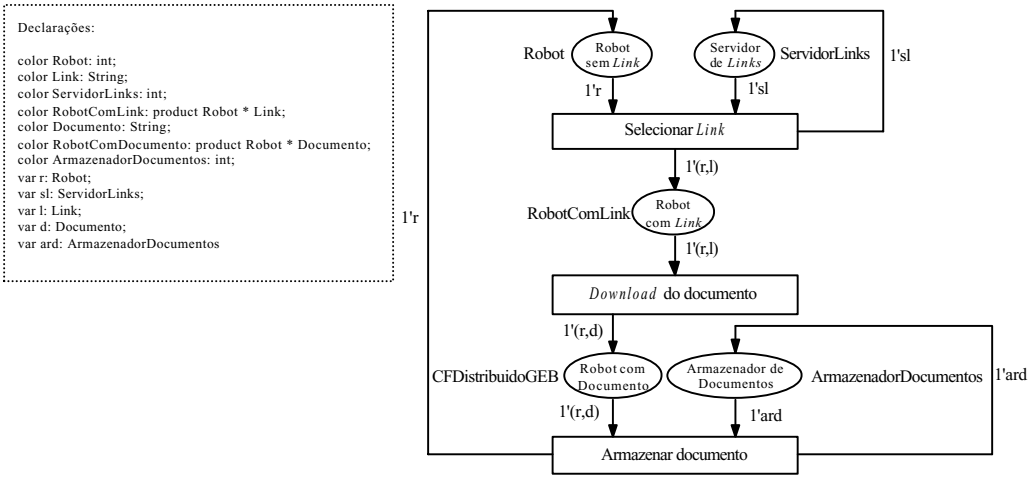

Figura 19 CPN ilustrando a coleta de páginas de um EB em um ambiente distribuído

O trabalho realizado pelos Robôs de um Engenho de Busca em ambiente distribuído é similar ao trabalho que o mesmo Engenho realizaria se estivesse em um ambiente isolado: o Robô seleciona um *link* (transição "Selecionar Link" da Figura 19), realiza o *download* do documento (transição "*Download* do documento" Figura 19), armazenando-o em seguida (transição "Armazenar documento" Figura 19).

Cada EB possui uma estratégia para percorrer a *Web* em busca das páginas. A única alteração reside no fato que a visão da *Web* do Robô pode ser diminuída com relação à visão original devido ao compartilhamento de indexação que o Engenho pode realizar. Isso ocorre se ele compartilhar esforço de indexação com outro e este indexar a área de interseção dos escopos.

Armazenamento de índices (indexação)

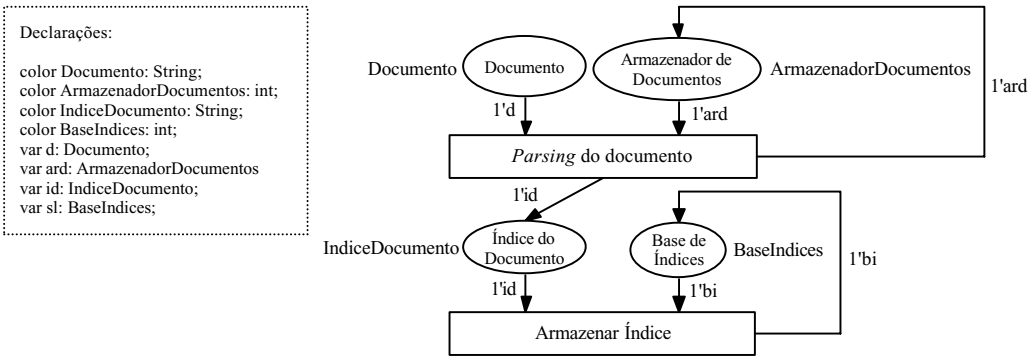

Figura 20 CPN ilustrando o armazenamento de índices de um EB em um ambiente distribuído

O Armazenador de Documentos realiza o *parsing* no documento, resultando no índice deste documento (transição "*Parsing* do documento" da Figura 20). O índice do documento é utilizado na inserção da Base de Índices (transição "Armazenar Índice" da Figura 20).

A inclusão de um Engenho em ambiente distribuído não afeta o armazenamento dos seus índices, dado que a forma de armazenar os índices é independente para cada EB. A alteração ocorre na determinação de quais índices serão armazenados. Supondo que EB_1 compartilha esforço de busca com EB_2 e EB_1 deixa a tarefa de indexar a interseção a cargo do EB_2, então EB_1 pode não manter uma cópia dessa interseção em sua base de índices local, economizando espaço em disco.

Envio de índices de páginas

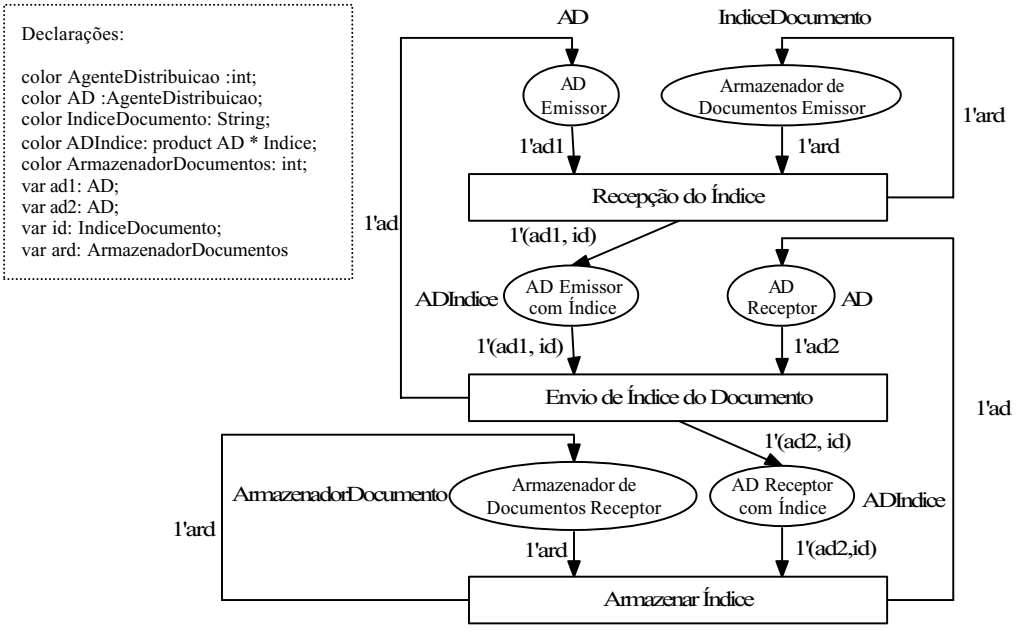

Figura 21 CPN ilustrando a envio de índices entre EBs em um ambiente distribuído

O envio de índices de páginas indexadas é necessário quando ocorre compartilhamento de indexação entre dois EBs e quando o EB que não indexa a interseção entre os escopos dos sistemas deseja manter uma cópia das páginas dessa interseção em sua Base de Índices. Essa tarefa é exclusiva a EBs envoltos em ambiente distribuído. O envio de páginas é realizado pelo Agente de Distribuição (AD), componente responsável pelas tarefas de distribuição de um EB.

Os índices do AD emissor provêm do Armazenador de Documentos do EB emissor (transição "Recepção do Índice" da Figura 21). De posse desses índices, o AD emissor os envia ao AD receptor com o qual compartilha índices (transição "Envio de Índice do Documento" da Figura 21). O AD receptor, por sua vez, armazena esses índices na Base de Índices do EB receptor (transição "Armazenar Índice" da Figura 21).

Há dois aspectos a serem considerados no envio de páginas: a abertura da informação e a economia de banda passante. A abertura dos dados é obtida através da formatação do pacote das páginas utilizando *interfaces* públicas e padrões de formatação de dados. Esta exigência possibilita uma fácil inserção de um Engenho no sistema distribuído. A

economia de banda passante ocorre no agrupamento e compactação dos índices enviados através da rede. Outra otimização seria o agendamento do horário do envio das páginas, a fim de se utilizar um horário com menor utilização da rede.

4.6 Conclusões do capítulo

Foram apresentados os aspectos referentes à solução adotada para a distribuição dos Engenhos de Busca, com os quais se deseja atingir alguns objetivos de *design*, que são:

- Escalabilidade: na medida que o número de documentos e o número de usuários existentes na *Web* pode crescer, aumenta-se organizadamente o número de EBs para coletar e indexar os documentos e processar as consultas dos usuários.

- Modularidade: a inserção do Agente de Distribuição no EB é realizada de forma modular, pois ele é um componente facilmente acoplado ao sistema. Caso um Engenho não deseje mais participar do ambiente distribuído, basta apenas desabilitar o AD.

- Abertura: houve a preocupação da definição de *interfaces* públicas e a utilização de tecnologias padrões, a fim de facilitar a extensão do sistema com a inserção de novos componentes.

Capítulo 5 O Protótipo

O Agente de Distribuição (AD) insere o Engenho de Busca em um ambiente de distribuição. Dentre as possibilidades de colaboração de recursos providas pelo AD, ele é responsável pelo envio dos índices das páginas; a estrutura de dados básica utilizada pelo Engenho de Busca, para montar a Base de Índices.

5.1 Introdução

Com a finalidade de validar as propostas contidas no capítulo anterior, um ambiente de distribuição foi implementado entre Engenhos de Busca. Além disso, testes e medições foram realizados, para avaliar a performance dos EBs nesse ambiente.

O capítulo está organizado nas seguintes seções: a seção 4.2 apresenta a arquitetura de distribuição da indexação; a seção 4.3 descreve o Agente de Distribuição, incluindo o seu funcionamento, suas características não-funcionais e o relato de um experimento avaliando o desempenho desse componente; a seção 4.4 apresenta um estudo de caso do sistema distribuído; a seção 4.5 descreve as tecnologias e as metodologias utilizadas no desenvolvimento do sistema e, finalmente, a seção 4.6 mostra as conclusões extraídas do capítulo.

