

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

**Extração de Informação sobre
Bases de Dados Textuais**

por

CHRISTIAN ZAMBENEDETTI

Dissertação submetida à avaliação, como requisito parcial para
a obtenção do grau de Mestre em
Ciência da Computação

Prof. Dr. José Palazzo M. de Oliveira
Professor Orientador

Porto Alegre, abril de 2002.

CIP - Catalogação da Publicação

Zambenedetti, Christian

Extração de Informação sobre Bases de Dados Textuais / por Christian Zambenedetti — Porto Alegre: PPGC da UFRGS, 2002.

142 f.: il.

Dissertação (mestrado) — Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR-RS, 2002. Orientador: Palazzo, José Moreira de Oliveira.

1. Extração de Informação. 2. Recuperação de Informação. 3. Bancos de Dados em Múltiplos Níveis. I. Palazzo, José Moreira de Oliveira. II. Título.

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Reitora: Prof^a. Wrana Maria Panizzi

Pró-Reitor de Ensino: Prof. José Carlos Ferraz Hennemann

Pró-Reitor Adjunto de Pós-Graduação: Prof. Jaime Evaldo Fensterseifer

Diretor do Instituto de Informática: Prof. Philippe Olivier Alexandre Navaux

Coordenador do PPGC: Prof. Carlos Alberto Heuser

Bibliotecária-Chefe do Instituto de Informática: Beatriz Haro

Agradecimentos

Pensei seriamente que nunca chegaria a ponto de escrever esta seção, mas agora aqui está. Com ela, estou terminando um projeto no qual investi muito tempo e gastei muita energia desde o início do Mestrado. Tenho que admitir que algumas vezes pensei que não o faria e estou orgulhoso agora de vê-lo concluído. Por causa disso, a primeira pessoa que tenho de mencionar é minha esposa, Tatiana, que desde o início teve uma fé inabalável em mim e demonstrou uma paciência sem fim em todos os momentos. Ela sempre me passou o incentivo necessário para a concretização deste trabalho. Tati, te amo muito, pra sempre!

Quero agradecer toda minha família, pai, mãe, irmã e cunhado, tias mais próximas, que deram todo suporte emocional durante o período de estudos.

Um agradecimento especial ao meu sócio na nossa empresa, Joacir, que também superou o seu desafio e já é Mestre. Iniciamos juntos na vida profissional e acadêmica de pós-graduação, e seguimos lado a lado rompendo todas barreiras, sempre unindo os esforços.

Agradeço muito a duas pessoas: meu orientador, Prof. Dr. José Palazzo M. de Oliveira, que sempre mostrou o melhor caminho a ser seguido e confiou plenamente em mim, e ao amigo que conheci durante este trabalho, pelos conhecimentos e dicas compartilhadas, Rui Gureghian Scarinci. Muito obrigado a vocês dois.

Para Leandro Krug Wives, Diego Schultz Trein, Alessander Pires Oliveira, Miguel Feldens e Hercules Antônio do Prado, um agradecimento pelas contribuições técnicas que ajudaram no desenvolvimento desta dissertação.

Para Leonardo Zanella, Tiago Migliavacca dos Santos, Antônio Carlos Carpenedo e Felipe Pasqualotto, um agradecimento pelas contribuições técnicas que ajudaram na implementação do protótipo. Ao Felipe, muito obrigado por tudo.

Finalmente, agradeço a Deus, por fazer eu acreditar em mim mesmo, na minha capacidade, e por colocar todas as pessoas que mencionei acima no meu caminho. Sem elas tenho certeza que não teria chegado até aqui.

Sumário

Lista de Abreviaturas	7
Lista de Figuras	8
Lista de Tabelas	10
Resumo	11
Abstract	12
1 Introdução	13
2 Recuperação de Informações.....	16
2.1 Paradigma da Recuperação de Informações.....	17
2.1.1 Abstração de Informações.....	17
2.1.2 Descrição da Necessidade do Usuário.....	18
2.1.3 O Processo de Matching.....	19
2.2 Técnicas Utilizadas em Recuperação de Informações.....	20
2.2.1 Identificação de Palavras	20
2.2.2 Remoção de Stop Words.....	21
2.2.3 Expansão Semântica	21
2.2.4 Relevance Feedback.....	21
2.2.5 Dicionários	22
2.3 Classificação das Técnicas de Recuperação segundo Peter	
Gloor	22
3 Extração de Informações	25
3.1 Técnicas Básicas para Extração de Informações	28
3.1.1 Análise Léxica.....	30
3.1.2 Reconhecimento de Nomes (Termos)	30
3.1.3 Estrutura Sintática	31

3.1.4 Cenário de Combinação de Padrões	32
3.1.5 Análise de Co-referência.....	33
3.1.6 Inferência e União de Eventos	34
3.2 Testes e Avaliações.....	35
3.2.1 Métricas.....	35
3.2.2 Avaliações	37
4 Extração de Informações em Correio Eletrônico	38
4.1 Mecanismos para Recuperação e Classificação de Mensagens de Correio Eletrônico	40
4.1.1 Conceitos Básicos.....	40
4.1.1.1 Mensagem.....	40
4.1.1.2 Folder.....	42
4.1.2 Modelo de Dados	42
4.1.3 Mecanismo de Recuperação.....	44
4.1.4 Mecanismo de Classificação	47
4.1.4.1 Gerência da Estrutura de Classificação.....	48
5 Extração de Informação de Alta Precisão.....	51
5.1 Protein Data Bank.....	51
5.2 HPIEW.....	53
5.2.1 Abordagem Detalhada.....	54
5.3 Extraíndo Valores R e R_{free}.....	56
5.4 Estimando as Medidas de Precision e Recall.....	57
6 Sistema de RI Baseado em Regras.....	59
6.1 Uma Abordagem Baseada em Conhecimento.....	60
6.2 Expressando Tópicos de Consulta como Regras	61
6.3 Processamento de Consultas	62
7 Sistema de Extração Semântica de Informações.....	65
7.1 Sistema Baseado em Conhecimento.....	65
7.1.1 Máquina de Inferência.....	65

7.1.2 Eventos de Entrada	66
7.1.3 Conclusão	66
7.1.4 Base de Conhecimento.....	66
7.2 Extração de Informações dos Arquivos de Entrada.....	67
7.3 Arquitetura do SES	69
7.4 SIRI	71
7.5 Extração em Múltiplos Níveis.....	72
8 Linguagem de Extração	75
8.1 Utilização da Linguagem de Extração.....	76
8.1.1 Aspectos Gerais.....	76
8.1.2 Aspectos Específicos.....	78
8.1.2.1 Nível de Classificação Estrutural – P1	78
8.1.2.2 Nível de Classificação por Domínio – P2.....	83
8.1.2.3 Nível de Análise Superficial – P3.....	86
8.2 Arquivo de Saída Final.....	99
9 Validação da Linguagem de Extração.....	101
9.1 Caso de Validação.....	102
9.1.1 Informações a serem Extraídas dos E-Mails	103
9.1.2 Ambiente de Validação	103
9.1.3 Bases de Conhecimento Criadas para a Validação	103
9.1.4 Avaliação do Resultado Final.....	113
10 Conclusões	117
10.1 Trabalhos Futuros.....	119
Anexo 1 – Projeto para Validação da Linguagem	121
Anexo 2 – E-Mail’s Utilizados na Validação	131
Bibliografia	138

Lista de Abreviaturas

ASCII	American Standard Code for Information Interchange
BD	Banco de Dados
BDMN	Base de Dados de Múltiplos Níveis
BDNE/SE	Bases de Dados Não Estruturadas e Semi-Estruturadas
EI	Extração de Informações
ETOS	Electronic Trading Opportunities System
HPIEW	High Precision Information Extraction Workbench
HTML	HyperText Markup Language
MUC	Message Understanding Conference
NLP	Natural Language Processing
PDB	Protein Data Bank
RFC	Request for Comment
RI	Recuperação de Informações
RUBRIC	RULE-Based Retrieval of Information by Computer
SA	Sociedade Anônima
SBC	Sistema Baseado em Conhecimento
SES	Sistema de Extração Semântica de Informações
SES-MN	Sistema de Extração Semântica de Informações em Múltiplos Níveis
SRI	Sistema de Recuperação de Informações

Lista de Figuras

FIGURA 2.1 – Paradigma da Recuperação de Informações	17
FIGURA 2.2 – Processo de Abstração.....	18
FIGURA 2.3 – Descrição de uma Consulta	19
FIGURA 2.4 – Identificação de Termos Válidos.....	20
FIGURA 2.5 – Identificação de Stop Words	21
FIGURA 2.6 – Classificação das Técnicas de Recuperação	23
FIGURA 3.1 – Template de um Sistema de Extração de Informações.....	26
FIGURA 3.2 – Estrutura de um Sistema de Extração de Informações.....	28
FIGURA 3.3 – Eventos Extraídos do Texto Dogs & Dogs SA.....	29
FIGURA 3.4 – Recall e Precision	35
FIGURA 3.5 – Relação entre Recall e Precision.....	36
FIGURA 4.1 – Soluções para Tratamento do Information Overload	39
FIGURA 4.2 – Estrutura de uma Mensagem Eletrônica com Cabeçalho Padrão...	41
FIGURA 4.3 – Estrutura de uma Mensagem Eletrônica.....	41
FIGURA 4.4 – Modelo de Dados (Classes Msg, Discarded, Fld)	43
FIGURA 4.5 – Hierarquia de Especialização de Folders Virtuais	43
FIGURA 4.6 – Interface da Subclasse Man_Fld	43
FIGURA 4.7 – Interface da Subclasse Aut_Fld.....	44
FIGURA 4.8 – Interfaces da Subclasse Sys_Fld e suas Especializações.....	44
FIGURA 4.9 – Sintaxe da Linguagem de Consulta	46
FIGURA 5.1 – Fluxograma do Processo de Extração com o Uso do HPIEW.....	55
FIGURA 5.2 – N° de Documentos Remanescentes X N° de Passos de Extração.....	57
FIGURA 6.1 – O Triângulo da Recuperação de Informações.....	59
FIGURA 6.2 – Árvore de Avaliação de Regras para Consulta sobre Terrorismo....	60
FIGURA 6.3 – Base de Regras para o Tópico World_Series	61
FIGURA 6.4 – Árvore de Avaliação de Regras para o Tópico World_Series	63
FIGURA 7.1 – Arquitetura Genérica de um Sistema Baseado em Conhecimento .	66
FIGURA 7.2 – Regra para Expressar Conhecimento.....	67
FIGURA 7.3 – Arquitetura do Sistema de Extração Semântica de Informações....	70
FIGURA 7.4 – Extração Semântica de Informações	72
FIGURA 7.5 – Sistema Baseado em Conhecimento no SES-MN	74
FIGURA 8.1 – Base de Dados Intermediária – Fase de Identificação	78

FIGURA 8.2 – Base de Dados Intermediária – Fase de Seleção	79
FIGURA 8.3 – Base de Dados Intermediária – Fase de Desdobramento.....	80
FIGURA 8.4 – Base de Dados Intermediária – Fase de Conversão	81
FIGURA 8.5 – Base de Dados Intermediária – Fase de Classificação	83
FIGURA 8.6 – Base de Dados do Nível de Classificação por Domínio – P2	86
FIGURA 8.7 – Estrutura do Arquivo de Saída.....	99
FIGURA 8.8 – Arquivo de Saída Exemplo para E-Mails do ETOS	100
FIGURA 9.1 – Tela Principal do Protótipo.....	104
FIGURA 9.2 – Visualização da Base de Dados do Nível 0.....	105
FIGURA 9.3 – Base de Dados da Sub-Fase Identificação.....	105
FIGURA 9.4 – Base de Dados da Sub-Fase Seleção	106
FIGURA 9.5 – Base de Dados da Sub-Fase Desdobramento	106
FIGURA 9.6 – Base de Dados da Sub-Fase Desdobramento	107
FIGURA 9.7 – Base de Dados da Sub-Fase Desdobramento	107
FIGURA 9.8 – Base de Dados da Sub-Fase Conversão.....	108
FIGURA 9.9 – Base de Dados da Sub-Fase Conversão.....	108
FIGURA 9.10 – Base de Dados da Sub-Fase Classificação.....	109
FIGURA 9.11 – Base de Dados da Sub-Fase Classificação.....	109
FIGURA 9.12 – Base de Dados do Nível 2 – Classificação por Domínio	110

Lista de Tabelas

TABELA 3.1 – Entidades Semânticas	32
TABELA 3.2 – Entidades Semânticas	32
TABELA 3.3 – Entidades Semânticas	33
TABELA 3.4 – Entidades Semânticas	33
TABELA 3.5 – Entidades Semânticas	34
TABELA 5.1 – Remark Records de alguns Arquivos do PDB.....	52
TABELA 5.2 – Padrões de Extração Desenvolvidos para Extrair o Valor R de Arquivos do PDB.....	57
TABELA 7.1 – Associação entre Extensões de Nomes de Arquivos e Bases de Conhecimento.....	69
TABELA 8.1 – Parâmetros das Funções de Pesquisa	90
TABELA 8.2 – Parâmetros das Funções de Verificação	93
TABELA 8.3 – Parâmetros das Funções de Extração.....	96
TABELA 8.4 – Parâmetros das Funções de Deslocamento.....	98
TABELA 9.1 – Variáveis do Resultado da Extração	112
TABELA 9.2 – Distribuição de E-Mails.....	113
TABELA 9.3 – Resultado Final do Processo de Extração	115

Resumo

Com a crescente popularização dos microcomputadores e da rede mundial de informação, Internet, uma enorme variedade e quantidade de informações estão se tornando acessíveis a um número cada vez maior de pessoas. Desta forma, também cresce a importância de se extrair a informação útil que está no grande conjunto das informações disponibilizadas.

Hoje há muito mais dados na forma de textos eletrônicos do que em tempos passados, mas muito disto é ignorado. Nenhuma pessoa pode ler, entender e sintetizar *megabytes* de texto no seu cotidiano. Informações perdidas, e conseqüentemente oportunidades perdidas, estimularam pesquisas na exploração de várias estratégias para a administração da informação, a fim de estabelecer uma ordem na imensidão de textos. As estratégias mais comuns são recuperação de informações, filtragem de informações e outra relativamente nova, chamada de extração de informações. A extração de informações tem muitas aplicações potenciais. Por exemplo, a informação disponível em textos não-estruturados pode ser armazenada em bancos de dados tradicionais e usuários podem examiná-las através de consultas padrão. Para isso, há um complexo trabalho de gerenciamento, que é conseqüência da natureza não estruturada e da difícil análise dos dados. Os dados de entrada, que são os textos semi ou não-estruturados, são manipulados por um processo de extração configurado através de bases de conhecimento criadas pelo usuário do sistema.

Esta dissertação tem como objetivo a definição de uma linguagem, com base em uma arquitetura de múltiplos níveis, para extrair satisfatoriamente as informações desejadas pelo usuário, presentes em bases de dados textuais. Também faz parte deste trabalho a implementação de um protótipo que utiliza a linguagem proposta.

Palavras-Chave: Extração de Informação, Bases de Dados Textuais, e Linguagem de Consulta.

TITLE: “INFORMATION EXTRACTION ON TEXTUAL DATABASES”

Abstract

With the growing use of microcomputers and of the Web, an enormous variety and amount of information is accessible to a larger number of persons. The importance of extracting the useful information among an enormous variety of raw data is of growing importance.

Today there are much more data in the form of electronic texts than in the past, but a lot of that information is not used, as it is lost in the larger amount of useless data. No one person can read, understand and synthesize megabytes of text in its daily one. Lost information, and consequently lost opportunities, stimulated researches in the exploration of several strategies for managing information to establish an order in the enormity of available texts. The most common strategies are information retrieval, information filtering and other relatively new area: information extraction. The information extraction has many potential applications, for example, the available information in unstructured texts can be stored in traditional databases and users can examine them through pattern query. For that application, there is a complex managing work, consequence of the nature and of the complex text analysis. An extraction process configured through knowledge bases made by the system user manipulates the input data, plain texts or semi-structured texts.

This dissertation has as objective the definition of a language, with base in multiple levels architecture, to acceptably extract the information wanted by the user from textual databases. Part of this work is the implementation of a prototype that uses the proposal language.

Keywords: Information Extraction, Textual Databases, and Query Language.

1 Introdução

A descoberta do conhecimento em bases de dados refere-se ao amplo processo de encontrar conhecimento a partir de dados. Ela tem por objetivo a extração do conhecimento implícito e a busca da informação potencialmente útil dos dados. Tipicamente este processo consiste em uma série de passos, executados de forma interativa e iterativa [FAY96], que inicia com a seleção de um conjunto ou amostra dos dados com os quais o processo de descoberta será realizado. Para isto utilizam-se técnicas de mineração que procuram por padrões e regularidades, normalmente em grandes volumes de dados. Por fim, as informações descobertas são interpretadas e avaliadas de forma que se selecione os conhecimentos úteis resultantes de todo este processo.

Estas técnicas estão sendo utilizadas com sucesso em diversas áreas do conhecimento, por exemplo, no comércio, onde podem auxiliar a compreender e prever o comportamento do consumidor. Conforme Miguel Feldens [FEL97], a aplicabilidade dos resultados de pesquisas na área é bastante ampla, podendo ser útil em qualquer área do conhecimento onde seja gerado um certo volume de informações.

Todos os sistemas de recuperação de informações possuem um único objetivo: fazer com que o usuário encontre a informação que está precisando rapidamente, de modo que este usuário não necessite analisar ele próprio todas as informações existentes na base de informações. O sistema de recuperação de informações deve realizar esta análise e fornecer aquelas informações que são mais relevantes para este usuário [WIV97].

A busca de informações textuais difere da busca de informações tradicional que é a realizada nos bancos de dados tradicionais, como o relacional. Os bancos de dados tradicionais preocupam-se com o armazenamento, manutenção e a recuperação de informações disponíveis explicitamente no sistema. Ao contrário dos bancos de dados textuais, onde a informação está implícita, muitas vezes escondida ou difícil de ser localizada. Neste último, não há campos, capazes de identificar os atributos específicos de determinados registros, ou seja, as informações não estão armazenadas em tabelas como em bancos de dados relacionais.

O estudo sobre extração de informações (EI) a partir de textos é um tópico de pesquisa de interesse mundial devido à importância dos documentos textuais [SCA97]. O problema central da EI é a dificuldade em descobrir a semântica de um texto por características superficiais como palavras individuais.

Sistemas de EI destinam-se a encontrar e relacionar informações relevantes a partir de documentos, enquanto ignoram informações estranhas e irrelevantes. Sistemas de EI não tentam interpretar o texto em todas as partes do documento de entrada, mas analisam as porções que contenham informações relevantes. A relevância é determinada por um domínio pré-definido de normas que devem especificar, com tanta exatidão quanto for possível, que tipos de informação o sistema espera encontrar. Um sistema de EI converte textos não estruturados em entradas estruturadas para uma base de dados. Estas entradas

podem ser geradas a partir de um conjunto fixo de valores ou de cadeias de caracteres retirados do texto original.

Um leitor adquire conhecimento a partir de um texto, facilmente e naturalmente, identificando as informações relevantes e memorizando-as. Contudo, automatizar esta atividade é tão complexo quanto construir um sistema que entenda a linguagem natural. A busca do conhecimento a partir de bases de dados não estruturadas e semi-estruturadas, como textos, exige o entendimento dos dados armazenados, transformando-os em informação de forma automática.

O sistema de EI apresentado por Rui Scarinci [SCA97, SCA97a], chamado de Sistema de Extração Semântica de Informações (SES), foi dividido em módulos, os quais refletem muito aproximadamente a estrutura trivial de um sistema baseado em conhecimento. Como a EI a partir de bases de dados não estruturadas e semi-estruturadas é uma tarefa de difícil e complexa solução, a modularização permite o tratamento do sistema como um conjunto de tarefas básicas independentes, facilitando o entendimento da arquitetura proposta. Após este trabalho, o mesmo autor, visando aprimorar o SES, apresentou uma nova arquitetura com o uso dos conceitos de Bases de Dados de Múltiplos Níveis [SCA2001]. Um processo de EI em múltiplos níveis realizará a extração de informações de forma progressiva, através de um escalonamento de rotinas de extração que podem atuar sobre diferentes características dos arquivos de entrada, e, principalmente, utilizar informações de todos os níveis de dados abaixo do atual. Nesta arquitetura de múltiplos níveis, o processo de EI pode ser dividido em fases mais simples e concisas.

A meta desta dissertação é definir uma linguagem, com base na arquitetura de múltiplos níveis, para extrair satisfatoriamente as informações desejadas por usuários de listas de distribuição de *e-mails*. A problemática encontrada nestas listas é a grande quantidade de mensagens que o usuário recebe, e que muitas vezes ultrapassa a sua capacidade de processá-las para absorver as informações que estão contidas, causando o fenômeno conhecido como *information overload*. Para atingir o objetivo traçado, o trabalho está estruturado da forma apresentada a seguir.

O capítulo 2 descreve o que é recuperação de informações, mostrando o paradigma envolvido no processo de RI, os pontos em que os problemas ocorrem e, conseqüentemente, onde a recuperação pode falhar, e também o que pode ser feito para evitar isso. Finalizando o capítulo, há algumas técnicas utilizadas em sistemas de RI.

O capítulo 3 descreve o que é extração de informações, apresentando as técnicas básicas do processo de EI e como um sistema de EI pode ser avaliado por sua eficiência, ressaltando quais as métricas utilizadas e testes realizados.

Nos capítulos 4, 5 e 6 são apresentados três sistemas de recuperação e extração de informações onde, através de exemplos, é detalhado o funcionamento dos mecanismos de recuperação e extração.

O capítulo 7 trata de alguns trabalhos já desenvolvidos dentro do projeto SES e que são a base fundamental para o objetivo desta dissertação, servindo como arquitetura sob a qual a linguagem proposta foi desenvolvida.

A descrição de toda linguagem proposta é apresentada no capítulo 8, detalhando a sintaxe e forma de utilização em todos os níveis da arquitetura através de exemplos. Os porquês da linguagem são destacados, mostrando os pontos positivos do seu uso. Já no capítulo 9, é feita a validação da linguagem através da aplicação de um caso real, utilizando os *e-mails* recebidos das listas do Sistema de Oportunidades de Comércio Eletrônico (*Electronic Trading Opportunities System – ETOS*). Esta validação é submetida ao protótipo que também foi implementado durante o desenvolvimento desta dissertação.

Finalmente, o capítulo 10 apresenta as conclusões alcançadas com base no estudo realizado na bibliografia, na elaboração da linguagem e conseqüente validação.

2 Recuperação de Informações

Todos os Sistemas de Recuperação de Informações (SRI) possuem um único objetivo: fazer com que o usuário encontre a informação que está precisando rapidamente, de modo que este usuário não necessite analisar todas as informações existentes na base de informações [WIV97].

O SRI deve realizar esta análise e fornecer aquelas informações que são mais relevantes. Determinar a relevância da informação depende muito de como o usuário expressa sua necessidade, isto é, depende de como ele formula sua consulta [WIV97]. É através das características fornecidas na consulta que o SRI vai determinar quais informações são mais relevantes. Já que não há como obter informações diretamente do usuário, a não ser através de uma expressão formal de consulta, o usuário deve expressar corretamente sua necessidade. Caso não seja feito isso, o resultado não será satisfatório. Muitas vezes o sistema tem como saber se a informação recuperada tem relação com a descrição feita pelo usuário. Não é possível saber se a consulta que o usuário elaborou descreve corretamente a sua necessidade. Portanto, em muitos casos, o sistema busca informações relevantes para a descrição do usuário, mas irrelevantes para a informação realmente desejada por ele, já que a necessidade não foi descrita corretamente.

Gerard Salton [SAL83] define um sistema de recuperação de informações como um sistema usado para armazenar itens de informação que precisam ser processados, pesquisados, recuperados e disseminados a várias populações de usuário.

Teoricamente não há nenhuma restrição quanto ao tipo e estrutura dos itens de informação a serem armazenados e recuperados com um sistema de RI. Na prática, entretanto, a maior parte dos sistemas de RI ainda estão processando principalmente informação textual. Se a informação é bem estruturada, sistemas de gerenciamento de banco de dados são usados para armazenar e acessar esta informação. Os dados, neste caso, normalmente estão estruturados na forma de redes (para bancos de dados em rede), hierarquicamente (para bancos de dados hierárquicos), em tabelas (para bancos de dados relacionais) ou objetos (para bancos de dados orientados a objetos). Ao contrário de bancos de dados, sistemas de RI clássicos se preocupam em armazenar e recuperar informações não-estruturadas ou narradas. Para que estas informações possam ser pesquisadas, elas devem ser armazenadas em um formato processado por computador.

Até recentemente, os sistemas de RI eram limitados a pesquisar informação textual. Com o advento da multimídia (por exemplo, bibliotecas digitais) a atenção se voltou para a recuperação de documentos que consistem em múltiplos tipos de mídia, incluindo o enfoque tradicional, em fontes textuais, e a ênfase crescente em mídia com propriedades de espaço e tempo (por exemplo, som, mapas, gráficos, imagens e vídeo). Aqui a discussão estará limitada à recuperação de informações baseada em textos.

2.1 Paradigma da Recuperação de Informações

A Figura 2.1 mostra o paradigma clássico, ou a estrutura geral de um SRI tradicional, que independe do tipo de informação utilizado. Os elementos da Figura 2.1 representam os objetos e a interação entre estes objetos no paradigma. Observando esta figura, pode-se notar que existem três pontos-chave que devem ser trabalhados com atenção.

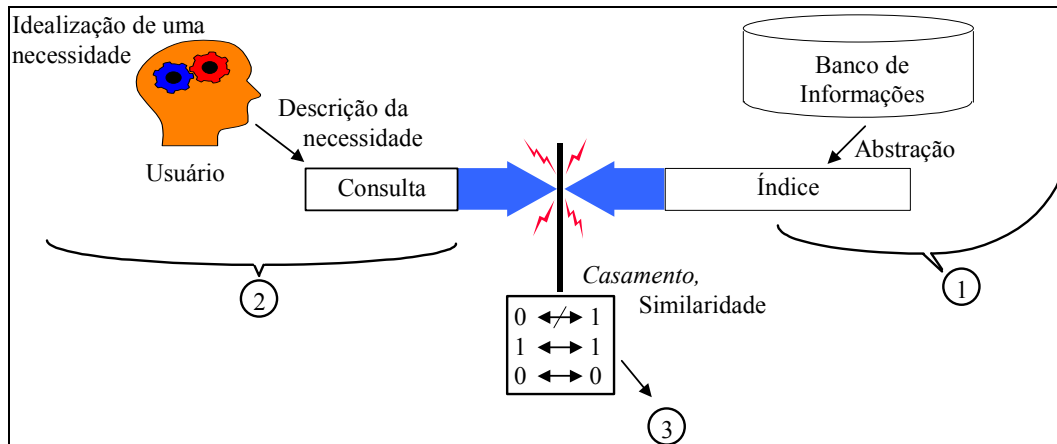


FIGURA 2.1 – Paradigma da recuperação de informações.

O primeiro é o processo de abstração de informações, determinado pela modelagem do sistema. O segundo é decorrente da abstração que o usuário faz ao descrever a informação de que necessita, em algum formalismo (linguagem de consulta). O último é o processo de casamento (*matching*) que o sistema faz entre a consulta do usuário e as informações do sistema, a fim de determinar quais informações são relevantes.

É nestes pontos que os problemas ocorrem e, conseqüentemente, onde a recuperação pode falhar. A seguir estes pontos serão detalhados.

2.1.1 Abstração de Informações

Em um SRI modelar a informação que o sistema irá tratar é extremamente importante. É necessário identificar o que é relevante em determinado tipo de informação, isto é, o que caracteriza determinada informação e pode ser utilizado para distingui-la entre outras informações. Ou seja, é preciso identificar o conteúdo real da informação, o conteúdo que realmente consegue descrevê-la. Para isto, pode ser importante descobrir como o usuário geralmente faz referência a esta informação.

Depois de identificado o conteúdo e as características de determinada informação, é necessário idealizar algum formalismo que possa descrevê-la e armazená-la. Escolher um modelo que consiga armazenar o conteúdo da informação como um todo é tarefa complexa e difícil, mas de extrema importância. Vários problemas podem surgir em decorrência de uma modelagem incorreta da

informação. Porém, depois de definidas as características da informação, o processo de modelagem pode ser realizado manualmente ou automaticamente.

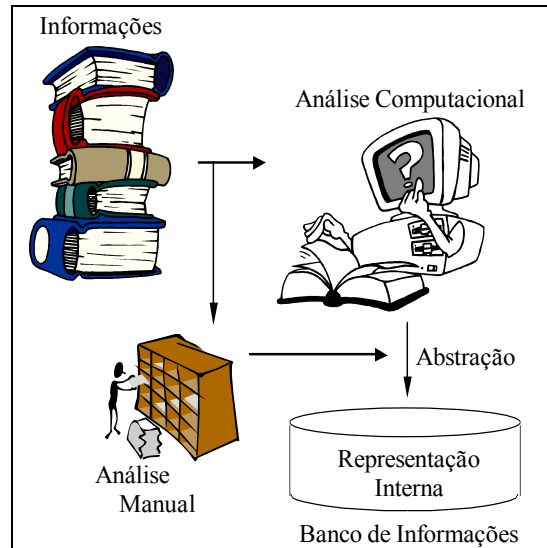


FIGURA 2.2 – Processo de abstração.

A figura 2.2 mostra o processo de abstração, onde as informações são analisadas manualmente ou automaticamente. Após a análise as características são armazenadas, conforme o modelo, em uma representação interna.

A indexação é uma das técnicas de abstração muito utilizada na área de RI. É através do índice que as informações são acessadas. Portanto o índice deve ser construído com muito cuidado. A não inclusão de uma característica importante de uma determinada informação no arquivo de índice fará com que o acesso a esta informação seja deficiente, ou até mesmo, dependendo da consulta, esta informação pode não ser recuperada. Logo, este processo de abstração tem seus problemas e deve ser tratado com cuidado.

2.1.2 Descrição da Necessidade do Usuário

Do mesmo modo que o sistema precisa estar preparado para trabalhar com abstração de informações, o usuário deve ser capaz de descrever a informação que ele necessita, identificando o maior número possível de características desta informação.

O processo de descrição da informação pelo usuário também é problemático por várias razões. A primeira em relação ao próprio usuário, pois devido à sua formação e seu conhecimento, pode não ser capaz de descrever a informação corretamente. A segunda em relação ao usuário não estar familiarizado com o sistema, não sabendo utilizar a linguagem de consulta, o tipo de informação armazenado ou o modelo de abstração utilizado pelo sistema. A última, em relação ao próprio sistema, onde a linguagem de consulta não permite que o usuário expresse corretamente suas intenções.

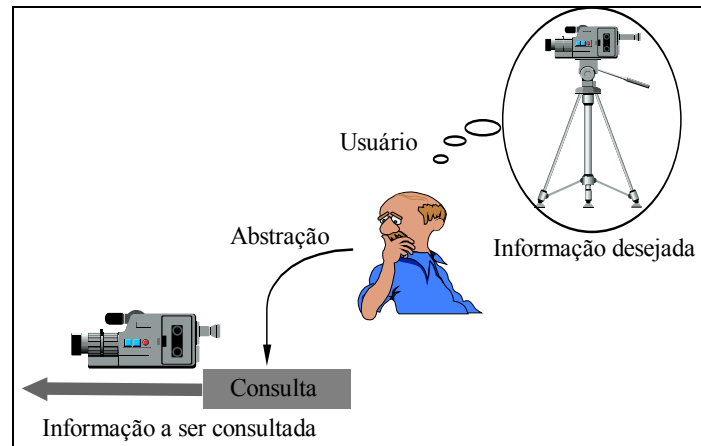


FIGURA 2.3 – Descrição de uma consulta.

Alguns destes problemas podem estar diretamente relacionados com o modelo utilizado pelo sistema para modelar as informações. Mesmo os usuários mais experientes costumam ter problemas quando o modelo de abstração utilizado não é adequado para o tipo de informação.

O usuário faz uma representação mental da informação que quer. Esta mentalização geralmente é feita levando em conta o formato original da informação. Quando o usuário utiliza outra forma de descrição, que não utilize o formato original da informação, os problemas surgem. Decorrente disso pode ocorrer de várias pessoas descreverem uma mesma imagem de formas diferentes, mesmo que vejam ou imaginem a mesma imagem. Este problema, onde vários usuários descrevem o mesmo objeto de formas diferentes é conhecido por problema do vocabulário.

Em muitos casos ocorre o cenário descrito na figura 2.3, onde o usuário deseja um tipo de informação, mas, devido aos problemas citados, a descrição do que ele está procurando não é completa ou é errônea. Neste caso sua consulta irá retornar informações irrelevantes.

2.1.3 O Processo de *Matching*

O último ponto-chave da recuperação de informações encontra-se justamente no mecanismo que faz o processo de identificação de quais informações são relevantes para a consulta do usuário. Este processo procura identificar a similaridade entre as informações armazenadas no sistema e a descrição de informação que o usuário deseja.

Muitos problemas surgem neste ponto. Primeiro, porque tanto o sistema quanto o usuário fazem abstrações da informação, o que pode acarretar numa perda de características importantes da informação. Além disso, muitos métodos utilizam um processo de comparação direta entre as características desejadas pelo usuário e as características das informações na base de dados. Aqui, incide novamente o problema do vocabulário, já que as pessoas que descrevem a informação podem não utilizar as mesmas características que o usuário.

Este fato ocorre muitas vezes em uma biblioteca, onde o usuário tem dificuldades para encontrar determinada informação porque o bibliotecário não a colocou na mesma categoria que o usuário pensa que ela deveria estar. Neste caso, duas pessoas diferentes, consultando, podem obter resultados diferentes, já que descrevem a informação de formas diferentes. Uma pode obter resultados melhores do que a outra. A utilização de um vocabulário controlado é uma solução que apresenta bons resultados, desde que os usuários sejam treinados para utilizar este vocabulário. Além disso, aqueles que procuram informações cujo assunto lhes é conhecido costumam obter melhores resultados, pois estão familiarizados com os termos utilizados. O usuário deve utilizar o maior número de características possíveis na descrição da informação que deseja, pois assim será maior a probabilidade do sistema localizar uma informação relevante.

2.2 Técnicas Utilizadas em Recuperação de Informações

A principal desvantagem das abordagens baseadas em palavras é que palavras sozinhas raramente são específicas o suficiente para precisamente discriminar documentos relevantes. Um método melhor é identificar grupos de palavras que criam frases com significado e, então, considerar não apenas as palavras na busca por informação relevante dentro de um documento, mas frases. Um bom exemplo é a frase “*Joint Venture*”, que pode ser muito mais significativa em um artigo financeiro do que as palavras “*Joint*” e “*Venture*” separadamente.

Em grandes bases de dados compreendendo milhares de documentos, o uso de termos frasais não é apenas desejável, mas se torna necessário. Os termos que serão utilizados para recuperar documentos relevantes normalmente são pré-processados através de algumas técnicas principais, como as mostradas a seguir.

2.2.1 Identificação de Palavras

A identificação de palavras nada mais é do que a análise da sequência de caracteres no texto. Este processo de validação torna-se bastante útil, especialmente quando o documento apresenta muitos caracteres inválidos ou palavras com erros gramaticais. As seqüências de caracteres inválidas devem ser eliminadas e as palavras com erros corrigidas. A figura 2.4 apresenta o trecho de um documento com diversas seqüências de caracteres. As seqüências marcadas são seqüências inválidas, que não devem passar pela fase de identificação de palavras. As demais podem ser identificadas como termos válidos. Os termos sublinhados são termos identificados como incorretos pelo dicionário e devem ser corrigidos. Os caracteres de pontuação são desprezados.

... ã± ∇ á` > ` ~ ` ÿb\$ Na maioria das vezes os documentos retornados pelas ferramentas de > ` recuperação de informacoes > ` envolvem um contexto mais amplo, fazendo com que o usuario tenha que garimpar, ou seja, especificar ou filtrar estes documentos (o que demanda tempo e conhecimento) a fim de obter a informação que ele realmente necessita ~` ...

FIGURA 2.4 – Identificação de termos válidos.