Arquitetura da Indexação Distribuída

A arquitetura de distribuição foi desenvolvida a partir da arquitetura do Engenho de Busca do Radix [17], mais especificamente nos módulos responsáveis por *crawling* e pela indexação. Inicialmente, esse Engenho tinha a seguinte arquitetura entre os módulos de indexação:

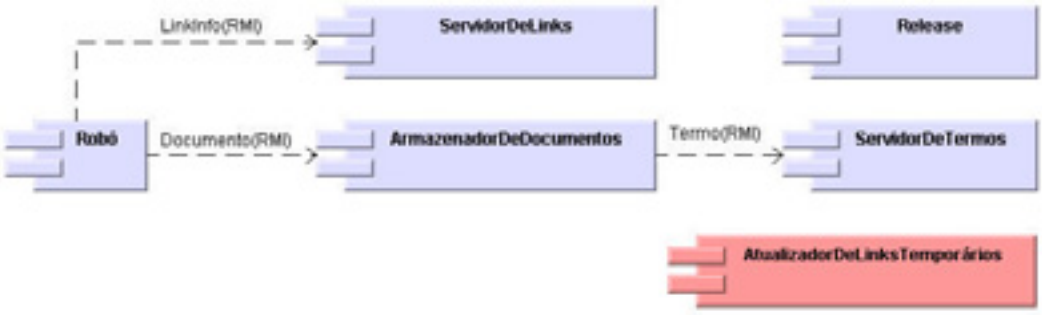

Figura 22 Arquitetura inicial (entre 1999 e 2000) do módulo de C*rawling e* Indexação de Engenho de Busca Radix

- Servidor de *Links*: responsável pelo fornecimento de *links* a serem processados pelo robô;

- Armazenador de Documentos: armazena os documentos processados pelo robô;

- Servidor de Termos: responsável pelo manipulação de todos os termos existentes nos documentos;

- Atualizador de *Links* Temporários: processa os *links* encontrados nos documentos;

- Robô: seleciona *links* no Servidor de *Links*, realiza o "*download*" do documento referenciado por esse *link*, e armazena-o no Armazenador de Documentos.

- Release: responsável por disponibilizar na busca todas as informações coletadas pelo processo de *crawling e* indexação.

Para evoluir a arquitetura acima, a fim de disponibilizar o Engenho de Busca em um ambiente de distribuição, fez-se necessário o desenvolvimento de um novo módulo, o Agente de Distribuição, ou simplesmente AD. Além disso, a inserção do AD exigiu mudanças na arquitetura da indexação.

O AD manipula índice dos documentos, estrutura que contém informações textuais desses documentos (ver Apêndice B). Essas informações textuais devem ser sintetizadas em um formato numérico para o processamento eficiente da indexação e busca. Dessa forma, todas as informações textuais do índice do documento devem ser convertidas por informações numéricas na inclusão da Base de Índices. Basicamente, as informações textuais convertidas são os termos do documento e os *links* encontrados no documento.

O AD acessa o módulo Armazenador de Documentos, que por sua vez acessa o módulo Servidor de Termos, responsável pela conversão numérica dos termos. Entretanto, o AD não consegue acessar o componente Armazenador de *Links* Temporários, responsável pela conversão dos *links*. Assim, houve a necessidade de modificação na forma como os *links* dos documentos são processados, a fim de disponibilizar um método de consulta imediato para a conversão numérica dos *links*. Da necessidade de acessar valores numéricos a partir de valores textuais surgiu o conceito de Função de Códigos, uma entidade capaz de realizar um mapeamento biunívoco entre um valor textual e um valor

numérico, ou seja, caso seja necessário o valor textual de um valor numérico ou vice-versa, basta consultar a Função de Códigos.

Da mesma forma que o Armazenador de Documentos acessa o Servidor de Termos (a Função de Códigos aplicada a termos), necessitou-se de um Armazenador de *Links* para acessar o Servidor de *URLs* (a Função de Códigos aplicada a *links*). Dessa necessidade realizaram-se as seguintes modificações na arquitetura de indexação: reconstruiu-se o Servidor de Termos, a fim de utilizar a Função de Códigos aplicada a termos; criou-se o Servidor de *URL*s e o Armazenador de *Link*s; removendo o módulo Atualizador de *Links* Temporários. A partir dessas alterações, é configurada a arquitetura atual da indexação do Engenho de Busca (Figura 6).

Figura 23 Arquitetura atual do módulo de indexação do Engenho de Busca Radix

Os seguintes módulos foram inseridos:

- Armazenador de *Links*: novo responsável pelo armazenamento dos *links* encontrados nos documentos;
- Servidor de *URL*s: "Função de Código" responsável pelo processamento numérico dos *links*;

Neste sistema proposto, o processo de indexação de documentos ocorre da seguinte forma:

O Robô solicita um *link* ao Servidor de *Links*; o Robô realiza o *download* do documento oriundo da *Web*; o Robô realiza uma análise no documento, a fim de extrair as

principais informações, gerando o seu índice; o Robô armazena índice do documento no Armazenador de Documentos.

O Armazenador de Documentos guarda as informações do índice em um meio persistente, utilizando o Servidor de Termos para resolver a informação numérica dos termos do índice e o Armazenador de *Links* para resolver os *links* que, por sua vez, utiliza o Servidor de *URL*s para armazenar a informação numérica dos *links* do índice.

5.2 AD (O Agente de Distribuição)

O AD é o elemento responsável pela conectividade dos Engenhos de Busca em ambientes de distribuição. Embora o AD trate todos os níveis de compartilhamento de esforços nas tarefas de um Engenho de Busca [seção Compartilhamento de recursos entre Engenhos de Busca], nessa dissertação, o Agente de Distribuição foi utilizado com a finalidade de compartilhar esforços de *crawler* e de indexação.

Um estudo sobre esse agente foi realizado, destacando os seus passos de funcionamento, seus requisitos não-funcionais e o experimento comprovante do atendimento a esses requisitos.

5.2.1 Passos do AD

O AD pode ser acoplado ao mecanismo de indexação de um EB, a fim de disponibilizar o compartilhamento de índices entre Engenhos de Busca (verificar Figura 24): o AD recebe os índices das páginas do Armazenador de Documentos; envia esses índices para outro AD de um Engenho de Busca colaborativo; e recebe os índices oriundos de outro AD e armazena-os no Armazenador de Documentos.

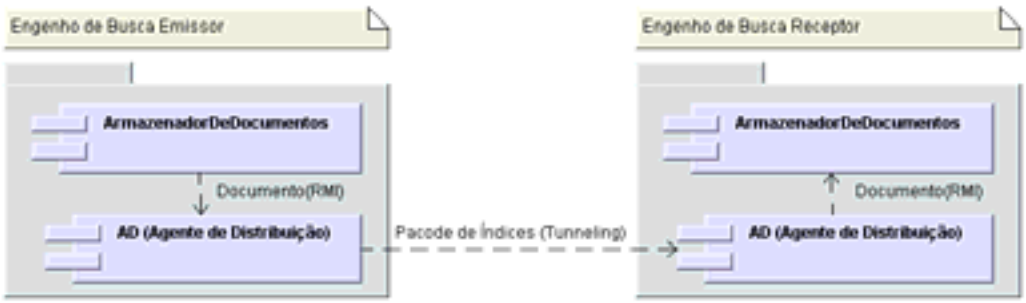

Figura 24 Passos realizados pelo AD na distribuição da indexação

A recepção dos índices de páginas pelo AD provenientes do Armazenador de Documentos

O AD possui acesso aos índices das páginas na medida em que eles são gerados, funcionando como um "observador" das inserções de páginas realizadas pelo Robô no Armazenador de Documentos [ver padrão *Observer* em **14**]. Em outras palavras, a medida que o Armazenador de Documentos recebe os índices provenientes do Robô, ele transmite estes índices ao AD, como ilustra a Figura 25.

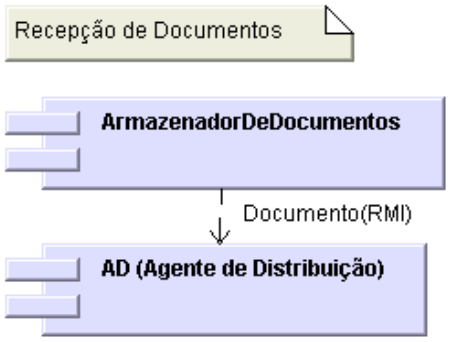

Figura 25 Recepção do índice de Documento pelo AD provenientes Armazenador de Documentos

De posse dos índices, o AD armazena-os temporariamente em um meio persistente, para, posteriormente, proporcionar envio desses índices.

Um aspecto importante a observar nesse envio de índices é em relação ao seu formato. Um formato proprietário para troca de informação prejudica o uso do mecanismo de envio de informação que o utiliza, pois obriga aplicações a utilizarem esse padrão. Caso o mecanismo utilize um formato aberto, aumentam as chances e as facilidades de adesão por conta de aplicações interessadas na utilização desse mecanismo. Assim, utilizou-se

XML como o protocolo para formatação de dados, a fim de possibilitar a abertura da informação a ser transmitida. O uso desse protocolo traz várias vantagens: as aplicações que processam os documentos XML compreendem a semântica das informações contidas no documento, por causa da DTD existente (ver descrição em XML do índice de uma página em XML no Apêndice); XML foi concebido como o protocolo para transmissão de documentos estruturados na *Web*, tornando-se o padrão para esse tipo de operação.

Com a finalidade de proporcionar economia no espaço de armazenamento dos índices e na utilização de recursos de rede, os dados foram compactados utilizando o algoritmo GZIP [61]. A escolha desse algoritmo ocorreu por ser amplamente utilizado e difundido entre aplicações e por gerar uma elevada taxa de compressão dos dados textuais a serem enviados.

O envio de índices de páginas

Como mostrado anteriormente sobre os estados do ambiente de Engenhos de Busca distribuídos [seção Definição do Framework], verifica-se que o envio de índices é exclusivo do ambiente distribuído. Nesse envio, ocorre a comunicação entre dois AD, um emissor e outro receptor dos índices (Figura 26).

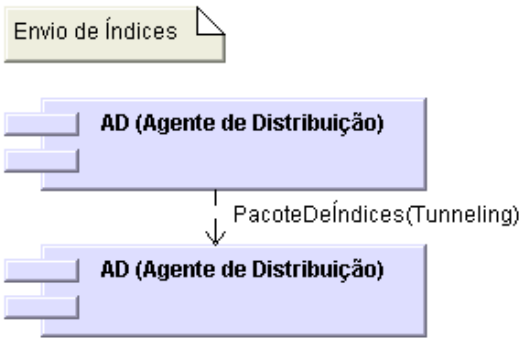

Figura 26 Comunicação entre ADs no envio de índices

No nível de aplicação, existe a preocupação em como a informação poderá ser transmitida. No ambiente da Internet, essa preocupação é direcionada para qual protocolo de aplicação deve-se utilizar para a transmissão de informações. Devido a restrições de segurança, a configuração dos protocolos de aplicações executados em rede varia bastante em cada servidor. Essas restrições são geralmente garantidas por

firewall, um conjunto de políticas (incluindo recursos de *hardware*, *software* e pessoal) que visam proteger e restringir acesso às informações disponíveis em uma rede de computadores.