2.2.2 Remoção de *Stop Words*

Palavras de alta frequência são eliminadas porque elas são supostamente muito comuns, e então, não significativas para a identificação de documentos relevantes [SPA97]. Esta técnica é normalmente implementada comparando o texto de entrada com uma “*stop list*” de palavras comuns. Palavras comuns podem ser, por exemplo: “sobre”, “em”, “nosso”, entre outros artigos e preposições. Além da remoção de palavras não significativas, que não interferem no processo de busca, outro benefício é a redução considerável do tamanho do texto original.

A figura 2.5 apresenta o documento resultante da etapa anterior após ser validado por uma *stop list*. Neste caso a lista de *stop words* contém artigos, preposições, conjunções e algumas seqüências de caracteres que não devem ser adicionadas ao índice, por possuírem frequência elevada.

...Na maioria das vezes os documentos retornados pelas ferramentas de recuperação de informações envolvem um contexto mais amplo, fazendo com que o usuário tenha que garimpar, ou seja, especificar ou filtrar estes documentos e que demanda tempo e conhecimento a fim de obter a informação que ele realmente necessita...

FIGURA 2.5 – Identificação de *stop words*.

2.2.3 Expansão Semântica

A técnica de expansão semântica tem por objetivo ampliar semanticamente as consultas informadas pelos usuários de sistemas de recuperação de informações. Expandir semanticamente uma palavra consiste em encontrar outras palavras relacionadas, utilizando então este conjunto maior para recuperação de informação.

Os principais métodos para implementação de técnicas de expansão semântica são:

- Remoção de sufixos;
- Detecção de raízes equivalentes;
- Adição de termos relacionados.

2.2.4 Relevance Feedback

Uma técnica bastante utilizada no refinamento de recuperação de informações é *relevance feedback*, uma técnica de *Machine Learning*. Neste processo, os usuários identificam documentos relevantes na lista de documentos recuperados, e então é criada uma nova pesquisa baseada nesta amostra de documentos relevantes.

O processo de *relevance feedback* se caracteriza por uma consulta inicial, descrevendo a informação de que se tem necessidade. A consulta é processada trazendo um conjunto de documentos recuperados. Tais documentos são classificados e ordenados por similaridade com a consulta. Neste momento, o

usuário tem a opção de selecionar alguns dos documentos melhor classificados, conforme a ordenação, normalmente 10 ou mais, de acordo com os que ele identifica como relevantes [SAL68].

Os documentos selecionados são analisados por um procedimento de aprendizado automático, normalmente supervisionado pelo usuário, que, juntamente com a consulta inicial, produz uma nova consulta. Esta nova consulta é utilizada para criar uma nova lista ordenada, normalmente mais relevante que a original [LEW95]. Este processo pode ser repetido diversas vezes e o resultado, a cada passo, produz uma modificação que aproxima a consulta dos termos relevantes e afasta dos irrelevantes [SAL68]. Desta forma, os documentos selecionados servem de padrão de busca para a próxima pesquisa, refinando, assim, a busca de informações.

2.2.5 Dicionários

Muitas das falhas do método sintático sugerem que a análise correta do conteúdo de um texto pode depender da possibilidade de informação adicional relacionada com os termos individuais e suas inter-relações [RIL93]. Uma possibilidade consiste no uso de descritores de termos contidos em dicionários, provendo uma maior exatidão dos termos que formam a frase. Dicionários, ou *thesaurus*, são ferramentas utilizadas para detectar relações entre palavras na linguagem natural. Tais relações podem ser utilizadas para criar um sistema de análise de linguagem, retirando a ambigüidade de significado de termos e, além disso, gerando grupos de similaridade entre palavras pela identificação de relações entre o contexto de várias entradas do dicionário [SAL68].

O principal objetivo dos dicionários é fornecer uma normalização da linguagem, para garantir que diferenças na terminologia e uso de palavras similares não afetará inadvertidamente o desempenho da recuperação. Um modo de assegurar isto é listar os identificadores de informação e fornecer uma definição que regulariza e controla seu uso, por exemplo, uma lista ou tabela, onde cada entrada tem o formato PALAVRA, DEFINIÇÃO/APLICAÇÃO [SAL68].

2.3 Classificação das Técnicas de Recuperação segundo Peter Gloor [GLO97]

Esta seção brevemente revisará uma classificação de técnicas de recuperação de texto que foram propostas por Nicholas Belkin e Bruce Croft [GLO97], mostradas graficamente na figura 2.6.

No nível mais alto, Belkin e Croft distinguem entre técnicas de combinação exatas e parciais. Técnicas de combinação exata são as usadas na maioria dos sistemas de RI convencionais. São formuladas consultas usando expressões Booleanas e o padrão de pesquisa dentro da consulta tem que combinar exatamente com a representação do texto dentro do documento a ser recuperado.

Nas técnicas de combinação parcial há muitas variações. As técnicas individuais pesquisam um único documento sem considerar a coleção de documentos como um todo. Para as técnicas baseadas em características,

documentos são representados por conjuntos de características ou termos de índice. O índice pode ser definido manualmente ou automaticamente. Entre as abordagens formais de recuperação temos os modelos vetor-espacial, o probabilístico e o *fuzzy*. Além dos sistemas formais de RI baseados em características, um grande número de medidas de similaridade entre a consulta e os documentos foram propostas, sendo chamadas, de maneira genérica, de *ad-hoc*. A maioria das abordagens baseadas em características tem o problema que pequenas diferenças nos pesos podem levar a significativas diferenças nos resultados de recuperação. O representante mais conhecido na categoria das abordagens formais é o modelo vetor-espacial. Neste modelo, cada documento é representado por um vetor de índice que contém um conjunto de pesos dos termos. Para cada consulta, os documentos são ordenados em ordem decrescente de similaridade. A abordagem probabilística é semelhante ao modelo vetor-espacial: o objetivo básico é recuperar documentos na ordem da sua probabilidade de relevância para a consulta. No modelo probabilístico, uma consulta digitada por um usuário para recuperar informação é tomada como uma lista de palavras ou frases não-estruturadas. Estes termos são então combinados aos documentos no banco de dados. Alguns autores também sugerem a abordagem *fuzzy* para sistemas de RI formais baseados em características.

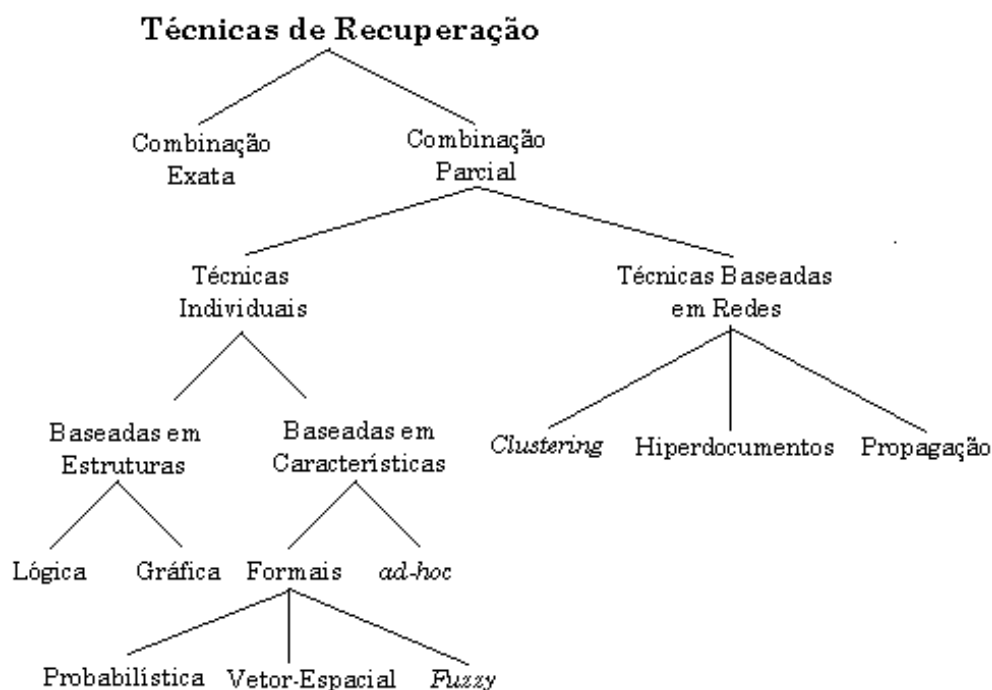


FIGURA 2.6 – Classificação das técnicas de recuperação.

Para técnicas baseadas em estruturas, os documentos são representados em uma estrutura mais complexa do que somente um conjunto de termos de índice, como as técnicas baseadas em características. É teoricamente possível representar os conteúdos das coleções de documentos em uma forma lógica. Sistemas nesta categoria podem, por exemplo, usar regras para descrever como fragmentos relevantes de um documento estão relacionados à consulta. Em

vez de usar lógica para representar os conteúdos de um documento, os conteúdos do documento podem ser descritos também como um gráfico onde as extremidades e nodos do gráfico representam idéias e relacionamentos contidos no documento.

Com as técnicas baseadas em redes, o conjunto de todos os documentos e seus relacionamentos são usados para encontrar os documentos mais relevantes com respeito à consulta. O método mais conhecido é o *clustering*, onde o objetivo é utilizar informações sobre as relações de similaridade existentes entre documentos e termos. Desta forma, a comparação da consulta não é feita em relação aos documentos da base, mas sim em relação às classes (clusters) de documentos [SPA97]. Segundo Robertson [ROB97], documentos similares tendem a ser relevantes para as mesmas consultas, e o método de *clustering*, portanto, prove um caminho de trazer juntos todos os documentos relevantes a uma consulta.

Se os documentos são representados como uma rede de nodos, o usuário também pode navegar pela rede com a ajuda do sistema. Através da interação com o usuário, o sistema pode usar a rede para construir um modelo de usuário. Baseado no modelo de usuário, um modelo necessário de informações do usuário pode ser construído, incluindo documentos relevantes encontrados durante a sessão de navegação. A propagação é semelhante à navegação em que, desde o nodo inicial, outros nodos conectados àquele nodo são ativados. Então os nodos ativados se propagam ou se espalham pela rede.

3 Extração de Informações

Hoje há uma grande quantidade de dados na forma de textos eletrônicos do que em qualquer tempo passado, mas muito disto é ignorado. Nenhuma pessoa pode ler, compreender e sintetizar *megabytes* de texto diariamente no seu cotidiano. Informações perdidas e, conseqüentemente, oportunidades desperdiçadas, têm estimulado pesquisadores a explorar várias estratégias de gerenciamento da informação para estabelecer uma ordem nesta imensidão textual. As estratégias mais comuns são recuperação de informações, filtragem de informações e outra relativamente nova, chamada de extração de informações.

Segundo Jim Cowie [COW96], sistemas de RI podem ser vistos como “colheitadeiras” que devolvem material útil de um vasto campo de materiais brutos. Com grandes quantidades de informações potencialmente úteis em mãos, um sistema de EI pode, então, transformar o material bruto refinando e reduzindo-o à idéia do texto original.

Supondo que se esteja investigando a produção de dispositivos semicondutores, o que se poderia desejar saber seria:

- Que substâncias químicas são colocadas para produzir camadas separadas;
- A densidade das camadas;
- A temperatura na qual as camadas são formadas;
- Quem usa o processo.

Tais informações freqüentemente estão em artigos de revistas e jornais, e sistemas de RI podem coletar os artigos com os textos relevantes. A extração de informações começa com a coleção de tais textos, transformando-os em informação que é mais facilmente compreendida e pronta para ser analisada. Os fragmentos de texto relevantes são isolados, a informação relevante é extraída destes fragmentos e então os pedaços são juntados coerentemente. Por exemplo, um artigo pode discutir gases químicos, temperaturas, processos e especificações de material, mas somente um ou dois destes itens podem interessar ao analista humano. O objetivo da pesquisa em extração de informações é construir sistemas que encontrem e liguem informações relevantes e ignorem informações estranhas e irrelevantes [COW96]. Para Ellen Riloff [RIL99] o propósito de sistemas de extração de informações é extrair informações de um domínio específico a partir de textos em linguagem natural. Sistemas de EI tipicamente contam com dois recursos específicos do domínio: um dicionário de padrões de extração e um dicionário semântico. Os padrões de extração podem ser construídos manualmente definindo palavras-chave ou modelos, ou podem ser gerados automaticamente com o uso de recursos especiais, como etiquetas com textos anotados em relação ao domínio específico. Os dicionários semânticos para extração de informação são quase sempre construídos manualmente porque fontes de propósito gerais não contêm o vocabulário específico do domínio necessário.

A extração de informações tem muitas aplicações potenciais. Por exemplo, a informação disponível em textos não-estruturados pode ser armazenada em bancos de dados tradicionais e usuários podem examiná-las através de consultas padrão. Para Ralph Grishman [GRI97], a idéia de reduzir a informação

em um documento para uma estrutura tabular não é nova. Sua viabilidade foi sugerida por Zellig Harris nos anos cinquenta, e uma implementação para textos médicos foi feita na Universidade de New York nos anos oitenta.

O crescente interesse no desenvolvimento de sistemas de EI é a indicação do grande volume de informações disponíveis. Este interesse representa a junção da necessidade e habilidade, observando o que é possível com a tecnologia atual de processamento da linguagem natural e como esta possibilidade pode realmente ser útil.

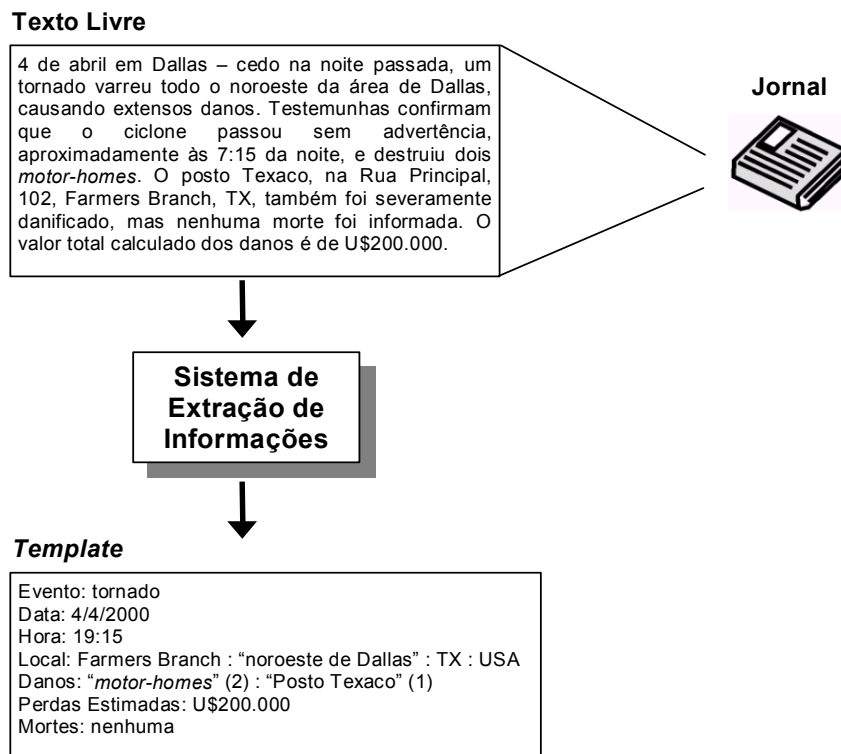


FIGURA 3.1 – *Template* de um sistema de extração de informações.

Uma enorme quantidade de informação existe somente na forma da linguagem natural. Se esta informação será automaticamente manipulada e analisada, ela deve primeiro ser separada em uma forma estruturada na qual os fatos individuais estarão acessíveis. A maioria das notícias mundiais, por exemplo, são relatadas em jornais, rádio ou redes de televisão. O que deveria ser oferecido é a recuperação de passagens pertinentes de um arquivo de texto. Se, por exemplo, se quer saber quem assinou contratos durante o último ano para a entrega de aviões ou gás natural, ou quais cidades ordenaram novas restrições em relação ao cigarro, trabalha-se em cima do conjunto de documentos recuperados. A extração de informações oferece um potencial de extração de tais dados com muito mais precisão, produzindo listas de empresas ou cidades em lugar de listas de documentos. Benefícios iguais resultariam do processamento de textos mais técnicos como jornais científicos, decisões legais ou relatórios hospitalares. Hospitais e médicos pesquisadores, em particular, enfrentam a necessidade de

executar uma análise retrospectiva em uma ampla faixa de coleções de relatórios que estão principalmente na forma da linguagem natural.

Grupos de pesquisa nos anos 50 e 60 já reconheciam o valor potencial de se estruturar automaticamente dados em linguagem natural, e projetos foram criados para executar tarefas como a transformação de entradas inteiras de enciclopédias para a forma estruturada. Tais projetos, porém, enfrentaram vários problemas no processamento da linguagem natural e representação do conhecimento, muitos dos quais ainda sem soluções efetivas. Mais recentemente foi reconhecido que estabelecendo como meta a estruturação de informação seletiva, pode-se obter um resultado que estaria mais ao alcance da tecnologia atual.

Uma tecnologia de extração de informação bem desenvolvida permitiria criar rapidamente sistemas de extração para novas tarefas cujo desempenho ficaria no mesmo nível que o desempenho humano, mas ainda não se está neste nível. Porém, sistemas com um desempenho mais modesto, que perdem alguns eventos e incluem alguns erros, podem ser proveitosos. Eles podem ser usados para extrair informação com uma posterior verificação manual, ou seja, um processo que pode oferecer benefícios quando comparado à extração puramente manual. Eles também podem ser úteis em circunstâncias onde não haveria tempo para revisar todos documentos fonte. Uma informação extraída automaticamente, mesmo que incompleta, é melhor do que nenhuma informação. Algumas pesquisas, estimuladas em parte pelas conferências *MUC (Message Understanding Conference* – ver seção 3.2.2) têm mostrado que tais sistemas de extração modestos podem, em alguns casos, ser implementados usando métodos de análise da linguagem natural relativamente simples. Estes métodos obtêm sucesso se as informações que serão extraídas estão dentro do contexto, e este foi expresso em um número pequeno de formas dentro do texto.

Na maioria dos sistemas de EI, o usuário define *templates* de extração, ou seja, modelos estruturados, como campos de uma base de dados, a serem preenchidos a partir do artigo original pelo processo de extração, como a figura 3.1 ilustra. A representação da informação em modelos tem a vantagem de ser altamente estruturada. Isto permite que, mais adiante, o processamento da informação possa ser feito por outras aplicações, como pacotes de bancos de dados tradicionais, aplicações comerciais e sistemas especialistas¹ ou de redes neurais [COS96]. Desta forma, está se provendo uma arquitetura de BD extensível para informações estruturadas extraídas de documentos semi-estruturados, com acesso rápido e preciso, bem como executando uma disseminação seletiva de documentos pertinentes, dependendo do critério de filtragem.

¹ Um sistema especialista é aquele que lida com problemas complexos do mundo real, que requeiram a interpretação de um especialista, e soluciona estes problemas através do uso de um modelo computacional do raciocínio de um especialista humano, chegando às mesmas conclusões que este especialista humano chegaria caso se defrontasse com um problema comparável [WEI88].

3.1 Técnicas Básicas para Extração de Informações

O processo de extração de informações tem duas partes principais. Primeiro, o sistema extrai fatos individuais do texto de um documento através de uma análise local do texto. Segundo, ele integra esses fatos, produzindo fatos maiores ou novos fatos, por inferência. Como passo final, depois de os fatos estarem integrados, os fatos pertinentes são traduzidos para o formato de saída requerido.

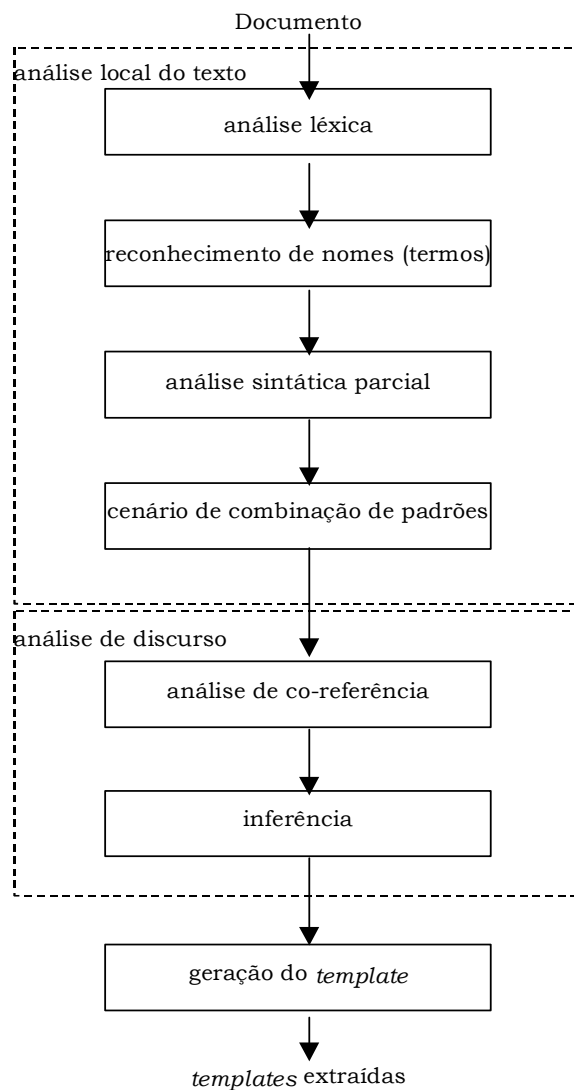


FIGURA 3.2 – Estrutura de um sistema de extração de informações.

Os fatos individuais são extraídos criando um conjunto de padrões que combinem com a lingüística real dos fatos. Por causa da complexidade da linguagem natural, não é prático descrever estes padrões diretamente como sucessões de palavras. Assim, como na maioria dos sistemas de processamento da linguagem natural, inicia-se estruturando a entrada, identificando vários níveis de componentes e relações, e então se determinam os padrões nos termos desses

componentes e relações. Este processo começa com a análise léxica e reconhecimento de nomes (termos), seguido por uma análise sintática completa (*parse*), ou por alguma forma de análise gramatical parcial para identificar grupos de substantivos, verbos e outras estruturas complementares. Depois que tudo está feito, os padrões são usados para identificar os fatos de interesse.

A fase de integração examina e combina fatos do documento inteiro. Soluciona relações de co-referência que podem variar do uso de pronomes a descrições múltiplas do mesmo evento. A figura 3.2 mostra o fluxo global do processo de extração de informações.

Seguindo a terminologia estabelecida pelos *MUCs*, a especificação dos eventos ou relações particulares a serem extraídas será chamada de cenário. Assim, será distinguido entre um domínio geral, como notícias financeiras, e um cenário particular, como *joint ventures* internacionais. No final, uma saída em formato tabular (*template*) será gerada com a informação extraída.

As próximas seções mostrarão cada estágio do processo de extração de informações. O cenário envolve a sucessão de um executivo em uma empresa, conforme o breve documento a seguir, e a figura 3.3 mostra a sua transformação em dois *templates*.

“Carlos Rodrigues aposentou-se como vice-presidente executivo de uma famosa fábrica de cachorro-quente, a Dogs & Dogs SA. Ele irá ser sucedido por Antônio Farias”.

EVENTO	Deixando o emprego
PESSOA	Carlos Rodrigues
POSIÇÃO	Vice-Presidente Executivo
EMPRESA	Dogs & Dogs SA

EVENTO	Começando no emprego
PESSOA	Antônio Farias
POSIÇÃO	Vice-Presidente Executivo
EMPRESA	Dogs & Dogs SA

FIGURA 3.3 – Eventos extraídos do texto Dogs & Dogs SA.

Em vários sistemas de extração, a maioria da análise no texto é realizada comparando-o contra um conjunto de expressões regulares. Se a expressão combina com um segmento de texto, é atribuído ao segmento de texto (componente) um rótulo, e possivelmente uma ou mais características associadas. Os padrões são organizados em conjuntos, e os rótulos componentes atribuídos em um conjunto padrão podem ser referenciados por padrões em conjuntos subseqüentes.

Associadas com alguns dos componentes estão as estruturas semânticas chamadas entidades e eventos. Estas estruturas serão usadas para construir os *templates* na última instância.

3.1.1 Análise Léxica

O texto é, primeiramente, dividido em orações e em *tokens*. Cada *token* é procurado no dicionário para determinar suas possíveis características. O dicionário do sistema Proteus [PRO2000] inclui um grande dicionário de inglês de propósito geral [GRI94], mais vários dicionários especiais, tais como dicionários com os nomes de cidades principais, as maiores empresas, primeiros nomes Americanos mais comuns e termos que são acrescentados aos nomes de empresas, como “Inc.”. No exemplo apresentado anteriormente, “Carlos” e “Antônio” seriam rotulados como primeiros nomes e “SA” seria rotulado como um termo que é acrescentado ao nome de uma empresa.

3.1.2 Reconhecimento de Nomes (Termos)

A próxima fase do processo identifica os vários tipos de nomes próprios e outras formas especiais, tais como datas e quantias em moeda. Como os nomes aparecem freqüentemente em muitos tipos de textos, identificá-los e classificá-los simplifica os processos futuros. Além disso, nomes são valores importantes como argumento para muitas tarefas de extração.

Os nomes são identificados por um conjunto de padrões (expressões regulares) que são declarados como partes da fala, características sintáticas e de ortografia. Nomes pessoais, por exemplo, podem ser identificados pela precedência de um título, como:

Sr. Henrique Dias,

por um simples primeiro nome, como:

João Vasconcelos,

por um sufixo, como:

Mário Dutra Jr.,

ou por uma inicial que está no meio, como:

Humberto S. Gomes.

Nomes de empresas podem, normalmente, ser identificados pelo seu *token* final, como:

Dogs & Dogs SA,
Dogs & Dogs Ltda.,
Dogs & Dogs Associados.

Porém, alguns nomes de grandes empresas (por exemplo, “General Motors”) podem aparecer sem qualquer dica, sendo importante ter também um dicionário com o nome de grandes empresas.

No parágrafo de exemplo, três nomes seriam identificados:

“[tipo de nome: pessoa]Carlos Rodrigues] aposentou-se como vice-presidente executivo de uma famosa fábrica de cachorro-quente, a [tipo de nome: empresa]Dogs & Dogs SA]. Ele irá ser sucedido por [tipo de nome: pessoa]Antônio Farias]”.

A identificação de nomes normalmente também inclui o processo para identificar apelidos (*aliases*) de um nome, ou seja, co-referência de nomes. Por exemplo, o sistema identificaria “João Vasconcelos” com uma menção subsequente de “Sr. Vasconcelos”, e “Hewlett-Packard Corp.” com “HP”. A identificação de apelidos também pode ajudar na classificação de nomes. Assim, se é lido “Hermes Macedo informou...” pode não ser possível saber se “Hermes Macedo” é uma pessoa ou uma empresa, mas se há uma referência subsequente a “Sr. Macedo”, a ambigüidade é resolvida.

3.1.3 Estrutura Sintática

Identificando alguns aspectos da estrutura sintática haverá uma simplificação nas fases seguintes. Depois de tudo, os argumentos a serem extraídos muitas vezes correspondem a substantivos no texto, e os relacionamentos a serem extraídos muitas vezes correspondem a relações funcionais gramaticais. Por outro lado, a identificação da estrutura sintática completa de uma sentença é uma tarefa difícil. Como resultado, há uma grande variação na quantia de estrutura sintática que é identificada explicitamente.

Alguns sistemas não têm qualquer fase separada de análise sintática. Outros tentam construir uma completa análise gramatical de uma oração. A maioria fica entre os dois tipos mencionados, fazendo análise gramatical em uma série de fragmentos. Em geral, eles só constroem estruturas sobre as quais podem ter bastante certeza, ou que tenham evidência sintática ou semântica. O sistema Proteus [PRO2000] constrói estruturas para grupos de substantivos (um substantivo mais os seus modificadores) e para grupos de verbos (um verbo com seus auxiliares), ambos usando na maioria dos casos apenas a informação sintática local. Além disso, pode construir frases maiores se tiver informação semântica que confirme a certeza da estrutura. Ao contrário de outros sistemas, nenhum procedimento especial é usado para análise gramatical.

O primeiro conjunto de rótulos possui os grupos e frases de substantivos (*np*). Isto inclui, no exemplo abaixo, os três nomes, o pronome e dois grupos de substantivos maiores. Após segue um conjunto de rótulos para grupos de verbos (*vg*). Depois da aplicação destes padrões, o texto é rotulado da seguinte maneira:

“[entidade np: e1] Carlos Rodrigues] [vg] aposentou-se] como [entidade np: e2] vice-presidente executivo] de [entidade np: e3] uma famosa fábrica de cachorro-quente], a [entidade np: e4] Dogs & Dogs SA]. [entidade np: e5] Ele] [vg] irá ser sucedido] por [entidade np: e6] Antônio Farias]”.

TABELA 3.1 – Entidades semânticas.

Entidade e1	Tipo: nome de pessoa: Carlos Rodrigues
Entidade e2	Tipo: valor da posição: Vice-Presidente Executivo
Entidade e3	Tipo: fabricante
Entidade e4	Tipo: nome da empresa: Dogs & Dogs SA
Entidade e5	Tipo: pessoa
Entidade e6	Tipo: nome de pessoa: Antônio Farias

Associadas com cada componente estão certas características que podem ser testadas por padrões nos estágios subseqüentes. No caso do *vg* incluí-se informações sobre o tempo (passado/presente/futuro), a voz (ativa/passiva) e a raiz do verbo. Para cada *np* o sistema cria uma entidade semântica, como é mostrado na tabela 3.1.

Os grupos de padrões fortalecem a estrutura das frases usando modificadores. Estas expressões não obedecem aos padrões normais. Há expressões que modificam os substantivos e outras não, pois têm o seu próprio significado. Para o exemplo que está sendo tratado, dois padrões serão relevantes: “descrição-empresa, nome-empresa” e a construção de frase preposicional “posição de empresa”. No segundo padrão, “posição” representa um elemento padrão que combina com qualquer *np* pertencente a uma entidade do tipo “posição” e “empresa” combina com qualquer *np* pertencente a uma entidade do tipo “empresa”. A partir de uma hierarquia semântica, qualquer subtipo de “empresa” (como por exemplo, fabricante) seria combinado. No primeiro padrão, “nome-empresa” especifica um *np* do tipo “empresa”, onde o termo principal é um nome, e “descrição-empresa” especifica um *np* do tipo “empresa”, onde o termo principal é um substantivo comum. Estes padrões produzem os seguintes rótulos:

“[*entidade np: e1* Carlos Rodrigues] [*vg* aposentou-se] como [*entidade np: e2* vice-presidente executivo de uma famosa fábrica de cachorro-quente, a Dogs & Dogs SA]. [*entidade np: e5* Ele] [*vg* irá ser sucedido] por [*entidade np: e6* Antônio Farias]”.

e as entidades são atualizadas como segue, na tabela 3.2:

TABELA 3.2 – Entidades semânticas.

Entidade e1	Tipo: nome de pessoa: Carlos Rodrigues
Entidade e2	Tipo: valor da posição: Vice-Presidente Executivo empresa: e3
Entidade e3	Tipo: nome do fabricante: Dogs & Dogs SA
Entidade e5	Tipo: pessoa
Entidade e6	Tipo: nome de pessoa: Antônio Farias

3.1.4 Cenário de Combinação de Padrões

Todo o processo até agora foi, de certo modo, preparatório para o cenário de combinação de padrões. O papel destes padrões é extrair os eventos ou relacionamentos relevantes ao cenário. Para o exemplo da sucessão executiva haverá dois padrões: pessoa que se aposenta em uma posição e pessoa é sucedida

por outra pessoa, onde pessoa e posição são elementos padrão que combinam *np*'s com o tipo associado. Se aposenta e é sucedida são elementos padrão que combinam grupos de verbos ativo e passivo, respectivamente. O resultado é um texto rotulado com duas cláusulas, cada uma apontando para estruturas de evento. Estas estruturas de evento apontam para as entidades previamente criadas:

“[cláusula de evento: e7 Carlos Rodrigues aposentou-se como vice-presidente executivo de uma famosa fábrica de cachorro-quente, a Dogs & Dogs SA]. [cláusula de evento: e8 Ele irá ser sucedido por Antônio Farias]”.

As entidades e eventos são atualizados como segue, na tabela 3.3:

TABELA 3.3 – Entidades semânticas.

Entidade e1	Tipo: nome de pessoa: Carlos Rodrigues
Entidade e2	Tipo: valor da posição: Vice-Presidente Executivo empresa: e3
Entidade e3	Tipo: nome do fabricante: Dogs & Dogs SA
Entidade e5	Tipo: pessoa
Entidade e6	Tipo: nome de pessoa: Antônio Farias
Evento e7	Tipo: pessoa deixando o trabalho: posição e1: e2
Evento e8	Tipo: sucede pessoa 1: e6 pessoa 2: e5

3.1.5 Análise de Co-referência

A análise de co-referência tem a tarefa de resolver referências anafóricas² provenientes de pronomes e substantivos. No texto da sucessão, o único exemplo é o pronome “ele” (entidade e5). A análise de co-referência irá procurar pela primeira entidade do tipo “pessoa” que aparecer antes de e5, e irá encontrar a entidade e1. Então, conseqüentemente, ao invés de referenciar a entidade e5 será referenciada a entidade e1, deixando as entidades e eventos como mostra a tabela 3.4:

TABELA 3.4 – Entidades semânticas.

Entidade e1	Tipo: nome de pessoa: Carlos Rodrigues
Entidade e2	Tipo: valor da posição: Vice-Presidente Executivo empresa: e3
Entidade e3	Tipo: nome do fabricante: Dogs & Dogs SA
Entidade e6	Tipo: nome de pessoa: Antônio Farias
Evento e7	Tipo: pessoa deixando o trabalho: posição e1: e2
Evento e8	Tipo: sucede pessoa 1: e6 pessoa 2: e1

Também, se uma referência a “a empresa” aparece no texto, ela iria referenciar a entidade e3.

² Anafórica: em que há anáfora. Anáfora é a repetição de uma palavra no princípio de diferentes frases ou de membros da mesma frase.

3.1.6 Inferência e União de Eventos

Em muitas situações, a informação parcial sobre um evento pode ser estendida para várias sentenças. Esta informação precisa ser combinada antes que um *template* possa ser gerado. Em outros casos, uma informação pode estar implícita, e é necessário torná-la explícita através do processo de inferência.

No domínio da sucessão executiva, é necessário determinar o quê o predicado “sucessão” implica, caso o desejo seja produzir *templates* que especifiquem quais indivíduos perderam ou adquiriram posições particulares. Por exemplo, se há o seguinte:

Carlos era o presidente. Ele foi sucedido por Antônio.

é deduzido que Antônio se tornará o presidente. Reciprocamente, se

Carlos será o presidente. Ele sucedeu Antônio.

é deduzido que Antônio era o presidente. Tais deduções podem ser implementadas por um sistema de regras, como:

deixando-trabalho(pessoa-X, trabalho-Y) & sucessão(pessoa-Z, pessoa-X)
 \Rightarrow começando-trabalho(pessoa-Z, trabalho-Y)
 começando-trabalho(pessoa-X, trabalho-Y) & sucessão(pessoa-X, pessoa-Z)
 \Rightarrow deixando-trabalho(pessoa-Z, trabalho-Y)

Isto deixará as entidades e eventos como mostra a tabela 3.5:

TABELA 3.5 – Entidades semânticas.

Entidade e1	Tipo: nome de pessoa: Carlos Rodrigues
Entidade e2	Tipo: valor da posição: Vice-Presidente Executivo empresa: e3
Entidade e3	Tipo: nome do fabricante: Dogs & Dogs SA
Entidade e6	Tipo: nome de pessoa: Antônio Farias
Evento e7	Tipo: pessoa deixando o trabalho: posição e1: e2
Evento e8	Tipo: sucede pessoa 1: e6 pessoa 2: e1
Evento e9	Tipo: pessoa começando-trabalho: posição e6: e2

que podem ser traduzidos nos *templates* mostrados anteriormente na figura 3.2.