Neste contexto, existem duas alternativas: criar um novo protocolo de aplicação ou utilizar um protocolo existente. No primeiro caso, o usuário da aplicação deveria obrigar mudanças na política de configuração de rede para a Internet, algo extremamente complicado. No segundo caso, aproveita-se uma infra-estrutura previamente aprovada e funcional.

Optou-se por utilizar um protocolo de aplicação já existente. A escolha foi HTTP [24], pois praticamente todos os servidores da Internet utilizam esse protocolo. Caso contrário, não participariam da *Web*, uma das principais aplicações da Internet. Resolvido o protocolo de aplicação a ser utilizado, resta identificar qual o método para encapsulamento dos dados a ser transmitido via HTTP. Esse método é o tulenamento [32].

Tunelamento é um método que utiliza a infra-estrutura de uma rede de trânsito qualquer para transferir dados entre duas outras redes, abstraindo toda a complexidade existente para lidar com a rede de trânsito. Os dados a serem transferidos podem ser quadros (ou pacotes) de outro protocolo. Ao invés de enviar um quadro como ele foi produzido pelo nó de origem, o protocolo de tunelamento encapsula o quadro em um cabeçalho adicional. Este cabeçalho fornece as informações necessárias para o roteamento de forma que os dados encapsulados possam trafegar através da rede intermediária.

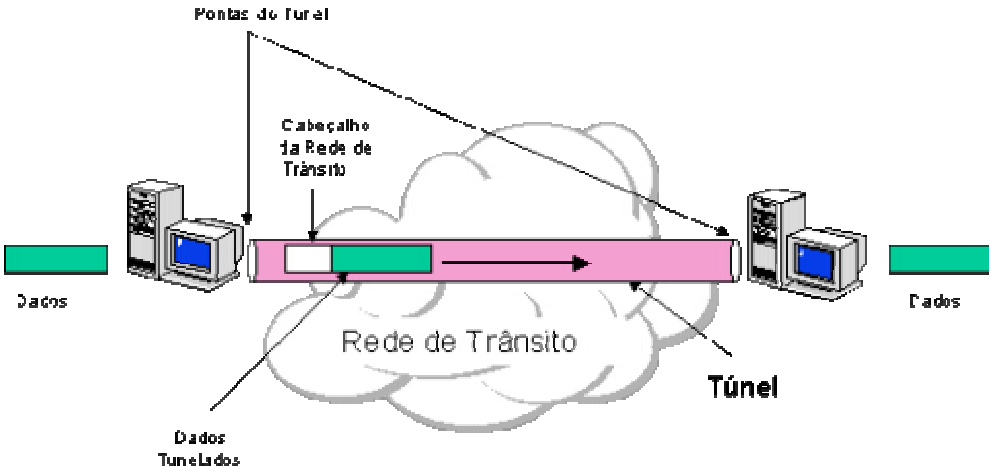

Figura 27 Funcionamento do método de tunelamento

O caminho lógico através do qual os dados encapsulados viajam pela rede de trânsito é chamado de túnel. Quando os dados encapsulados chegam ao seu destino, o cabeçalho adicional é retirado e os quadros originais seguem para o seu destino final. O processo de tunelamento envolve encapsulamento, transmissão e desencapsulamento dos pacotes.

Realizando uma analogia entre o tunelamento, mostrado na Figura 27, e o tunelamento do mecanismo de envio de índices de páginas, verifica-se que a rede de trânsito dos dados corresponde à Internet; os dados tunelados, aos índices das páginas; as pontas do túnel, aos Agentes de Distribuição; sendo o túnel gerado pelo protocolo HTTP, que também cria o cabeçalho adicional.

Assim, o meio para transmissão de dados utilizado no sistema é o **Tunelamento via HTTP**.

Vale salientar também uma opção de envio dos índices: a programação do horário de envio dos índices. Essa facilidade possibilita a utilização da rede em momentos de menor tráfego.

A recepção de índices de páginas pelo AD provenientes de outro AD

À medida que o AD recebe índices de páginas provenientes de um AD cooperante, esse agente processa-os da seguinte forma: descompactam-se os dados, utilizando o algoritmo GZIP [61]; realiza-se uma leitura do documento de índices no formato XML,

recriando os índices de forma a ser processado pelo EB; e insere esses índices no Armazenador de Documentos do EB receptor.

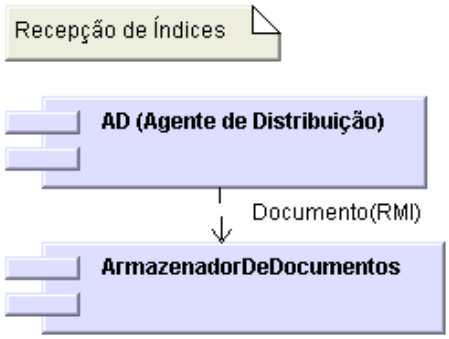

Figura 28 Recepção dos índices pelo AD e seu posterior armazenamento no Armazenador de Documentos

5.2.2 Requisitos Não-Funcionais do AD

O desenvolvimento do AD ocorreu seguindo várias exigências não-funcionais de software, adjetivando-o com várias características:

- É fracamente acoplado: de fato, esse módulo funciona como uma extensão do Engenho de Busca, não sendo essencial à execução. Em outras palavras, caso um Engenho não deseje mais participar do ambiente de distribuição, basta apenas desabilitar o funcionamento do AD;

- Não consome muitos recursos computacionais;

- Não prejudica a velocidade da indexação (quantidade de documentos indexados em um intervalo de tempo): dessa afirmação decorre que o AD não executa como um processo limitante de desempenho para o sistema;

- É facilmente instalado e configurável: todas as propriedades desse módulo são variáveis, até mesmo a possibilidade de sua execução, funcionando como um "*plugin*";

- É facilmente monitorável: a detecção de falhas em sua operação é identificada por um sistema de monitoramento, responsável por tomar as devidas medidas de recuperação a falhas;

- Funciona com a técnica "push". Considere dois sistemas, em particular dois Engenhos de Busca, com a seguinte característica: o sistema A detém informações úteis ao sistema B. A comunicação entre A e B pode ocorrer de duas formas. Na primeira possibilidade, B requisita essas informações ao sistema A (técnica de "pull"). Na segunda possibilidade, B é informado por A (técnica de "push"). A adoção da segunda alternativa no sistema de indexação distribuída torna-se mais interessante, pois as informações podem ser processadas na medida da sua geração.

5.2.3 Experimento de Custos Computacionais

O funcionamento não eficiente do AD de índices compromete o funcionamento do ambiente distribuído, pois através dele as informações das páginas indexadas são compartilhadas entre Engenhos de Busca colaborativos. Com o intuito de descrever a performance desse módulo, realizou-se o experimento a seguir.

Sejam as situações no processo de indexação: o caso A, um Engenho de Busca indexando páginas isoladamente; o caso B, um Engenho de Busca indexando páginas oriundas da colaboração de indexação de outro Engenho. Os passos necessários para realizar a indexação nos casos citados são ilustrados na Figura 29.

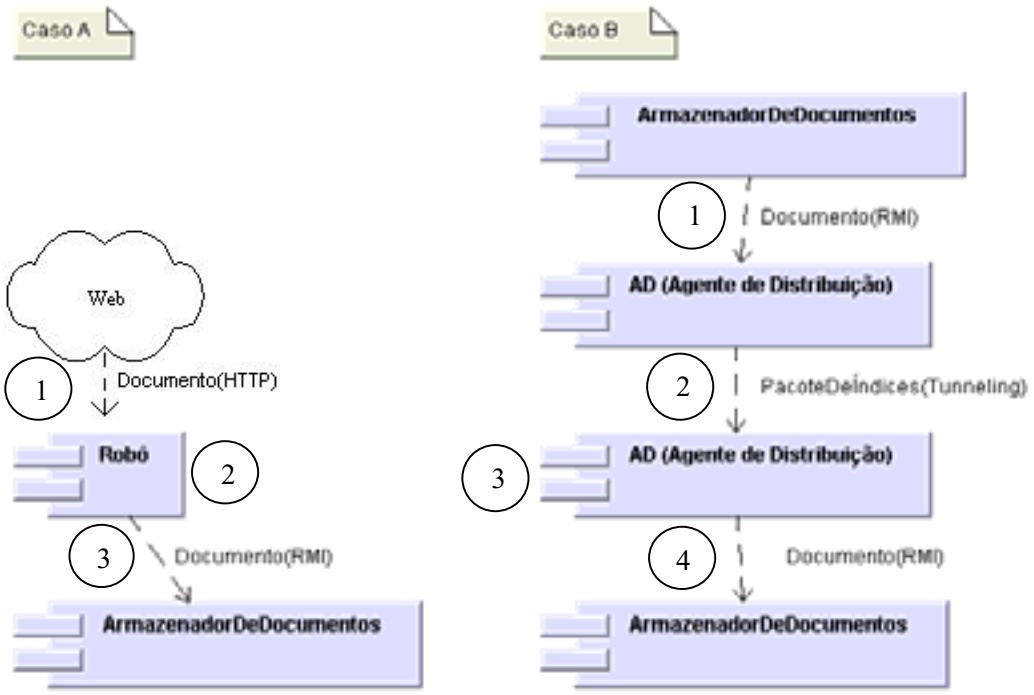

Figura 29 Processo de indexação em dois ambientes: centralizado (Caso A) e distribuído (Caso B)

A determinação de qual caso é mais eficiente no processo de indexação consiste em verificar quais passos são realizados para se completar o processo de indexação em cada caso, medindo os custos computacionais associados. Abaixo, as tarefas realizadas em cada caso do experimento (algumas das etapas do processo de indexação são similares, e outras, distintas em negrito):

Caso A

1 - Realizar o *download* das páginas;

2 - Realizar o *parsing* do documento, gerando o seu índice;

3 - Armazenar os índices das páginas no Armazenador de Documentos.