O simples cenário mostrado aqui não exige que se leve em conta questões de tempo de cada evento. Porém, para muitos cenários o tempo é importante: quando o tempo deve ser relatado explicitamente ou a seqüência de eventos é significativa. Em tais casos, a informação de tempo pode ser derivada de muitas fontes, incluindo datas e tempos absolutos (“em 11 de janeiro de 2001”), datas e tempos relativos (“semana passada”), tempos de verbo e conhecimento sobre a sucessão natural de eventos. Considerando que a análise de tempo pode interagir com outras inferências sobre o discurso, normalmente ela será executada como parte desta fase do processo.

3.2 Testes e Avaliações

Sabendo-se que a saída de informação de um sistema de extração é uma base de dados estruturada, é possível comparar a saída de um determinado sistema com uma base de dados ideal a fim de determinar quão bem o sistema executou suas tarefas. Enquanto os testes realizados em conferências sobre Extração de Informação têm demonstrado que é possível avaliar rigorosamente alguns aspectos de um sistema de extração de informações, é difícil expor completamente os níveis de performance dos atuais sistemas de extração: a performance depende, pelo menos, da complexidade relativa da tarefa de extração, da qualidade das bases de conhecimento disponíveis, da complexidade sintática e semântica dos documentos a serem processados, e da regularidade da linguagem existente nestes documentos.

3.2.1 Métricas

Um sistema de extração de informações pode ser avaliado por sua eficiência em filtrar e selecionar as informações desejadas pelo usuário, ou seja, em outras palavras, a validade dos dados recuperados segundo o usuário. Uma recuperação eficiente depende de dois fatores principais, conforme Gerard Salton [SAL87]:

- Os itens relevantes para as necessidades do usuário devem ser recuperados;
- Os itens estranhos, ou não relevantes, devem ser rejeitados.

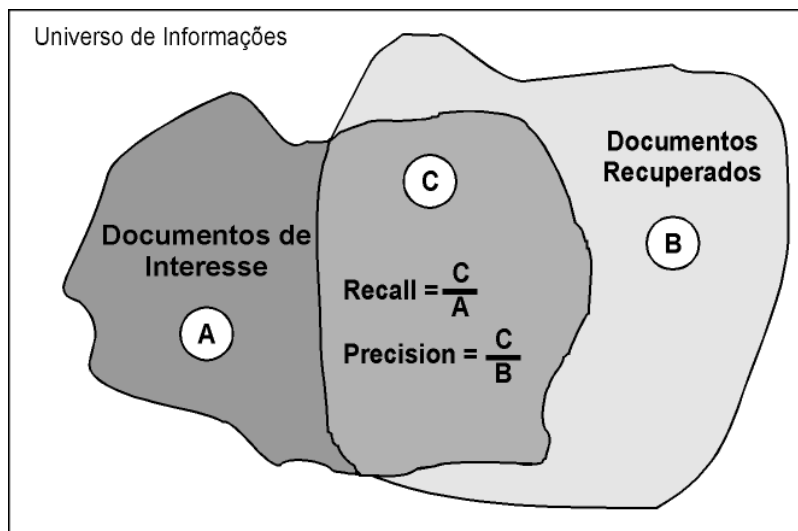


FIGURA 3.4 – Recall e Precision.

Duas medidas são normalmente utilizadas para descrever a habilidade de um sistema na recuperação dos itens relevantes e na rejeição dos itens não relevantes de uma base de dados: *recall* e *precision* (figura 3.4). *Recall* é a proporção de itens relevantes recuperados, sendo calculado pela divisão do número de itens relevantes recuperados pelo número total de itens relevantes na base de dados. Esta medida define quão completo ou compreensivo o sistema é na extração de informações relevantes [HOB96]. *Precision* é a proporção de itens

recuperados que são relevantes, sendo calculado pela divisão do número de itens relevantes recuperados pelo número total de itens recuperados. Ela define a precisão dos sistemas, ou seja, sua capacidade de extrair informações corretas.

Em princípio é preferível que um sistema produza alto índice de *recall*, pela recuperação de tudo que é relevante, e também alto índice de *precision*, pela rejeição de todos os itens que são estranhos às necessidades do usuário. No entanto, tais condições de *recall* e *precision* são conflitantes [RIL94], pois quando se tem a recuperação de muitos itens relevantes (*recall*), ocorre também a recuperação de dados não relacionados à pesquisa realizada. Por outro lado, quando a maioria dos itens recuperados é relevante (*precision*), muita informação relevante acaba não sendo recuperada. Na prática, sabe-se que *recall* e *precision* tendem a variar inversamente, pois é muito difícil recuperar tudo que é relevante, desprezando tudo o que não é relevante. A Figura 3.5 mostra a relação entre recuperação e precisão.

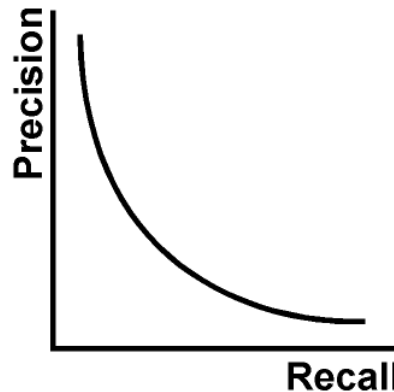


FIGURA 3.5 – Relação entre *Recall* e *Precision*.

A função de *recall* na recuperação de dados é normalmente melhor satisfeita pelo uso de termos amplos, ou de alta frequência, que ocorrem em vários documentos da base de dados. Termos com estas características podem ser usados para colocar de fora da seleção várias informações de um arquivo de entrada (ou todo o arquivo). Contudo, nesta exclusão, podem estar incluídos vários dados relevantes que deveriam ser recuperados.

Por outro lado, o fator *precision* é normalmente melhor satisfeito pelo uso de termos restritos, ou altamente específicos, que são capazes de isolar poucos itens relevantes a partir da massa dos não relevantes.

No momento de avaliar um sistema, é importante interpretar os resultados de *precision* frente ao número de textos relevantes em cada conjunto de documentos de teste [RIL94]. Se usarmos como exemplo dois conjuntos de textos, cada um contendo 100 documentos, sendo o primeiro com 69 textos relevantes, e o segundo com somente 55, um algoritmo de extração constante, que classifica todos os textos como relevantes, irá apresentar 69% de *precision* no primeiro conjunto e 55% no segundo. Neste sentido, a quantidade de informações relevantes disponíveis é fundamental para calcular-se a medida de *precision*.

Além das medidas acima, uma medida combinada, chamada *F-score*, é freqüentemente usada. Esta medida é um padrão de métrica para EI que combina as medidas de *recall* e de *precision* dentro de um único valor [HOB96]. A *F-score* aceita um valor B que ajusta o peso relativo (importância) de *recall* e de *precision* para o cálculo do valor desta medida [RIL94a]. Se $B = 1$, os pesos de *recall* e de *precision* são iguais. Se $B > 1$, *precision* é mais significativo, e se $B < 1$, *recall* tem um peso maior. Desta forma, o fator peso pode variar conforme o objetivo do sistema a ser avaliado. Caso este tenha sido desenvolvido para atuar sobre domínios específicos com alta precisão, pode-se usar $B = 2$, por exemplo, aumentando a importância da medida de *precision* na avaliação do sistema.

A fórmula para calcular o *F-score* é:

$$F = \frac{(B^2 + 1) P R}{B^2 P + R}$$

onde P é *precision*, R é *recall*, e B é o parâmetro de importância relativa entre *recall* e *precision* [HOB96].

3.2.2 Avaliações

Como foi mencionado no início deste capítulo, um aspecto que estimulou e modelou a área de EI nas últimas duas décadas foram uma série de avaliações, as *Message Understanding Conferences (MUC)* ou Conferências de Entendimento de Mensagens. Estas conferências foram criadas para avaliar e fomentar a pesquisa em análise automática de dados textuais, inicialmente, contidos em mensagens militares. Embora chamada de “conferência”, a característica que distingue os *MUCs* não são as conferências propriamente ditas, mas as avaliações às quais os sistemas dos participantes são submetidos a fim de fazerem parte destas [GRI95].

Em uma avaliação pelo *MUC*, inicialmente os participantes recebem uma descrição detalhada do domínio, ou seja, a informação a ser extraída, juntamente com um conjunto de documentos e os *templates* a serem extraídos destes documentos (os arquivos de treinamento). Os desenvolvedores de sistemas têm, então, um tempo (de um a seis meses) para adaptarem seus sistemas ao novo domínio. Depois deste tempo, cada participante recebe um novo conjunto de documentos (os arquivos de teste), usa seu sistema para extrair informações destes documentos e devolve os *templates* extraídos à organização da conferência. Os organizadores possuem o conjunto dos *templates* que deveriam ter sido extraídos (as respostas), preenchidos manualmente a partir dos arquivos de teste, e comparam com os extraídos por cada sistema, atribuindo uma pontuação que segue as métricas mostradas anteriormente.

Os *MUCs* são notáveis quanto ao seu papel de incentivo à pesquisa. Eles têm formatado substancialmente o programa de pesquisa em EI e trazido esta área para o corrente estado. Uma análise mais detalhada sobre a evolução dos *MUCs* e dos sistemas participantes, desde a sua primeira edição em 1987, pode ser vista nos trabalhos de Ralph Grishman [GRI96] e Rui Scarinci [SCA2000].

4 Extração de Informações em Correio Eletrônico

Com o surgimento da Internet, difundiu-se também o uso de um novo meio de comunicação: o correio eletrônico (*e-mail*). Inicialmente o *e-mail* era um recurso utilizado por poucos devido à limitação imposta pela pouca disponibilidade de acesso e abrangência das redes de computadores existentes. Mais tarde, com a popularização do uso dos computadores, a proliferação das redes locais e o crescimento das redes internacionais, o *e-mail* adquiriu uma abrangência mundial, sendo considerado um dos meios de comunicação mais eficientes.

Entre os benefícios oferecidos pelo correio eletrônico, destacam-se a facilidade de multiplicação de informações, a possibilidade de endereçamento de mensagens a grandes grupos de usuários e a velocidade de transmissão. Com isso, os sistemas de *e-mail* passaram de simples sistemas de transmissão e recepção de mensagens para uma importante ferramenta de divulgação e busca de informações, e também gerenciamento e delegação de tarefas. Entretanto, estes benefícios são também a origem de um dos maiores problemas no uso de *e-mail*, que é o excessivo volume de mensagens recebidas. Mike Robinson [ROB91] diz que há estimativas de que usuários regulares recebam de 30 a 100 mensagens por dia. Muitas delas não apresentam informações representativas no campo “Assunto” e outras podem não interessar ao usuário, o que dificulta a localização das mensagens relevantes. Assim sendo, muitas vezes a taxa de recebimento de mensagens ultrapassa a capacidade do usuário em analisá-las, provocando um grande acúmulo de mensagens em sua caixa postal. Este fenômeno é conhecido como *information overload* [HIL85].

David Goldberg [GOL92] diz que uma maneira para controlar grandes volumes de mensagens (figura 4.1a) é fornecer listas de mensagens, permitindo aos usuários a subscreverem-se somente às listas de seu interesse. Porém, como mostra a figura 4.1b, o conjunto de documentos de interesse de um usuário em particular raramente combina com as listas existentes. Uma solução melhor seria o usuário especificar um filtro que rastreasse todas as listas, selecionando os documentos interessantes, não importando a lista em que eles estejam (figura 4.1c). Mas David Goldberg propõem uma filtragem mais efetiva, envolvendo pessoas no processo, chamada de filtragem colaborativa (figura 4.1d). Filtragem colaborativa significa que as pessoas irão colaborar, ajudando umas as outras, executando uma filtragem e registrando seu parecer em relação aos documentos lidos. Tais pareceres podem indicar que um documento é interessante ou não. Estes pareceres, geralmente chamados de anotações, podem ser acessados por outros filtros ou pessoas.

Dependendo do objetivo e da forma de utilização dos sistemas de *e-mail* os usuários apresentam necessidades diferentes para combater o *information overload*. Wendy Mackay [MAC88] identificou nos usuários duas estratégias de utilização, nas quais eles foram classificados como: priorizadores e arquivadores. Os usuários considerados como priorizadores são aqueles interessados em limitar o tempo despendido com o uso do *e-mail*, procurando otimizar a sua utilização. Os usuários arquivadores utilizam o *e-mail* como fonte de informação e, portanto, dispõem-se a dedicar mais tempo à utilização do *e-mail* para evitar a possibilidade de deixar de ler alguma mensagem importante ou interessante. Estes

comportamentos não são mutuamente exclusivos, mas normalmente existe uma tendência a manifestar preferência por um deles.

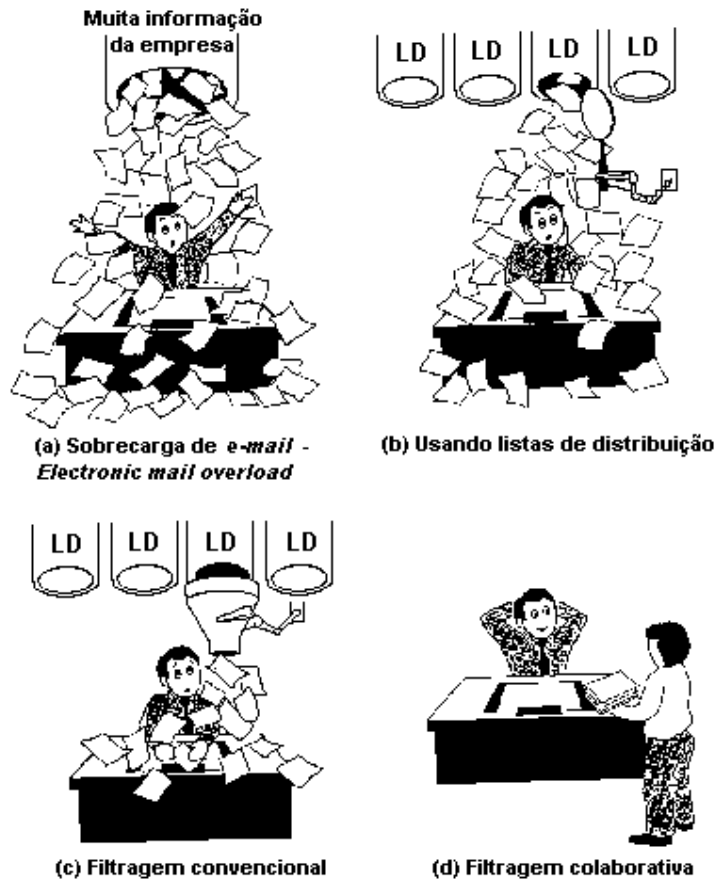


FIGURA 4.1 – Soluções para tratamento do *information overload* [GOL92].

Ainda Wendy Mackay diz que os usuários arquivadores não consideram interessante a possibilidade de eliminar automaticamente mensagens inúteis (para um usuário priorizador esta abordagem seria bastante satisfatória), através de recursos como filtragem e processamento automático, por dois motivos: (1) mensagens ora inúteis podem, com o passar do tempo, se tornar úteis e, (2) a dificuldade de caracterizar semanticamente, por meio de regras o que poderia ser considerado como uma mensagem inútil. Estes usuários afirmam: “Eu não confio em uma fórmula para organizar minhas mensagens antes de eu vê-las. Tenho medo de que elas sejam organizadas de uma forma na qual eu nunca mais vá encontrá-las. Prefiro primeiro lê-las e depois, então, organizá-las manualmente” [MAC88]. Percebe-se, então, que para estes usuários a necessidade de recursos consistentes para a classificação e o armazenamento de mensagens, com o intuito de facilitar sua recuperação posterior, é muito importante.

Devido aos problemas colocados anteriormente e com base nas necessidades dos usuários arquivadores, Simone Ferreira [FER97, FER97a] propõem mecanismos para a manipulação de grandes volumes de mensagens

eletrônicas, considerando que toda a informação recebida é relevante. Nas próximas seções esta proposta será apresentada.

4.1 Mecanismos para Recuperação e Classificação de Mensagens de Correio Eletrônico

Para evitar que o usuário tenha que descartar informações, que eventualmente poderão vir a ser úteis, devido à dificuldade na gerência de sua caixa postal, foram desenvolvidos um mecanismo de recuperação de informações e um mecanismo de classificação, através dos quais ele poderá utilizar o *e-mail* como um sistema de informações.

O mecanismo de recuperação baseia-se no uso de técnicas de recuperação utilizadas por sistemas de recuperação de textos, permitindo a recuperação de informações não estruturadas com eficiência. O mecanismo de classificação é baseado no conceito de *folder* virtual, que permite a associação lógica entre mensagens e *folders*, garantindo maior flexibilidade na classificação e recuperação de mensagens. *Folders* (unidades físicas de classificação) são pastas eletrônicas que possibilitam a organização das mensagens segundo alguma classificação estabelecida pelo usuário, onde pode ser definido um conjunto ou uma hierarquia de *folders* organizando as mensagens em diferentes categorias, onde ficam disponíveis para referências futuras.

Dois benefícios relevantes que os mecanismos propostos apresentam são destacados por Simone Ferreira:

- O mecanismo de recuperação permite que o usuário localize e recupere informações facilmente, não se limitando à recuperação de mensagens, mas permitindo também a recuperação de informações sobre as mensagens;
- O mecanismo de classificação permite a organização e a reorganização de mensagens em *folders*, permitindo a manutenção da consistência entre a intenção de classificação e o conjunto de mensagens que o *folder* representa, através da introdução do conceito de *folder* virtual.

4.1.1 Conceitos Básicos

O desenvolvimento dos mecanismos propostos foi estruturado sobre os conceitos de mensagem e *folder* virtual.

4.1.1.1 Mensagem

Segundo os padrões definidos para a Internet, descritos no RFC #822 (*Request for Comment*) [CRO82], uma mensagem consiste de cabeçalho (*header*) estruturado em campos e, opcionalmente, um corpo. O corpo é simplesmente uma sucessão de linhas que contém caracteres ASCII e está separado do cabeçalho por uma linha nula. A figura 4.2 mostra um exemplo de mensagem eletrônica.

Cabeçalho	De: <wellmade@jlonline.com> Para: <offer-mail@heuristic.untppdc.org> Assunto: OFFER: [CN] Kitchen cabinet doors Data: segunda-feira, 21 de agosto de 2000 14:28
Corpo	This ETO come from Global Commerce System, http://www.tradepost-chat.com Please visit our web site thanks. http://www.wellmadecorp.com We make and export kitchen cabinet doors and other doors. The wood is oak, ash or birch. The panel can be solid wood or mdf covered with veneer. w/ or w/o frame, coated by lacquer or uncoated as your prefer. Your designs are welcome. All samples and detail information will be sent you asap upon your request. fax:0086 25 6613758

FIGURA 4.2 – Estrutura de uma mensagem eletrônica com cabeçalho padrão.

As informações presentes no cabeçalho podem variar dependendo do sistema de *e-mail* usado. Porém, algumas informações, como as que aparecem no cabeçalho da mensagem mostrada na figura 4.2, são padronizadas pelo RFC #822. Outras informações adicionais que podem aparecer no cabeçalho de uma mensagem são mostradas na figura 4.3.

Cabeçalho	Date: 27 Aug 76 0932 PDT From: Ken Davis <KDavis@This-Host.This-net> Subject: Re: The Syntax in the RFC Sender: KSecy@Other-Host Reply-To: Sam.Irving@Reg.Organization To: George Jones <Group@Some-Reg.An-Org>, Al.Neuman@MAD.Publisher cc: Important folk: Tom Softwood <Balsa@Tree.Root>, "Sam Irving"@Other-Host,, Standard Distribution: /main/davis/people/standard@Other-Host, "<Jones>standard.dist.3"@Tops-20-Host>; In-Reply-To: <some.string@DBM.Group>, George's message Message-ID: 4231.629.XYzi-What@Other-Host
Corpo	Sam is away on business. He asked me to handle his mail for him. He'll be able to provide a more accurate explanation when he returns next week.

FIGURA 4.3 – Estrutura de uma mensagem eletrônica.

O corpo de uma mensagem padrão possui formato livre e constitui-se exclusivamente de texto desprovido de qualquer formatação. Porém, pode conter documentos formatados, gráficos, imagens, sons e arquivos anexados (*attachments*). Entretanto, a estrutura básica da mensagem permanece sendo composta de uma parte estruturada e outra não estruturada, uma vez que os objetos anexados são posteriormente extraídos das mensagens e tratados pelas ferramentas correspondentes.

4.1.1.2 *Folder*

Folders ou Unidades Físicas de Classificação são pastas eletrônicas que possibilitam a organização das mensagens segundo alguma classificação estabelecida pelo usuário [LAQ93]. Na proposta de Simone Ferreira, o *folder* não foi implementado como uma unidade física de armazenamento de mensagens. As mensagens são associadas de forma lógica aos *folders*, o que permite uma maior flexibilidade na classificação, evitando que uma mesma mensagem tenha que ser copiada para vários *folders* por atender a mais de um critério de classificação. Os *folders* são estruturas virtuais, ou seja, o usuário pode definir quantos *folders* forem convenientes sem que haja prejuízo em relação ao espaço de armazenamento. Foram definidos três tipos de *folders* virtuais:

- Manual – a classificação é feita através de manipulação direta pelo usuário, ficando implícita a definição e manutenção dos critérios estabelecidos;
- Automático – a classificação é feita através de uma expressão de recuperação, ficando explícita a definição e manutenção dos critérios estabelecidos;
- Sistema – são utilizados para o gerenciamento de mensagens, sendo sua manipulação feita exclusivamente pelos procedimentos de controle do sistema.

Além dos *folders* virtuais, um *folder* físico foi definido para o gerenciamento das mensagens descartadas, a fim de fornecer facilidades de recuperação destas mensagens quando houver necessidade.

4.1.2 Modelo de Dados

Para representação dos conceitos utilizados, foi adotado um modelo orientado a objetos. Segundo Simone Ferreira, a escolha deste modelo foi motivada pelas facilidades oferecidas para representar:

- Atributos multivalorados, necessários para a caracterização das propriedades de mensagens e *folders*;
- Propriedades opcionais, presentes nos descritores das mensagens;
- Objetos complexos, necessários para melhor representar a associação entre mensagens e *folders*;
- Comportamento associado a classes de objetos, permitindo uma melhor especificação de funcionalidades, tanto para mensagens como para *folders*.

O modelo de dados foi definido sobre três classes de objetos: Mensagens, “Lixeira” (*folder* DISCARDED) e *Folders* Virtuais. Estas classes estão descritas na figura 4.4.

A classe MSG é caracterizada pelos campos definidos no RFC #822 e pelas operações aplicadas com maior frequência sobre mensagens de *e-mail*, como enviar, descartar e responder. A classe DISCARDED representa o *folder* físico que receberá as mensagens eliminadas pelo usuário, fornecendo apenas as operações para remoção física destas mensagens ou sua reintegração à caixa postal. A classe FLD modela os *folders* virtuais definidos na seção 4.1.1.2. Por possuírem características funcionais distintas, cada um dos tipos é modelado através de

subclasses definidas para a classe FLD. A subclasse que representa os *folders* do sistema é ainda especializada em outras subclasses. A figura 4.5 ilustra a hierarquia de especializações.

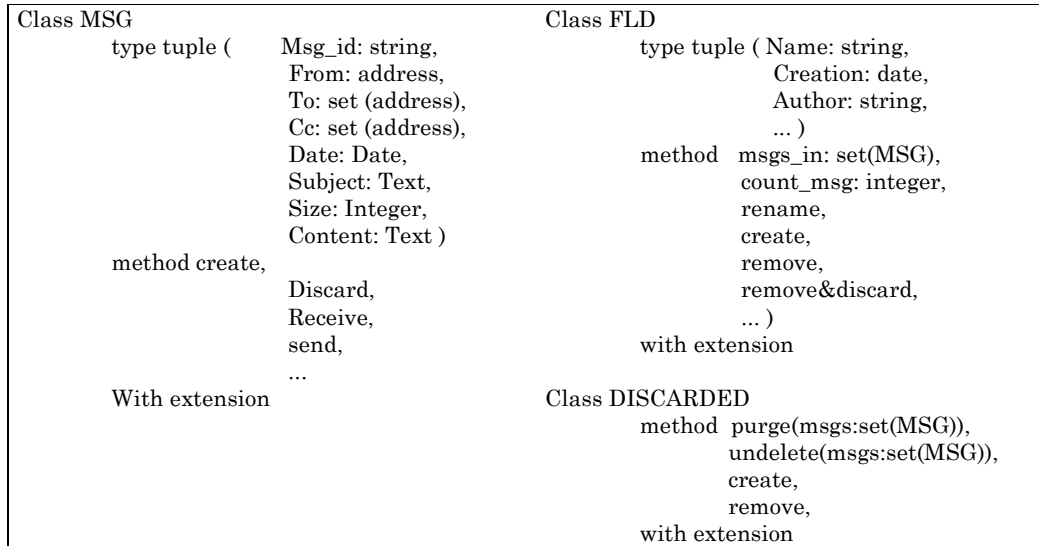


FIGURA 4.4 – Modelo de dados (classes MSG, DISCARDED, FLD) [FER97].

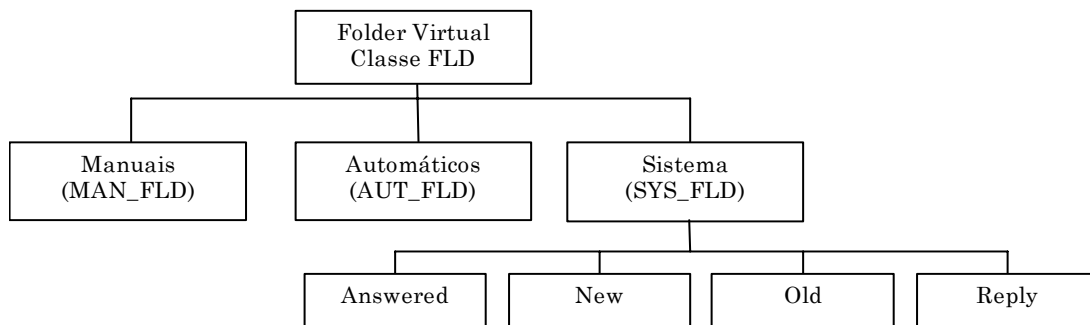


FIGURA 4.5 – Hierarquia de especialização de *folders* virtuais [FER97].

Os folders manuais, automáticos e do sistema são representados respectivamente pela extensão das subclasses MAN_FLD, AUT_FLD e SYS_FLD, ilustradas nas figuras 4.6, 4.7 e 4.8.

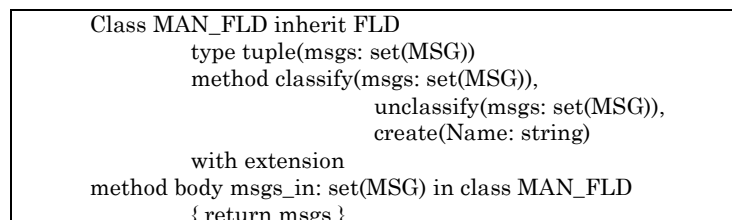


FIGURA 4.6 – Interface da subclasse MAN_FLD [FER97].

```

Class AUT_FLD inherit FLD
    type tuple(expression: Query)
    method set_expression(exp: Query),
        create_aut_fld(Name: string, exp: Query)
    with extension
    method body msgs_in: set(MSG) in class AUT_FLD
        { AUT_FLD f return expression.execute }

```

FIGURA 4.7 – Interface da subclasse AUT_FLD [FER97].

<pre> Class SYS_FLD inherit FLD Type tuple(msgs: set(MSG)) Method sys_classify(msgs: set(MSG)), sys_unclassify(msgs: set(MSG)), with extension end method body set_expression(exp: string) in class AUT_FLD { AUT_FLD f f.expression = exp; return } </pre>	<pre> Class NEW inherit SYS_FLD with extension end Class OLD inherit SYS_FLD with extension end Class REPLY inherit SYS_FLD with extension end Class ANSWERED inherit SYS_FLD with extension end </pre>
---	---

FIGURA 4.8 – Interfaces da subclasse SYS_FLD e suas especializações [FER97].

4.1.3 Mecanismo de Recuperação

O mecanismo de recuperação proposto por Simone Ferreira tem como principais objetivos minimizar as dificuldades de localização e recuperação de mensagens e ampliar as facilidades fornecidas pelas ferramentas de correio eletrônico. Para isso foi desenvolvida uma linguagem de consulta tomando como base as características estabelecidas para a linguagem O2SQL, de propósito específico e com sintaxe bastante simplificada, a qual é mostrada na figura 4.9, para o usuário descrever suas consultas.

As consultas podem ser compostas através da utilização de operadores de conjuntos sobre consultas atômicas, respeitando-se a compatibilidade entre tipos resultantes. Uma consulta atômica é estruturada em quatro cláusulas básicas (SELECT, FROM FOLDER, FROM MESSAGE e WHERE). Aos resultados das consultas podem ser atribuídos identificadores, permitindo sua posterior utilização em outras consultas, fornecendo uma estratégia de consulta baseada em sucessivos refinamentos.

- **SELECT:** na cláusula *SELECT* é especificado o tipo de resultado devolvido pela consulta. O resultado de uma consulta pode ser um conjunto de mensagens (*msglist*), um conjunto de *folders* (*folderlist*), uma lista de atributos de mensagens e/ou atributos de *folders* (*infolist*).
- **FROM FOLDER:** permite ao usuário especificar quais entre os *folders* existentes serão considerados no processamento da consulta (*flds*). Caso não seja especificado, todos os *folders* existentes serão considerados. Opcionalmente, restrições adicionais podem ser estabelecidas aplicando-se operadores relacionais, lógicos ou de texto através da sub-cláusula *WITH*. A cláusula *FROM FOLDER* é opcional, sendo necessária apenas quando são

desejadas informações sobre *folders*, com base em atributos de *folders*, ou ainda, quando se deseja estabelecer condições adicionais através do uso da cláusula *WHERE*.

- **FROM MESSAGE:** permite ao usuário especificar o conjunto de mensagens a ser considerado no processamento da consulta (*msgs*). Caso não seja especificado, todas as mensagens existentes serão consideradas. Pode-se especificar o conjunto de duas maneiras: ou usando apenas as mensagens associadas aos *folders* definidos na cláusula *FROM FOLDER*, especificadas na expressão *f.msgs_in*, ou usando-se um conjunto de mensagens obtido em uma consulta prévia (*msglist*). Assim como na cláusula *FROM FOLDER*, também podem ser definidas restrições através da sub-cláusula *WITH*. A cláusula *FROM MESSAGE* também é opcional, sendo necessária apenas quando são desejadas informações sobre mensagens, ou quando se deseja estabelecer condições adicionais através do uso da cláusula *WHERE*.
- **WHERE:** pode ser utilizada quando se deseja estabelecer condições de seleção comparando atributos de mensagens e atributos de *folders*. As condições são especificadas através do uso de operadores relacionais, lógicos, de conjunto e para recuperação de textos.

A seguir, a utilização da linguagem é mostrada através de exemplos.

- a) Quais as mensagens novas enviadas por John Smith?

```
SELECT m
FROM FOLDER f IN NEW
FROM MESSAGE m IN f.msgs_in WITH m.from HAS 'John Smith'
```

- b) Quais as mensagens classificadas no *folder* manual '*call for papers*' falam sobre inteligência artificial e foram recebidas antes de 10/10/2000?

```
SELECT m
FROM MESSAGE m IN MSG WITH m.content HAS 'AI' AND
                        m.date < 10/10/2000
```

- c) Onde está a mensagem que recebi de John na qual ele falava sobre os preços do sistema? Eu lembro que a movi para um *folder* com o mesmo nome do assunto da mensagem, mas não consigo lembrar qual era...

```
SELECT m
FROM FOLDER f IN MAN
FROM MESSAGE m IN f.msgs_in
    WITH m.content HAS ('sys*' AND '$*') AND m.from HAS 'John'
WHERE m.subject = f.name
```

Query	::=	<Single_Query> UNION <Single_Query> <Single_Query> INTERSEC <Single_Query> <Single_Query> MINUS <Single_Query>
<Single_Query>	::=	SELECT <result> FROM FOLDER f IN <flds> [WITH <fld_cond>] [FROM MESSAGE m IN <msgs> [WITH <msg_cond>] [WHERE <comp_cond>]] SELECT <result> FROM MESSAGE m IN <msgs> [WITH <msg_cond>]
<result>	::=	<folderlist> <msglist> <msg_att> <fld_att> TUPLE((msg_att fld_att)+)
<folderlist>	::=	(fld)+ /*onde fld é uma instância da classe FLD */
<msglist>	::=	(msg)+ /*onde msg é uma instância da classe MSG */
<flds>	::=	FLD MAN_FLD AUT_FLD NEW OLD ANSWERED REPLY DISCARDED named-folder-query
<msgs>	::=	MSG f.msgs_in named-folder-query
<fld_cond>	::=	(<fld_cond>) NOT <fld_cond> <fld_cond> AND <fld_cond> <fld_cond> OR <fld_cond>
<fld_cond>	::=	f.<fld_att> <rel_op> value f.<fld_att> <rel_op> f.<fld_att> f.<fld_att> HAS <text_exp>
<msg_cond>	::=	(<msg_cond>) NOT <msg_cond> <msg_cond> AND <msg_cond> <msg_cond> OR <msg_cond>
<msg_cond>	::=	m.<msg_att> <rel_op> m.<msg_att> m.<msg_att> <rel_op> valor m.<msg_att> HAS <text_exp> m.<msg_set_att> HAS <text_exp> value <set_op1> m.<msg_set_att> m.<msg_att> <set_op1> m.<msg_set_att> m.<msg_set_att> <set_op2> m.<msg_set_att> m.<msg_set_att> <set_op2> set value
<comp_cond>	::=	(<comp_cond>) NOT <comp_cond> <comp_cond> AND <comp_cond> <comp_cond> OR <comp_cond>
<comp_cond>	::=	f.<fld_att> <rel_op> m.<msg_att> m <set_op1> f.msgs
<rel_op>	::=	> < = <> >= <=
<set_op1>	::=	∈ ∉
<set_op2>	::=	⊃ ⊇ ⊂ ⊆
<text_exp>	::=	<text> <text> AND <text> <text> OR <text> NOT <text>
<text>	::=	prefixo string frase
<fld_att>	::=	Name Type Creation Count_msg Update
<msg_att>	::=	Msg_id From Date Subject Size Content
<msg_set_att>	::=	To Cc

FIGURA 4.9 – Sintaxe da linguagem de consulta [FER97].

- d) Quais são os *folders* que possuem mais de 500 mensagens?

```
SELECT f
FROM FOLDER f IN MAN WITH f.msg_count > 500
```

- e) Quais os nomes dos *folders* manuais que contêm mais de 50 mensagens com pelo menos uma mensagem recebida hoje?

```
SELECT f.name
FROM FOLDER f IN MAN_FLD WITH f.msg_count > 50
FROM MESSAGE m IN f.msgs_in WITH m.date = hoje
```

- f) Quem enviou hoje mensagens que ainda não foram lidas?

```
SELECT m.from
FROM FOLDER f IN NEW
FROM MESSAGE m IN f.msgs_in WITH m.date = hoje
```

- g) Quando John Smith enviou mensagens importantes e quais eram os assuntos destas mensagens?

```
SELECT TUPLE (m.date,m.subject)
FROM FOLDER f IN AUT WITH f.name = 'important'
FROM MESSAGE m IN f.msgs_in WITH m.from HAS 'John Smith'
```

4.1.4 Mecanismo de Classificação

O mecanismo de classificação está baseado no conceito de *folders* virtuais, apresentando as seguintes vantagens em relação ao uso de *folders* como unidades físicas de armazenamento: (1) a possibilidade de associar uma mesma mensagem a diversos *folders*, estabelecendo assim múltiplas perspectivas de classificação, sem que para isto seja necessária a duplicação da mensagem e, (2) a possibilidade de localização de mensagens sem que seja necessária a inspeção do conteúdo de diferentes *folders*, quando a forma de classificação original é desconhecida. Além destas vantagens, o uso de *folders* automáticos traz outras. Como as mensagens são associadas dinamicamente nestes *folders*, através da execução de uma expressão de consulta definida para cada *folder*, disparada somente no momento em que se deseja manipular ou examinar o conteúdo deste, a consulta devolve o conjunto de mensagens que atende aos critérios especificados naquele dado momento. Também usando *folders* automáticos há a possibilidade de alteração da estrutura de classificação de mensagens apenas através da alteração das expressões de consulta que definem estes *folders*.

Quando não for desejável explicitar os critérios de classificação, o uso de *folders* manuais possibilita a classificação arbitrária de mensagens, agrupando-as através de critérios subjetivos.

4.1.4.1 Gerência da Estrutura de Classificação

A estrutura de classificação é o conjunto de todos os *folders* manuais, automáticos e do sistema existentes em um dado momento. Para o usuário gerenciar esta estrutura foram criados métodos para criação, alteração e remoção de *folders* manuais e automáticos. Para os *folders* do sistema estas operações são disparadas por procedimentos sob controle do próprio sistema.

A seguir, a utilização dos métodos para a gerência da estrutura de classificação é mostrada através de exemplos.

- a) Criação de um *folder* manual que receberá o nome ‘urgente’:

```
MAN_FLD.create('urgente')
```

- b) Criação de um *folder* automático que receberá o nome ‘conferência’ e reunirá mensagens que possuem a palavra ‘conferência’ em seu conteúdo:

```
AUT_FLD.create('conferência',
SELECT m
FROM MESSAGE m IN MSG
WITH m.text HAS 'conferência')
```

- c) Um *folder* automático pode ser definido com base em outros *folders*, sejam eles automáticos, manuais ou do sistema. Neste exemplo, o *folder* automático ‘importante_esta_semana’ será criado com a definição de uma expressão de consulta que reúne todas as mensagens classificadas manualmente como ‘urgente’ até o momento da consulta a este *folder* e também todas aquelas classificadas automaticamente como ‘conferência’:

```
AUT_FLD.create('importante_esta_semana',
SELECT m
FROM FOLDER f IN MAN_FLD WITH f.name = 'urgente'
FROM MESSAGE m IN f.msgs_in
UNION
SELECT m
FROM FOLDER f IN AUT_FLD WITH f.name = 'conferência'
FROM MESSAGE m IN f.msgs_in)
```

- d) Este exemplo mostra como é simples redefinir a estrutura de classificação de acordo com os novos interesses do usuário. Considerando a aplicação da operação *set_expression* para o *folder* ‘importante_esta_semana’ criado no exemplo anterior, é simples alterar a semântica associada a um *folder* automático, tornando desnecessário remanejar mensagens entre os *folders* envolvidos na alteração. Depois desta alteração, sempre que o usuário fizer acesso ao conjunto de mensagens importantes para a semana, terá a sua disposição as

mensagens de seu projeto e não mais aquelas pertencentes ao *folder* ‘conferência’, como era conveniente na semana anterior:

```
set_expression(SELECT m
                FROM FOLDER f IN MAN WITH f.name = ‘urgente’
                FROM MESSAGE m IN f.msgs_in
                UNION
                SELECT m
                FROM MESSAGE m IN MSG
                WITH m.subject HAS ‘projeto X’)
```

A remoção de *folders* pode ser feita de duas maneiras: sem a remoção das mensagens associadas (método *remove* da classe FLD) ou com a remoção das mensagens associadas (método *remove&discard* da classe FLD). No primeiro caso, apenas o *folder* é eliminado e as mensagens associadas a ele no momento permanecem no sistema, mantendo os *links* existentes com os demais *folders*. No segundo caso, a eliminação do *folder* é acompanhada da remoção das mensagens por ele representadas, menos aquelas associadas a ele que tenham sido classificadas manualmente em outros *folders*, que manterão os vínculos originais.

As operações para associação e dissociação manual de mensagens têm seu uso restrito apenas aos *folders* manuais através dos métodos *classify* e *unclassify* da subclasse MAN_FLD. Embora estes métodos não desfrutem dos mesmos benefícios da subclasse AUT_FLD, a tarefa de reestruturação é facilitada pela possibilidade de aliar os recursos de consulta aos métodos *classify* e *unclassify*. No exemplo a seguir, que cria *folders* manuais a partir de outros *folders*, através dos recursos de recuperação e classificação manual pode-se estabelecer uma relação correlata àquela apresentada no exemplo c) desta seção, uma vez que o resultado de consultas sobre o conteúdo de um *folder* automático ou do sistema pode ser associado a um identificador e depois utilizado na aplicação do método *classify*.

```
X := SELECT m
      FROM FOLDER f IN ANSWERED
      FROM MESSAGE m IN f.msgs_in
      WITH m.from = ‘John Smith’
      reply_john_10/03.classify(X)
```

Neste exemplo, são explicitamente movidas para o *folder* manual *reply_john_10/03* aquelas mensagens recebidas de John Smith que já haviam sido respondidas naquele dia, no caso 10/03, criando assim uma visão estática das instâncias de MSG que satisfaziam a condição especificada no momento da consulta.

Sobre a dissociação de uma mensagem de um *folder* e a eliminação desta mensagem há diferença no que se refere à classificação manual. No primeiro caso, que usa o método *unclassify* da subclasse MAN_FLD, as mensagens são dissociadas do *folder* manual em que o método é aplicado. No segundo caso, que usa o método *discard* da classe MSG, as mensagens são dissociadas de todos os *folders* aos quais encontram-se associadas e são movidas para o *folder*

DISCARDED, onde permanecem até que sua remoção física seja determinada, ficando inacessíveis a qualquer expressão de consulta.

5 Extração de Informação de Alta Precisão

Extração de informação precisa (*accurate*) requer a maximização de ambos *precision* e *recall*. A extração de informação manual é muito precisa, mas se torna difícil quando há um grande conjunto de documentos. A extração de informação automática é mais fácil, mas tipicamente é menos precisa. Infelizmente, alguns problemas são ainda grandes e requerem uma precisão muito alta. Por exemplo, um projeto de bioinformática desenvolvido pelos autores do artigo de Rich Caruana [CAR2000] juntamente com a Universidade de Pittsburgh e a Universidade Carnegie Mellon, o qual usa aprendizado de máquina para resolver o problema do desdobramento de proteínas, requer que seja extraído vários valores numéricos importantes de milhares de arquivos do *Protein Data Bank* (PDB). O projeto requer uma taxa de erro menor que 0,1% nos valores extraídos. O esforço necessário para fazer a extração manual é enorme. Como o método de extração automática baseado em expressões regulares foi incapaz de alcançar uma precisão aceitável, o fato levou ao desenvolvimento de uma ferramenta de extração semi-automática, chamada *High Precision Information Extraction Workbench* (HPIEW) [CAR2000], que amplifica a habilidade de um especialista no domínio na extração dos valores. Usando o HPIEW, um usuário foi capaz de extrair valores de 5.000 arquivos do PDB em uma única tarde.

A extração de informações com o HPIEW é um processo interativo onde o especialista humano usa sua experiência para extrair a informação a partir de textos. A abordagem apresentada não depende de aprendizado de máquina e, ao invés, ela enfatiza o uso de um especialista humano para a tomada de decisões críticas. Enquanto isso torna o processo menos automático, também ajuda a obter o alto nível de precisão necessário ao problema.

Nas próximas seções será apresentado o HPIEW. Este método é conduzido por um especialista humano e, então, é semi-automático.

5.1 *Protein Data Bank*

O PDB é um repositório internacional único para dados, descrevendo a estrutura 3-D de macromoléculas biológicas. Ele foi estabelecido no *Brookhaven National Laboratories* em 1971 e, inicialmente, possuía sete estruturas. Durante os anos 80 começou um crescimento dramático que continua até hoje a uma taxa exponencial. Atualmente o PDB é gerenciado pelo *Research Collaboratory for Structural Bioinformatics* e contém mais de 12.000 entradas.

Cada entrada do PDB contém uma lista de coordenadas atômicas para uma entidade molecular e informações adicionais tais como procedimentos experimentais, referências bibliográficas, nomes de autores, comentários e anotações, etc. Muito dessa informação adicional foi originalmente representada como texto livre em registros de observações (*REMARK RECORDS*). O formato não-uniforme dos dados foi reconhecido por um longo tempo como o maior obstáculo na introdução da capacidade de realizar consultas avançadas no PDB.

TABELA 5.1 – *Remark records* de alguns arquivos do PDB [CAR2000].

from PDB 1ldm:					
REMARK	3	REFINEMENT. BY THE RESTRAINED LEAST SQUARES		1LDM	150
REMARK	3	PROCEDURE OF J. KONNERT AND W. HENDRICKSON		1LDM	151
REMARK	3	(PROGRAM *PROLSQ*). THE R VALUE IS 0.173 FOR		1LDM	152
REMARK	3	REFLECTIONS IN THE RESOLUTION RANGE 6.0 TO 2.1		1LDM	153
REMARK	3	ANGSTROMS. ATOMS WITH THERMAL FACTORS WHICH		1LDM	154
REMARK	3	CALCULATE LESS THAN 2.00 ARE ASSIGNED THIS VALUE.		1LDM	155
REMARK	3	THIS IS THE LOWEST VALUE ALLOWED BY THE REFIN. PROGR.		1LDM	156
from PDB 3c2c:					
REMARK	3	REFINEMENT. RESTRAINED PARAMETER LEAST-		3C2C	25
REMARK	3	SQUARES METHOD OF KONNERT AND HENDRICKSON.		3C2C	26
REMARK	3	THE FINAL R-FACTOR IS 0.175.		3C2C	27
from PDB 3cyt:					
REMARK	3	REFINEMENT. SIMULTANEOUS MINIMIZATION OF		3CYT	60
REMARK	3	ENERGY AND R-FACTOR (SEE A.JACK,M.LEVITT, ACTA		3CYT	61
REMARK	3	CRYST., V. A34, P. 931, 1978).		3CYT	62
from PDB 1cvc:					
REMARK	3	PROGRAM	PROLSQ	1CVC	20
REMARK	3	AUTHORS	KONNERT,HENDRICKSON	1CVC	21
REMARK	3	R VALUE	0.170	1CVC	22
REMARK	3	RMSD BOND DISTANCES	0.014 ANGSTROMS	1CVC	23
REMARK	3	RMSD BOND ANGLES	2.6 DEGREES	1CVC	24
from PDB 4gsp:					
REMARK	3	FIT TO DATA USED IN REFINEMENT.			
REMARK	3	CROSS-VALIDATION METHOD	: THROUGHOUT		
REMARK	3	FREE R VALUE TEST SET SELECTION	: RANDOM		
REMARK	3	R VALUE (WORKING SET)	: 0.159		
REMARK	3	FREE R VALUE	: 0.197		
REMARK	3	FREE R VALUE TEST SET SIZE (%)	: 10.0		
REMARK	3	FREE R VALUE TEST SET COUNT	: 1169		
REMARK	3	ESTIMATED ERROR OF FREE R VALUE	: 0.02		
REMARK	3	FIT IN THE HIGHEST RESOLUTION BIN.			
REMARK	3	REFLECTIONS IN BIN (WORKING SET)	: 1207		
REMARK	3	BIN R VALUE (WORKING SET)	: 0.278		
REMARK	3	BIN FREE R VALUE	: 0.265		
REMARK	3	BIN FREE R VALUE TEST SET SIZE (%)	: 11.6		
REMARK	3	BIN FREE R VALUE TEST SET COUNT	: 158		
REMARK	3	ESTIMATED ERROR OF BIN FREE R VALUE	: NULL		
from PDB 1jdo:					
REMARK	3	FIT TO DATA USED IN REFINEMENT (NO CUTOFF).			
REMARK	3	R VALUE (WORKING + TEST SET, NO CUTOFF)	: 0.1550		
REMARK	3	R VALUE (WORKING SET, NO CUTOFF)	: 0.1531		
REMARK	3	FREE R VALUE (NO CUTOFF)	: 0.2059		
REMARK	3				
REMARK	3	FIT/AGREEMENT OF MODEL FOR DATA WITH F>4SIG(F).			
REMARK	3	R VALUE (WORKING + TEST SET, F>4SIG(F))	: 0.1378		
REMARK	3	R VALUE (WORKING SET, F>4SIG(F))	: 0.1363		
REMARK	3	FREE R VALUE (F>4SIG(F))	: 0.1813		
REMARK	3	FREE R VALUE TEST SET SIZE (%, F>4SIG(F))	: 9.8		
REMARK	3	FREE R VALUE TEST SET COUNT (F>4SIG(F))	: 1361		
REMARK	3	TOTAL NUMBER OF REFLECTIONS (F>4SIG(F))	: 13842		
from PDB 1lhc:					
REMARK	3	FIT TO DATA USED IN REFINEMENT.			
REMARK	3	CROSS-VALIDATION METHOD	: NULL		
REMARK	3	FREE R VALUE TEST SET SELECTION	: NULL		
REMARK	3	R VALUE (WORKING + TEST SET)	: 0.204		
REMARK	3	R VALUE (WORKING SET)	: NULL		
REMARK	3	FREE R VALUE	: NULL		
REMARK	3				
REMARK	3	FIT/AGREEMENT OF MODEL WITH ALL DATA.			
REMARK	3	R VALUE (WORKING + TEST SET, NO CUTOFF)	: NULL		
REMARK	3	R VALUE (WORKING SET, NO CUTOFF)	: NULL		
REMARK	3	FREE R VALUE (NO CUTOFF)	: NULL		

Duas informações importantes que estão presentes nos *REMARK RECORDS* são o R e o R_{free} . O R é um fator que indica a qualidade das estruturas biológicas e o R_{free} é um outro fator introduzido mais recentemente, que é mais seguro e menos suscetível à manipulação. A extração dos valores de R e R_{free} do PDB é um pré-requisito para a comparação da qualidade das estruturas entre todas as entradas do PDB. Tal tarefa é complicada devido a duas razões: (1) Os valores são relacionados em formatos variados, como mostra a tabela 5.1. Arquivos antigos como o **1ldm** e **3c2c** contêm o valor de R embutido em texto livre e o arquivo **3cyt** nem faz referência ao valor, mas sim a uma referência bibliográfica. (2) Há variações nos métodos experimentais para refinamento das estruturas. Algumas vezes vários valores diferentes de R e R_{free} são relacionados ao mesmo experimento, representando aspectos diferentes deste experimento, como os arquivos **4gsp** e **1jdo**. Para estes dois exemplos, com a finalidade de avaliação de qualidade, um especialista selecionaria o valor de R 0.159 da linha 4 do arquivo **4gsp** e 0.1531 da linha 3 do arquivo **1jdo**. Seria difícil projetar um *parser* que pudesse identificar aquelas duas linhas corretamente entre o grande número de linhas contendo padrões de texto similares. Outras vezes um único valor de R ou R_{free} é relacionado, como no arquivo **1lhc**. Até mesmo nestes casos o método de cálculo do valor não é único e então a linha correta que contém o valor precisa ser identificada.

Desta forma, a extração dos valores de R e R_{free} do PDB traz a seguinte questão: é relativamente fácil para um especialista decidir quais valores escolher para avaliar a qualidade, porém, não é prático processar manualmente o banco de dados inteiro. Por outro lado, é difícil desenvolver um procedimento automatizado que possa extrair os valores desejados e confiáveis. Para resolver esta situação, Rich Caruana propõem uma estratégia que combine a habilidade do especialista em relação ao domínio e compreensão de texto livre com uma forma computacional semi-automatizada que aumentará a velocidade de extração que ele pode atingir no processo.

5.2 HPIEW

A abordagem fundamental para extração de informação de alta precisão é criar uma base de conhecimento que amplifique a velocidade e a precisão de especialistas humanos, quando executando a mesma tarefa. O sistema, mantendo a pessoa no ciclo, pode render uma precisão comparável a extração manual pura e ainda ser mais rápido. Além disso, o projeto permite a incorporação de algumas técnicas de extração automática.

O método depende de duas capacidades do usuário. Primeiro, o sistema depende da habilidade do usuário em gerar expressões regulares como padrões, que combinem um subconjunto de documentos que o usuário tenha selecionado. Os padrões gerados não precisam ser perfeitos. Pelo contrário, o sistema permite que o usuário interaja refinando e fazendo experimentos com cada padrão até que esteja satisfeito com ele.

Para suportar os experimentos, o sistema depende da habilidade do usuário em rastrear rapidamente tabelas/listas de itens, verificando se os padrões estão presentes em um formato apropriado. Isso permite ao usuário rapidamente

gerar, testar e refinar padrões antes de confiar neles. Além disso, o sistema não requer que um super-padrão seja criado para combinar/extrair todos documentos.

Pelo contrário, o usuário necessita somente encontrar padrões que combinem bastante com os documentos que valham a pena. A extração de um conjunto de documentos inteiro é realizada pela aplicação de uma seqüência de padrões construída pelo usuário na ordem em que eles foram criados. Isso quebra o processo de extração em porções modulares que são facilmente geradas e verificadas.

5.2.1 Abordagem Detalhada

A figura 5.1 mostra o fluxograma do processo de extração. Considerando a extração de valores numéricos para R_{free} de um grande conjunto de arquivos do PDB, a extração começa colocando todos os arquivos do PDB (cerca de 5.000) dentro de um único diretório. Cada arquivo tem um nome único que identifica a entrada no PDB. O HPIEW seleciona um pequeno número de arquivos do PDB randomicamente e mostra-os ao especialista. O especialista rastreia os arquivos do PDB mostrados e localiza o valor R_{free} desejado nas observações em cada arquivo. Uma vez que o R_{free} tenha sido identificado, o usuário constrói uma expressão regular que combina com um ou mais dos valores R_{free} . O objetivo é construir o máximo de expressões regulares específicas que irão combinar com as ocorrências de R_{free} em outros arquivos do PDB que são formatados similarmente ao formato em alguns dos arquivos mostrados. A alta especificação das expressões regulares (REGEX) é importante porque não se quer uma *regex* para combinar falsamente qualquer valor que não seja R_{free} . Se a *regex* combina, tem que combinar com um valor correto de R_{free} .

O usuário tem à sua disposição facilidades na elaboração da expressão regular que fornecem um conjunto conveniente de combinações e operações de correção. O usuário é também capaz de especificar uma faixa e tipo de checagem sobre os valores combinados. Por exemplo, ele pode especificar que o valor combinado deve ser um número e deve ter um valor entre 0,1 e 1,0. Combinações que não satisfaçam essas restrições são sinalizadas para ajudar o usuário a refinar a *regex*.

Uma vez aceitável e específica ao máximo, o sistema rapidamente aplicará a expressão regular construída para todos os arquivos do PDB no diretório. O sistema mostra com que freqüência a *regex* combina, com quais arquivos do PDB ela combina, que valores seriam extraídos de cada arquivo onde ela combina, e o texto que circunda a combinação em cada arquivo. Isso permite ao usuário rapidamente avaliar a precisão e cobertura da expressão regular proposta. Se a *regex* combina com poucos ou nenhum arquivo é necessário voltar e tentar generalizar a *regex* para, assim, combinar com mais casos. Se a *regex* produz falsas combinações, ou seja, valores combinados que não são o valor R_{free} correto em alguns arquivos do PDB, um refinamento da *regex* é necessário, de forma que ela não produza qualquer valor falso.

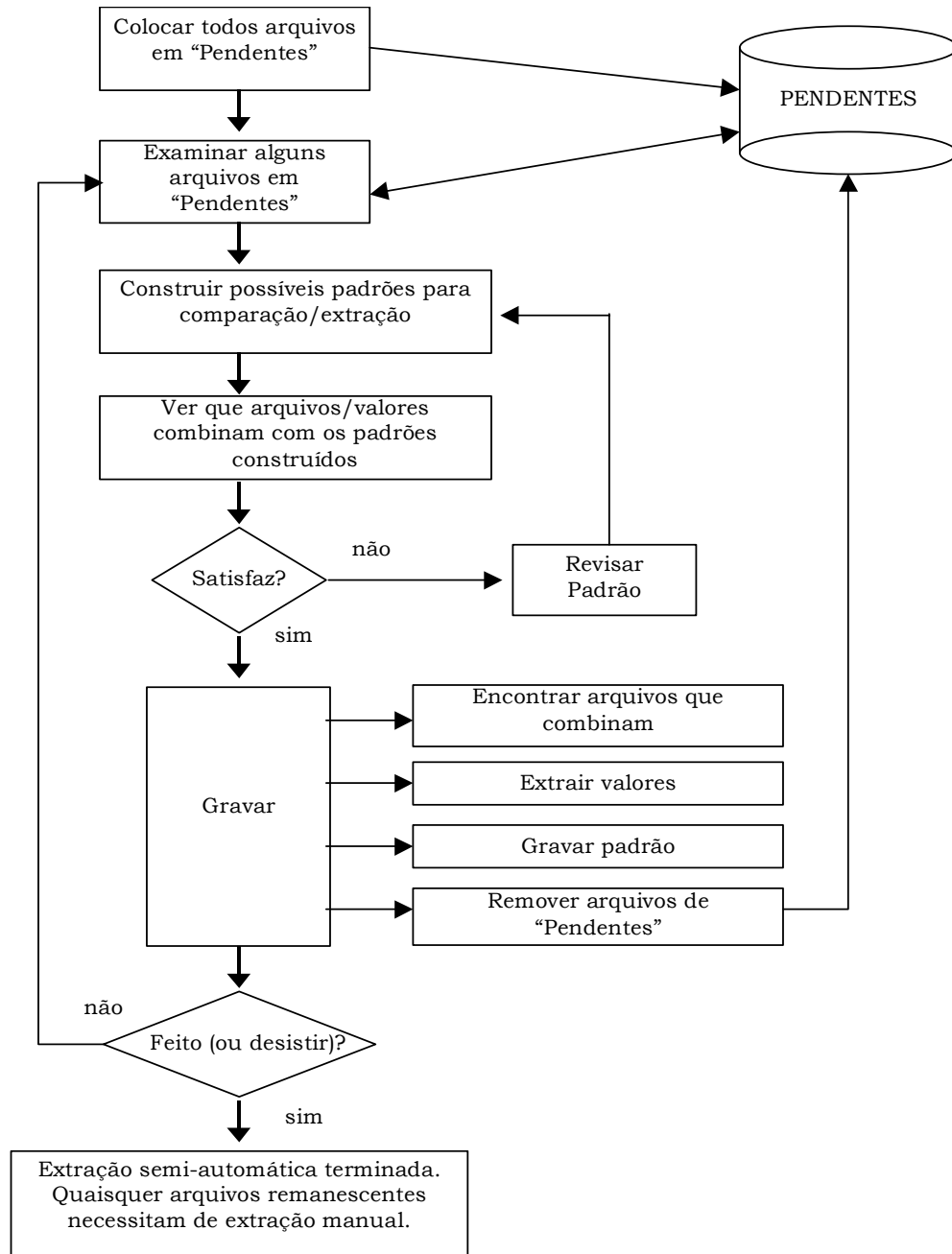


FIGURA 5.1 – Fluxograma do processo de extração com o uso do HPIEW [CAR2000].

Depois de várias repetições deste processo, o usuário converge em uma expressão regular que geralmente combina com 1% a 20% de arquivos do PDB com nenhuma combinação falsa. Uma vez satisfeito, o usuário confia nas combinações. Esta confiança dispara a execução de vários passos. Primeiro, ele encontra todos arquivos do PDB que combinam com a *regex* e extrai o valor combinado do arquivo. Os valores combinados são colocados em um banco de dados junto com o nome do arquivo do PDB. Então, depois da extração dos valores, ele remove todos arquivos do PDB do diretório de arquivos temporários que combinam

com a *regex* e os coloca em um diretório de arquivos de PDB finais. Finalmente, a *regex* é salva em um arquivo crescente de expressões regulares. Isto é feito para documentar o processo de extração e tornar possível a posterior aplicação dos mesmos passos de extração para futuras versões de PDB, quando mais arquivos estarão disponíveis.

Uma vez que uma expressão regular é gravada, todos arquivos do PDB em que ela combinou terão o valor extraído e serão removidos da lista de arquivos pendentes. Os arquivos restantes ainda serão analisados gramaticalmente e ainda ficarão pendentes. O usuário tentará encontrar uma nova expressão regular que irá combinar com os arquivos restantes do PDB. O usuário também não precisa se preocupar com a nova *regex* se ela combinar com casos que já tiveram seu valor extraído com sucesso, pois uma vez que um caso tenha combinado com uma *regex* ele será movido do diretório. Isso permite ao usuário focar-se na busca de padrões que combinem com os atuais arquivos não-combinados. Também significa que a sequência na qual as expressões regulares são aplicadas e salvas durante o progresso da extração é muito importante. Se a extração será repetida em futuros casos do PDB, deverá ser feita na mesma sequência, caso contrário combinações falsas podem ocorrer.

O usuário continua o ciclo de procura por expressões regulares e extração de valores dos arquivos pendentes do PDB que não foram combinados com sucesso com os padrões anteriores. Cada vez que o usuário grava uma nova *regex*, o número de arquivos pendentes do PDB que restam para serem processados é reduzido.

Depois que aproximadamente 10 a 20 expressões regulares tenham sido gravadas, os arquivos pendentes do PDB que restam podem ser tão diferentes entre si que é difícil achar padrões que combinem com mais alguns deles. Neste ponto não vale mais nenhum esforço para tentar inventar expressões regulares que possam somente combinar com poucos arquivos, e o usuário tem a opção de parar o processo de extração semi-automática e continuar com a extração manual.

5.3 Extraíndo Valores R e R_{free}

Nesta seção será demonstrada a aplicação do HPIEW na extração de dois valores do PDB: R e R_{free} . Inicialmente foram tomados aproximadamente 5.200 arquivos de interesse do PDB que representam estruturas de proteínas.

A tabela 5.2 mostra as expressões regulares desenvolvidas para a extração de R . Esses padrões de extração foram desenvolvidos por um biólogo e um cientista em computação trabalhando lado-a-lado usando a base de conhecimento de extração de informação. Como se observa na tabela 5.2, a linguagem usada para representar os padrões de extração é enigmática e não ajuda a interpretação. Há sutilezas escondidas nos padrões de extração que provavelmente não serão óbvias a leitores que não estejam presentes durante o desenvolvimento destes padrões, ou seja, eles são uma forma pobre de documentação.

Estas expressões regulares utilizam a sintaxe de ferramentas UNIX, como o “sed”, que lidam com texto e as aceitam como argumento de buscas. Um

exemplo de uso seria a troca de todas as ocorrências do tipo “axur” para “AXUR” em um arquivo qualquer.

Para ilustrar o processo de extração, o padrão 1 da tabela 5.2 extrai o arquivo **4gsp** do PDB da tabela 5.1, mas não combina com a entrada muito semelhante do arquivo **1lhc** da mesma tabela, pois o valor é “NULL” para aquela entrada. Neste caso o padrão 3 faz a extração para o arquivo **1lhc**. Um caso mais desafiador é representado pelo arquivo **1ldm** da tabela 5.1, que não combina com nenhum dos padrões de 1 a 9, mas é finalmente extraído pelo padrão 10 da tabela 5.2.

TABELA 5.2 – Padrões de extração desenvolvidos para extrair o valor R de arquivos do PDB [CAR2000].

```

1 s/REMARK 3 R VALUE (WORKING SET) : \([0-9\.\]*\)\.*/\1/p
2 s/REMARK 3 R VALUE \([0-9\.\]*\)\
3 s/REMARK 3 R VALUE (WORKING + TEST SET) : \([0-9\.\]*\)\.*/\1/p
4 s/REMARK 3 R VALUE (WORKING SET, NO CUTOFF) : \([0-9\.\]*\)\.*/\1/p
5 s/REMARK 3 R VALUE (NO SIGMA CUTOFF) : \([0-9\.\]*\)\.*/\1/p
6 s/REMARK 3 R VALUE (WITH SIGMA CUTOFF) : \([0-9\.\]*\)\.*/\1/p
7 s/.*THE R VALUE IS \([0-9]*\.\[0-9]*\)\.*/\1/p
8 s/REMARK 3 [ ]*R[ -]*VALUE[ ]*[:]*[ ]*\([0-9]*\.\[0-9]*\)\.*/\1/p
9 s/REMARK 3 R VALUE (WORKING + TEST SET, NO CUTOFF) : \([0-9][0-9]*\.\[0-9][0-9]*\)\.*/\1/p
10 s/.*R [ ^ z]*zREMARK 3 [ ]*VALUE IS \([0-9][0-9]*\.\[0-9][0-9]*\)\.*/\1/p
11 s/.*R[ -]*VALUE [ ^ z]*zREMARK 3 [ ]*IS \([0-9][0-9]*\.\[0-9][0-9]*\)\.*/\1/p
12 s/.*R[ -]*VALUE IS [ ^ z]*zREMARK 3 [ ]*\([0-9][0-9]*\.\[0-9][0-9]*\)\.*/\1/p
13 s/.*R[ -]*VALUE IS \([0-9][0-9]*\.\[0-9][0-9]*\)\.*/\1/p

```

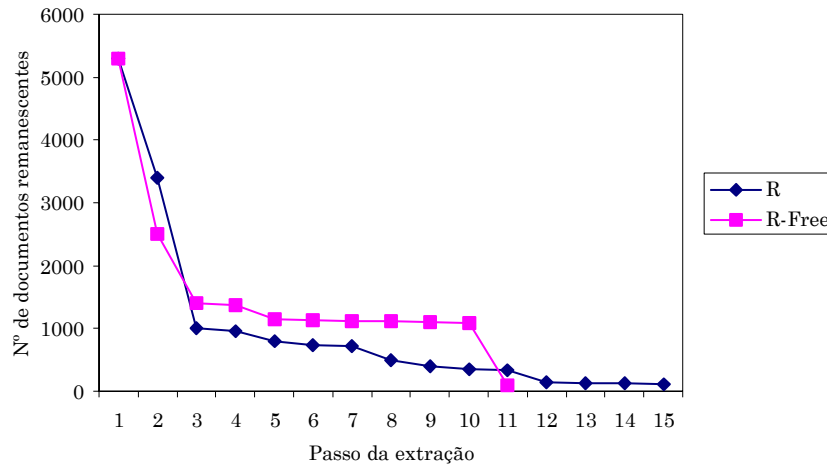


FIGURA 5.2 – Nº de documentos remanescentes X Nº de passos de extração [CAR2000].

A figura 5.2 mostra o número de arquivos pendentes do PDB que restaram depois de cada passo do processo de extração. No caso do *R*, a primeira expressão regular desenvolvida e gravada combinou com 1/3 dos documentos. Depois do segundo passo de extração somente 20% dos documentos restaram no estado pendente. Como mostrado na figura 5.2, os subseqüentes padrões de expressões regulares extraíram mais uns poucos documentos. Depois de 14 passos, 127 arquivos ainda não tinham tido seu valor *R* extraído. Eles eram tão heterogêneos, o que tornou difícil encontrar padrões que combinassem com mais do

que poucos documentos ao mesmo tempo. Depois de navegar pelos arquivos pendentes e fazer várias tentativas sem sucesso para gerar padrões utilizáveis, o usuário decide que o conjunto pendente é muito pequeno e que a extração manual é mais fácil e segura.

A principal diferença na extração do R_{free} foi devido ao fato de que apenas 1/3 dos arquivos do PDB tinham este valor gravado. Uma das razões para isso é que o R_{free} era um parâmetro que tinha sido introduzido recentemente, e uma grande parte dos arquivos tinham sido incluídos antes disso. A segunda razão é que muitas vezes os valores de R_{free} não estavam calculados, como por exemplo no arquivo **1lhc** da tabela 5.1. Então, o primeiro passo no processo de extração foi usar uma expressão regular para eliminar todos arquivos que não tivessem a palavra “*FREE*” nos *REMARK RECORDS*, deixando menos que 50% dos arquivos no estado pendente. Depois, com a aplicação de mais alguns padrões de extração, restaram poucos arquivos para serem processados manualmente.

5.4 Estimando as Medidas de *Precision* e *Recall*

Avaliar as medidas de *precision* e *recall* de um processo de extração é uma tarefa desafiadora. No HPIEW, enquanto está se desenvolvendo um padrão, são apresentadas ao usuário tabelas compactas dos valores que seriam extraídos se aquele padrão fosse gravado, bem como uma apresentação compacta do texto ao redor daquele valor extraído. Isso traz uma facilidade para os usuários avaliarem e controlarem a precisão do processo de extração enquanto ele está em progresso. Como o usuário pode examinar algumas combinações, tendo atenção e perícia apropriada, ele é capaz de manter a taxa de erro abaixo de 1%. Esta fase de extração de alta precisão, onde há a combinação repetitiva de padrões, é seguida por um processo manual dos documentos residuais. A fase manual assegura que um alto *recall* no processo de extração será mantido.

Uma maneira para avaliar a precisão do processo de extração é randomizar a amostra de documentos, extrair os valores manualmente, e comparar esses valores extraídos manualmente com aqueles extraídos usando o HPIEW. Em testes feitos pelos autores do HPIEW, para algumas centenas de arquivos PDB não foram encontrados erros nos valores extraídos. Também, nenhum valor foi perdido. A extração manual é tão tediosa e consumidora de tempo que não é prático fazer isso para uma amostra muito grande para detectar uma taxa de erro menor do que aproximadamente 0.5%, visto que as taxas de erro atuais são menores do que 0.1%.

6 Sistema de RI Baseado em Regras

As três abordagens mais comuns para recuperação de informações textuais, segundo Brian McCune [MCC85], e mostradas nos vértices do triângulo na figura 6.1, quando usadas isoladamente, sofrem problemas de *precision* e *recall*, compreensão e escopo da aplicabilidade. Sistemas de RI através de palavras-chave operam a um nível léxico e, conseqüentemente, ignoram muita informação sintática, semântica ou contextual disponível. O raciocínio que está por trás das respostas de sistemas de recuperação estatísticos é difícil de explicar a um usuário de um modo compreensível e intuitivo. Sistemas que contam com compreensão semântica da linguagem natural que está presente em documentos devem restringir o vocabulário e estilos de documento permitidos.

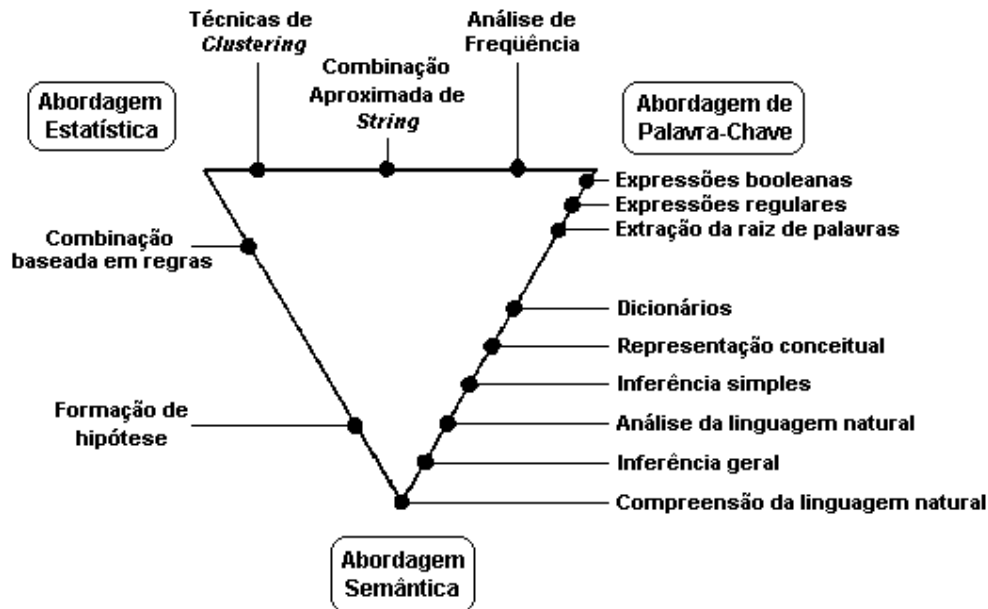


FIGURA 6.1 – O triângulo da recuperação de informações [MCC85].

Hoje, além dos especialistas terem acesso a grandes repositórios de documentos *on-line*, via redes de computadores, usuários relativamente ingênuos também podem acessá-los. Para ambas as classes de usuários, é importante que os sistemas de recuperação possuam as seguintes características:

- As consultas devem ser propostas ao nível conceitual do usuário, usando seu vocabulário de conceitos e sem requerer programação complexa;
- O número de documentos recuperados deve ser dependente das necessidades do usuário (por exemplo, restrições de tempo em lê-los);
- Uma explicação lógica e compreensível do porquê que cada documento foi recuperado deve estar disponível;
- O usuário deve ser capaz de fazer experiências facilmente a fim de controlar seus interesses ou discordâncias.