Caso B

1 - Formatar e enviar os índices das páginas;

2 - Realizar o *download* dos índices das páginas;

3 - Desempacotar os índices das páginas;

4 - Armazenar os índices das páginas no Armazenador de Documentos.

No caso A, é necessário realizar o *download* do documento da *Web,* e o *parsing* desse documento; no caso B, o Engenho receptor de índices necessita receber o os índices e desempacotá-los, existindo também um custo adicional de formatação dos índices dos documentos por parte do EB emissor de índices.

Uma vez identificados os passos do processo de indexação, sejam consumidos, para ambos os casos, os seguintes recursos computacionais:

- Memória Principal: não há utilização de grande quantidade de memória principal nos passos da indexação citados;

- Memória Secundária: no caso B, embora seja utilizada memória secundária para armazenar temporariamente os índices das páginas, a quantidade de espaço utilizado é pequena em relação à quantidade de espaço utilizado pelo Engenho de Busca;

- Processos: no caso B existe o acréscimo de dois processos, responsáveis pela execução dos ADs nos EB emissor e receptor, embora não comprometam a execução da indexação;

- Processador e Rede: esses dois componentes computacionais são bastante utilizados nos casos A e B.

Dessa forma, focaliza-se a atenção em dois recursos: processador e utilização de rede. Dadas as variáveis computacionais a serem consideradas e os passos diferenciadores dos casos, determinam-se as seguintes medições:

No caso A, mediu-se:

- Consumo de rede: consumo de banda passante para realizar o *download* das páginas;

- Tempo de Processamento: consumo de processamento para realizar o *parsing* do documento;

No caso B, mediu-se:

- Consumo de rede: consumo de banda passante para realizar o *download* dos índices;

- Tempo de Processamento: consumo de processamento para gerar o pacote de índices a ser enviado e consumo de processamento para desempacotá-los.

As páginas utilizadas na realização do experimento foram provenientes do serviço de consultas a notícias existentes no Engenho de Busca Radix [17]. Esse serviço indexa diariamente páginas das principais fontes de informação do Brasil. Essas fontes de informação, escolhidas por jornalistas, consistem em uma lista dos cem principais *sites* de informação do Brasil (ver lista dos *sites* no Apêndice B). A partir da página inicial desses *sites*, um processo chamado de Varredura de Sites realiza uma busca no grafo de *links* do documento até um certo nível de profundidade N. Como resultado, tem-se um conjunto de *URL*s com profundidade menor que N a partir da página inicial. Uma característica marcante desses *sites* é a alta taxa de atualização. Além disso, quanto menor o nível de profundidade, maior a chance de se encontrar páginas atualizadas. No estudo realizado, foi utilizada profundidade de nível três. Essa característica de alta atualização das páginas encontradas no serviço de Notícias determinou a escolha dessas páginas nos experimentos, pois foram processadas páginas distintas e com tamanhos diferentes em cada medição.

O experimento durou sessenta dias (ver resultados pormenorizados no Apêndice B) em uma máquina com as seguintes características técnicas: processador Pentium II 550 Mhz, 512 Mb de memória RAM, máquina virtual Java® 1.3 da SUN [16], obtendo os seguintes valores médios:

	Caso A	**Caso B**
Consumo de Rede (em bytes)	19603 (tamanho de uma página)	2556 (tamanho do índice de uma página)
Tempo de Processamento (em milissegundos)	106 (tempo para se realizar o *parsing* de uma página)	53 + 263 = 316 (tempo para inserir o índice em um pacote de índices acrescido do tempo para extrair o índice de pacote de índices)

Tabela 2 Valores médios encontrados durante o experimento

Dos valores obtidos, seguem algumas considerações:

- Consumo de Rede: ocorreu um ganho substancial no consumo de banda passante no ambiente distribuído. O consumo diminuiu, em média, de 19603 bytes para 2556, correspondendo a uma economia de 86,96 %;

- Tempo de Processamento: houve um acréscimo da utilização de processamento no ambiente distribuído (Caso B). O tempo de processamento aumentou, em média, 2.99 vezes, passando de 53 para 263 milissegundos. Além de existir um *overhead* de 53 milissegundos de processamento por parte do Engenho que envia os índices.

Análise de Custos e Benefícios

Surge a necessidade de descobrir qual dos recursos computacionais é mais importante nesse ambiente: poder de processamento ou consumo de banda passante. Segue a seguinte análise:

- Consumo de rede: o custo atual é de R$ 26.000,00 mensais por uma conexão de 2 megabits por segundo através da operadora Embratel [15].

- Tempo de processamento: o custo atual é de R$ 4.000,00 no caso da máquina utilizada no experimento.

Seja considerada a suposição: deseja-se manter o mesmo poder de processamento entre o caso A e o caso B (ambiente distribuído). No caso A, não há necessidade de comprar máquina, mas paga-se a quantia de R$ 26.000,00 mensais pela conectividade. No caso B, deve-se adquirir 2.99 máquinas, gastando R$11.960,00, além do custo associado à depreciação do equipamento, valor de 50% em 18 (dezoito) meses. Entretanto, economiza-se 86.96% de consumo de banda passante e, considerando a proporção entre o valor cobrado e o uso da conexão de rede, pagar-se-ia apenas R$ 3.390,40. Assim, calcula-se a previsão do gasto em dezoito meses em cada caso:

	Caso A:	**Caso B:**
Máquina	Não há gasto adicional	2.99 X 4.000 = 11.960,00 11.960,00 + 50% = 17.940,00
Rede	26.000 X 18 = R$ 468.000,00	3.390,40 X 18 = 61.027,20
Total	R$ 468.000,00	R$ 78.967,20

Tabela 3 Previsão de valores de custos computacionais

Em dezoito meses de operação, o gasto extra com processamento seria pago pela economia no custo com banda passante no Caso B, significando uma economia de 83.13%.

Uma medição foi omitida na realização desse experimento: o tempo de *download* dos dados. No caso A, esse tempo corresponde ao tempo necessário no *download* dos documentos e, no caso B, o tempo necessário no *download* dos índices. Para a realização dessas medidas, seria necessário que esse processo fosse realizado através da Internet. Como o AD emissor de índices e o AD receptor de índices estão presentes na mesma rede local, foi impossível realizar o cálculo do tempo no Caso B. Muito embora não realizadas essas medições, torna-se claro que o tempo necessário na primeira medição (caso A) seria maior que na segunda medição (caso B), pois existe uma quantidade significativamente maior de dados a serem transmitidas.

5.3 Estudo de Caso

O Agente de Distribuição está em operação no *software* de indexação e busca (conhecido como *SiteSearch*) da empresa Radix [17]. Esse *software* é capaz de manter uma base de consultas para um determinado *site*. O AD é responsável por enviar os índices dos documentos das páginas do *site* para outro Engenho de Busca que possua interseção com o seu escopo.

Atualmente, o portal Radix [17] possui um Engenho de Busca responsável por manter um índice das páginas contidas no escopo ".br". A operação desse Engenho é realizada por um conjunto de máquinas executando ininterruptamente. Esse índice possui dois anos de funcionamento, contendo mais de doze milhões de documentos, sendo atualizado a cada oitenta dias. A partir do escopo ".br", extraem-se vários escopos nele contidos, para a realização desse estudo de caso:

Escopo	Cobertura (em número de Documentos)	Tempo para Atualização (em dias)
".cesar.org.br"	131	80
".cin.ufpe.br"	15.590	80
".ufpe.br"	18.510	80
".hpg.com.br"	383.407	80

Tabela 4 Valores de cobertura e tempo de atualização para alguns escopo extraídos no Engenho de Busca Radix

Paralelamente ao funcionamento do Engenho no Radix, o *software SiteSearch* foi instalado nos seguintes *sites*: hpG [47], CESAR [10], UFPE [67] e Cin [11]. Esses Engenhos proporcionaram a criação de um ambiente distribuído, utilizado para compartilhamento de esforços de *crawler* e indexação. Vale salientar as seguintes relações entre os escopos desses Engenhos: todos possuem interseção com o escopo do Radix, possibilitando a colaboração entre eles com o Engenho Radix; e o escopo do CIn possui interseção com o escopo da UFPE, possibilitando a colaboração entre os Engenhos CIn e UFPE.

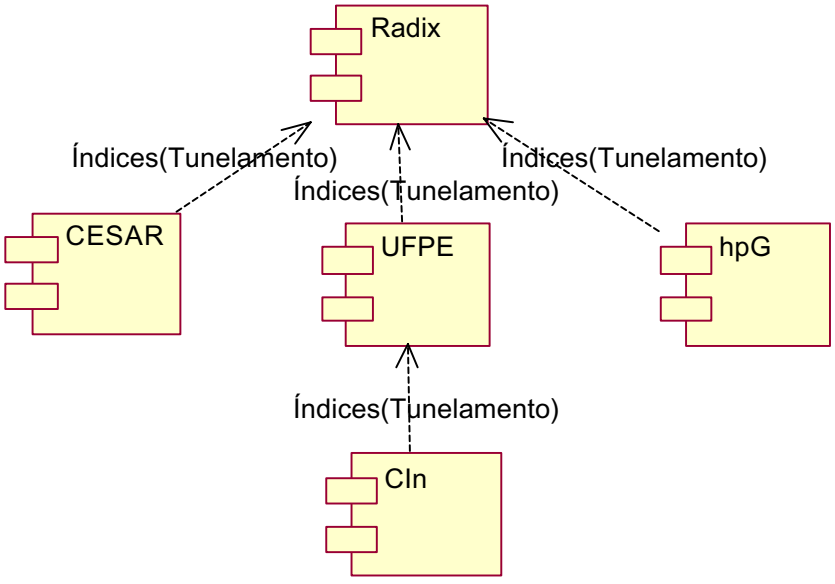

Figura 30 Ambiente de Distribuição entre Engenhos de Busca

A seguinte constatação determinou a forma de comunicação entre os integrantes do ambiente distribuído: Engenhos especializados, ou seja, cujo escopo é mais específico,

são mais eficientes para manter suas bases de índices com boa cobertura e atualidade, pois tendem a implementar eficientes estratégias de visitas de páginas. Por exemplo, no caso do portal hpG, o seu Engenho de Busca é notificado diariamente sobre quais páginas foram modificadas, inseridas ou removidas. Entretanto, devido à distribuição, essa informação pode ser repassada a Engenhos com escopos mais genéricos, como o do portal Radix.