Dentro da classificação das técnicas de RI mostradas na seção 2.3, os sistemas especialistas baseados em regras são representantes clássicos de um sistema baseado em estrutura. Os sistemas baseados em estrutura necessitam de conhecimento sobre a coleção de documentos a ser pesquisada. O sistema descrito na próxima seção faz raciocínio baseado em regras, que formam a base de conhecimento sobre a coleção de documentos.

6.1 Uma Abordagem Baseada em Conhecimento

Para prover as características citadas anteriormente, Brian McCune criou um protótipo de sistema de RI baseado em conhecimento chamado RUBRIC (*RULE-Based Retrieval of Information by Computer*), integrando algumas características das três abordagens presentes na figura 6.1. As consultas são representadas através de um conjunto de regras (ver seção 7.1.4) que permitem ao usuário definir melhor os critérios de recuperação. Um exemplo de regra para a recuperação de textos que falem de terrorismo seria:

se (o texto contém o termo “bomba”),
 então (se é sobre um dispositivo explosivo)
 para grau 0.6;
 mas se também (menciona “caixa de fósforos”),
 então (reduza a força da conclusão)
 para grau 0.3

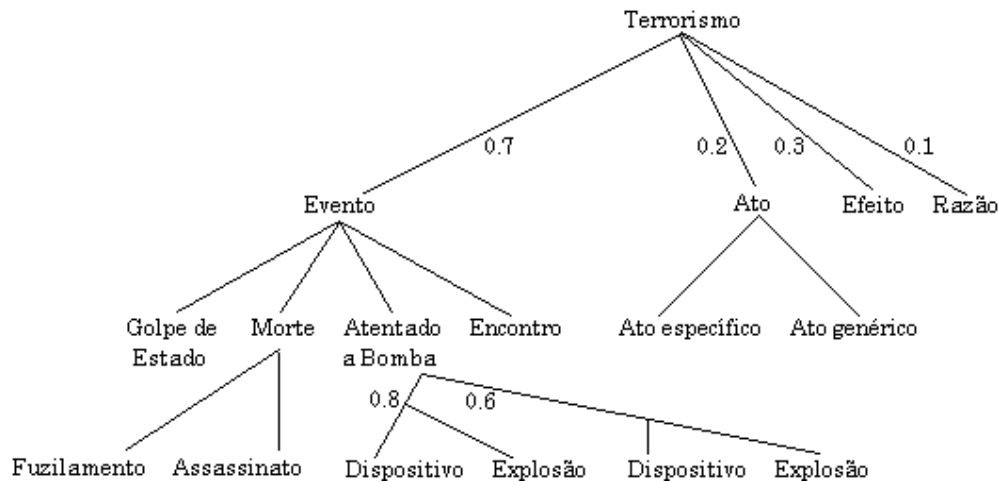


FIGURA 6.2 – Árvore de avaliação de regras para consulta sobre terrorismo.

As regras definem uma hierarquia de tópicos (ou conceitos) e sub-tópicos de recuperação. Definindo um único tópico, o usuário executa automaticamente uma busca em toda árvore montada com os sub-tópicos usados para definir este tópico. Cada regra pode ter um peso definido pelo usuário. Este peso define quanto o usuário confia na regra que indica a presença do sub-tópico da regra.

Para encontrar um texto sobre “Atos Violentos de Terrorismo”, o RUBRIC tenta construir uma representação interna de como um texto sobre este

assunto deveria ser, na forma de uma estrutura em árvore, definindo nas suas folhas que padrões de palavras deveriam estar presentes nos documentos, e em que combinações. Por exemplo, quatro elementos poderiam ser incluídos em tal representação, como é mostrado na figura 6.2: um evento violento real, um ato terrorista, o efeito que um evento pode causar e a razão do evento. Cada elemento na árvore tem um valor de relevância associado, que poderá determinar se um documento será ou não recuperado.

6.2 Expressando Tópicos de Consulta como Regras

Um exemplo de conjunto de regras que definem o tópico *World Series of Baseball* é mostrado na figura 6.3. Estas 15 regras definem um tópico principal, chamado *World_Series*, e vários sub-tópicos. Os sub-tópicos são usados para definir o tópico principal, mas também podem ser usados como tópicos de consulta deles mesmos ou como sub-tópicos de outros tópicos principais.

```
team | event => World_Series
St._Louis_Cardinals | Milwaukee_Brewers => team
"Cardinals" => St._Louis_Cardinals (0.7)
Cardinals_full_name => St._Louis_Cardinals (0.9)
saint & "Louis" & "Cardinals" => Cardinals_full_name
"St." => saint (0.9)
"Saint" => saint
"Brewers" => Milwaukee_Brewers (0.5)
"Milwaukee Brewers" => Milwaukee_Brewers (0.9)
"World Series" => event
baseball_championship => event (0.9)
baseball & championship => baseball_championship
"ball" => baseball (0.5)
"baseball" => baseball
"championship" => championship (0.7)
```

FIGURA 6.3 – Base de regras para o tópico *world_series* [MCC85].

Cada regra define uma implicação lógica, isto é, a existência do padrão no lado esquerdo da seta (\Rightarrow) indica a existência do tópico nomeado no lado direito. Assim, uma regra define o tópico ou conceito nomeado no lado direito. O lado esquerdo de uma regra é o seu corpo, que define um padrão a ser combinado. Poderá ser o tópico nomeado no lado direito de outra regra, uma expressão de referência de texto ou uma expressão composta de dois ou mais outros tópicos de regra de expressões de referência de texto, definida pelo “E” (&) ou o “OU” (|) lógico. O texto explícito a ser combinado, sem que haja interpretação adicional, é colocado entre aspas, e os nomes de tópicos e referências de texto não. O último elemento em uma regra é o seu peso, que é um número real no intervalo [0,1]. Este peso representa a confiança de quem definiu as regras, na existência de um documento com o padrão definido no lado esquerdo da regra, indicando que o documento é sobre o tópico nomeado no lado direito da regra. Se um peso é omitido, será assumido 1.0, isto é, confiança absoluta. Um peso é um número definido pelo

usuário, baseado na sua experiência e perspicácia, e não uma quantidade estatística.

Uma expressão de referência de texto pode ser uma única palavra-chave ou frase, ou um contexto léxico dentro do qual devem ser achadas duas palavras-chave ou frases (por exemplo, palavra adjacente, mesma oração e mesmo parágrafo). Assim, por exemplo, a pessoa pode especificar que dois padrões só são de interesse se eles acontecem na mesma oração.

As regras também podem definir, freqüentemente, termos, frases e ortografias para o mesmo conceito. Assim, as regras fornecem um dicionário hierárquico simples, com pesos variáveis que definem o grau de certeza com que uma variável particular é combinada. Por exemplo, “*St.*” em inglês é usado como a abreviação para ambas “*Saint*” (São) e “*Street*” (Rua), e assim “*St.*” tem seu peso menor do que a palavra-chave “*Saint*”, como é mostrado na figura 6.3. As regras também podem ajudar na recuperação de informações em múltiplas línguas. Se o banco de dados contém texto em múltiplos idiomas, então o nível mais baixo das regras pode definir sinônimos em cada idioma de interesse. As regras dos níveis mais altos na hierarquia permanecem inalteradas.

6.3 Processamento de Consultas

Um conjunto de regras define uma hierarquia lógica de tópicos e sub-tópicos de recuperação, conforme mostra a figura 6.4, formando uma árvore de padrões de recuperação. O nodo raiz desta árvore representa um tópico ou conceito semântico que o usuário quer recuperar. Os nodos-pai abaixo da raiz representam tópicos intermediários, com os quais o tópico raiz está definido, e os nodos-folha da árvore representam padrões de palavras que serão procurados no banco de dados. Cada arco da árvore, que é a linha que liga dois nodos, tem um peso que, combinado com os pesos dos tópicos intermediários e expressões de palavras-chave, contribui na formação do grau de confiança global, indicando que o tópico raiz também foi encontrado no texto (arcos sem valor têm um peso implícito de 1.0). Arcos que representam uma expressão “E” são unidos próximo a sua base comum, como nos tópicos *baseball_championship* e *Cardinals_full_name* da figura 6.4.

Tomando como base as figuras 6.3 e 6.4, será descrito a seguir como o RUBRIC processa uma consulta. Quando o usuário fizer uma consulta conceitual por *World_Series*, o RUBRIC procura na sua base de regras por todas as regras que tenham definições para este tópico, ou seja, que tenham *World_Series* no seu lado direito. Há somente uma regra na figura 6.3 que tem *World_Series* no lado direito. Assim, o RUBRIC expande aquela regra de acordo com o seu lado esquerdo. O resultado é que os nodos *World_Series*, *team* e *event*, na figura 6.4, são criados, bem como os dois arcos entre eles. Como *team* e *event* são nomes de tópicos e não padrões textuais, o RUBRIC procura na sua base de regras as definições para eles. Este processo continua recursivamente, até que todos os nodos-folha da árvore contenham padrões textuais.

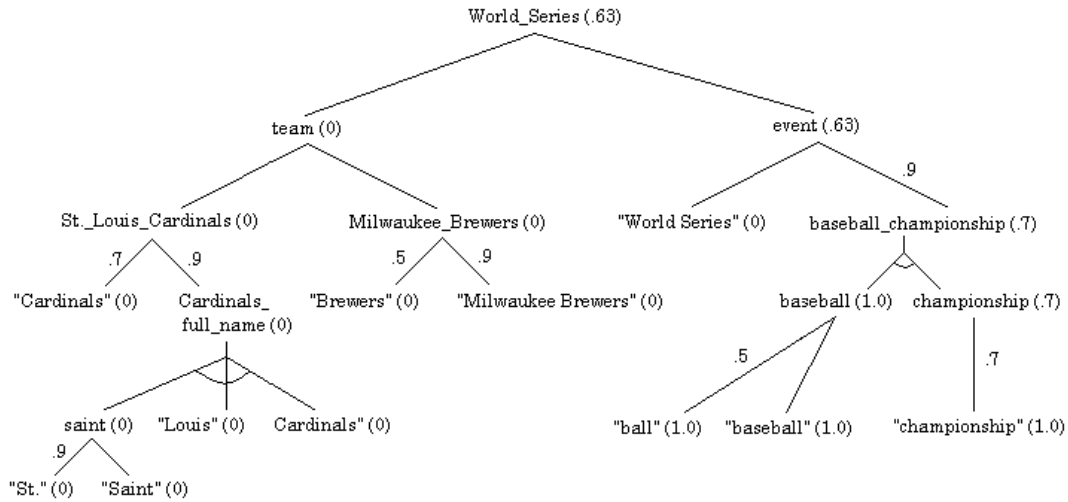


FIGURA 6.4 – Árvore de avaliação de regras para o tópico *world_series* [MCC85].

Neste ponto, cada documento no banco de dados é comparado com todas as frases dos nodos-folha da árvore. Para um determinado documento, se uma frase é encontrada em algum lugar do documento, é atribuído o valor 1.0 (um) ao nodo correspondente na árvore, caso contrário é atribuído o valor 0 (zero). Então, os pesos dos nodos-folha são combinados e propagados aos níveis superiores da árvore, para determinar o peso global a ser nomeado para este documento.

Por exemplo, se um documento contém as palavras “ball”, “baseball” e “championship”, e nenhuma outra palavra é referenciada na base de regras, então os nodos da árvore seriam nomeados com os pesos mostrados entre parênteses na figura 6.4. Os nodos-folha “ball”, “baseball” e “championship” recebem peso 1.0, e todos os outros nodos-folha recebem peso zero. O nodo *baseball* recebe o peso 1.0 porque é o máximo entre 1.0 multiplicado por 0.5 e 1.0 multiplicado por 1.0. Semelhantemente, o nodo *championship* recebe o valor 0.7. O nodo *baseball_championship*, que é um nodo “E”, recebe o peso 0.7, que é 1.0 multiplicado pelo mínimo entre 1.0 e 0.7. O nodo *event* recebe o peso 0.63, que é o máximo entre zero multiplicado por 1.0 e 0.7 multiplicado por 0.9. Desde que não exista nenhuma palavra-chave no documento que dê suporte ao sub-tópico *team*, o peso global do tópico *World_Series*, neste documento, é 0.63 (1.0 multiplicado pelo máximo entre zero e 0.63).

Depois de rastrear todo banco de dados, cada documento que foi recuperado tem um peso associado que representa a confiança do sistema de que o documento é relevante em relação ao tópico definido pelo usuário. Uma necessidade que o usuário tem é de visualizar somente os documentos com peso mais alto. O RUBRIC ordena estes documentos em ordem decendente, baseado em seus pesos, e os agrupa aplicando técnicas de *clustering* aos pesos. Então, são apresentados ao usuário os documentos que estão situados em um *cluster* que contém pelo menos um documento com um peso dentro do limite mínimo fornecido pelo usuário (por exemplo, 0.8 ou mais).

O RUBRIC pode explicar porque um documento em particular foi recuperado. Esta capacidade é muito importante para que o usuário confie mais nos resultados obtidos, ajudando também na aquisição de experiência para as operações futuras do sistema na elaboração de novas regras de recuperação. Para isso, o RUBRIC pode exibir cada regra que resulta em um peso diferente de zero, que será propagada aos níveis superiores, e o valor do peso. O RUBRIC também pode mostrar cada tentativa de combinação de uma palavra ou frase com o documento, juntamente com o resultado dizendo se houve ou não combinação.

Sistemas como o RUBRIC permitem a construção de modelos próximos do mundo real e assim oferecem consultas de alta precisão. Porém, uma desvantagem destes sistemas é a necessidade da descrição da base de regras, que é construída manualmente.

7 Sistema de Extração Semântica de Informações

Uma pessoa pode adquirir conhecimento a partir de textos identificando e memorizando as informações relevantes. A automatização desta atividade é bastante complexa. A busca do conhecimento a partir de bases de dados não estruturadas e semi-estruturadas (BDNE/SE), como textos, exige o entendimento dos dados armazenados, transformando-os em informação de forma automática. Desta maneira, um processo de extração da informação é necessário a fim de simular, com a maior proximidade possível, o papel que a pessoa sozinha desempenha na busca das informações relevantes.

Neste sentido, Rui Scarinci [SCA97, SCA97a] propõe um método de extração do conhecimento (SES – Sistema de Extração Semântica de Informações) não visando a construção de um processo que entenda a linguagem natural, mas, ao contrário, visando através da convergência das técnicas de sistemas baseados em conhecimento (SBC) e sistemas de extração de informações, uma maneira automatizada para auxiliar o usuário na extração de informações a partir de BDNE/SE. Este processo foi chamado de “Extração Semântica”, visto que apresenta ao usuário um conjunto de dados menor que o armazenado, permitindo-lhe avaliar e entender o conteúdo das bases de dados armazenadas, não tendo como objetivo o entendimento semântico de um texto.

A extração semântica fornecerá ao usuário os dados contidos nas BDNE/SE conforme uma filtragem prévia, seleção e processamento destes dados. Com isto, o método de extração é, na realidade, auxiliado pelo próprio usuário, pois a análise e o entendimento dos dados resultantes do sistema ficam a cargo deste. É com este objetivo que o próprio conhecimento do usuário de como extrair as informações relevantes a partir dos arquivos de entrada é utilizado: “conhecimento para extrair conhecimento” [SCA97].

7.1 Sistema Baseado em Conhecimento

Segundo Howard Turtle [TUR91], sistemas de recuperação e extração de informações muitas vezes requerem a adaptação de modelos de inferência de sistemas baseados em conhecimento para a tarefa de análise de documentos. A partir disto, em um SBC tem-se um conjunto de “Eventos de Entrada” que, processados por uma “Máquina de Inferência”, geram uma “Conclusão”, a qual é extraída a partir dos conhecimentos armazenados na “Base de Conhecimento”, como mostra a figura 7.1.

7.1.1 Máquina de Inferência

A máquina de inferência é representada por um módulo que realiza a extração semântica. Este módulo tem por finalidade analisar os arquivos de entrada (BDNE/SE) conforme o conteúdo das bases de conhecimento definidas pelo usuário. Como resultado, será gerado um arquivo de saída.

A máquina de inferência apresenta entre suas funções, a tarefa de interpretar regras, as quais servem para armazenar o conhecimento necessário à extração. Tais regras contêm conhecimento sobre características léxicas e

sintáticas que definem as informações a serem selecionadas e filtradas a partir dos arquivos de entrada.

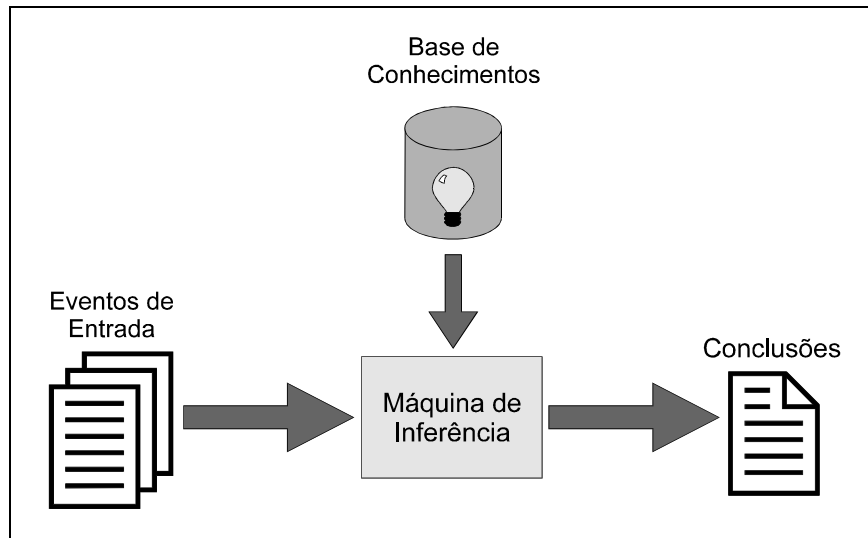


FIGURA 7.1 – Arquitetura Genérica de um Sistema Baseado em Conhecimento [SCA97].

7.1.2 Eventos de Entrada

Os eventos de entrada são representados pelo conjunto de arquivos de entrada que compõem a base de dados a ser analisada pela máquina de inferência. Cada arquivo contém um conjunto de características e informações que serão analisadas conforme as definições do usuário, contidas na base de conhecimento, as quais darão margem ao processo de inferência para filtragem e seleção da informação.

7.1.3 Conclusão

A conclusão é caracterizada por um arquivo de saída. Este arquivo no formato texto contém o resultado da análise de todos os arquivos que compõem a base de dados de entrada. Desta forma, dentro do arquivo de saída existirão as informações filtradas e selecionadas pela máquina de inferência de maneira estruturada.

7.1.4 Base de Conhecimento

A base de conhecimentos é a parte mais importante de um SBC, pois ela contém os conceitos básicos do domínio, relações, fatos e regras usadas no processo decisório. Por isso, deve ser de boa qualidade, completa e íntegra. Ela é composta por um conjunto de arquivos de dados que armazenam o conhecimento de como realizar a extração de informações a partir de BDNE/SE. É possível criar diversas bases de conhecimento, podendo assim, definir múltiplos domínios de extração, utilizados conforme as necessidades do usuário.

A representação do conhecimento de como filtrar e selecionar as informações dos arquivos de entrada é realizada de uma forma procedimental. Uma técnica comum para a representação de conhecimento é através de regras. Esta representação provê um conjunto modular de regras na forma situação/ação [WEI88]. Um exemplo de representação comum e muito natural de expressar conhecimento são regras do tipo “Se [condição] Então [ação]”. A figura 7.2 mostra um exemplo de regra aplicada no processo de EI dos *e-mails* do ETOS.

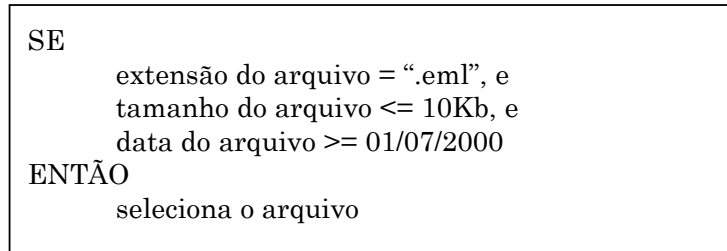


FIGURA 7.2 – Regra para expressar conhecimento.

As regras, na realidade, representam conhecimentos empíricos, gerados a partir do conhecimento do usuário. Estes conhecimentos referem-se a utilização de um grupo de características léxicas e sintáticas exploradas para a seleção das informações de entrada. Desta forma, o sistema processa a análise dos textos levando em conta o conhecimento envolvido na leitura de um texto por um ser humano.

7.2 Extração de Informações dos Arquivos de Entrada

Com o objetivo da exploração léxica e sintática do arquivo de entrada através do uso do conhecimento do próprio usuário do sistema em extrair dados, as regras armazenadas na base de conhecimento permitem a configuração do extrator semântico para a extração das informações desejadas.

Quanto às características léxicas, o extrator semântico buscará obter o conhecimento a partir de termos, ou seja, as palavras que compõem os arquivos de entrada. Assim, o texto é percorrido palavra a palavra, analisando-as uma a uma. As análises realizadas para a extração das características léxicas dos arquivos de entrada são:

- **Simple ocorrência:** cada palavra extraída do texto deve ser comparada com a base de dados definida pelo usuário, verificando se esta palavra serve para identificação e seleção do conteúdo do texto. Desta forma, a simples ocorrência de uma palavra pode auxiliar na extração de informações armazenadas nas BDNE/SE, trazendo consigo um significado semântico importante dentro do contexto global do texto;
- **Características léxicas:** existem cadeias de caracteres ou palavras cujo significado só pode ser extraído a partir de um processo de análise mais profundo, caractere a caractere, e/ou realizando a comparação com padrões pré-definidos de palavras. Por exemplo, a cadeia de caracteres “10/10/2000”, nesta formatação, pode ser identificada como uma data, atribuindo-lhe um importante conhecimento semântico;

- **Termos compostos:** muitas palavras podem ter seu significado obtido de forma correta somente quando combinadas com palavras posteriores ou anteriores. Individualmente, elas podem não apresentar um significado relevante na extração do conhecimento ou até podem gerar uma análise errônea.

O objetivo da análise das características sintáticas é encontrar as propriedades estruturais de como um termo individual está inserido dentro de um documento, ampliando seu significado através da extração das informações sintáticas associadas a ele. Várias palavras iguais podem ocorrer em documentos diferentes, combinadas de formas distintas, ou seja, dois textos ou duas sentenças contêm os mesmos termos mas não se referem ao mesmo conceito. Neste caso, o significado de cada uma das ocorrências destes termos depende muito da relação entre os mesmos em cada ocasião e da relação com outros termos próximos. A cadeia de caracteres “10/10/2000”, identificada como uma data, deve ter um significado no contexto global do texto sob análise. Com o uso das características sintáticas, pode-se associar esta data com a palavra que a antecede, por exemplo, atribuindo-lhe um significado semântico maior. O mesmo ocorre com palavras que aparecem muitas vezes no texto, e seu significado será melhor definido quando houver a associação com outras palavras dependentes, como: Data de Inscrição, Data de Entrega ou Data de Avaliação.

Existe uma interdependência entre os procedimentos envolvidos na análise léxica e sintática do texto. O analisador sintático é requisitado pelo léxico, sempre que este identificar no arquivo de entrada um termo dado como relevante pelo usuário através da base de conhecimento. No exemplo da data, o analisador léxico recebe do leitor de arquivos a palavra “10/10/2000”, destacando-a como relevante e realizando a interpretação do significado semântico associado ao termo. A partir deste significado inicial, o analisador léxico requisita uma análise sintática, ampliando o entendimento semântico do termo inicial e de outros termos relacionados a este.

Durante a análise do texto, as palavras que devem ser encontradas estão armazenadas na base de conhecimento. Tais palavras podem ser declaradas de duas formas: diretamente nos comandos de representação do conhecimento (Se [palavra] então [ação]) e através do uso de uma base de dados, denominada Dicionário de Palavras do Sistema, composta por um conjunto de arquivos onde estão armazenadas todas as palavras identificáveis pelo analisador léxico. Cada um destes arquivos pode conter um grupo de palavras relacionadas entre si, como por exemplo, nome de países, formando um *thesaurus*.

Outra análise a ser feita nos arquivos de entrada antes de analisá-los internamente, chamada de avaliação semântica prévia, é em relação ao seu nome. Cada um dos arquivos analisados é separado de acordo com sua extensão de nome, por exemplo, DOC, TXT, PS, EML e outras, criando grupos de arquivos que viabilizam o uso de regras de extração específicas para cada um dos tipos, aumentando a eficiência do processamento.

Para a avaliação semântica prévia é definida uma base de conhecimento auxiliar, com o objetivo de guardar informações, associando as extensões dos nomes de arquivos de entrada com as bases de conhecimento

(conjunto de regras) correspondentes, necessárias ao processo de extração. A tabela 7.1 mostra um exemplo desta base de conhecimento auxiliar, onde o próprio usuário fornece os dados.

TABELA 7.1 – Associação entre extensões de nomes de arquivos e bases de conhecimento.

Extensão	Arquivo de Conhecimento
C	Conhecimento para arquivo fonte da linguagem C
EML	Conhecimento para arquivo de <i>e-mail</i>
HTM	Conhecimento para arquivo HTML
TXT	Conhecimento para arquivo texto

Além da avaliação semântica prévia há mais uma fase de pré-processamento sobre os arquivos de entrada, onde eles poderão ser expandidos e convertidos. A expansão é realizada em arquivos que contêm outros arquivos inseridos dentro de sua estrutura, como por exemplo os arquivos comprimidos no formato ZIP e ARJ. Já outros arquivos, como por exemplo os arquivos *PostScript*, devem ser convertidos para o formato texto, a fim de possibilitar a extração de informações. Este processo continua recursivamente até não encontrar mais arquivos a serem expandidos e convertidos.

7.3 Arquitetura do SES

A fim de atingir os objetivos desejados para a extração de informações, Rui Scarinci definiu a arquitetura da sua proposta e, com base nela, implementou o sistema SES. A figura 7.3 mostra a divisão dos módulos, os quais refletem muito aproximadamente a estrutura trivial de um sistema baseado em conhecimento.

- **Arquivos de entrada:** são os vários arquivos de dados com as informações armazenadas de forma textual não estruturada ou semi-estruturada, que serão analisados e processados pelo extrator semântico, sendo selecionados pelo próprio usuário do sistema;
- **Arquivo de saída:** é um arquivo estruturado e em formato texto que contém o resultado da análise e processamento dos arquivos de entrada. Este arquivo pode ser importado por outros sistemas que utilizarão as informações extraídas;
- **Extrator semântico:** é o núcleo do SES, sendo responsável pela extração dos dados armazenados nos arquivos de entrada e geração do arquivo de saída. A extração é realizada conforme as regras armazenadas na base de conhecimento;
- **Classificador:** responsável pela avaliação prévia dos arquivos de entrada, antes de serem processados pelo extrator semântico. O classificador verifica a extensão do arquivo de entrada, identificando seu tipo e selecionando o conjunto de regras de extração, que está na base de conhecimento, ao qual será submetido;

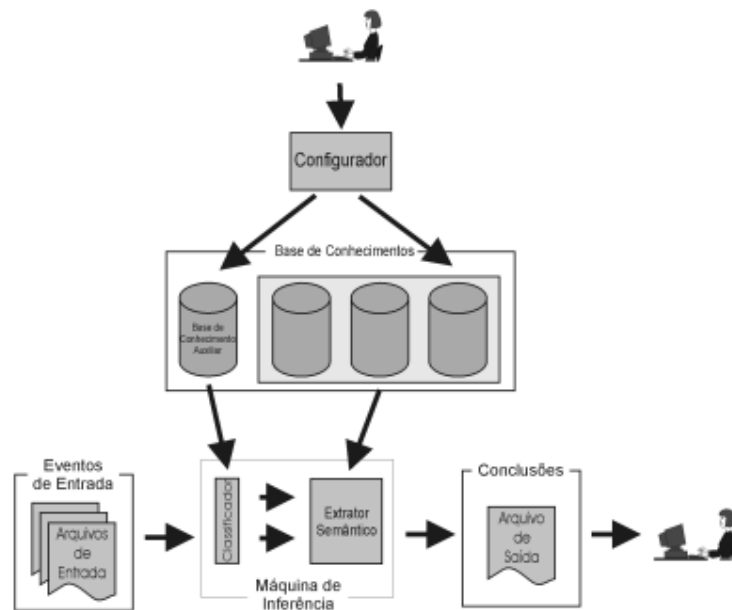


FIGURA 7.3 – Arquitetura do Sistema de Extração Semântica de Informações [SCA97].

- **Base de conhecimento:** conjunto de arquivos nos quais são armazenadas as regras de extração configuradas pelo usuário. Cada arquivo poderá se aplicar a um determinado tipo de arquivo de entrada, sendo sua seleção feita pelo classificador;
- **Base de conhecimento auxiliar:** é formada apenas por um arquivo, no qual é armazenado o relacionamento entre os arquivos de regras e os arquivos de entrada, usado pelo classificador na seleção do arquivo de regras definido conforme a extensão do arquivo de entrada;
- **Configurador:** módulo que permite a geração da base de conhecimento, bem como sua manutenção, através de uma interface de acesso a tal conhecimento;
- **Usuário:** caracterizado pela pessoa que configura o sistema ou usufrui os resultados do mesmo. Pode haver dois tipos de usuário, com papéis distintos, os quais podem ser assumidos pela mesma pessoa. O primeiro tipo é o configurador do sistema, que é responsável pela geração da base de conhecimento, devendo compreender a forma de representação do conhecimento usada no sistema, bem como ter um raciocínio formal com relação à estruturação deste. O segundo tipo é o usuário final, que somente ativa o extrator semântico, indicando quais arquivos devem ser analisados para obter o arquivo de saída.

7.4 SIRI

O projeto denominado Extração Semântica de Informações, que iniciou com o sistema SES [SCA97], já produziu outros trabalhos. Um deles é o Sistema Inteligente de Recuperação de Informações (SIRI) [BEC97]. O SIRI, em combinação com o SES, busca através de um processo automatizado facilitar a manipulação dos dados armazenados a nível local. Suas características são:

- **Realiza a seleção dos arquivos de entrada para o SES:** a partir dos critérios de seleção fornecidos pelo usuário, como por exemplo, uma ou mais palavras-chave, o SIRI seleciona somente os documentos ou textos que validem tais critérios. Isto diminui significativamente a quantidade de arquivos de entrada no SES, aumentando a velocidade de extração;
- **Trabalha com diversos formatos de arquivos de dados:** através de conhecimentos específicos de cada arquivo, pré-definidos nas bases de conhecimento, é possível trabalhar com diversos formatos de arquivos, como por exemplo, arquivos compactados e/ou em formato diferente do ASCII;
- **Apresentar uma saída de dados compatível com o SES:** gera os dados de saída no formato ASCII, permitindo uma integração transparente com o SES;
- **Classificar os arquivos por assuntos:** permite ao SES trabalhar apenas com os arquivos relacionados com determinado assunto;
- **Melhorar a eficiência e a eficácia do SES:** auxilia no processo de extração de informações, permitindo uma melhora no tempo de extração, bem como o aumento da qualidade dos dados extraídos.

Conforme mostra a figura 7.4, o SIRI tem como uma de suas principais características a interatividade com o usuário. Os resultados obtidos através desta interação permitem a este sistema realizar um processo de desdobramento e conversão sobre os mais variados tipos de arquivos para, posteriormente, selecionar apenas os arquivos de interesse do usuário.

O SIRI se divide em três fases distintas:

- a) **Processo de identificação:** realiza a identificação do formato do arquivo, permitindo ao sistema selecionar, na base de conhecimentos, a regra específica para o tratamento do mesmo. A base de conhecimentos utilizada neste processo é criada com informações fornecidas pelo usuário. Para cada tipo de arquivo existe uma regra associada que indica um programa (externo ao sistema) para tratá-lo;
- b) **Processo de tratamento de arquivos:** utiliza a regra identificada na fase anterior, efetuando chamadas aos programas externos que deverão ser capazes de tratar o arquivo. Ao final deste processo, todos os arquivos processados estarão fisicamente separados e no formato texto;

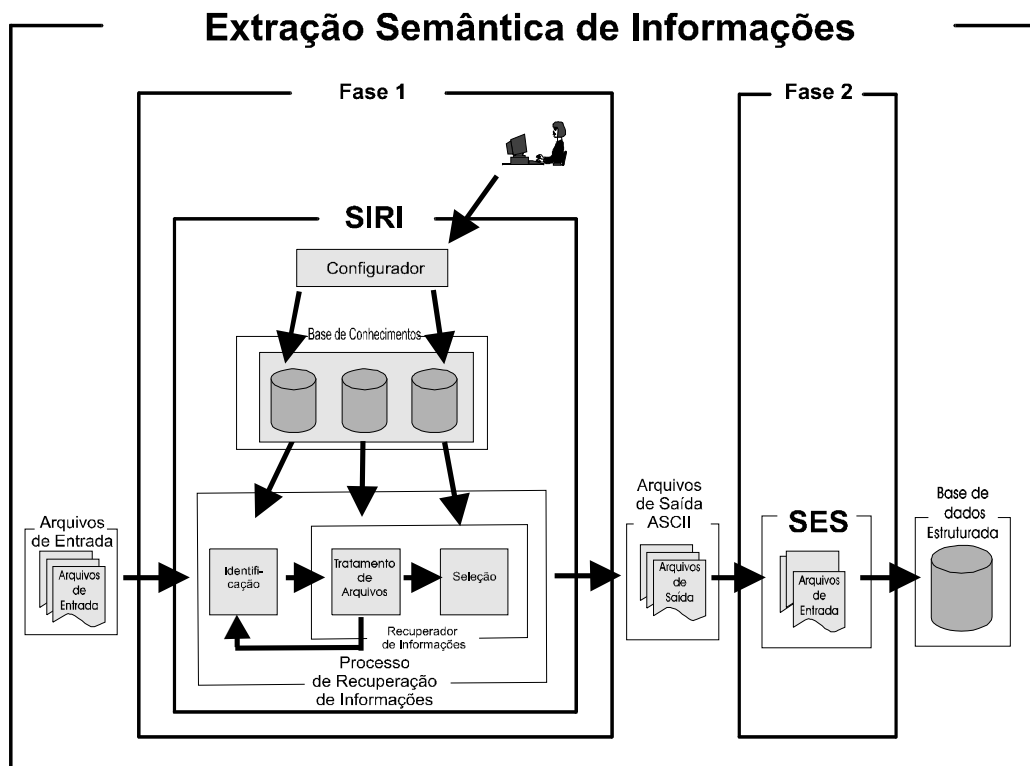


FIGURA 7.4 – Extração Semântica de Informações [BEC97].

- c) **Processo de seleção:** é realizada uma filtragem nos arquivos tratados na fase anterior. Para isso, são feitas consultas à base de conhecimentos, conforme critérios informados pelo usuário (dados do problema). Esses critérios estarão relacionados a assuntos de interesse do usuário, sendo que para cada assunto existe uma base de conhecimentos associada. Por exemplo, ao tópico “CONGRESSO” poderia estar associado o critério: selecionar o arquivo se a palavra “CONGRESSO” for referida no texto pelo menos 4 vezes. A seleção também pode ser feita por um ou mais tipos de arquivos, sendo que estes também são informados pelo usuário.

Cabe salientar que o SIRS cria uma tabela com o nome do arquivo de saída, o assunto em que o arquivo de saída está classificado, o nome do arquivo de entrada que originou o arquivo de saída selecionado e o diretório onde ele se encontra, para que estes fiquem disponíveis ao SES.

7.5 Extração em Múltiplos Níveis

Com a grande expansão da computação, ocorrida na última década, a descoberta eficaz e eficiente de informações interessantes em grandes bases de dados tornou-se essencial. Diversas técnicas de recuperação, extração e descoberta de conhecimento emergiram como uma solução para o problema da análise de dados enfrentado por muitas organizações. Contudo, os estudos anteriores nestas áreas foram focados no tratamento de informações em um único nível conceitual: o

mais primitivo ou o mais alto. Entretanto, muitas vezes há o desejo de descobrir conhecimento em múltiplos níveis conceituais, fornecendo uma compreensão dos dados analisados do nível mais genérico ao mais específico.

Uma Base de Dados de Múltiplos Níveis (BDMN) é uma base de dados composta de vários níveis de dados. O nível mais baixo é o nível zero ou primitivo, correspondendo à informação primitiva armazenada na base de dados inicial ou global, pois armazena toda a informação bruta, sem ainda nenhum tratamento. Os níveis altos, isto é, o nível um e os demais, armazenam informações extraídas dos níveis mais baixos [HAN94].

Em uma arquitetura de múltiplos níveis, o processo de extração de informações pode ser dividido em fases mais simples e concisas, onde em cada uma destas fases a meta é extrair um determinado tipo de informação desejado pelo usuário. Por outro lado, a divisão do processo de extração em fases mais estanques permitirá de forma facilitada o uso e a combinação de diferentes técnicas de EI dentro do sistema de extração como um todo. Tais técnicas serão usadas em fases diferentes, pesquisando, cada uma, parcialmente os dados desejados pelo usuário. Desta forma, espera-se obter as melhores vantagens de cada técnica de extração empregada no processo como um todo.

A formação de uma BDMN é realizada pela generalização e transformação de informações, nível-a-nível, iniciando a partir da base de informação original (nível zero ou primitivo) [ZAI97]. Técnicas de EI e mineração de dados podem ser usadas para extrair e transformar a informação dos níveis mais baixos da base de dados, gerando os níveis mais altos [HAN95].

A filosofia que está por trás da construção de uma BDMN é a abstração de informações, na qual assume-se que a maioria dos usuários não gosta de ler os detalhes de grandes volumes de informação, como documentos na íntegra, mas gosta de varrer uma descrição geral da informação.

Geralmente, o nível mais alto apresenta a informação mais estruturada. Pela transformação de informações globais desestruturadas em bases de dados relativamente estruturadas, muitas tecnologias de banco de dados já existentes podem ser aplicadas para manusear e recuperar os dados nestes níveis mais altos [ZAI97]. A arquitetura de uma BDMN transforma uma volumosa e desestruturada base de dados global em bases progressivamente menores e melhor estruturadas, sobre as quais a boa tecnologia de banco de dados pode ser aplicada [HAN95].

No entanto, cabe salientar que mesmo uma base de dados de alto nível sendo geralmente muito menor que as bases de dados de níveis baixos, o nível um é resultante direto do nível zero, e a extração de informações será realizada a partir de um elevado volume de dados, podendo o tamanho da base de dados do nível um continuar ainda elevado. Neste sentido, um bom processo de extração e uma boa definição do domínio de extração serão a chave para a construção de uma BDMN de qualidade.

O trabalho de Diego Trein [TRE2000] também está inserido no projeto denominado Extração Semântica de Informações, visando aperfeiçoar o que foi apresentado por Rui Scarinci. A proposta de Diego Trein é, a partir da

combinação de técnicas de EI com BDMN, melhorar a arquitetura e, conseqüentemente, o processo de extração do SES [SCA97]. Estes trabalhos formam a base desta dissertação, fundamentando o desenvolvimento de uma linguagem de extração de informações que utilizará BDMN e também a implementação de um protótipo. A junção destes trabalhos forma o Sistema de Extração Semântica de Informações em Múltiplos Níveis –SES-MN.

A estrutura de um SBC estará presente em todas as fases do processo de extração. Assim sendo, cada fase de extração será gerida por uma base de conhecimentos configurada pelo usuário do sistema, as quais servirão de guia para o processo como um todo, como mostra a figura 7.5. Neste sentido, o SES-MN irá combinar técnicas distintas de extração na busca de resultados de melhor qualidade.

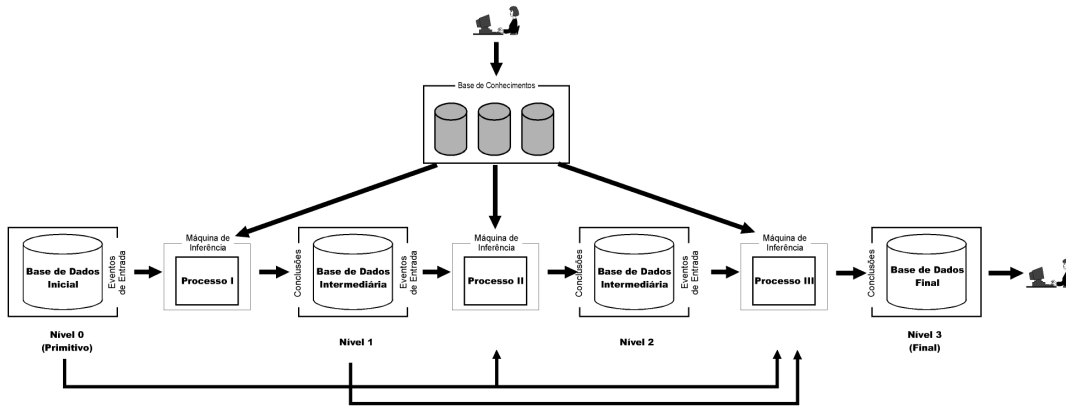


FIGURA 7.5 – Sistema baseado em conhecimento no SES-MN.

8 Linguagem de Extração

A linguagem proposta nesta dissertação é baseada na linguagem de extração definida por Rui Scarinci [SCA97, SCA97a], tendo como objetivo aumentar o poder de extração em relação a esta, além de tornar mais fácil a criação de um conjunto de regras para extrair determinado tipo de informação. A linguagem é utilizada para a representação do conhecimento na filtragem e seleção das informações dos arquivos de entrada, que fazem parte dos diferentes níveis da arquitetura também proposta por Rui Scarinci [SCA2001]. Uma técnica comum para a representação de conhecimento é a das regras do tipo “Se [condição] Então [ação]”, as quais codificam o conhecimento para a tomada de decisão sob a forma de regras de raciocínio altamente modulares.

Sholom Weiss [WEI88] diz que a maior parte do poder de um sistema especialista provém da aplicação apropriada de boas técnicas de raciocínio a uma grande reserva de conhecimentos referentes a problemas específicos. Os métodos de raciocínio não podem ser completamente independentes da classe de problemas que eles têm que resolver. Contudo, ao mesmo tempo, os procedimentos de raciocínio não devem ser tão específicos para determinado problema que se tornem inaplicáveis em outros casos. Na investigação do raciocínio especializado são procurados métodos poderosos, ou seja, aqueles que tem uma capacidade maior de lidar com grande número de variáveis e/ou de casos, e, além disso, sejam aplicáveis de maneira genérica na solução e descrição de uma ampla classe de problemas úteis. É esperado que a linguagem proposta nesta dissertação possa se encaixar no exposto acima. A forma geral de uma regra é:

SE Uma condição lógica é satisfeita
ENTÃO Realize determinada ação.

É possível estender este conceito para, caso a condição lógica não seja satisfeita, ser executada outra ação. Neste caso, seria usado o comando SENÃO e a regra teria a seguinte forma geral:

SE Uma condição lógica é satisfeita
ENTÃO Realize determinada ação
SENÃO Realize outra ação.

Há também a possibilidade da criação de regras aninhadas, ou seja, dentro de um ENTÃO e/ou SENÃO podem ser colocadas, além de ações, novas condições SE. Assim, a regra teria a seguinte forma geral:

SE Uma condição lógica é satisfeita ENTÃO
SE Uma condição lógica é satisfeita ENTÃO
Realize determinada ação
SENÃO
Realize outra ação
SENÃO
SE Uma condição lógica é satisfeita ENTÃO
Realize determinada ação
SENÃO
Realize outra ação.

O protótipo implementado para utilização e validação desta linguagem gera, a partir dos arquivos de entrada semi ou não-estruturados, um arquivo de saída estruturado, no formato texto, com as informações extraídas, que pode ser facilmente exportado para um programa de banco de dados, planilhas, editor de texto ou outro programa qualquer. As informações são extraídas de acordo com uma pré-especificação do que deve ser considerado “relevante”, ou seja, um usuário define um conjunto de regras que será utilizado pelo sistema para realizar a extração das informações contidas nos arquivos de entrada.

Além do arquivo de saída com as informações extraídas ao final de todos os processos que compõem a arquitetura do SES-MN, há outras bases de dados intermediárias, também no formato texto, que são geradas durante a execução das suas sub-fases, que contribuem para o sucesso do resultado final.

Os campos que compõem cada base de dados gerada em um arquivo texto, seja ela intermediária ou final, são separados pelo caractere ponto-e-vírgula (;), facilitando a exportação das informações extraídas.

8.1 Utilização da Linguagem de Extração

A seguir será mostrada a sintaxe da linguagem proposta, que é utilizada nas diferentes fases dos níveis de extração que compõem a arquitetura. Primeiramente serão abordados os aspectos gerais e, depois, características específicas dos níveis.

8.1.1 Aspectos Gerais

- **Convenção de Notações**

[]	Colchetes indicam uma cláusula opcional.
< >	Indica uma instrução obrigatória.
()	Parênteses encapsulam um ou mais parâmetros.
“ ”	Aspas duplas indicam, em alguns casos, uma constante do tipo string, uma ação de seleção de arquivo ou uma chamada a um programa externo.
,	Vírgula separa os parâmetros de uma função.
;	Ponto-e-vírgula indica o final de uma linha lógica.

- **Estrutura de Controle**

A seguir é mostrada a sintaxe utilizada na estrutura SE...ENTÃO...SENÃO das regras nas sub-fases seleção, desdobramento e conversão do Nível de Classificação Estrutural – P1, que segue o padrão da linguagem Pascal [BOR97]. A sintaxe para montagem das regras de análise superficial difere em alguns detalhes, que serão abordados posteriormente.

IF <condição> THEN <ação> [ELSE <ação>];

Primeiramente, é testada a condição apontada pelo *IF*. Caso ela seja verdadeira, a ação indicada pelo *THEN* é executada. Se a condição for falsa, a ação indicada pelo *ELSE* será executada. Nas cláusulas “ação” pode ter uma ou mais

ações e também outras condições, caracterizando a possibilidade de criar regras aninhadas, como por exemplo:

```
IF <condição> THEN
  IF <condição> THEN
    IF <condição> THEN
      <ação>;
```

Caso haja mais de uma ação após o *THEN* ou a condição *IF* tenha um *ELSE*, devem ser usados os comandos *BEGIN* e *END*, sinalizando o início e fim de um bloco de comandos, como por exemplo:

```
IF <condição> THEN
  IF <condição> THEN
    BEGIN
      <ação>
      IF <condição> THEN
        <ação>
      ELSE
        <ação>;
    END;
```

- **Operadores**

Duas ou mais condições podem ser unidas por um dos operadores:

- **Operador OR**

Caso duas condições estejam unidas por um operador *OR*, é suficiente que uma das duas condições esteja correta para validar toda expressão.

Exemplo: Considere a seguinte condição:
Se A OR B

E as seguintes ações a serem executadas:
Então Ação1
Senão Ação2

Caso A e/ou B sejam verdadeiros, a Ação1 é executada. Caso A e B sejam falsos, a Ação2 é executada.

- **Operador AND**

Caso duas condições estejam unidas por um operador *AND*, é necessário que as duas condições estejam corretas para validar toda expressão.

Exemplo: Considere a seguinte condição:
Se A AND B

E as seguintes ações a serem executadas:

Então Ação1

Senão Ação2

Caso A e B sejam verdadeiros, a Ação1 é executada. Caso A e/ou B sejam falsos, a Ação2 é executada.

8.1.2 Aspectos Específicos

Os aspectos específicos podem se somar aos aspectos gerais e também diferir em algum detalhe. As próximas seções tratarão disto.

8.1.2.1 Nível de Classificação Estrutural – P1

- **Identificação**

Os arquivos a serem processados são o ponto de partida para o processo de EI. A localização dos arquivos no disco (diretórios ou pastas) fica a cargo do usuário, sendo necessário uma especificação exata. Neste sentido, o sistema disponibiliza um ambiente para configurações, onde são informados os diretórios onde estarão armazenados os arquivos. A partir desta informação, o sistema saberá onde buscar os arquivos e também já poderá extrair a extensão do nome, tamanho e data de modificação ou criação de cada arquivo, para realizar o processamento nas fases posteriores.

A base de conhecimento desta fase guarda, então, uma lista com todos os caminhos completos onde estarão os arquivos a serem processados, como por exemplo:

C:\Meus Documentos;
C:\Trabalho\Fontes;
D:\E-mail\ETOS;

Arquivo Original	Extensão	Tamanho	Data e Hora
C:\Meus Documentos>manual.pdf	.pdf	244334	10/10/2000 15:15:33
C:\Meus Documentos\extração.ps	.ps	103415	11/11/2000 14:46:23
C:\Meus Documentos\artigo.ps	.ps	133495	02/03/2001 09:08:21
C:\Meus Documentos\readme.txt	.txt	88475	18/03/2001 19:35:31
C:\Meus Documentos\agenda.txt	.txt	123657	18/05/2001 19:15:33
C:\Meus Documentos\arquivos.zip	.zip	60152	22/04/2001 14:56:34
C:\Trabalho\Fontes\clientes.pas	.pas	6657	10/03/2001 08:05:22
C:\Trabalho\Fontes\estoque.pas	.pas	2344	22/03/2001 11:34:45
D:\E-mail\ETOS\email01.eml	.eml	9374	01/03/2001 11:23:45
D:\E-mail\ETOS\email02.eml	.eml	12837	01/03/2001 11:34:21
D:\E-mail\ETOS\email03.eml	.eml	3747	10/04/2001 16:19:37
D:\E-mail\ETOS\email04.eml	.eml	6246	11/04/2001 15:56:43
D:\E-mail\ETOS\email05.eml	.eml	4897	15/04/2001 10:20:39
D:\E-mail\ETOS\email06.eml	.eml	2241	10/05/2001 17:55:23
D:\E-mail\ETOS\email07.eml	.eml	4897	18/01/2001 11:12:11

FIGURA 8.1 – Base de dados intermediária – fase de identificação.

Um exemplo de base de dados intermediária gerada nesta fase é mostrado na figura 8.1 e contém as seguintes informações referentes a cada arquivo: localização original, extensão, tamanho em *bytes* e data e hora de criação/modificação.

- **Seleção**

Na base de conhecimento da fase de seleção, o usuário poderá criar regras que selecionem os arquivos desejados de acordo com as suas características externas, extraídas na fase anterior. A sintaxe destas regras é a seguinte:

IF <característica> <operador> <valor> THEN
 <ação>;

onde,

<característica> pode ser:

C1SIZE retorna o tamanho do arquivo;
C2EXT retorna a extensão do arquivo;
C3DATE retorna a data de modificação ou criação do arquivo.

<operador> pode ser:

= igual;
< > diferente;
< menor;
> maior;
<= menor ou igual;
>= maior ou igual.

<valor>: indicado pelo usuário, que será comparado com a característica informada. No caso de ser um tamanho de arquivo é colocado um número de *bytes*. No caso de ser uma extensão de arquivo é colocada a extensão desejada com um ponto antes, tudo entre aspas duplas. No caso de ser uma data é colocada a função *STRTODATE()*, que recebe uma *string*, entre aspas duplas, no formato de data “DD/MM/AAAA”

<ação>: nesta cláusula a função disponível é a *EXECUTE()*, que tem como objetivo selecionar um determinado arquivo, com a passagem do parâmetro *SELECT*, que deverá estar entre aspas duplas.

Arquivo Original	Extensão	Tamanho	Data e Hora
C:\Meus Documentos>manual.pdf	.pdf	244334	10/10/2000 15:15:33
C:\Meus Documentos\artigo.ps	.ps	133495	02/03/2001 09:08:21
C:\Meus Documentos\readme.txt	.txt	88475	18/02/2000 19:35:31
C:\Meus Documentos\agenda.txt	.txt	123657	18/12/2000 19:15:33
C:\Meus Documentos\arquivos.zip	.zip	60152	22/02/2001 14:56:34
D:\E-mail\ETOS\email01.eml	.eml	9374	01/03/2001 11:23:45
D:\E-mail\ETOS\email02.eml	.eml	12837	01/03/2001 11:34:21
D:\E-mail\ETOS\email03.eml	.eml	2747	10/04/2001 16:19:37
D:\E-mail\ETOS\email04.eml	.eml	1246	11/04/2001 15:56:43
D:\E-mail\ETOS\email05.eml	.eml	4897	15/04/2001 10:20:39

FIGURA 8.2 – Base de dados intermediária – fase de seleção.

Alguns exemplos:

```

if C2EXT = ".pdf" then
    execute("select");

if (C2EXT = ".ps") and (C3DATE >= strtodate("01/02/2001")) then
    execute("select");

if (((C2EXT = ".eml") or (C2EXT = ".txt") or
    (C2EXT = ".zip"))and (C3DATE >= strtodate("01/03/2001"))
    and (C1SIZE >= 3000)) then
    execute("select");
  
```

Um exemplo de base de dados intermediária gerada nesta fase é mostrado na figura 8.2 e contém as seguintes informações referentes a cada arquivo: localização original, extensão, tamanho em *bytes* e data e hora de criação/modificação.

- **Desdobramento**

Na base de conhecimento da fase de desdobramento, o usuário poderá criar regras que serão aplicadas somente sobre os arquivos agrupados que foram selecionados na fase anterior. Conforme a extensão do arquivo agrupado, um programa externo será executado para proceder com o desdobramento. Cabe lembrar que, arquivos agrupados podem conter em sua estrutura outros arquivos agrupados e também arquivos que seriam desprezados pelas regras da fase de seleção. Portanto, estas fases requerem um processo recursivo.

A sintaxe das regras desta fase é a mesma da fase de seleção. Apenas há uma diferença na cláusula <ação>, aonde o parâmetro que vai à função *EXECUTE()* é o *EXPAND*. Já estão pré-definidos alguns tipos de arquivos compactados que serão processados, entre os quais aqueles com extensão ARC, ARJ, CAB, TAR e ZIP.

Arquivo Original	...	Herança	Arquivo Destino
C:\Meus Documentos>manual.pdf	...		
C:\Meus Documentos\artigo.ps	...		
C:\Meus Documentos\readme.txt	...		
C:\Meus Documentos\agenda.txt	...		
redação.pdf	...	Arquivos.zip	C:\SES-MN\Trabalho\redação.pdf
aulas.txt	...	Arquivos.zip	C:\SES-MN\Trabalho\aulas.txt
semestre.txt	...	Arquivos.zip	C:\SES-MN\Trabalho\semestre.txt
programa.ps	...	Arquivos.zip	C:\SES-MN\Trabalho\programa.ps
D:\E-mail\ETOS\email01.eml	...		
D:\E-mail\ETOS\email02.eml	...		
D:\E-mail\ETOS\email03.eml	...		
D:\E-mail\ETOS\email04.eml	...		
D:\E-mail\ETOS\email05.eml	...		

FIGURA 8.3 – Base de dados intermediária – fase de desdobramento.

Alguns exemplos:


```

if C2EXT = ".arj" then
    execute("expand");

if (C2EXT = ".zip") or (C2EXT = ".cab") then
    execute("expand");

```

Um exemplo de base de dados intermediária gerada nesta fase é mostrado na figura 8.3 e contém as seguintes informações referentes a cada arquivo: localização original, extensão, tamanho em *bytes*, data e hora de criação/modificação, herança e localização da cópia do arquivo original. As colunas extensão, tamanho em *bytes* e data e hora de criação/modificação foram suprimidas da figura por motivo de espaço. A herança indica em qual arquivo agrupado está o arquivo original.

Os arquivos provenientes de um arquivo agrupado, ou seja, aqueles que possuem herança, são copiados para um diretório auxiliar de trabalho, não ocorrendo nenhuma alteração no arquivo original. Os demais arquivos, que não estão agrupados, permanecem apenas em seu local original.

- **Conversão**

Na base de conhecimento da fase de conversão, o usuário poderá criar regras que serão aplicadas sobre todos os arquivos selecionados e desdobrados que passaram pelas fases anteriores. Conforme a extensão do arquivo, um programa externo será executado para proceder com a conversão para arquivo texto. Cabe lembrar que esta conversão para arquivo texto é imprescindível para as fases posteriores.

A sintaxe das regras desta fase é a mesma da fase de seleção. Apenas há uma diferença na cláusula <ação>, aonde o parâmetro que vai à função *EXECUTE()* é o *CONVERT* mais o indicativo de qual programa externo será executado, o que depende do tipo de arquivo colocado nas regras, separado pelo caractere “_” (*underline*).

Arquivo Original	...	Herança	Arquivo Destino
C:\Meus Documentos>manual.pdf	...		C:\SES-MN\Trabalho\d1_manual.txt
C:\Meus Documentos\artigo.ps	...		C:\SES-MN\Trabalho\d2_artigo.txt
C:\Meus Documentos\readme.txt	...		
C:\Meus Documentos\agenda.txt	...		
redação.pdf	...	Arquivos.zip	C:\SES-MN\Trabalho\d5_redação.txt
aulas.txt	...	Arquivos.zip	C:\SES-MN\Trabalho\aulas.txt
semestre.txt	...	Arquivos.zip	C:\SES-MN\Trabalho\semestre.txt
programa.ps	...	Arquivos.zip	C:\SES-MN\Trabalho\d8_programa.txt
D:\E-mail\ETOS\email01.eml	...		
D:\E-mail\ETOS\email02.eml	...		
D:\E-mail\ETOS\email03.eml	...		
D:\E-mail\ETOS\email04.eml	...		
D:\E-mail\ETOS\email05.eml	...		

FIGURA 8.4 – Base de dados intermediária – fase de conversão.

Alguns exemplos:

```
if C2EXT = ".doc" then
    execute("convert_doc");

if (C2EXT = ".ps") then
    execute("convert_ps");
```

Um exemplo de base de dados intermediária gerada nesta fase é mostrado na figura 8.4 e contém as seguintes informações referentes a cada arquivo: localização original, extensão, tamanho em *bytes*, data e hora de criação/modificação, herança e localização da cópia do arquivo original. As colunas extensão, tamanho em *bytes* e data e hora de criação/modificação foram suprimidas da figura por motivo de espaço.

Os arquivos que não estão em formato texto são convertidos pelos programas externos e gravados em um diretório auxiliar de trabalho, não ocorrendo nenhuma alteração no arquivo original. Os demais arquivos, que estão em formato texto, permanecem apenas em seu local original.

- **Classificação**

Na base de conhecimento da fase de classificação, o usuário poderá criar regras que serão aplicadas sobre todos os arquivos selecionados, desdobrados e convertidos que passaram pelas fases anteriores. Os arquivos que chegaram nesta fase obrigatoriamente estarão no formato texto, para que o processamento nos níveis posteriores seja possível, como já foi mencionado na fase anterior. A fase de classificação, então, tem o objetivo de classificar estes arquivos texto conforme a estrutura interna de cada um, definida pela respectiva extensão, através de classes que permitirão o uso de regras de extração mais adequadas ao seu tipo nos próximos níveis. A sintaxe das regras da fase de classificação é a seguinte, com a utilização da função **DefClasse**:

Sintaxe:

DefClasse(<nome da classe>,<descrição da classe>[<extensões>])

onde,

<nome da classe>: identificador que será usado nas regras dos próximos níveis para a seleção de um outro conjunto de regras específicas.

<descrição da classe>: texto que descreve a classe.

<extensões>: uma ou mais extensões, colocadas entre colchetes, que irão definir quais arquivos farão parte da determinada classe. Deve ser colocado um ponto antes da extensão e, caso tenha duas ou mais extensões, separa-las por vírgula. Na sintaxe desta regra os colchetes são obrigatórios, não indicando cláusula opcional como em outras regras.

Alguns exemplos:

```
DefClasse(Documentos,Arquivos de Documentos do Word[.doc])
DefClasse(Fontes,Arquivos-Fonte de Linguagens[.pas,.c,.prg])
```

Um exemplo de base de dados intermediária gerada nesta fase é mostrado na figura 8.5 e contém as seguintes informações referentes a cada arquivo: localização original, extensão, tamanho em *bytes*, data e hora de criação/modificação, herança, localização da cópia do arquivo original e classe. As colunas extensão, tamanho em *bytes*, data e hora de criação/modificação e herança foram suprimidas da figura por motivo de espaço.

A base de dados intermediária gerada nesta fase também é a base de dados final do nível de classificação estrutural – P1, primeiro nível da arquitetura do SES-MN.

Arquivo Original	...	Arquivo Destino	Classe
C:\Meus Documentos>manual.pdf	...	C:\SES-MN\Trabalho\d1_manual.txt	AC_READER
C:\Meus Documentos\artigo.ps	...	C:\SES-MN\Trabalho\d2_artigo.txt	POSTSCRIPT
C:\Meus Documentos\readme.txt	...		TEXTO
C:\Meus Documentos\agenda.txt	...		TEXTO
redação.pdf	...	C:\SES-MN\Trabalho\d5_redação.txt	AC_READER
aulas.txt	...	C:\SES-MN\Trabalho\aulas.txt	TEXTO
semestre.txt	...	C:\SES-MN\Trabalho\semestre.txt	TEXTO
programa.os	...	C:\SES-MN\Trabalho\d8_programa.txt	POSTSCRIPT
D:\E-mail\ETOS\email01.eml	...		EMAIL
D:\E-mail\ETOS\email02.eml	...		EMAIL
D:\E-mail\ETOS\email03.eml	...		EMAIL
D:\E-mail\ETOS\email04.eml	...		EMAIL
D:\E-mail\ETOS\email05.eml	...		EMAIL

FIGURA 8.5 – Base de dados intermediária – fase de classificação.

8.1.2.2 Nível de Classificação por Domínio – P2

As regras que compõem a base de conhecimento do nível de classificação por domínio têm o objetivo de identificar a que categoria determinado arquivo pertence, utilizando como base o seu conteúdo. Para isso, a metodologia empregada na arquitetura do SES-MN utiliza a técnica de similaridade de vetores, explicada por Rui Scarinci [SCA2001]. O processo de classificação por domínio pode ser dividido em diversas etapas intermediárias ou sub-processos e, para cada um destes, há uma linguagem específica para a montagem das regras.

- **Definição de *stop words***

Um texto é formado por muitas palavras, chamadas de *stop words*, que contribuem pouco para o seu significado geral, pois não servem para caracterização do seu conteúdo. Desta forma, pode-se eliminar tais palavras. As *stop words* podem ser, por exemplo, pronomes, artigos, conjunções, verbos auxiliares, adjetivos quantitativos, palavras estranhas à linguagem utilizada ou, até mesmo, estranhas ao contexto tratado na coleção de documentos que está sendo processada. A definição das *stop words* é através da função **DefStopWords**, que

permite a criação de grupos de *stop words* que poderão ser usadas para cada arquivo, dependendo da classe que lhe foi atribuída no nível de classificação estrutural.

Sintaxe:

```
DefStopWords(<nome do grupo>[<stopword>,<stopword>,<...>])
```

onde,

<nome do grupo>: identificador que será usado na função **DefDominio** para a seleção de um conjunto de *stop words*.

<stopword>: uma ou mais palavras, colocadas entre colchetes, que são consideradas *stop words* e fazem parte de um determinado grupo. No caso de várias *stop words*, elas são separadas por vírgula. Na sintaxe desta regra os colchetes são obrigatórios, não indicando cláusula opcional como em outras regras.

Alguns exemplos:

```
DefStopWords (artigos [a, as, o, os, um, uns, uma, umas] )
DefStopWords (preposições [com, de, para] )
DefStopWords (outras [sim, não, é, pois] )
```

- **Definição das categorias**

As categorias são os assuntos estipulados pelo usuário que os textos em análise poderão tratar. A criação das categorias é através da função **DefCategoria**, que permite ao usuário definir quais palavras irão compor cada categoria. Associado a cada palavra, o usuário também define um peso que indica a relevância da palavra dentro do texto. A análise de similaridade, que irá definir o assunto tratado em cada texto, é feita entre estas categorias e os atributos relevantes identificados em cada texto de entrada, quando estes foram lidos palavra por palavra.

Sintaxe:

```
DefCategoria(<nome da categoria>,
             <descrição da categoria>[<palavra>,<peso>,<palavra>,<peso>,<...>,<...>])
```

onde,

<nome da categoria>: identificador que será usado na função **DefDominio** para a seleção de um conjunto de palavras e pesos que definem uma categoria.

<descrição da categoria>: texto que descreve a categoria.

<palavra>: palavra ou atributo que o usuário define como identificador do assunto (categoria) tratado em cada arquivo.

<peso>: valor, de 0.01 a 1, que o usuário atribui a cada palavra colocada no parâmetro <palavra> que determina o seu grau de importância na definição do assunto que o arquivo trata.

Os pares <palavra>,<peso> são separados por vírgula. Na sintaxe desta regra os colchetes são obrigatórios, não indicando cláusula opcional como em outras regras.

Alguns exemplos:

```
DefCategoria(informática,textos sobre informática
             [processador,0.60,internet,0.90,disquete,0.80])
DefCategoria(flores,textos sobre jardinagem
             [semente,0.85,girassol,0.70])
DefCategoria(alimentação,textos sobre alimentos
             [arroz,0.90,batata,0.90,feijão,0.90])
```

- **Definição dos identificadores**

Na discriminação dos atributos de um documento textual é gerada uma lista com todas as palavras presentes no texto, menos as *stop words*, juntamente com sua frequência relativa. O usuário, através da função **DefIdentificadores**, determina a quantidade máxima de palavras que identificarão o texto.

Sintaxe:

```
DefIdentificadores(<número de identificadores>)
```

onde,

<número de identificadores>: número máximo de palavras a serem utilizadas para caracterizar o conteúdo do arquivo.

Um exemplo:

```
DefIdentificadores(10)
```

- **Definição da classificação**

Através da função **DefDominio** o usuário combina as classes, *stop words* e categorias, definidas previamente, que irão ser utilizadas em cada arquivo.

Sintaxe:

```
DefDominio(<classe>,<classe>,<...>),(<stopwords>,<stopwords>,<...>),
          (<categoria>,<categoria>,<...>)
```

onde,

<classe>: nome da classe atribuída aos arquivos na fase de classificação do P1, dependendo da extensão de cada um.

<stopwords>: nome de um grupo de *stopwords* definido previamente.

<categoria>: nome do assunto, definido previamente, que os textos em análise poderão tratar.

Alguns exemplos:

```
DefDominio(texto), (artigos, preposições), (flores, alimentação)
DefDominio(email), (outras), (informática)
```

A base de dados final do nível de classificação por domínio – P2, é mostrada na figura 8.6 e contém as seguintes informações referentes a cada arquivo: nome e localização, nome da categoria (assunto) e grau de similaridade entre o conteúdo do arquivo processado e a categoria definida pelo usuário que lhe foi atribuída.

Arquivo Pós-Processo	Categoria	Similaridade
C:\SES-MN\Trabalho\d1_manual.txt	flores	0,445767
C:\SES-MN\Trabalho\d2_artigo.txt	flores	0,761423
C:\Meus Documentos\readme.txt	flores	0,101022
C:\Meus Documentos\agenda.txt	flores	0,334477
D:\E-mail\ETOS\email01.eml	informática	0,883645
D:\E-mail\ETOS\email02.eml	informática	0,902635
D:\E-mail\ETOS\email03.eml	informática	0,661859
D:\E-mail\ET OS\email04.eml	informática	0,229374
D:\E-mail\ETOS\email05.eml	informática	0,957654

FIGURA 8.6 – Base de dados do nível de classificação por domínio – P2.

8.1.2.3 Nível de Análise Superficial – P3

As regras armazenadas nas bases de conhecimento referentes a este nível, permitem a configuração para a análise superficial dos textos de entrada, conforme o conjunto de características léxicas e sintáticas que o usuário irá explorar para a extração das informações desejadas. Utilizando a base de dados do nível de classificação por domínio – P2, onde os arquivos estão classificados por assunto (categoria), e a base de dados do nível de classificação estrutural – P1, onde se encontra a classe de cada um destes mesmos arquivos, associa-se a cada combinação destas classificações as regras de extração léxica e sintática. Desta forma, o domínio de extração torna-se mais restrito, facilitando a configuração das bases de conhecimento e aumentando a qualidade dos dados extraídos. Arquivos de classificações não interessantes ao usuário podem ser desprezados por este processo.

- **Definição da extração**

Através da função **DefExtracao** o usuário determina qual arquivo de regras será utilizado na análise superficial de cada texto, de acordo com a sua classe e categoria.

Sintaxe:

```
DefExtracao(<classe>,<classe>,<...>),(<categoria>,<categoria>,<...>),
(<arquivo de regras>)
```

onde,

<classe>: nome da classe atribuída aos arquivos na fase de classificação do P1, dependendo da extensão de cada um.

<categoria>: nome do assunto, definido previamente, que os textos em análise poderão tratar.

<arquivo de regras>: nome do arquivo TXT que contém as regras para a análise superficial dos textos.

Alguns exemplos:

```
DefExtracao(texto), (flores, alimentação), (Teste de Mesa.TXT)
DefExtracao(email), (informática), (RegrasETOS.TXT)
```

- **Definição do arquivo de saída final**

Determina o nome e caminho completo onde está o arquivo de saída final, que contém o resultado de todo processo de extração.

Um exemplo:

```
C:\SES-MN\SaidaFinal.txt
```

- **Montagem das regras de análise superficial**

Como foi colocado anteriormente, para cada combinação de classes e categorias de textos, podem ser criadas regras de análise superficial específicas. Para isso, basta criar diferentes arquivos TXT, onde cada um será uma base de conhecimento onde o usuário irá inserir as regras. A sintaxe da estrutura de controle *IF...THEN...ELSE* difere do padrão Pascal em alguns detalhes, como nos casos em que há mais de uma ação na mesma regra, regras aninhadas e regras com a cláusula *ELSE*.

Nas regras que possuem mais de uma ação não é necessário colocar um ponto-e-vírgula (;) no final de cada linha que possui a ação, como por exemplo:

```
IF <condição> THEN
BEGIN
    <ação1>
    <ação2>
    <ação3>
END;
```

Nas regras aninhadas, para cada condição é necessário colocar um *BEGIN...END*, exceto na última condição, se esta tiver apenas uma ação. O ponto-e-vírgula (;) irá somente após o último *END*, que fecha a primeira condição, como por exemplo:

```

IF <condição1> THEN
BEGIN
    IF <condição2> THEN
    BEGIN
        IF <condição3> THEN
            <ação>
        END
    END
END;

```

Nas regras que possuem condição com a cláusula *ELSE*, é necessário colocar um *BEGIN...END* para o *IF* e outro para o *ELSE*, inclusive se estes tiverem apenas uma ação. O ponto-e-vírgula (;) irá somente após o último *END*, que fecha a primeira condição, como por exemplo:

```

IF <condição> THEN
BEGIN
    IF <condição> THEN
    BEGIN
        <ação1>
    END
    ELSE
    BEGIN
        <ação2>
    END
END;

```

A leitura do arquivo de entrada irá gerar os *tokens* necessários para a verificação das regras existentes nas bases de conhecimento do nível de análise superficial. Para isso, há um ponteiro, chamado de ponteiro global, que indicará a posição do próximo caractere a ser lido. Desta forma, é com base no andamento do ponteiro global que a leitura do texto de entrada é realizada e os *tokens* são gerados. Um *token* é formado quando o ponteiro global encontra um caractere que foi pré-definido como separador de *token* pelo usuário em um cadastro de separadores, como por exemplo, um ponto (.), uma barra (/) e outros.

No processo de extração, é importante que o usuário tenha o controle do ponteiro global, pois lhe será permitido controlar o andamento da leitura do arquivo de entrada. Assim, pedaços do arquivo poderão ser ignorados diretamente no processo de leitura, reduzindo o tempo de extração, pois as palavras deste pedaço não serão testadas pelas regras da base de conhecimento. Além disso, pode ser necessário extrair informações que já foram lidas, fazendo isso através do atraso do ponteiro global.