De fato, ficou acordada a seguinte colaboração de indexação entre os Engenhos: Engenho CIn colabora índices com o Engenho UFPE que, juntamente com os Engenhos CESAR e hpG, colaboram índices com o Engenho Radix. Assim, não há mais necessidade dos Robôs do Engenho UFPE visitarem páginas no *site* CIn, assim como os Robôs do Engenho Radix não mais visitarão páginas nos *sites* hpG, CESAR, UFPE e CIn , sendo este último por transitividade.

Após dois meses de operação, foram obtidos os seguintes valores de cobertura e atualidade [ver definições em de cobertura e atualidade na seção 2.3] dos escopos da Tabela 5 no Engenho Radix:

Engenho	**Cobertura**	**Atualidade**
CESAR	252	Diária
Cin	17.995	Diária
UFPE	22.212	Diária
HpG	1031.603	Diária

Tabela 5 Valores de cobertura e tempo de atualização para alguns escopos extraídos no Engenho de Busca Radix após distribuição

Após a instalação do *SiteSearch* no portal hpG, foram verificadas as seguintes medidas:

- Cobertura: antes de instalação do SiteSearch no portal hpG, existiam 383.407 documentos do escopo hpG na base de índices do portal Radix, operando há dois anos. Atualmente, após dois meses de colaboração, existem mais de um milhão de documentos, significando um aumento de mais de 260 % na cobertura que o Engenho de Busca do Radix possui no escopo "hpG.com.br". Já a cobertura do site CIn passou de 15.590 documentos para 17.995, significando um aumento de 15 % na cobertura no mesmo intervalo de tempo;

- Atualidade: a base de índices do Radix demora cerca de oitenta dias para ser totalmente atualizada. Atualmente, os índices das páginas no Engenho do CIn, CESAR e hpG são atualizados diariamente. Com a colaboração de indexação existente entre estes EBs e o Radix, esses índices são enviados para o Engenho de Busca Radix, que não necessita mais indexar páginas dos escopos dos EBs.

5.4 Desenvolvimento do protótipo

O protótipo de Engenho de Busca foi elaborado pela equipe de desenvolvimento da empresa Radix[17]. É enumerado um conjunto de ferramentas, técnicas, linguagens e metodologias, ou seja, as principais tecnologias utilizadas como padrão no desenvolvimento de sistemas por esta empresa.

- Java [64]: foi a plataforma de programação utilizada. Vários fatores influenciaram a adoção dessa linguagem: é orientada a objetos, possui bibliotecas apropriadas a utilização da Internet, possui uso público e gratuito, embora a patente pertença à empresa Sun Microsystems [16] .

- Objetos Distribuídos: a tecnologia de objetos distribuídos nasceu da união de duas outras tecnologias: a orientação a objetos com a tradicional arquitetura cliente/servidor, a fim de unir as vantagens de cada uma para o desenvolvimento de aplicações distribuídas. Objetos distribuídos baseiam-se no conceito de *middleware*. O *middleware* é uma categoria de *software* que permite a conexão entre aplicações distribuídas buscando transparência nas comunicações, ou seja, abstraindo complexidades sobre protocolos de rede, sistemas operacionais ou linguagens de implementação entre as aplicações que trocam informações. [20] Na construção de aplicações distribuídas com objetos distribuídos, as aplicações são representadas por conjuntos de objetos, entidades que encapsulam e servem como unidades básicas de informação para estas aplicações. Tais objetos interagem entre si através da troca de mensagens, no estilo pedido/resposta, tradicional da arquitetura cliente/servidor, porém sem a necessidade de objetos estarem localizados em uma mesma aplicação, nem em uma única máquina, deixando as tarefas de localização dos objetos e tráfego de requisições para um

middleware. A implementação de objetos distribuídos utiliza foi RMI [49]. RMI (*Remote Method Invocation*) é o mecanismo que a linguagem Java possui para permitir a invocação de métodos de objetos remotos. Após um estudo das tecnologias CORBA, JINI e RMI, esta última foi adotada na elaboração dos experimentos. A escolha por RMI como a plataforma de objetos distribuídos decorreu por vários fatores: a abstração de objetos que a linguagem possibilita; o conhecimento existente sobre essa tecnologia na empresa onde o sistema foi desenvolvido; e dependência exclusiva da linguagem Java.

- *Servlets* [55]: é a solução da tecnologia Java que provê um mecanismo consistente para estender a funcionalidade de um servidor *Web*. Um *Servlet* pode ser imaginado como uma *applet* executando na máquina servidora, excluindo-se a *interface* com o usuário. *Servlets* provêm um método independente de plataforma e de servidores para o desenvolvimento de aplicações *web*. *Servlets* equivalem à tecnologia CGI (*Common Gateway Interface*), porém eliminando algumas limitações: *Servlets* possuem melhor performance, pois são invocados como processos leves (*threads)*, além de manter objetos em memória após várias invocações da aplicação.

- XML [18]: é uma linguagem de "*tags*" ou "*markup*" para documentos que contêm informação estruturada proposta pelo Consórcio "*World-Wide-Web*" (W3C) [66]. Informação estruturada contém contexto (palavras, figuras, etc.) e alguma indicação de qual regra se aplica o contexto.Em HTML [23], a semântica do "*tag*" e o conjunto de "*tags*" são fixos. Um "*tag*" <h1> é sempre um cabeçalho de primeiro nível e um "*tag*" <ati.product.code> não possui significado. O consórcio W3C, em conjunto com vendedores de navegadores, estão em constante trabalho para estender a definição de HTML, a fim de permitir novas "*tags*" para atender à mudança de tecnologia e oferecer variação de apresentação ("*stylesheets*") para *Web*. Dessa forma, as mudanças no protocolo HTML visão a adequada implementação dos navegadores. XML não especifica semântica nem um conjunto de "*tags*". De fato, XML é uma "meta-linguagem" para uma linguagem de descrição de "*markup*". Em outras palavras, XML provê uma facilidade para definir "*tags*" e relacionamento estrutural entre eles. Já que não há um conjunto de "*tags*" predeterminado, não pode haver um prévio conceito

semântico para um conjunto de "*tags*". Toda a semântica de um documento XML é definida pelas aplicações que o processa.

- HTTP [24]: (*Hypertext Transfer Protocol*) é um protocolo de aplicação utilizado por sistemas de informação distribuídos e colaborativos. HTTP foi inicialmente utilizado pelo consórcio "*World-Wide-Web*" em 1990. HTTP é utilizado também como um protocolo genérico para comunicação entre agentes, *proxies* e *gateways* existentes na Internet, além de ser um meio de acesso a recursos para aplicações.

- PIM [42]: é uma metodologia para desenvolvimento de software cuja finalidade é adicionar progressivamente distribuição, persistência e concorrência a aplicações. Mais especificamente, essa metodologia abrange o padrão de desenvolvimento de objetos distribuídos em Java definido em [68]. Esse padrão permitiu o desenvolvimento de um *software* modular, extensível e reusável.

- *Extreme Programming* [19]: é uma metodologia para desenvolvimento de software criada em 1990 por Kent Beck. *Extreme Programming,* ou resumidamente XP, foi desenvolvida para disponibilizar o *software* para os usuários na medida de suas necessidades. XP está baseado em vários preceitos: programadores XP comunicam-se freqüentemente com os usuários do sistema e com outros programadores, mantêm o *design* do sistema simples, recebem *feedback* dos usuários e testam exaustivamente o *software*. O *software* é geralmente disponibilizado para os usuários o quanto antes. Além disso, os programadores XP são motivados a responder por mudanças na tecnologia ou nos requisitos do sistema. Embora XP não seja recomendado para equipes de trabalho com mais de dez profissionais, essa restrição não foi relevante na utilização desse método, pois foram utilizadas equipes menores que dez profissionais. Duas técnicas se destacam em XP: Pair Programming [46] e Refactoring [40]. Pair Programming é uma técnica que define a programação aos pares. Muitos são os benefícios de se utilizar essa técnica: encontra-se uma quantidade de erros de programação durante a época de escrita do código; o *design* do código fica melhor e a quantidade de linhas de programação menor, devido as discussões em pares; uma quantidade maior de problemas é resolvido; os programadores aprendem

sobre o *software* desenvolvido e sobre desenvolvimento de sistemas; aumenta-se o fluxo de informação e a dinâmica do grupo; o grupo sente-se mais satisfeito no trabalho. O custo para se obter todas as vantagens citadas acima é cerca de 15% comparando-se o trabalho individual [2]. Esse custo adicional é compensado por um período menor de testes e pela garantia de maior qualidade do sistema. Foi constatada uma satisfação em trabalhar com *pair programming*. Refactoring é uma técnica que permite a alteração do código de um *software* de forma a não alterar o seu comportamento externo, promovendo uma reestruturação interna do sistema. Trata-se de um processo disciplinado para organizar o código, minimizando as chances de ser introduzir erros. A aplicação de *refactoring* sobre um sistema melhora o *design* do código fonte escrito. A fim de inserir o Agente de Distribuição no Engenho de Busca de forma modular, foi necessária a reestruturação de parte do código fonte. Como era desejado reescrever o código sem alterar a sua funcionalidade e sem inserir erros, foi utilizado o *refactoring* na realização dessa tarefa.

5.5 Conclusões do capítulo

Nesse capítulo foi apresentada uma aplicação do ambiente de distribuição proposto por esse trabalho.

No experimento realizado, aferiu-se a eficiência do mecanismo de colaboração de *crawling* e indexação. Com ele, foi otimizado o uso do recurso computacional mais onerosa, a banda passante. Além de existir um menor consumo, o envio de informações é realizado em horário conveniente, ou seja, de menor utilização da rede.

Executando em um caso real, a tarefa de indexação do Engenho de Busca do portal Radix foi comparada antes e depois de participar de um ambiente distribuído. Quando participando de um ambiente distribuído, recebendo colaboração de indexação de vários Engenhos de Busca, verificou-se uma melhora na cobertura e na taxa de atualização (diminuiu de oitenta dias para diária) da Base de Índices do Engenho Radix com relação ao escopo dos Engenhos colaboradores.

Na construção do Engenho de Busca, um complexo sistema de informação não-trivial, um vasto amparo tecnológico foi utilizado: ferramentas, técnicas e metodologias de desenvolvimento de software. A correta aplicação dessas tecnologias proporcionou o desenvolvimento de um sistema modular.