Conforme o modelo condicional de regra utilizado na linguagem proposta, uma ação A é executada quando a condição que compõe a regra for verdadeira (então); ou, caso contrário, uma ação B pode ser executada quando esta condição for falsa (senão). Contudo, neste trabalho, a primeira condição de cada regra não deverá possuir a ação B, que seria executada quando esta condição fosse falsa (senão). Esta restrição existe pois, caso contrário, as regras da base de conhecimento cuja primeira condição retornasse falso, teriam a ação de senão (negativa) executada. Como, normalmente, a grande maioria das palavras do texto

não estão previstas nas regras da base de conhecimento, as ações negativas de todas as regras seriam quase sempre executadas.

Como em cada regra podem existir várias ações, há um outro ponteiro, chamado de ponteiro local, que será usado somente nas funções de extração, tendo o objetivo de controlar se após a extração de um pedaço de texto, a próxima função de extração, dentro da mesma regra, irá extrair a partir do ponto inicial do pedaço já extraído ou do ponto final.

A seguir são explicadas todas as funções que podem ser utilizadas na montagem das regras de análise superficial.

- **Comentários**

Sempre que se está utilizando qualquer linguagem de programação, é importante a possibilidade da inserção de comentários dentro do código fonte. Estes comentários são ignorados no processo de compilação e servirão para a documentação do sistema em consultas futuras, podendo esclarecer os porquês de determinadas linhas de código. Na linguagem aqui proposta também há a possibilidade da inserção de comentários dentro das bases de conhecimento que contêm as regras de análise superficial. Para isso, o texto que será comentário deverá ser colocado entre chaves {}.

Um exemplo:

```
if PesqBase(endereco,CxLivre,8,100) then
  Copia(3,Linha,TOT,NALT,Endereco);

{Aqui é um comentário - Validação para a Dissertação}

if PesqString(telefone,CxLivre,8,100) and {também é comentário}
  VerifCaracter(:,CxLivre,1,1,Palavra,DEP) then
  Copia(1,Linha,DEP,NALT,Telefone);
```

- **Cláusula Condição**

As condições contidas nas regras definem quando as ações associadas serão executadas. Neste sentido, cada *token* extraído do arquivo de entrada é comparado com todas as condições definidas pelo usuário, verificando se existe alguma ação a ser realizada. Para uma extração de melhor qualidade e precisão, a “condição” pode ter duas partes: condições de pesquisa e condições de verificação.

a) Condições de Pesquisa

Cada *token* de entrada é comparado com todas as condições de pesquisa existentes nas regras. Os termos de comparação contidos nas condições de pesquisa podem ser declarados através de quatro funções, sendo que todas elas irão retornar ou um valor verdadeiro ou um valor falso. Caso existam condições de verificação, estas só serão testadas nas condições de pesquisa que retornarem verdade. Em todas as funções é permitido definir uma faixa, em linhas, que indicará o escopo de validade para a comparação entre o termo declarado na função e o *token* recebido do texto. Em três funções também é permitido definir se a caixa

dos caracteres que compõem o termo é livre, ou seja, não influi no processo de comparação, ou conforme especificada pelo usuário na regra. A tabela 8.1 mostra os valores de cada parâmetro que devem ser passados em cada função.

TABELA 8.1 – Parâmetros das funções de pesquisa.

Parâmetro	Tipo/Valor
Termo ou Base	(Caractere) <i>String</i> ou nome da base de dados a ser comparada com o <i>token</i> recebido do texto de entrada.
Caixa	(Caractere) Caixa do termo. “CxLivre” para caixa livre ou “CxEsp” para caixa especificada. Caso o termo seja uma base de dados, será a caixa como cada termo foi gravado na base de dados.
Início	(Inteiro) Linha inicial da faixa de pesquisa.
Fim	(Inteiro) Linha final da faixa de pesquisa.
Tipo	(Caractere) Quando o termo for uma <i>string</i> deve ser informado “String”, e quando for uma base de dados deve ser informado “Base”.
Delimit1	(Caractere) Caractere que antecede o termo.
Delimit2	(Caractere) Caractere que sucede o termo.

String: termo declarado diretamente na condição de pesquisa para comparação com os *tokens* de entrada.

Sintaxe:

PesqString(Termo,Caixa,Início,Fim)

Alguns exemplos:

```
if PesqString(Assunto,CxEsp,1,5) then...
if PesqString(endereço,CxLivre,1,30) then...
```

Base de Dados ou Dicionário: os termos para comparação com os *tokens* de entrada estão declarados em um dicionário de termos. Caso o *token* de entrada exista no dicionário de termos a condição é verdadeira. O objetivo é facilitar a comparação das palavras do texto com um grupo de palavras relacionadas entre si, como por exemplo: nomes de países.

Sintaxe:

PesqBase(Base,Caixa,Início,Fim)

Alguns exemplos:

```
if PesqBase(países,CxEsp,1,50) then...
if PesqBase(frutas,CxLivre,1,50) then...
```

Data: esta função verifica se o *token* de entrada é uma data, independente de formatos pré-estabelecidos. Uma data poderá ser composta

somente de números, como 20/10/2000, e também com o nome do mês por extenso, como 20 de outubro de 2000. Além disso, podem aparecer datas que trazem o dia da semana por extenso. Para determinar se uma data é válida, o sistema utiliza três variáveis do tipo inteiro e duas do tipo caractere, fazendo o seguinte processo: quando um *token* é recebido do texto de entrada, e este *token* é um número, ele é colocado em uma das variáveis do tipo inteiro. Se os próximos dois *tokens* forem números, também serão colocados nas variáveis do tipo inteiro. Só poderá haver uma variável com valor maior do que 31 e também somente duas com valor maior do que 12, sendo que a variável com valor maior do que 31 não poderá aparecer na posição do meio da data. As variáveis caractere receberão os *tokens* que sejam nome de meses ou dias da semana, que serão comparados com aqueles que deverão ser pré-cadastrados nas bases de dados específicas para este fim. Uma data só com números será válida se as três variáveis do tipo inteiro forem preenchidas corretamente. Uma data por extenso será válida com uma variável do tipo inteiro e uma variável caractere preenchidas corretamente.

Sintaxe:

PesqData(Início,Fim)

Um exemplo:

```
if PesqData(1,5) then...
```

String entre Caracteres: esta função verifica se o *token* recebido do texto de entrada é igual ao termo declarado diretamente na condição de pesquisa, que pode ser uma string ou uma base de dados, e se está entre os caracteres especificados.

Sintaxe:

Pesq(Tipo,Termo,Delimit1,Delimit2,Caixa,Início,Fim)

Alguns exemplos:

```
if Pesq(String,remetente,"",CxEsp,1,10) then...
if Pesq(Base,Países,[,],CxEsp,1,30) then...
```

Cabe salientar que uma condição de pesquisa pode ser composta, ou seja, podem ser especificadas várias condições de pesquisa no mesmo *IF*, ligando-as pelo operador *OR*. Desta forma é possível comparar um *token* recebido do texto de entrada com várias condições. Caso uma condição de pesquisa retorne verdade, já é o suficiente para a continuidade da avaliação da expressão.

Alguns exemplos:

```
if PesqString(Telefone,CxLivre,1,5) or
   PesqString(Fone,CxLivre,1,5) or
   PesqString(Phone,CxLivre,1,5) then...
if PesqString(endereço,CxLivre,1,30) or
   PesqBase(Endereco,CxEsp,1,30) then...
```

Uma função de pesquisa que é colocada em um *IF* de primeiro nível, ou seja, não em um *IF* aninhado, recebe o *token* do texto para comparação com seu termo ou base. Quando esta comparação retorna verdade e a regra possuir outras funções de pesquisa aninhadas, estas próximas funções começarão a ler o texto a partir do ponto seguinte ao termo ou base da função de pesquisa anterior, respeitando a faixa de pesquisa. Então, ao contrário de uma função de pesquisa de primeiro nível, as funções de pesquisa que estão aninhadas não recebem o *token* do texto e sim buscam um *token* no texto. A utilização de regras deste tipo é para casos em que se está pesquisando termos compostos, como por exemplo, *call for papers*. Para isso, a seguinte regra poderia ser criada:

```
if PesqString(call,CxLivre,5,100) then
begin
  if PesqString(for,CxLivre,5,100) then
  begin
    if PesqString(papers,CxLivre,5,100) then
      Cópia(1,Linha,DEP,NALT,CONGRESSO)
    end
  end
end;
```

b) Condições de Verificação

As condições de verificação servem como complemento das condições de pesquisa, sendo testadas somente quando, pelo menos, uma condição de pesquisa for verdadeira. As condições de verificação sempre são ligadas às condições de pesquisa pelo operador *AND*. O objetivo é permitir uma melhor avaliação do *token* de entrada quanto a outros termos relacionados ao termo testado na condição de pesquisa. Por exemplo, pode-se ter uma condição de pesquisa que tenha identificado uma data. No entanto, esta data pode ter um significado no contexto global do texto sob análise. Com o uso das condições de verificação, o usuário do sistema busca associar esta data com outros termos relacionados existentes no texto, como por exemplo, a palavra que a antecede. Isto atribui, a partir do contexto, um significado semântico maior à data encontrada, viabilizando, posteriormente, um banco de dados (arquivo de saída) mais consistente com relação ao texto original.

As condições de verificação sempre estão associadas às condições de pesquisa, sendo a verificação relacionada com o termo encontrado na condição de pesquisa, na busca de uma extração mais precisa. Neste sentido, não existe, normalmente, validade ou necessidade de análises realizadas por condições de verificação em escopos distantes do termo inicialmente identificado pela condição de pesquisa. Sendo assim, as funções de comparação em condições de verificação apresentam delimitadores da distância de verificação, tendo como referência a condição de pesquisa associada.

Este controle de distância permite ajustar o nível de relacionamento entre termos do arquivo de entrada dentro do processo de extração semântica. Distâncias maiores ampliam o escopo de relacionamento das palavras, mas podem gerar avaliações erradas, associando palavras com pouca ou nenhuma relação. Verificações em distâncias menores retornam palavras, normalmente, mais fortemente relacionadas, devido a sua proximidade no texto. Contudo, outras

palavras também relacionadas com o termo localizado na condição de pesquisa e importantes no contexto semântico do texto podem ser excluídas do processo de extração.

O controle da distância é realizado dentro de cada função de verificação, permitindo que para cada verificação associada a uma mesma condição de pesquisa, existam distâncias diferentes, conforme as características a serem avaliadas pelo usuário do sistema no processo de extração. As distâncias, visando a obtenção de uma maior flexibilidade de uso, podem ser medidas nas seguintes unidades:

- Palavras;
- Linhas (até um caractere de quebra de linha);
- Frases (até um caractere “.”);
- Parágrafos (até encontrar dois ou mais caracteres de quebra de linha subsequentes).

Intrinsecamente à distância, está associado o sentido, onde é caracterizado se a condição de verificação deve ser testada na parte do texto de entrada anterior ou posterior à condição de pesquisa avaliada como verdadeira. Sendo assim, o usuário, juntamente com a distância, deverá configurar o sentido na geração das condições de verificação que compõem a base de conhecimento.

Por outro lado, é importante salientar que algumas características do texto só poderão ser válidas ao processo de extração quando ocorrerem mais de uma vez, como, por exemplo, na seguinte situação: verificada a palavra “período” em uma condição de pesquisa, é desejado que a condição de verificação associada avalie a existência de duas datas até o final da linha. Estas datas informariam ao usuário o início e o fim do período sob extração.

TABELA 8.2 – Parâmetros das funções de verificação.

Parâmetro	Tipo/Valor
Termo	(Caractere) Caractere, <i>string</i> ou nome da base de dados a ser verificada caso a condição de pesquisa tenha retornado verdade.
Caixa	(Caractere) Caixa do termo. “CxLivre” para caixa livre ou “CxEsp” para caixa especificada.
Vezez	(Inteiro) Quantidade de ocorrências do termo que deverá ser verificada.
Distância	(Inteiro) Limite para verificar a ocorrência do termo.
Unidade	(Caractere) Unidade que determina o limite para verificar a ocorrência do termo. Pode ser “Palavra”, “Linha”, “Frase” ou “Parágrafo”.
Sentido	(Caractere) Sentido de verificação da ocorrência do termo. “ANT” para anterior e “DEP” para depois.

Também é permitido definir se a caixa dos caracteres que compõem o termo da condição de verificação, caso estes sejam letras, poderá ser livre, ou seja, não influirá no processo de verificação, ou conforme especificada pelo usuário.

Os termos contidos nas condições de verificação podem ser declarados de quatro maneiras diferentes, conforme as funções a seguir, sendo que todas elas irão retornar ou um valor verdadeiro ou um valor falso. A tabela 8.2 mostra os valores de cada parâmetro que devem ser passados em cada função.

Caractere: permite trabalhar com uma granularidade a nível de caractere, para uma análise mais detalhada do texto de entrada.

Sintaxe:

VerifCaracter(Termo,Caixa,Vezes,Distância,Unidade,Sentido)

Exemplo: verificar se após a palavra “De” existe o caractere “:”, uma vez, com uma distância máxima da condição de pesquisa de uma palavra depois.

```
if PesqString(De,CxEsp,1,5) and
    VerifCaracter(:,CxLivre,1,1,Palavra,DEP) then...
```

String: termo declarado diretamente na condição de verificação para comparação com os *tokens* de entrada.

Sintaxe:

VerifString(Termo,Caixa,Vezes,Distância,Unidade,Sentido)

Exemplo: verifica a existência de uma palavra “*deadline*” antes de uma data, à distância de uma palavra.

```
if PesqData(1,50) and
    VerifString(deadline,CxLivre,1,1,Palavra,ANT) then...
```

Base de Dados ou Dicionário: o funcionamento desta função de verificação é semelhante à função de pesquisa PesqBase, ou seja, os termos para comparação com os *tokens* de entrada devem estar declarados em um dicionário de termos. Caso o *token* de entrada exista no dicionário de termos especificado, a condição de verificação será verdadeira. Para tal, todas as palavras do dicionário são comparadas com o *token*.

Sintaxe:

VerifBase(Base,Caixa,Vezes,Distância,Unidade,Sentido)

Exemplo: verifica se após a palavra “local”, até uma distância de 3 palavras, existe o nome de uma cidade cadastrada na base de dados CIDADES.

```
if PesqString(local,CxLivre,1,50) and
    VerifBase(Cidades,CxLivre,1,3,Palavra,DEP) then...
```

Data: esta função verifica se o *token* de entrada é uma data válida, independente de formatos, seguindo o mesmo processo da função de data na condição de pesquisa.

Sintaxe:

VerifData(Vezes,Distância,Unidade,Sentido)

Exemplo: verifica se após a palavra “aniversário”, até uma distância de 3 palavras, existe uma data.

```
if PesqString(aniversário,CxLivre,1,50) and
    VerifData(1,3,Palavra,DEP) then...
```

Cabe salientar que uma condição de verificação pode ser combinada com outras, ou seja, podem ser especificadas várias condições de verificação no mesmo *IF*, ligando-as pelos operadores *AND* ou *OR*. Desta forma é possível verificar se ocorrem várias características em relação às condições de pesquisa.

Alguns exemplos:

```
if PesqString(telefone,CxLivre,1,50) and
    VerifCaracter(:,CxLivre,1,1,Palavra,DEP) or
    VerifCaracter(-,CxLivre,1,1,Palavra,DEP) then...
if PesqString(período,CxLivre,1,30) and
    VerifCaracter(:,CxLivre,1,1,Palavra,DEP) and
    VerifData(2,1,Frase,DEP) then...
```

- **Cláusula Ação**

Conforme o modelo condicional de regra utilizado, uma ação A é executada quando a condição que compõe a regra for verdadeira (então), ou, caso contrário, uma ação B pode ser executada quando esta condição for falsa (senão). A seguir são detalhadas as funções que podem ser usadas na cláusula ação.

a) Ações de Extração

As ações de extração têm o objetivo de copiar o conteúdo selecionado e filtrado dos arquivos de entrada para o arquivo de saída. Para isto, foram criadas quatro funções de extração, existindo em todas a informação de um grupo de assunto (parâmetro ASSUNTO), definido pelo usuário, com o objetivo de classificar e organizar a informação extraída, que poderá ser lida por outros sistemas. A tabela 8.3 mostra os valores de cada parâmetro que devem ser passados em cada função.

Copia: realiza a extração do conteúdo do arquivo de entrada baseado em uma distância informada pelo usuário. Desta forma, todo conteúdo entre a posição atual do ponteiro global e o limite da distância informado é copiado para o arquivo de saída, no grupo de assunto especificado no parâmetro ASSUNTO. Caso a função “copia” venha logo depois de outra função “copia” ou de uma função “copia até”, que estejam na mesma regra, deve-se observar se o ponteiro local foi alterado. Caso tenha sido alterado, o início do conteúdo a ser copiado será o final do conteúdo copiado pela função anterior, senão, será o mesmo início da função anterior.

TABELA 8.3 – Parâmetros das funções de extração.

Parâmetro	Tipo/Valor
Distância	(Inteiro) Quantidade da unidade informada para extração.
Unidade	(Caractere) Unidade que determina o limite da distância informada. Pode ser “Palavra”, “Linha”, “Frase” ou “Parágrafo”.
Sentido	(Caractere) Sentido da extração. “ANT” para anterior, “DEP” para depois, “ANTINC”, que é igual ao “ANT” mas com a inclusão do <i>token</i> recebido do texto de entrada, “DEPINC”, que é igual ao “DEP” mas com a inclusão do <i>token</i> recebido do texto de entrada e “TOT”, que indica conteúdo total da unidade selecionada, exceto para a unidade “Palavra”.
Ponteiro	(Caractere) Para alteração do ponteiro local deve ser informado “ALT” e para não alterar deve ser informado “NALT”.
Tipo	(Caractere) Quando o termo for uma <i>string</i> deve ser informado “String”, quando for uma base de dados deve ser informado “Base” e quando for uma data deve ser informado “Data”.
Termo	(Caractere) Para o tipo <i>String</i> especifica-se a <i>string</i> desejada, para o tipo Base o nome da base de dados e para o tipo Data o parâmetro fica em branco.
Veze	(Inteiro) Quantidade de ocorrências do termo que deverá ser verificada.
Caixa	(Caractere) Caixa do termo. “CxLivre” para caixa livre ou “CxEsp” para caixa especificada. Caso o termo seja uma base de dados, será a caixa como cada termo foi gravado na base de dados.
Conclusão	(Caractere) Comentário que será gravado no <i>template</i> de saída.
Assunto	(Caractere) Grupo de assunto para onde será enviado o resultado da extração.

As distâncias a serem copiadas são medidas da mesma forma utilizada nas funções de verificação, ou seja, em palavras, linhas, frases e parágrafos. Junto com a distância está associado o sentido da cópia, caracterizando se a extração do conteúdo deverá ser anterior (ANT) ou posterior (DEP) ao ponteiro. Ainda no sentido, pode ser informado o parâmetro ANTINC, que tem a característica do ANT e determina que inclusive a palavra recebida do texto de entrada, que está na condição de pesquisa, será copiada. O parâmetro DEPINC funciona da mesma forma que o ANTINC, mas extraíndo no sentido para frente. Pode ser usado também no parâmetro “sentido” o valor TOT, indicando que todo conteúdo da unidade escolhida onde o ponteiro está será extraído, ou seja, toda linha, frase ou parágrafo. Para a unidade “Palavra”, o uso do valor TOT não causa efeito.

Sintaxe:

Copia(Distância,Unidade,Sentido,Ponteiro,Assunto)

Um exemplo:

```
if <condição> then
  Copia(1,Linha,DEP,NALT,CONTATO);
```


Copia Até: realiza a extração do conteúdo do arquivo de entrada baseado em um ponto de referência informado pelo usuário. Este ponto de referência pode ser uma *string*, uma data ou um termo que está em uma base de dados. Desta forma, todo conteúdo entre a posição atual do ponteiro global e o ponto desejado pelo usuário é copiado para o arquivo de saída, no grupo de assunto especificado no parâmetro ASSUNTO. O funcionamento do ponteiro local é igual ao da função anterior.

Nesta função, como na anterior, também devem ser especificados uma distância limite e um sentido, para verificação da existência do ponto de referência. Caso o ponto de referência não seja encontrado no limite informado, o conteúdo copiado será até este limite. Adicionalmente, pode ser especificado pelo usuário o número de vezes que uma determinada referência deve ser encontrada no texto como, por exemplo, copiar o conteúdo do texto até a terceira data encontrada. Também pode ser especificada a caixa da referência procurada.

Sintaxe:

CopiaAte(Tipo,Termo,Vezes,Caixa,Distância,Unidade,Sentido,Ponteiro,Assunto)

Um exemplo:

```
if <condição> then
    CopiaAte(String, assunto, 1, CxLivre, 2, Linha, DEP, NALT, ASSUNTO) ;
```

Copia Condição: realiza a cópia do termo testado na condição de pesquisa para o arquivo de saída, no grupo de assunto especificado no parâmetro ASSUNTO. Muitas vezes a simples existência de um termo, indicada pela condição de pesquisa, já significa uma informação completa, não sendo necessária a extração de termos relacionados a este.

Sintaxe:

CopiaCond(Assunto)

Um exemplo:

```
if <condição> then
    CopiaCond(REMETENTE) ;
```

Copia Texto: permite que o usuário coloque no arquivo de saída, informações não provenientes do texto sob processo de extração. O objetivo é a inclusão de conclusões geradas a partir de dados existentes no arquivo de entrada. Desta forma, o usuário, através das regras, define que, sempre que uma determinada condição ocorrer, uma determinada informação deve ser colocada no arquivo de saída.

Sintaxe:

CopiaTexto(Conclusão,Assunto)

Um exemplo:

```
if <condição> then
  CopiaTexto(Pessoa Jurídica, REMETENTE);
```

TABELA 8.4 – Parâmetros das funções de deslocamento.

Parâmetro	Tipo/Valor
Distância	(Inteiro) Quantidade da unidade informada para deslocamento do ponteiro de extração.
Unidade	(Caractere) Unidade que determina o limite da distância informada. Pode ser “Palavra”, “Linha”, “Frase” ou “Parágrafo”.
Sentido	(Caractere) Sentido do deslocamento do ponteiro de extração. “ANT” para anterior e “DEP” para depois.
Tipo	(Caractere) Quando o termo for uma <i>string</i> deve ser informado “String”, quando for uma base de dados deve ser informado “Base” e quando for uma data deve ser informado “Data”.
Termo	(Caractere) Para o tipo <i>String</i> especifica-se a <i>string</i> desejada, para o tipo Base o nome da base de dados e para o tipo Data o parâmetro fica em branco.
Vezes	(Inteiro) Quantidade de ocorrências do termo que deverá ser verificada.
Caixa	(Caractere) Caixa do termo. “CxLivre” para caixa livre ou “CxEsp” para caixa especificada. Caso o termo seja uma base de dados, será a caixa como cada termo foi gravado na base de dados.

b) Ações de Deslocamento

Como explicado no início desta seção (item 8.1.2.3), é importante que o usuário tenha controle sobre o ponteiro de leitura global, ampliando, assim, o controle sobre a geração dos *tokens*. Desta forma, duas funções de deslocamento do ponteiro global podem ser usadas nas regras que compõem a base de conhecimento. A tabela 8.4 mostra os valores de cada parâmetro que devem ser passados em cada função.

Desloca: realiza o deslocamento do ponteiro de leitura global baseado em uma distância e sentido informados pelo usuário. As distâncias e sentidos são os mesmos utilizados nas funções de verificação.

Sintaxe:

Desloca(Distância, Unidade, Sentido)

Um exemplo:

```
if <condição> then
  Desloca(1, Frase, DEP);
```

Desloca Até: realiza o deslocamento do ponteiro de leitura global baseado em um ponto de referência informado pelo usuário. Este ponto de referência pode ser uma *string*, uma data ou um termo que está em uma base de dados.

Nesta função, como na anterior, também devem ser especificados uma distância limite e um sentido, para verificação da existência do ponto de referência. Caso o ponto de referência não seja encontrado no limite informado, o deslocamento do ponteiro de leitura global será até este limite. Adicionalmente, pode ser especificado pelo usuário o número de vezes que uma determinada referência deve ser encontrada no texto como, por exemplo, deslocar do ponteiro de leitura global até a terceira data encontrada. Também pode ser especificada a caixa da referência procurada.

Sintaxe:

DeslocaAte(Tipo,Termo,Vezes,Caixa,Distância,Unidade,Sentido)

Um exemplo:

```
if <condição> then
  DeslocaAte(String, assunto, 1, CxLivre, 2, Linha, DEP);
```

8.2 Arquivo de Saída Final

A base de dados final do nível de análise superficial – P3, contém o resultado da análise e processamento de todos os arquivos que compõem a base de dados de entrada, que foram submetidos às regras especificadas nas diversas fases dos três níveis da arquitetura. Ao final do último nível, as informações filtradas e selecionadas estão associadas ao nome do arquivo de origem, como mostra a figura 8.7.

Contudo, juntamente à associação da informação extraída ao nome de seu arquivo de origem, é importante poder classificar as informações em grupos de assuntos definidos pelo usuário. Tais grupos permitem uma melhor organização da informação, visando facilitar o entendimento da mesma pelo usuário. Adicionalmente, esta característica de estruturação do arquivo de saída gera um relacionamento entre as informações de um mesmo grupo, sendo que, a partir desta classificação, pode-se gerar um banco de dados relacional. Cada grupo de assunto da base de dados de saída define um campo da tabela e as informações formam os registros.

Arquivo	Grupo	Informação
Arquivo 1	Grupo A	Informação 1
Arquivo 1	Grupo B	Informação 2
Arquivo 2	Grupo A	Informação 1
Arquivo 2	Grupo B	Informação 2
Arquivo 2	Grupo B	Informação 3
Arquivo 3	Grupo A	Informação 1

FIGURA 8.7 – Estrutura do arquivo de saída.

O arquivo de saída é um arquivo texto, como mostra a figura 8.8, facilitando a exportação dos dados para outros sistemas, além de permitir sua leitura direta pelo usuário.

Arquivo	Grupo	Informação
EMAIL001	E-mail	<amittatia@hotmail.com>
EMAIL001	Assunto	DEMAND: [AE] Paraffin Wax of Chinese Origin
EMAIL001	Data	terça-feira, 22 de agosto de 2000 00:31
EMAIL002	E-mail	<defreitassl@candw.ag>
EMAIL002	Assunto	DEMAND: [AG] Furniture
EMAIL002	Data	domingo, 20 de agosto de 2000 23:55
EMAIL003	E-mail	<r2123@ihug.com.au>
EMAIL003	Assunto	DEMAND: [AU] REFURBISHED MOBILE PHONES
EMAIL003	Data	segunda-feira, 21 de agosto de 2000 23:52

FIGURA 8.8 – Arquivo de saída exemplo para *e-mails* do ETOS.

9 Validação da Linguagem de Extração

A fim de atingir os objetivos desejados e descritos nesta dissertação, pondo em prática o embasamento conceitual apresentado anteriormente para a busca do conhecimento a partir de BDNE/SE, utilizando a convergência de técnicas de RI, EI e SBC, aplicadas em uma arquitetura de múltiplos níveis [SCA2001], realizou-se a implementação do sistema SES-MN.

Para o desenvolvimento do protótipo do sistema SES-MN foi utilizada a ferramenta de programação Delphi 3 [BOR97, BOR97a, BOR97b, CAN96]. O uso desta ferramenta facilitou o processo de programação e aumentou a qualidade do protótipo gerado. Gerando programas para o ambiente Windows, o Delphi permitiu que o protótipo final se beneficiasse ao máximo das características deste sistema operacional, principalmente quanto à interface. O uso deste ambiente gráfico, já difundido mundialmente, facilita a manipulação do sistema pelos seus usuários.

Algumas características da ferramenta que contribuíram para a sua escolha são:

- Possui um ambiente de desenvolvimento integrado para Windows, com características de programação visual. Isto significa que, a maior parte do programa é feita graficamente com o uso do *mouse*, interagindo com os objetos em tempo de projeto, sem a necessidade de compilação para ver os resultados a cada vez que é feita uma alteração, ganhando tempo e economizando esforços;
- A linguagem utilizada, que é o Pascal, foi aumentada e melhorada, sobretudo para adaptá-la à programação com objetos. A programação é mais intuitiva e simples com o uso de vários componentes visuais;
- Assistentes de codificação (*wizards*) que aceleram a digitação de comandos e funções, além de mostrar a sua sintaxe correta e indicar os parâmetros necessários. Com isso há uma redução de erros e, conseqüentemente, otimização do tempo de desenvolvimento;
- Aumento de produtividade com o uso de um editor de código profissional completamente integrado ao ambiente Windows, mensagens que indicam claramente a localização de erros de sintaxe e *debugger* para a execução passo-a-passo do código na busca de erros lógicos;
- O compilador gera rápida e eficientemente código executável nativo 32 *bits*, sem a necessidade de bibliotecas de *run-time*;
- Boa performance do código executável gerado, proporcionando uma boa velocidade na extração de informações dos textos;
- A manutenção das aplicações geradas fica simplificada com a redução do tempo de programação.

Em geral, qualquer ferramenta RAD (*Rapid Application Development* – Desenvolvimento Rápido de Aplicações) é preferível aos velhos

compiladores, onde era necessário escrever o programa inteiro, imaginando como ele deveria ser, até chegar ao fim depois de uma longa série de compilações que consumiam muito tempo e paciência. Com as ferramentas de desenvolvimento visual pode-se ver o aspecto do programa acabado e até mesmo um pouco de funcionalidade enquanto se está programando, proporcionando uma diferença muito importante de tempo e esforço.

Devido ao exposto acima, o porquê da escolha do Delphi para a implementação do protótipo pode ser resumido em duas palavras: velocidade e facilidade de uso.

O aproveitamento de todas as potencialidades e facilidades oferecidas por uma aplicação, depende fundamentalmente do nível de conhecimento que o usuário possui a respeito do sistema, bem como da experiência prática por ele adquirida com a contínua e freqüente utilização do mesmo. Este capítulo tem o objetivo de apresentar a validação da linguagem proposta através da utilização do protótipo implementado para um estudo de caso.

9.1 Caso de Validação

O Sistema de Oportunidades de Comércio Eletrônico (*Electronic Trading Opportunities System – ETOS*), desenvolvido pelo *United Nations Trade Point Development Center (UNTPDC)*, é um conjunto de listas de distribuição de *e-mails* que tratam de *e-commerce*. O ETOS capacita pontos de comércio, os *Trade Points*, para a troca de informações em uma forma padronizada, sobre uma base global, usando texto semi-estruturado transmitido por *e-mail*. Estas mensagens, que consistem basicamente de oportunidades de comércio, ofertas, informações de produtos e listas de preço, estão disponíveis para distribuição entre os diferentes *Trade Points*. O ETOS conecta 135 *Trade Points* e 10.000 corporações ligadas ao comércio em 75 países desenvolvidos, 60 países em desenvolvimento e 20 países menos desenvolvidos. Atualmente, o ETOS gera mais de 130.000 registros mensais que resultam em quase 13 GB de mensagens todos os meses e transfere mais de 2 milhões de *e-mails* por dia entre todos participantes [ELE2000]. O ambiente descrito oferece um potencial enorme de oportunidades de comércio para todas empresas, estando elas em países desenvolvidos, em desenvolvimento e menos desenvolvidos.

Além dos arquivos de *e-mail* do ETOS, que definem o domínio da validação, há alguns arquivos estranhos a este domínio que foram utilizados nos testes. Assim, houve a possibilidade de se verificar e validar algumas características da linguagem de extração proposta, como:

- Seleção de arquivos de tipos específicos;
- Seleção de arquivos com tamanho e data de criação/modificação específicas;
- Possibilidade de análise em arquivos que fazem parte de um arquivo agrupado;
- Definição do assunto tratado no texto;
- Conteúdo a ser extraído dos textos diferenciado conforme o tipo de arquivo e assunto tratado;
- Adaptação de uso em dados estruturados (cabeçalho do *e-mail*) e não-estruturados (corpo do *e-mail*);

- Utilização para extração de informações em diferentes idiomas, no caso específico, o inglês;
- Extração de informações temporais;
- Possibilidade de disponibilizar as informações extraídas a outros aplicativos.

9.1.1 Informações a serem Extraídas dos *E-Mails*

A partir da definição de três assuntos (alimentos, celulares e informática) que os *e-mails* do ETOS devem tratar, pretende-se extrair as seguintes informações:

- Para os três assuntos:
 - Remetente do *e-mail*;
 - Data de envio do *e-mail*;
 - Tipo do ETO, que pode ser ou oferta ou procura;
 - País de origem (sigla);
 - Assunto (*subject*);
 - Produto.
- Para o assunto “alimentos”:
 - Fax.
- Para o assunto “celulares”:
 - Endereço físico;
 - Telefone.
- Para o assunto “informática”:
 - Endereço físico;
 - Telefone;
 - Endereço do *website*;
 - Data da validade da oferta ou procura.

9.1.2 Ambiente de Validação

As características do computador em que os testes de validação da linguagem com o protótipo se realizaram são as seguintes:

- Processador AMD K6-II 450Mhz;
- 64 Mb de memória RAM;
- Sistema Operacional Windows 98.

No momento dos testes de validação somente o protótipo do SES-MN estava sendo executado no computador.

9.1.3 Bases de Conhecimento Criadas para a Validação

A codificação das bases de conhecimento criadas para a validação da linguagem proposta são mostradas no Anexo 1. Há um arquivo, chamado Validação.sp, que é o projeto onde estão todas as regras que compõem os três níveis da arquitetura do SES-MN, e também três arquivos texto que contêm as regras

para análise superficial dos textos que tratam dos assuntos pré-definidos (alimentos, celulares e informática).

A figura 9.1 mostra a tela principal do protótipo SES-MN, com o projeto Validação.sp aberto.

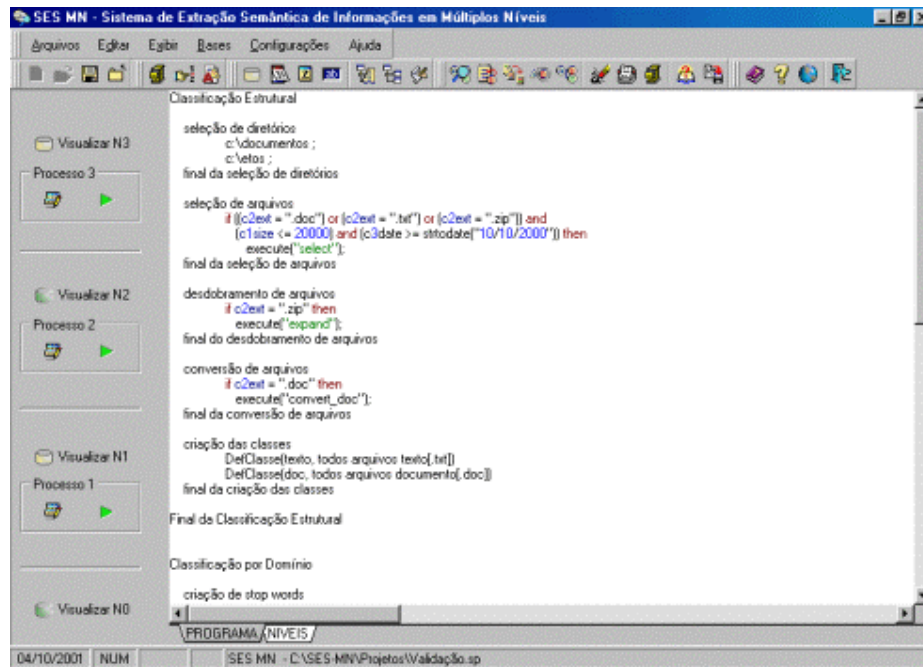


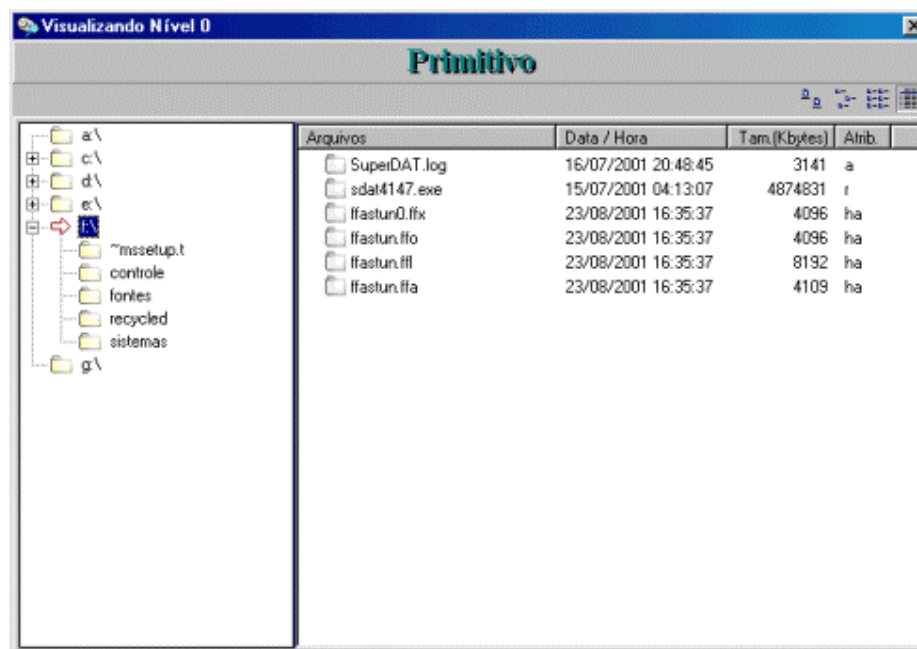
FIGURA 9.1 – Tela principal do protótipo.