Capítulo 6 Conclusão

Essa dissertação fornece uma solução para o problema de escalabilidade em Engenhos de Busca da Internet através de aplicações de técnicas de distribuição.

Os Engenhos de Busca são um dos principais meios de auxílio aos usuários na tarefa de realizar busca por informações na *Web*. O Engenho de Busca deve manter em sua Base de Índices a maior quantidade possível de páginas relevantes às requisições dos usuários, ou seja, uma alta taxa de cobertura, além de manter esses índices atualizados com os seus valores reais, ou seja, uma alta taxa de atualidade.

Nessa dissertação, um estudo foi realizado sobre a área de Recuperação de Informações, cujos esforços focalizaram sobre o funcionamento do mecanismo de *crawling* e indexação do Engenho de Busca.

Foi elaborado também um estudo sobre sistemas distribuídos. Sabe-se que a utilização de técnicas de distribuição é requisito não-funcional das aplicações, ou seja, não se torna, a princípio, requisito exigido pelos usuários de sistemas de software. Entretanto, devido a várias características inerentes da distribuição, tais como compartilhamento de recursos, abertura, concorrência, escalabilidade, tolerância a falhas e transparência, a sua utilização torna-se essencial para o funcionamento adequado dos sistemas computacionais, principalmente na Internet, um ambiente naturalmente distribuído.

Sobre essas duas áreas de conhecimento, formulou-se um ambiente de distribuição para Engenhos de Busca, que apresenta as seguintes características: fácil possibilidade de inserção, adaptação e remoção de um Engenho de Busca nesse ambiente; é extensível e escalável; e possibilita compartilhamento de recursos, escalabilidade e abertura.

Nesse ambiente de distribuição, há a possibilidade de consultar Engenhos de Busca com escopos diferenciados. Por exemplo, são consultados Engenhos com domínio específico como a busca no portal hpG ou são realizadas consultas em um portal mais genérico, como o portal Radix.

Embora a arquitetura de distribuição possibilite vários níveis de colaboração de recursos, houve o interesse em distribuir os processos de *crawler* e indexação, através do mecanismo de envio de índices das páginas. Nos experimentos realizados, esse mecanismo torna os processos computacionalmente mais eficientes, economizando significantemente o recurso mais importante na Internet, a banda passante.

Com a finalidade de otimizar o envio de índices, envia-se o índice de várias páginas conjuntamente. Verificou-se que, quanto maior o número de índices das páginas a ser

enviada em uma transmissão de dados, maior a economia de banda passante, pois era atingida uma melhor taxa de compactação dos índices. Entretanto, o tamanho do "pacote de índices" não pode ser excessivamente grande, pois fatalmente provocaria fragmentação de pacotes no nível de camada de enlace da rede. Na Internet, existem várias soluções na camada de enlace. Uma interessante extensão desse trabalho seria descobrir um tamanho ótimo para o pacote a ser transmitido. Caso descoberto esse valor, ficaria fácil determinar a quantidade ideal de índices a serem enviados em uma transmissão de dados.

Com relação à abertura de dados, a utilização de XML para transmissão de índices é um passo importante, pois padroniza a formatação das informações dos índices. Seria interessante também a existência de um padrão para especificar as principais estruturas de um documento. Com ele, não haveria a necessidade de aplicações que processam documentos (por exemplo, *crawler* de Engenhos de Busca) realizarem o *download* de documento, mas apenas de suas principais informações. Atualmente, não existem padrões para descrever as "meta-informações" de um documento HTML, apesar de existirem esforços relacionados [22].

O desenvolvimento de um protótipo executando em um ambiente de distribuição visando a distribuição de *crawling* e indexação entre Engenhos na *Web* favoreceu vários aprendizados: o estudo das principais tecnologias ligadas à distribuição de objetos, tais como RMI, CORBA e JINI; o contato com modernas metodologias de desenvolvimento de *software*, tais como *extreme programming*, *pair programming* e *refactory;* um melhor entendimento da arquitetura do Engenho e das várias possibilidades de configuração de sua operação, gerando um sistema modular.

O ambiente de distribuição, quando comparado a um ambiente tradicional, mostrou vários ganhos, possibilitando um aumento significativo na cobertura e na atualidade da Base de Índices do Engenho. Esses resultados validam e encorajam a utilização do ambiente distribuído proposto nessa dissertação.

No protótipo desenvolvido, o Agente de Distribuição auxilia apenas a colaboração de esforço de *crawler* e indexação. Como trabalho futuro, seria interessante implementar o mecanismo de colaboração de busca descrito no ambiente de distribuição.

Esse trabalho concentrou-se em utilizar técnicas de distribuição para desenvolver um ambiente colaborativo entre Engenhos de Busca. Entretanto, a utilização dessas técnicas aplica-se também à operação do próprio Engenho, otimizando o funcionamento de cada um dos seus componentes.

Na elaboração do estudo de caso, cinco Engenhos foram utilizados, aferindo vantagens para cada um deles. De fato, não foi determinado quantos Engenhos podem participar do ambiente distribuído sem existir desvantagens para nenhum deles. Teoricamente, esse valor é supostamente infinito. Entretanto, seria um trabalho interessante calcular o valor prático, normalmente não-infinito, sobre determinada configuração do ambiente. Além disso, alterações no ambiente proposto poderiam ser introduzidas a fim de aumentar o valor prático encontrado.

Várias áreas de conhecimento da computação foram pesquisadas, tratando-se de uma dissertação multidisciplinar. Embora a dissertação focaliza principalmente em Recuperação de Informação e Sistemas Distribuídos, foi necessário o conhecimento sobre outros assuntos, como Redes de Computadores e Banco de Dados.

A realização desse trabalho ocorreu em conjunto com a iniciativa privada, causando amadurecimento profissional, pois, embora inicialmente árduo, foi de grande aprendizado conciliar os interesses acadêmicos e os interesses do mercado. Além disso, uma infra-estrutura para trabalhar com problemas reais estava disponível.

De forma geral, o processo da pesquisa é desafiante, porque muitas dificuldades são encontradas, desde a aprendizagem no processo de pesquisa até a organização pessoal. Além disso, o trabalho tornou-se gratificante, pois foram enfrentados problemas reais e soluções foram desenvolvidas e utilizadas nesse ambiente.

Bibliografia

As referências bibliográficas que aparecem com *links* (*URL*s) para a *Web* foram testadas à data da elaboração desta dissertação. Alguns desses *links* podem mudar ou deixar de serem acessíveis sem prévia notificação.

[1] A. Arasu, J. Cho, H. Garcia-Molina, A. Paepcke and S. Raghavan. Searching the *Web*.

[2] A. Cockburn and L. Williams. The Costs and Benefits of Pair Programming. <http://collaboration.csc.ncsu.edu/laurie/Papers/XPSardinia.PDF>

[3] A. K. A. Medeiros. Mecanismo de Interação para um Modelo de Redes de Petri Orientado a Objetos. Dissertação de Mestrado, Departamento de Informática, Campus II, Universidade Federal de Paraíba, Campina Grande, agosto de 2000.

[4] Advanced Micro Devices, Inc. (AMD) <http://*www*.amd.com>

[5] B. Brewington and G. Cybenko. How dynamic is the *web*?

[6] B. Frakes, Ricardo Baeza. Information Retrieval Data Structures & Algorithms, Prentice Hall, Upper Saddle River, New Jersey 1992.

[7] Bruce Eckel. Thinking in Java. Prentice Hall, Upper Saddle River, New Jersey, 1998.

[8] C. S. Horstmann and G. Cornell. Core Java 1.1 Volume II Advanced Features, Prentice Hall, December 1998.

[9] C. S. Horstmann and G. Cornell. Core Java 1.2 Volume 1 : Fundamentals 4/e, Prentice Hall, January 1999.

[10] Centro de Estudos e Sistemas Avançados do Recife. <http://*www*.cesar.org.br>

[11] Centro de Informática da Universidade Federal de Pernambuco. <http://*www*.cin.ufpe.br>

[12] Common Object Request Broker Architecture (CORBA). <http://*www*.corba.org>

[13] Distributed Component Object Model (DCOM). <http://*www*.microsoft.com/com/tech/dcom.asp>

[14] E. Gamma, R. Helm, R. Johnson and J. Vlissides. Designing Patterns – elements of reusable Object-Oriented Software. Addison-Wesley Publishing Company, 1998.

[15] Empresa Embratel. <http://*www*.embratel.com.br>

[16] Empresa Sun Microsystems. <http://*www*.sun.com>

[17] Engenho de Busca Radix. <http://*www*.radix.com>

[18] Extensible Markup Language (XML). http://*www*.w3.org/XML

[19] Extreme Programming. <http://*www*.extremeprogramming.org>

[20] F. A. M. Trinta. Arquiteturas Distribuídas para Co-Autoria Cooperativa de Aulas na Internet. Dissertação de Mestrado, Centro de Informática, Universidade Federal de Pernambuco, Brasil, 2000.

[21] G. Coulouris, J. Dollimore and T. Kindberg. Distributed Systems – Concepts and Design, 2ª edição, editora Addison-Wesley, 1994.

[22] Harvest Summary Object Interchange Format (SOIF). <http://*www*.landfield.com/rfcs/rfc2656.html>

[23] HyperText Markup Language HomePage (HTML). <ttp://*www*.w3.org/MarkUp/>

[24] HyperText Transfer Protocol (HTTP). <http://*www*.w3c.org/Protocols>

[25] J. A. S. Cardoza. Filtragem e Recomendação de Documentos na *Web:* Uma Abordagem usando Java. Dissertação de Mestrado, Centro de Informática, Universidade Federal de Pernambuco, Brasil, 1998.

[26] J. Cardoza, P. F. Gonçalves, A. Lima, L. Valadares, C. Tercero, S. L. Meira, A. C. Salgado and F. Q. B. Silva. A Framework for Developing Information Indexing, Filtering and Searching Tools in the Internet.

[27] J. Cho and H. G. Molina. Synchronizing a database to Improve Freshness.

[28] J. Cho and H. G. Molina. The Evolution of the *Web* and Implications for Incremental Crawler.

[29] J. L. Peterson. Petri Nets. Department of Computer Sciences, The University of Texas, Austin, Texas, 1977.

[30] JINI HomePage <http://*www*.sun.com/jini>

[31] K. Jensen. Coloured Petri Nets – Basic Concepts, Analysis Methods and Practical Use. Volume 1, Editora Springer-Verlag, 1992.

[32] *Link*s sobre definição de tulenamento. <http://*www*.rnp.br/newsgen/9811/vpn.shtml> and <http://*www*.di.ufpe.br/~flash/ais98/vpn/Vpn.html>

[33] *Link* sobre pesquisa que reflete o tamanho da *Web*. <http://completeplanet.com/Tutorials/Deep*Web*/contents04.asp>

[34] *Links* sobre pesquisa que reflete o tamanho dos Engenhos de Busca. <http://searchenginewatch.com/reports/sizes.html>

[35] *Links* sobre pesquisas que refletem o uso da Internet. <http://*www*.nua.ie/surveys/index.cgi?f=VS&art_id=905356447&rel=true> and http://*www*.searchenginewatch.com/sereport/01/02-keen.html

[36] M. Kobayashi and K. Takeda. Information Retrieval on the *Web*, 2000. <http://citeseer.nj.nec.com/kobayashi00information.html>

[37] *Links* sobre pesquisas que refletem o uso dos Engenhos de Busca. <http://*www*.searchenginewatch.com/sereport/00/06-realnames.html> and <http://*www*.searchenginewatch.com/reports/gvu.html>

[38] *Links* sobre pesquisas sobre tempo de resposta para sistemas *Web*. <http://*www*.useit.com/alertbox/9703a.html> and <http://*www*.useit.com/papers/responsetime.html>

[39] M. Baker, J. Kiniry, N. Minar and R. Resnich. The Best of Distributed Objects.

[40] M. Fowler. Refactoring: Improving the Design of Existing Code. Published by Addison Wesley, 1999.

[41] Mic. C. Bowman, Peter B. Danzig, Darren R. Hardy, Udi Manber and Michael F. Schwartz. The Harvest Information Discovery and Access System. <http://*www*.ncsa.uiuc.edu/SDG/IT94/Proceedings/Searching/schwartz.harvest/schwartz.harvest.html>

[42] P. Borba, S. Araújo, H. Bezerra, M. Lima and S. Soares. Progressive Implementation of Distributed Java Applications.

[43] P.F. Gonçalves, S. L. Meira and A. C. Salgado] Digital Neighbourhoods: Partitioning the *Web* for Information Indexing and Searching". In Olivé, A., Pastor, J.A.(Eds.) Springer Verlag, Lecture Notes in Computer Science 1250, pp. 289-302, June 1997.

[44] P. F. Gonçalves and S. L. Meira. The Fundamental Limits of Indexing the World-Wide *Web*: A Preliminary Discussion.

[45] P. F. Gonçalves, S. L. Meira and A. C. Salgado. Bright: A Distributed System for *Web* Information Indexing and Searching.

[46] Pair Programming. <http://*www*.pairprogramming.com>

[47] Portal Home Page Grátis (hpG). <http://*www*.hpg.com.br>

[48] Projeto piloto para distribuição de indexação CHIC-PILOT. <http://*www*.terena.nl/projects/chic-pilot/>

[49] Remote Method Invocation (RMI). <http://*www*.java.sun.com/rmi>

[50] RFC CIP Architecture. <ftp://ftp.ietf.org/rfc/rfc2651.txt>

[51] Ricardo Baeza and Berthier Ribeiro Neto. Modern Information Retrieval, ACM Press, New York, 1998.

[52] Robert Korfrage. Information Storage and Retrieval, Wiley Computer Publishing, USA, 1997.

[53] S. Brin and L. Page. The Anatomy of a Large-Scale Hypertextual *Web* Search Engine.

[54] S. Russel and P. Norvig. Artificial Intelligence – A Modern Approach. Prentice Hall Series in Artificial Intelligence, New Jersey, 1995.

[55] Servlet Home Page <http://java.sun.com/products/servlet/index.html>

[56] Site de busca Cadê?. <http://*www*.cade.com>

[57] Site de busca Google. <http://*www*.google.com>

[58] Site de busca Altavista. <http://*www*.altavista.com>

[59] Site de busca Yahoo!. <http://*www*.yahoo.com>

[60] Site do Jornal do Comércio. <http://*www*.jc.com.br>

[61] The GZIP home page. <http://www.gzip.org>

[62] The Linux Home Page. <http://*www*.linux.org>

[63] The Object Management Group. <http://*www*.omg.org>

[64] The Source for Java(TM) Technology. <http://*www*.java.sun.com>

[65] The Whois++ Testbed Project. <http://*www*.ucdavis.edu/whoisplus/index.html>

[66] The World Wide *Web* Consortium. <http://*www*.w3.org>

[67] Universidade Federal de Pernambuco. <http://*www*.ufpe.br>

[68] V. R. Alves. Definition of a method for the implementation of distributed Java applications. Dissertação de Mestrado, Centro de Informática, Universidade Federal de Pernambuco, Brasil, 2001.

Apêndice A

Exemplo e Template do Documento de Índices

O índice de um documento representa uma estruturação textual das principais informações contidas no documento. Por exemplo, o índice de uma página *Web* é composto pelo endereço (*url*), título, data de visitação, data de modificação, tamanho, parágrafo inicial, conjunto de palavras e conjunto de *links*. A construção da Base de Índices de um Engenho de Busca é realizada a partir do índice de um documento. Índices de documentos distintos determinam um pacote de índices de documentos. Nesta dissertação, o protocolo XML foi utilizado para formatar o documento de pacote de índices. Segue um exemplo de pacote de índices, contendo páginas cujos endereços são: http://*www*.cin.ufpe.br e http://*www*.radix.com.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE packet [
<!ELEMENT packet (webview, page*, URL*)>
<!ELEMENT webview (CDATA) >
<!ELEMENT page (address, visitedDate, modifiedDate, size, title, paragraph, word*, link*) >
<!ELEMENT address (CDATA) >
<!ELEMENT visitedDate (CDATA) >
<!ELEMENT modifiedDate (CDATA) >
<!ELEMENT size (CDATA) >
<!ELEMENT title (CDATA) >
<!ELEMENT paragraph (CDATA) >
<!ELEMENT word (name,occurrency,positions)>
<!ELEMENT name (CDATA) >
<!ELEMENT occurrency (CDATA) >
<!ELEMENT positions (CDATA) >
<!ELEMENT link (CDATA) >
<!ELEMENT URL (CDATA) >
]>
<packet>
 <webview>.br</webview>
 <page>
  <address>http%3A%2F%2Fwww.cin.ufpe.br</address>
  <visitedDate>1000240397000</visitedDate>
  <modifiedDate>999781132000</modifiedDate>
  <size>6192</size>
  <title>Centro+de+Inform%E1tica</title>
  <paragraph>Inscri%E7%E3o%0Ana%0AP%F3s-
Gradua%E7%E3o+em+Ci%EAncia+da%0AComputa%E7%E3o+Novo+curso+de+grad
ua%26ccedil%E3o%0Aem+Engenharia%0Ada+Computa%E7%E3o+</paragraph>
  <word>
   <name>de</name>
   <occurrency>11</occurrency>
   <positions>4,15</positions>
  </word>
```

```
<word>
 <name>%23cin.ufpe.br%23</name>
 <occurrency>10</occurrency>
 <positions>1</positions>
</word>
<word>
 <name>centro</name>
 <occurrency>10</occurrency>
 <positions>3</positions>
</word>
<word>
 <name>%23ufpe.br%23</name>
 <occurrency>10</occurrency>
 <positions>1</positions>
</word>
<word>
 <name>inform%E1tica</name>
 <occurrency>10</occurrency>
 <positions>5</positions>
</word>
<word>
 <name>%23br%23</name>
 <occurrency>10</occurrency>
 <positions>1</positions>
</word>
<word>
 <name>%23www.cin.ufpe.br%23</name>
 <occurrency>10</occurrency>
 <positions>1</positions>
</word>
<word>
 <name>informatica</name>
 <occurrency>10</occurrency>
 <positions>5</positions>
</word>
<word>
 <name>ufpe</name>
 <occurrency>5</occurrency>
 <positions>2</positions>
</word>
<word>
 <name>cin</name>
 <occurrency>5</occurrency>
 <positions>1</positions>
</word>
<word>
 <name>da</name>
 <occurrency>2</occurrency>
 <positions>11,19</positions>
</word>
```

<word>
<name>computacao</name>
<occurrency>2</occurrency>
<positions>12,20</positions>
</word>
<word>
<name>computa%E7%E3o</name>
<occurrency>2</occurrency>
<positions>12,20</positions>
</word>
<word>
<name>em</name>
<occurrency>2</occurrency>
<positions>9,17</positions>
</word>
<word>
<name>gradua%26ccedilao</name>
<occurrency>1</occurrency>
<positions>16</positions>
</word>
<word>
<name>pos-graduacao</name>
<occurrency>1</occurrency>
<positions>8</positions>
</word>
<word>
<name>gradua%26ccedil%E3o</name>
<occurrency>1</occurrency>
<positions>16</positions>
</word>
<word>
<name>p%F3s-gradua%E7%E3o</name>
<occurrency>1</occurrency>
<positions>8</positions>
</word>
<word>
<name>ci%EAncia</name>
<occurrency>1</occurrency>
<positions>10</positions>
</word>
<word>
<name>inscricao</name>
<occurrency>1</occurrency>
<positions>6</positions>
</word>
<word>
<name>ciencia</name>
<occurrency>1</occurrency>
<positions>10</positions>
</word>

```
  <word>
   <name>inscri%E7%E3o</name>
   <occurrency>1</occurrency>
   <positions>6</positions>
  </word>
  <word>
   <name>engenharia</name>
   <occurrency>1</occurrency>
   <positions>18</positions>
  </word>
  <word>
   <name>novo</name>
   <occurrency>1</occurrency>
   <positions>13</positions>
  </word>
  <word>
   <name>curso</name>
   <occurrency>1</occurrency>
   <positions>14</positions>
  </word>
  <word>
   <name>na</name>
   <occurrency>1</occurrency>
   <positions>7</positions>
  </word>
  <link>http%3A%2F%2Fwww.ufpe.br%2F</link>
  <link>http%3A%2F%2Fwww.cin.ufpe.br%2Fcentro%2Findex.html</link>
  <link>http%3A%2F%2Fwww.cin.ufpe.br%2Fdocentes%2Findex.html</link>
  <link>http%3A%2F%2Fwww.cin.ufpe.br%2F%7Egraduacao</link>
  <link>http%3A%2F%2Fwww.cin.ufpe.br%2F%7Eposgraduacao%2Finf-
geral.html</link>
  <link>http%3A%2F%2Fwww.cin.ufpe.br%2F%7Egraduacao%2Feng</link>
  <link>http%3A%2F%2Fwww.cin.ufpe.br%2F%7Eposgraduacao</link>
  <link>http%3A%2F%2Fwww.cin.ufpe.br%2Fextensao</link>
  <link>http%3A%2F%2Fwww.cin.ufpe.br%2Fgrupos%2Findex.html</link>
  <link>http%3A%2F%2Fwww.cin.ufpe.br%2Fservicos%2F</link>
  <link>http%3A%2F%2Fwww.cin.ufpe.br%2Feventos%2Findex.html</link>
  <link>http%3A%2F%2Fwww.cin.ufpe.br%2Flocalizacao.html</link>
  <link>mailto%3Awebmaster%40cin.ufpe.br</link>
  <link>http%3A%2F%2Fwww.designers.com.br%2F</link>
 </page>
 <page>
  <address>http%3A%2F%2Fwww.radix.com</address>
  <visitedDate>1000240053000</visitedDate>
  <modifiedDate>992330670000</modifiedDate>
  <size>506</size>
  <title>RADIX+-+A+Internet+na+sua+l%EDngua%21</title>
  <paragraph></paragraph>
  <word>
   <name>radix</name>
```

<occurrency>15</occurrency>
<positions>1,2</positions>
</word>
<word>
<name>lingua</name>
<occurrency>10</occurrency>
<positions>8</positions>
</word>
<word>
<name>a</name>
<occurrency>10</occurrency>
<positions>4</positions>
</word>
<word>
<name>%23*www*.radix.com%23</name>
<occurrency>10</occurrency>
<positions>1</positions>
</word>
<word>
<name>%23com%23</name>
<occurrency>10</occurrency>
<positions>1</positions>
</word>
<word>
<name>internet</name>
<occurrency>10</occurrency>
<positions>5</positions>
</word>
<word>
<name>sua</name>
<occurrency>10</occurrency>
<positions>7</positions>
</word>
<word>
<name>-</name>
<occurrency>10</occurrency>
<positions>3</positions>
</word>
<word>
<name>%23radix.com%23</name>
<occurrency>10</occurrency>
<positions>1</positions>
</word>
<word>
<name>na</name>
<occurrency>10</occurrency>
<positions>6</positions>
</word>
<word>
<name>l%EDngua</name>

```
   <occurrency>10</occurrency>
   <positions>8</positions>
  </word>
 </page>
</packet>
```

Além do exemplo acima, vale salientar que todos os pacotes de índices são gerados a partir de um mesmo modelo (template).

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE packet [
<!ELEMENT packet (webview, page*, URL*)>
<!ELEMENT webview (CDATA) >
<!ELEMENT page (address, visitedDate, modifiedDate, size, title, paragraph, word*, link*) >
<!ELEMENT address (CDATA) >
<!ELEMENT visitedDate (CDATA) >
<!ELEMENT modifiedDate (CDATA) >
<!ELEMENT size (CDATA) >
<!ELEMENT title (CDATA) >
<!ELEMENT paragraph (CDATA) >
<!ELEMENT word (name,occurrency,positions)>
<!ELEMENT name (CDATA) >
<!ELEMENT occurrency (CDATA) >
<!ELEMENT positions (CDATA) >
<!ELEMENT link (CDATA) >
<!ELEMENT URL (CDATA) >
]>
<packet>
 <webview>${web_view}</webview><list pages as p>
 <page>
  <address>${p.address}</address>
  <visitedDate>${p.visitedDate}</visitedDate>
  <modifiedDate>${p.modifiedDate}</modifiedDate>
  <size>${p.size}</size>
  <title>${p.title}</title>
  <paragraph>${p.paragraph}</paragraph><list p.words as w>
  <word>
   <name>${w.name}</name>
   <occurrency>${w.occurrency}</occurrency>
   <positions>${w.positions}</positions>
  </word></list><list p.links as l>
  <link>${l.link}</link></list>
 </page></list><list URLs as u>
 <URL>${u.URL}</URL></list>
</packet>
```

Apêndice B

Valores Experimentais

Abaixo seguem duas tabelas. A primeira contém a lista dos *sites* visitados e a segunda, uma tabela contendo os valores medidos durante os sessenta dias de experimento, onde foram processadas as páginas do serviço de busca em Notícias do Engenho de Busca Radix.

Lista de *Sites* de Visitação	
http://vejaonline.uol.com.br	http://*www*.jornaldocommercio.com.br
http://*www*.acritica.com.br	http://*www*.jt.com.br
http://*www*.agestado.com.br	http://*www*.lancenet.com.br
http://*www*.an.com.br	http://*www*.meioemensagem.com.br
http://*www*.atribuna.com.br	http://*www*.no.com.br
http://*www*.brasilnorte.com.br	http://*www*.odiariodemogi.inf.br
http://*www*.computerworld.com.br	http://*www*.oglobo.com.br
http://*www*.correiodabahia.com.br	http://*www*.oimparcial.com.br
http://*www*.correiodaparaiba.com.br	http://*www*.ojornal-al.com.br
http://*www*.correiodopovo.com.br	http://*www*.oliberal.com.br
http://*www*.correio*web*.com.br	http://*www*.opopular.com.br
http://*www*.dgabc.com.br	http://*www*.opovo.com.br
http://*www*.diario.com.br	http://*www*.otempo.com.br
http://*www*.diariodecuiaba.com.br	http://*www*.pcworld.com.br
http://*www*.diariodenatal.com.br	http://*www*.tribunademinas.com.br
http://*www*.dpnet.com.br	http://*www*.tribunadonorte.com.br
http://*www*.emtempo.com.br	http://*www*.un.com.br
http://*www*.epoca.com.br	http://*www*.uol.com.br/folha
http://*www*.estado.com.br	http://*www*.uol.com.br/gazeta-oam
http://*www*.estaminas.com.br	http://*www*.uol.com.br/gazetapr
http://*www*.folhadamanha.com.br	http://*www*.uol.com.br/imprensa
http://*www*.gazeta.com.br/*web*news	http://*www*.uol.com.br/jornaldodia
http://*www*.gazetadesergipe.com.br	http://*www*.uol.com.br/mundo
http://*www*.gazetadopovo.com.br	http://*www*.uol.com.br/np
http://*www*.gazetaesportiva.com.br	http://*www*.uol.com.br/odia
http://*www*.gazetamercantil.com.br	http://*www*2.uol.com.br/elle
http://*www*.gazetaonline.com.br	http://*www*2.uol.com.br/exame
http://*www*.ibusiness.com.br	http://*www*2.uol.com.br/info
http://*www*.in.gov.br	http://*www*2.uol.com.br/observatorio
http://*www*.interesportes.com.br	http://*www*2.uol.com.br/veja
http://*www*.jb.com.br	

Tabela 6 Lista de *sites* de conteúdo de informação

Data	Quantidade de Páginas	Consumo de Banda Passante (em bytes)		Tempo de Processamento (em milissegundos)		
		Caso A	Caso B	Caso A	Caso B	
		Tamanho Paginas HTML	Tamanho Índice	Tempo Parsing HTML	Tempo Empacotação	Tempo Desempacotação
26_03	13750	17819	2245	89	47	229
27_03	14200	18118	2364	88	50	244
28_03	13500	19998	2702	95	55	274
29_03	13900	17791	2375	84	49	244
04_04	12550	17299	2233	85	46	228
05_04	13500	17698	2351	79	49	238
06_04	13100	17836	2327	84	47	240
07_04	13900	17818	2535	61	52	258
08_04	13550	18291	2535	88	53	256
09_04	11800	17453	2363	76	48	243
10_04	12300	17519	2345	95	49	241
11_04	12250	17169	2321	86	48	236
12_04	12550	15630	2157	69	43	221
13_04	12800	18178	2403	95	49	245
14_04	13100	17978	2326	101	50	239
15_04	12900	17446	2303	104	47	238
16_04	11900	17013	2303	90	51	292
17_04	13250	17888	2355	88	50	249
18_04	12600	17207	2251	87	45	232
19_04	12500	16267	2214	96	44	226
20_04	12900	17713	2328	102	48	239
21_04	14250	15996	2128	112	44	221
22_04	14950	16918	2292	121	47	239
23_04	10800	17484	2391	144	49	248
24_04	12800	19221	2653	92	58	277
25_04	12850	18417	2402	77	50	247
26_04	12750	18499	2421	76	50	249
27_04	14350	18083	2407	142	50	244
28_04	12850	18403	2401	107	50	249
29_04	13550	17345	2299	92	49	236
30_04	12750	17043	2258	85	46	232
03_05	12250	18579	2429	86	48	244
04_05	11350	19255	2556	127	53	264
05_05	10250	17451	2277	74	46	236
06_05	11700	19595	2573	130	53	260
07_05	11350	18547	2380	106	49	245
08_05	10800	17663	2363	116	49	242
09_05	12400	20381	2774	141	55	281
10_05	12450	19400	2526	128	53	258
11_05	11700	17814	2344	101	48	245
12_05	11550	18138	2456	112	53	258
13_05	12650	18662	2573	131	61	287
14_05	12050	19103	2523	132	53	275
15_05	12100	19635	2549	143	52	260

16_05	12850	19267	2577	110	52	264
17_05	11050	19530	2516	146	56	289
17_05	11050	19530	2516	146	56	289
23_05	14350	20696	2560	118	51	259
24_05	13800	21273	2556	127	53	262
27_05	22900	27409	3349	123	71	334
28_05	19950	23996	2963	116	61	299
30_05	20750	28748	3577	125	76	350
31_05	20650	28836	3578	123	77	349
01_06	21650	28165	3497	127	74	337
02_06	19100	30162	3764	141	80	363
03_06	21600	28712	3542	126	75	351
04_06	16800	22888	2968	104	61	300
05_06	20350	28917	3605	125	73	350
07_06	15450	27545	3635	142	76	348
08_06	21000	20729	2583	105	54	267
Média	14010	19803	2585	107	54	266
Desvio Padrão	3152	3767	423	23	9	39

	Caso A / Caso B	**Caso B / Caso A**
Razão Consumo Banda Passante	7,66	0,13
Razão Tempo de Processamento	0,33	2,99

Tabela 7 Valores de utilização de banda passante e de processamento