A figura 9.2 mostra a visualização da base de dados inicial de todo processo de extração, que é definido como nível 0 ou primitivo. O conteúdo desta base de dados é toda a árvore de diretório do computador onde o protótipo está sendo executado.

Criado o projeto, o usuário deve iniciar com a codificação do processo 1, que é o processo de classificação estrutural. Este processo é dividido em cinco sub-fases que têm o objetivo de extrair informações sobre as características externas dos arquivos (extensão de nome, tamanho, data de criação/modificação), bem como desdobrar arquivos agrupados, converter arquivos que não estão em formato texto para este formato, que é uma necessidade para os próximos processos, e definir a qual classe o arquivo pertence através da sua extensão de nome, que pode indicar como é sua estrutura interna. Todas as cinco sub-fases geram bases de dados intermediárias que auxiliam a construção da base de dados final do processo 1.

Todas bases de dados geradas são mostradas nas figuras a seguir através de grades com as informações extraídas. Cada linha da grade corresponde a um registro e cada coluna a um campo. Nas figuras aparecem apenas algumas linhas e colunas, sendo que as demais são visualizadas utilizando-se as barras de rolagem.

A figura 9.3 mostra a base de dados gerada pela sub-fase “seleção de diretórios”. O objetivo desta sub-fase é atingido pois somente constam os arquivos presentes nos diretórios selecionados, ou seja, a quantidade aproximada de arquivos que o computador onde a validação foi realizada possui é de 48.000 em 2.200 diretórios, totalizando 4.3Gb, e o resultado retornado foi de 259 arquivos em dois diretórios, totalizando 1.1Mb aproximadamente.



Arquivos	Data / Hora	Tam.(Kbytes)	Atrib.
SuperDAT.log	16/07/2001 20:48:45	3141	a
sdat4147.exe	15/07/2001 04:13:07	4874831	i
ffastun0.ffx	23/08/2001 16:35:37	4096	ha
ffastun.flo	23/08/2001 16:35:37	4096	ha
ffastun.fll	23/08/2001 16:35:37	8192	ha
ffastun.fla	23/08/2001 16:35:37	4109	ha

FIGURA 9.2 – Visualização da base de dados do nível 0.



Nº	Arquivo original	Ext.	Tamanho	Data
23	c:\documentos\formatado.doc	.doc	19194	10/05/2001 16:24:
24	c:\documentos\arquivos.zip	.zip	3743	17/01/2001 21:56:
25	c:\documentos\outros.zip	.zip	2086	12/07/1999 22:01:
26	c:\vetos\vetos.zip	.zip	16315	21/09/2001 09:33:
27	c:\vetos\cra.html	.html	4571	19/10/2000 12:05:
28	c:\vetos\exemplo.zip	.zip	2049	21/09/2001 09:56:
29	c:\vetos\about eto system.htm	.htm	18043	23/07/2000 15:45:

Total de diretórios selecionados: 2 Total de arquivos scaneados: 259 Total em bytes: 1164921 (~1137k)

FIGURA 9.3 – Base de dados da sub-fase identificação.

A figura 9.4 mostra a base de dados gerada pela sub-fase “seleção de arquivos”. O objetivo desta sub-fase é atingido pois somente constam os arquivos que possuem as extensões de nome, limite de tamanho e data de criação/modificação especificadas nas regras, a partir da base de dados da sub-fase anterior. Pode-se observar que o total de arquivos recuperados diminuiu para 231,

totalizando 357Kb aproximadamente, pois haviam, por exemplo, arquivos com extensão de nome “html”, que não está especificada nas regras, bem como arquivos com extensão de nome “zip”, que embora seja uma extensão que está especificada nas regras, a data de criação/modificação é inferior a estipulada, tornando a expressão lógica falsa e, conseqüentemente, não selecionando os arquivos.

Seleção Tempo de execução: 00:00:02

Nº	Arquivo original	Ext.	Tamanho	Data
3	c:\documentos\aviso.doc	.doc	15547	13/08/2001 16:30:01
4	c:\documentos\formatado.doc	.doc	19194	10/05/2001 16:24:21
5	c:\documentos\arquivos.zip	.zip	3743	17/01/2001 21:56:11
6	c:\vetos\vetos.zip	.zip	16315	21/09/2001 09:33:41
7	c:\vetos\exemplo.zip	.zip	2049	21/09/2001 09:56:11
8	c:\vetos\ie4 error log.txt	.txt	1213	24/03/2001 17:31:51

Total de arquivos selecionados: 231 Total em bytes: 365985 (~357k)

FIGURA 9.4 – Base de dados da sub-fase seleção.

Cabe ressaltar que as sub-fases de seleção de diretórios e arquivos não alteram nem duplicam os arquivos selecionados, mantendo apenas um *link* em suas bases de dados que apontam ao local original.

Desdobramento Tempo de execução: 00:00:01

Primeiro Estágio Segundo Estágio

B5_arquivos.zip

Nº	Arquivo	Tam	Data	Herança
0	LMC.TXT	2415	29/12/2000	B5_arquivos.zip
1	Linguagem.txt	1031	21/11/2000	B5_arquivos.zip
2	Mensagem.txt	3202	28/08/2000	B5_arquivos.zip
3	DEMAND SN COMF	934	13/08/2001	B6_etos.zip
4	DEMAND AU COMF	1795	13/08/2001	B6_etos.zip
5	DEMAND AU COMF	914	13/08/2001	B6_etos.zip
6	DEMAND CL CELLL	883	13/08/2001	B6_etos.zip
7	DEMAND CL USED	926	13/08/2001	B6_etos.zip
8	DEMAND CN CD-R	1179	13/08/2001	B6_etos.zip

Total de arquivos selecionados: 250 Total em bytes: 370236 (~361k)

FIGURA 9.5 – Base de dados da sub-fase desdobramento.

As figuras 9.5, 9.6 e 9.7 mostram a base de dados gerada pela sub-fase “desdobramento de arquivos”, que é dividida em dois estágios. A figura 9.5 representa o primeiro estágio e o segundo estágio é representado pelas figuras 9.6 e 9.7, para que possam ser visualizadas todas as colunas da grade. O objetivo desta sub-fase é atingido desdobrando todos arquivos agrupados de acordo com as extensões de nome especificadas pelo usuário nas regras, fazendo isso recursivamente até que não sejam mais encontrados arquivos agrupados dentro de

arquivos agrupados. No caso desta validação, somente arquivos agrupados com extensão “zip” são desdobrados. Pode-se perceber no primeiro estágio (figura 9.5) que três arquivos agrupados (arquivos.zip, etos.zip e exemplo.zip) passaram pela sub-fase anterior de seleção e todos seus arquivos estão na grade ao lado, mesmo aqueles que não satisfaçam as condições impostas naquela sub-fase. Cada arquivo tem também a informação da sua herança, ou seja, de qual arquivo agrupado foi originado.

Desdobramento Tempo de execução: 00:00:01

Primeiro Estágio Segundo Estágio

Nº	Arquivo original	Ext.	Tamanho	Data	H
224	c:\etos\offer us used ibm p133 computers .txt	.txt	1057	14/08/2001 09:1	
225	c:\etos\offer us used laptops .txt	.txt	704	14/08/2001 09:1	
226	c:\etos\offer us used p200 computers .txt	.txt	1050	14/08/2001 09:1	
227	c:\etos\offer us used telephones .txt	.txt	1164	13/08/2001 22:4	
228	c:\etos\offer us used toshiba notebooks pentium 133 mhz .txt	.txt	1036	14/08/2001 09:1	
229	Linguagem.txt	.txt	1031	21/11/2000 21:5 B	
230	DEMAND SN COMPUTER PRODUCTS AND OFFICE EQUIP	.txt	934	13/08/2001 22:5 B	
231	DEMAND AU COMPUTER .txt	.txt	1795	13/08/2001 22:5 B	
232	DEMAND AU COMPUTER PRODUCTS .txt	.txt	914	13/08/2001 22:5 B	
233	DEMAND CL CELLULAR PHONE ACCESORIES . .txt	.txt	883	13/08/2001 22:4 B	
234	DEMAND CL USED CELLULAR PHONES AMPS AND TDMA	.txt	926	13/08/2001 22:4 B	
235	DEMAND CN CD-ROM .txt	.txt	1179	13/08/2001 22:5 B	
236	DEMAND CN COMPIETERS PARTS CD-ROM/DISK ACCESS	.txt	1227	13/08/2001 22:5 B	

Total de arquivos selecionados: 250 Total em bytes: 370236 (~361k)

FIGURA 9.6 – Base de dados da sub-fase desdobramento.

Desdobramento Tempo de execução: 00:00:01

Primeiro Estágio Segundo Estágio

Nº	Data	Herança	Arquivo destino
224	14/08/2001 09:1		
225	14/08/2001 09:1		
226	14/08/2001 09:1		
227	13/08/2001 22:4		
228	14/08/2001 09:1		
229	21/11/2000 21:5 B5_arquivos.zip		C:\SES-MN\Trabalho\Linguagem.txt
230	13/08/2001 22:5 B6_etos.zip		C:\SES-MN\Trabalho\DEMAND SN COMPUTER PRODUCTS A
231	13/08/2001 22:5 B6_etos.zip		C:\SES-MN\Trabalho\DEMAND AU COMPUTER .txt
232	13/08/2001 22:5 B6_etos.zip		C:\SES-MN\Trabalho\DEMAND AU COMPUTER PRODUCTS J
233	13/08/2001 22:4 B6_etos.zip		C:\SES-MN\Trabalho\DEMAND CL CELLULAR PHONE ACCES
234	13/08/2001 22:4 B6_etos.zip		C:\SES-MN\Trabalho\DEMAND CL USED CELLULAR PHONES
235	13/08/2001 22:5 B6_etos.zip		C:\SES-MN\Trabalho\DEMAND CN CD-ROM .txt
236	13/08/2001 22:5 B6_etos.zip		C:\SES-MN\Trabalho\DEMAND CN COMPIETERS PARTS CD

Total de arquivos selecionados: 250 Total em bytes: 370236 (~361k)

FIGURA 9.7 – Base de dados da sub-fase desdobramento.

O segundo estágio (figuras 9.6 e 9.7) terá todos os arquivos que satisfaçam as condições impostas nas regras da sub-fase anterior, sejam eles

originados de arquivos agrupados ou não, neste caso, arquivos com extensão “doc” e “txt” que possuem tamanho e data de criação/modificação conforme especificado.

No segundo estágio (figura 9.6) pode-se perceber que alguns arquivos possuem na coluna “arquivo original” o caminho completo da sua localização e outros apenas o nome. Os mesmos que possuem somente o nome, a coluna “herança” e “arquivo destino” (figura 9.7) está preenchida com o arquivo agrupado a que ele pertence e o caminho da localização da cópia feita. Somente os arquivos que provêm de um arquivo agrupado serão duplicados, evitando um aumento maior na área de armazenamento ocupada e alterações no arquivo original. Também percebe-se que, devido aos arquivos agrupados, o total de arquivos selecionados aumentou de 231 para 250 e o total em *Kbytes* de 357 para 361.

Conversão Tempo de execução: 00:00:01

Nº	Arquivo original	Ext	Tamanho	Data	Orig
1	c:\documentos\lattes.txt	.txt	259	28/08/2001 15:5	
2	c:\documentos\relatorio.txt	.txt	491	24/08/2001 08:4	
3	c:\documentos\aviso.doc	.doc	15547	13/08/2001 16:3	
4	c:\documentos\formatado.doc	.doc	19194	10/05/2001 16:2	
5	c:\vetos\ie4 error log.txt	.txt	1213	24/03/2001 17:3	
6	c:\vetos\license.txt	.txt	12223	28/03/2001 22:3	
7	c:\vetos\demand ca usedrefurbised.. p-ii laptops p-classic and g	.txt	1215	13/08/2001 22:5	
8	c:\vetos\demand [ae] sugar.txt	.txt	1215	31/08/2001 18:1	
9	c:\vetos\demand [cy] pet preforms.txt	.txt	1319	31/08/2001 18:1	
10	c:\vetos\demand [cz] crude oil.txt	.txt	720	31/08/2001 18:1	

Total de arquivos selecionados: 250 Total em bytes: 370236 (~361k)

FIGURA 9.8 – Base de dados da sub-fase conversão.

Conversão Tempo de execução: 00:00:01

Nº	Data	Origem	Arquivo destino
1	28/08/2001 15:5		
2	24/08/2001 08:4		
3	13/08/2001 16:3		C:\SES-MN\Trabalho\D3_aviso.txt
4	10/05/2001 16:2		C:\SES-MN\Trabalho\D4_formatado.txt
5	24/03/2001 17:3		
6	28/03/2001 22:3		
7	13/08/2001 22:5		
8	31/08/2001 18:1		
9	31/08/2001 18:1		
10	31/08/2001 18:1		

Total de arquivos selecionados: 250 Total em bytes: 370236 (~361k)

FIGURA 9.9 – Base de dados da sub-fase conversão.

As figuras 9.8 e 9.9 mostram a base de dados gerada pela sub-fase “conversão de arquivos”, para que possam ser visualizadas todas as colunas da grade. O objetivo desta sub-fase é atingido convertendo todos arquivos que tiverem sua extensão de nome indicada pelo usuário nas regras para o formato texto, que é

uma necessidade para a continuidade do processo de extração. Conforme o tipo de arquivo, indicado pela sua extensão, deve haver uma regra que faça a chamada a um programa externo que fará a conversão para texto (extensão TXT). No caso desta validação há arquivos no formato documento (extensão DOC) que foram convertidos para texto. Para estes arquivos, a coluna “arquivo destino” (figura 9.8) está preenchida com o caminho e novo nome da cópia. Somente os arquivos convertidos são duplicados, evitando um aumento maior na área de armazenamento ocupada e alterações no arquivo original. Nesta sub-fase não há diminuição ou aumento no total de arquivos selecionados e total em *Kbytes*.

Classificação Tempo de execução: 00:00:00

Nº	Arquivo original	Ext	Tamanho	Data	Arq
1	c:\documentos\lattes.txt	.txt	259	28/08/2001 15:5	
2	c:\documentos\relatorio.txt	.txt	491	24/08/2001 08:4	
3	c:\documentos\aviso.doc	.doc	15547	13/08/2001 16:3	
4	c:\documentos\formatado.doc	.doc	19194	10/05/2001 16:2	
5	c:\vetos\ie4 error log.txt	.txt	1213	24/03/2001 17:3	
6	c:\vetos\license.txt	.txt	12223	28/03/2001 22:3	
7	c:\vetos\demand ca usedrefurbised. p-ii laptops p-classic and p	.txt	1215	13/08/2001 22:5	
8	c:\vetos\demand [ae] sugar.txt	.txt	1215	31/08/2001 18:1	
9	c:\vetos\demand [cy] pet preforms.txt	.txt	1319	31/08/2001 18:1	
10	c:\vetos\demand [cz] crude oil.txt	.txt	720	31/08/2001 18:1	
11	c:\vetos\demand [cz] unsalted butter and skim milk powder.txt	.txt	618	31/08/2001 18:1	
12	c:\vetos\demand [in] mild steel billets-10000 mt monthly.txt	.txt	1828	31/08/2001 18:2	
13	c:\vetos\demand [in] molasses in bulk.txt	.txt	2650	31/08/2001 18:1	

Total de arquivos selecionados: 250 Total em bytes: 370236 (~361k)

FIGURA 9.10 – Base de dados da sub-fase classificação.

Classificação Tempo de execução: 00:00:00

Nº	Data	Arquivo Pai	Arquivo destino	Classe
1	28/08/2001 15:5			texto
2	24/08/2001 08:4			texto
3	13/08/2001 16:3		C:\SES-MN\Trabalho\D3_aviso.txt	doc
4	10/05/2001 16:2		C:\SES-MN\Trabalho\D4_formatado.txt	doc
5	24/03/2001 17:3			texto
6	28/03/2001 22:3			texto
7	13/08/2001 22:5			texto
8	31/08/2001 18:1			texto
9	31/08/2001 18:1			texto
10	31/08/2001 18:1			texto
11	31/08/2001 18:1			texto
12	31/08/2001 18:2			texto
13	31/08/2001 18:1			texto

Total de arquivos selecionados: 250 Total em bytes: 370236 (~361k)

FIGURA 9.11 – Base de dados da sub-fase classificação.

As figuras 9.10 e 9.11 mostram a base de dados gerada pela sub-fase “criação de classes”, para que possam ser visualizadas todas as colunas da grade. A base de dados desta sub-fase também é a base de dados final do nível 1 – classificação estrutural. O objetivo desta sub-fase e, por consequência, de todo nível 1 é atingido com a execução dos comandos e funções da linguagem proposta pois, para cada arquivo selecionado, foi atribuída uma classe que determina a sua estrutura interna, possibilitando a criação das regras do nível 3 – análise superficial, específicas para cada classe. Esta classe também serve de rótulo do arquivo pois, após a sub-fase de conversão, todos arquivos possuem a mesma estrutura e extensão, ou seja, todos são arquivos texto. Com isso, o rótulo indica qual é o tipo original de cada arquivo, permitindo um tratamento posterior diferenciado.

No caso desta validação foram criadas duas classes. A classe “DOC” engloba todos arquivos com extensão “doc” e a classe “TEXTO” todos arquivos com extensão “txt”. As informações extraídas ao final do nível 1 são, então, localização original, extensão de nome, tamanho, data de criação/modificação e classe de cada arquivo.

As regras do nível 2 – classificação por domínio, estão divididas em três sub-fases: criação de *stop words*, categorias e domínios. Apenas uma base de dados é gerada ao final da execução das três sub-fases, a qual é mostrada na figura 9.12.

Classificação por Domínio Tempo de execução: 00:04:52

Nº	Arquivo Processado	Categoria	Similaridade
120	c:\vetos\offer cn computer relevant products .txt	informática	0,045620
121	c:\vetos\offer cn computer shaped electronic calendar with mult .txt	informática	0,077563
122	c:\vetos\offer cn ginkgo terpenoids (ginkgolide) .txt		
123	c:\vetos\offer cn graphite electrode .txt		
124	c:\vetos\offer cn mobile phone accessories .txt	celular	0,140423
125	c:\vetos\offer cn mobile phone accessories wholesale center .txt	celular	0,140284
126	c:\vetos\offer cn mobile phone battery and accessories .txt	celular	0,323635
127	c:\vetos\offer cn mobile phone battery pack .txt	celular	0,157830
128	c:\vetos\offer cn used computer and parts .txt	informática	0,053861

FIGURA 9.12 – Base de dados do nível 2 – classificação por domínio.

O objetivo deste nível é determinar dentro de qual assunto, que aqui é chamado de categoria, entre os especificados pelo usuário, cada arquivo que foi selecionado se encaixa utilizando, como já foi mencionado anteriormente, a técnica de similaridade de vetores, explicada por Rui Scarinci [SCA2001]. A linguagem proposta possui as funções necessárias para trabalhar com a técnica utilizada, sendo elas de fácil compreensão e utilização, como pode-se perceber no projeto desta validação.

A definição das *stop words* pode ser feita através de somente um grupo, mas também são permitidos vários grupos como é o caso do projeto em questão, onde foram criados grupos de *stop words* que são pronomes, artigos, palavras que ocorrem frequentemente em todo tipo de texto, entre outros. Também foram criados grupos de *stop words* que aparecem mais especificamente nos textos

alvo do projeto, que são os *e-mails* do ETOS. As *stop words* que estão no grupo “etos” foram determinadas através de uma análise manual em uma amostra de *e-mails*, onde observou-se que elas apareciam freqüentemente e não iriam contribuir para a definição do assunto tratado nos textos. As *stop words* que estão no grupo “etos português” foram inseridas, pois indicam as palavras que aparecem no cabeçalho de todos *e-mails*. Como a ferramenta utilizada para a leitura dos *e-mails* está no idioma português, os dados do cabeçalho ficam neste idioma. Cabe ressaltar que, se os dados do cabeçalho estivessem em outro idioma, bastaria acrescentar as respectivas palavras em um grupo de *stop words*.

Na sub-fase “criação das categorias” o usuário define os assuntos que ele espera encontrar nos textos selecionados, juntamente com as palavras que podem indicar o assunto e seus pesos de importância. Também é definido quantas palavras servirão para a identificação do assunto de um texto. No caso desta validação foi definido que as quinze palavras mais freqüentes em um texto podem identificar seu assunto.

As palavras da categoria “alimentação” foram definidas ao acaso, sem nenhuma metodologia ou análise manual nos textos. As palavras da categoria “informática” foram definidas através de uma análise manual em uma amostra de *e-mails*, bem como algumas palavras da categoria “celular”. As palavras “*mobile*”, “*nokia*”, “*cellular*”, “*batt*” e “*accessories*” da categoria “celular” foram definidas através da metodologia de clusterização, trabalho desenvolvido por Leandro Wives [WIV99]. Para isso, uma amostra de aproximadamente 160 *e-mails* do ETOS foi submetida ao processo, resultando 10 *clusters* com 4 palavras em cada um. Neste conjunto de 40 palavras identificou-se subjetivamente quais poderiam tratar mais de celulares, sendo consideradas as mais relevantes por serem as mais freqüentes. Quanto aos pesos atribuídos às palavras, todos foram definidos subjetivamente.

Na sub-fase “criação dos domínios” é feita a ligação entre as classes, *stop words* e categorias definidas anteriormente. Para cada classe de arquivo selecionado (DOC e TEXTO) foram escolhidos os grupos de *stop words* a serem utilizados e qual categoria deverá ser atribuída ao arquivo (ou alimentação ou celular ou informática). Quando em uma função **DefDominio** há mais de uma categoria, a categoria atribuída ao arquivo será aquela que resultar o maior grau de similaridade. Para os arquivos que obtiverem um grau de similaridade igual a zero, nenhuma categoria lhe será atribuída, sendo, conseqüentemente, descartado do nível de análise superficial.

Percebe-se, então, que ao final de todo processamento do nível 2 o objetivo foi alcançado, visto que foi possível determinar dentro de qual assunto, entre os especificados pelo usuário, cada arquivo que foi selecionado e que tem um grau de similaridade com algum dos assuntos se encaixa.

As regras do nível 3 – análise superficial, estão divididas em duas sub-fases: definições e arquivo de saída. O objetivo deste nível é fazer a ligação entre a classe, informação extraída no nível 1, e a categoria, informação extraída no nível 2, de cada arquivo selecionado para que possam ser escolhidas regras de análise superficial específicas, determinar onde será gravado o resultado final de todo processo de extração e executar as regras de análise superficial.

Pode-se perceber na sub-fase “definições” a possibilidade que a linguagem forneça de determinar que arquivos da mesma classe, mas de categorias diferentes, utilizem uma base de regras de análise superficial diferenciada. O mesmo aconteceria para arquivos de categorias iguais e classes diferentes. Também há a possibilidade de, em uma só função **DefExtracao**, determinar que arquivos de várias classes e categorias utilizem a mesma base de regras de análise superficial.

Para o projeto “validação.sp” foram criadas três bases de conhecimento com as regras de análise superficial. Cada base está em um arquivo no formato texto e o conteúdo de todas elas está no Anexo 1. A base de conhecimento “BC Celular.TXT” irá tratar dos arquivos cujo assunto pré-determinado seja relacionado a telefones celulares, “BC Informática.TXT” tratará dos arquivos relacionados à informática e “BC Alimentos.TXT” tratará dos arquivos relacionados à alimentos. Nas regras de análise superficial também são utilizadas três bases de dados, que são os chamados dicionários, através da função **PesqBase** e o conteúdo delas está no Anexo 1. As palavras que estão nestas bases de dados foram escolhidas depois de ser feita uma análise manual em uma amostra de *e-mails* do ETOS, onde verificou-se que elas apareciam freqüentemente como indicativo de um possível endereço, produto ou país.

Há ainda três bases de dados auxiliares que servem para guardar especificamente os caracteres que serão considerados separadores de *tokens*, termos que indicam nomes de meses e nomes dos dias da semana. As duas últimas são utilizadas internamente pelas funções que tratam com datas (**PesqData**, **VerifData**, **CopiaAte** e **DeslocaAte**). O conteúdo destas três bases é mostrado no Anexo 1. Cabe ressaltar que nas bases de meses e dias da semana foram colocados os nomes completos e abreviados apenas nos idiomas português e inglês, podendo ser acrescentado outros idiomas conforme a necessidade do usuário.

TABELA 9.1 – variáveis do resultado da extração.

<i>Informação</i>	<i>Variável</i>
Remetente do <i>e-mail</i>	EMAIL
Data de envio do <i>e-mail</i>	DATAENVIO
Tipo do ETO	TIPO
País de origem	SIGLAPAI
Assunto	ASSUNTO
Produto	PRODUTO1
Produto	PRODUTO2
Produto	PRODUTO3
Produto	PRODUTO4
Produto	PRODUTO5
Produto	PRODUTO6
Fax	FAX
Endereço físico	ENDERECO_BASE
Endereço físico	ENDERECO
Telefone	TELEFONE
Endereço do <i>website</i>	WEBSITE
Data da validade do ETO	VALIDADE

Dos 250 arquivos que estão na base de dados do nível 2, 89 obtiveram algum grau de similaridade com os assuntos pré-definidos, sendo estes que irão ser submetidos à análise superficial. A categoria “alimentação” não recebeu nenhum arquivo, a categoria “celular” recebeu 35 arquivos e a categoria “informática” recebeu 54 arquivos.

O parâmetro “Assunto” das funções de extração, que é a variável para onde será enviado o resultado da extração, terá os nomes que estão na tabela 9.1, conforme as informações especificadas no item 9.1.1.

Para a informação “Produto” há seis variáveis, pois foram criadas seis regras diferentes com o objetivo de extrair possíveis características referentes ao produto ou aos produtos tratados no *e-mail*. A criação de variáveis diferentes para a mesma informação não é uma obrigatoriedade da linguagem, mas sim uma opção para que o usuário saiba por qual regra a informação foi extraída. O mesmo acontece com a informação “Endereço físico”, que possui duas variáveis. As informações “Endereço do *website*” e “Data da validade do ETO” são extraídas somente para os *e-mails* da categoria “informática”.

O Anexo 2 contém uma amostra de 5 *e-mails* do ETOS, retirada dos 89 arquivos que obtiveram algum grau de similaridade com os assuntos pré-definidos. Destes 5 *e-mails*, 2 são da categoria “celular” e utilizarão a base de conhecimento “BC Celular.TXT” e 3 são da categoria “informática”, utilizando a base de conhecimento “BC Informática.TXT”. Logo após os *e-mails* há o arquivo “ResultadoFinal.TXT”, onde está o resultado final do todo processo de EI realizado pelo SES-MN. Os arquivos transcritos no Anexo 2 reproduzem fielmente os originais, exceto um deles, devido a explicação colocada a seguir.

No final da base de conhecimento “BC Celular.TXT”, após uma linha de comentário, há cinco regras que foram criadas apenas para mostrar o uso de algumas funções, em situações específicas, que não foram utilizadas para a extração de informações relacionadas aos arquivos do ETOS. Para isso, um *e-mail* que está no Anexo 2, com o título “*DEMAND CL REFURBISHED CELLULAR PHONES .txt*”, teve seu conteúdo original alterado, com a inserção de algumas informações que são extraídas por estas cinco regras.

9.1.4 Avaliação do Resultado Final

A avaliação do resultado final apresentada aqui considera 87 dos 89 arquivos que obtiveram algum grau de similaridade com os assuntos pré-definidos. Os dois arquivos restantes não são *e-mails* do ETOS e, por isso, não foram considerados. A tabela 9.2 mostra a distribuição dos 87 *e-mails* entre os assuntos “celular” e “informática” e os tipos “oferta” e “procura”.

TABELA 9.2 – distribuição de *e-mails*.

	<i>Celular</i>	<i>Informática</i>	<i>TOTAIS</i>
<i>Oferta</i>	18	28	46
<i>Procura</i>	16	25	41
<i>TOTAIS</i>	34	53	87

Os 87 arquivos de *e-mail* considerados para a análise superficial foram analisados também manualmente e, neste processo manual, todas as informações que deveriam ser extraídas foram marcadas para posterior comparação com as informações extraídas pelo protótipo. O resultado desta comparação é mostrado na tabela 9.3 conforme as seguintes opções, colocadas em letras e números por motivos de espaço nos títulos das linhas e colunas.

- Variável **A** – Extraída corretamente – informação que foi extraída conforme o que estava marcado no *e-mail*;
- Variável **B** – Extraída incorretamente – informação diferente que foi extraída, sem ser aquela que estava marcada no *e-mail*;
- Variável **C** – Extraída com mais texto do que deveria – informação que foi extraída com algum texto a mais em relação ao que estava marcado no *e-mail*;
- Variável **D** – Extraída com menos texto do que deveria – informação que foi extraída incompleta, ou seja, com texto a menos em relação ao que estava marcado no *e-mail*;
- Variável **E** – Extraída sem ter nada marcado no *e-mail* – informação que foi extraída em um *e-mail* que não tinha nada marcado para extração;
- Variável **F** – Tinha a informação no *e-mail* e não foi extraída – informação que estava marcada no *e-mail* para extração e não foi extraída;
- Variável **G** – Número de itens relevantes recuperados – este valor corresponde ao somatório das variáveis A, C e D;
- Variável **H** – Número total de itens relevantes na base de dados – este valor corresponde ao somatório de todas as variáveis, ou seja, A, B, C, D, E e F;
- Variável **I** – Número total de itens recuperados – este valor corresponde ao somatório das variáveis A, B, C, D e E;
- Variável **R** – Índice percentual de *recall* – segundo a definição que consta na seção 3.2.1, $R = G / H$;
- Variável **P** – Índice percentual de *precision* – segundo a definição que consta na seção 3.2.1, $P = G / I$;
- Variável **1** – Informação endereço base;
- Variável **%1** – Percentuais da informação endereço base;
- Variável **2** – Informação endereço;
- Variável **%2** – Percentuais da informação endereço;
- Variável **3** – Informação telefone;
- Variável **%3** – Percentuais da informação telefone;
- Variável **4** – Informação website;
- Variável **%4** – Percentuais da informação website;
- Variável **5** – Informação validade;
- Variável **%5** – Percentuais da informação validade;
- Variável **6** – Informação produto;
- Variável **%6** – Percentuais da informação produto.

As informações remetente e data de envio do *e-mail*, tipo do ETO, país de origem e assunto, por estarem todas no cabeçalho do *e-mail*, que segue uma estrutura padrão, tiveram um índice de *recall* e *precision* de 100%. Isto confirma a ótima adaptação de uso da linguagem em dados estruturados.

Com relação às demais informações, que estão todas no corpo dos *e-mails* e a tabela 9.3 mostra os resultados, cabe ressaltar alguns comentários e conclusões.

A extração das informações “endereço base” e “endereço” tem o mesmo objetivo, ou seja, extrair o endereço físico da pessoa ou empresa que está oferecendo ou procurando determinado produto. Porém, para cada uma, foi criada uma regra de extração diferente para observar aspectos da linguagem. As duas regras procuram por um termo que possa indicar um endereço, como rua, avenida, bloco e outros. Mas, a diferença entre elas é que aquela que extrai “endereço”, encontrando o termo indicativo, extrai três linhas de informação, não importando se todo o conteúdo destas três linhas faz parte do endereço, o que ocasiona a extração, em alguns casos, de conteúdo indesejado. Já a regra que extrai “endereço base”, além de encontrar o termo indicativo, também verifica se nas próximas linhas há um termo que indique um País, situação que ocorre frequentemente quando se especifica um endereço, e o conteúdo extraído irá somente até a ocorrência deste País. Isto faz com que haja menos extração de conteúdo indesejado. Os resultados comprovam a afirmação (9,59% contra 57,33% na variável C) mas também mostram consequências, ou seja, a informação “endereço base” obteve um índice de precisão melhor e, por outro lado, o índice de *recall* baixou devido ao fato de que podem existir endereços sem a indicação do País que não serão extraídos (20,55% contra 6,67% na variável F).

TABELA 9.3 – resultado final do processo de extração.

	1	%1	2	%2	3	%3	4	%4	5	%5	6	%6
A	42	57,53	23	30,67	62	91,18	18	78,26	3	100,00	189	61,36
B	0	0,0	1	1,33	0	0,00	1	4,35	0	0,00	14	4,55
C	7	9,59	43	57,33	5	7,35	0	0,00	0	0,00	47	15,26
D	9	12,33	3	4,00	1	1,47	0	0,00	0	0,00	8	2,60
E	0	0,0	0	0,00	0	0,00	0	0,00	0	0,00	3	0,97
F	15	20,55	5	6,67	0	0,00	4	17,39	0	0,00	47	15,26
Total	73		75		68		23		3		308	
G	58		69		68		18		3		244	
H	73		75		68		23		3		308	
I	58		70		68		19		3		261	
R		79,45		92,00		100,00		78,26		100,00		79,22
P		100,00		98,57		100,00		94,74		100,00		93,49

As informações de telefone para contato e validade da oferta ou procura, embora estejam no corpo dos *e-mails*, onde não há uma estrutura tão bem definida quanto no cabeçalho, obtiveram ótimos índices de *recall* e *precision* pois normalmente há termos que indicam a ocorrência delas.

Para a informação “produto”, o objetivo é extrair características adicionais do produto que está sendo oferecido ou procurado, pois a indicação de qual produto se trata já é extraída na informação “assunto”. Para isso, foram criadas várias regras diferentes porque estas características podem se apresentar de diversas formas, dificultando o processo de extração pelo fato de estarem em qualquer ponto do corpo do *e-mail*, sem nenhuma estrutura. Também, para cada *e-mail*, podem ser extraídas várias características relacionadas ao produto ou aos

produtos, o que explica o valor de 308 informações mostrado na linha de total da tabela 9.3 para o produto, embora a análise tenha sido feita em 87 *e-mails*. Pelos números obtidos, pode-se observar que o resultado alcançado para a informação “produto” foi bom, com um alto índice de precisão (93,49%). O índice de *recall*, mesmo ficando em quase 80%, poderia ser melhorado com a inserção de mais termos na base de dados “produto”. É importante destacar também que os índices das variáveis B e E ficaram bem baixos, mostrando que poucas informações foram extraídas incorretamente. Finalmente, um outro ponto a salientar, é que em 26 *e-mails* onde não havia nenhuma indicação de características do produto, em 23 nada foi extraído, comprovando ainda mais o baixo índice de extração de informações incorretas.

10 Conclusões

A presente dissertação definiu e validou uma linguagem de propósito específico para, com base em uma arquitetura de múltiplos níveis, extrair satisfatoriamente as informações desejadas por usuários de listas de distribuição de *e-mails*. O objetivo deste trabalho é desenvolver uma ferramenta para reduzir o tempo utilizado na análise de informações. É importante ressaltar que o processo de extração não substitui a leitura de um texto, mas ajuda a demonstrar se este texto é interessante ou não ao usuário, que assim decidirá se o texto merece ou não uma leitura mais aprofundada.

Os primeiros capítulos serviram para uma apresentação das áreas de recuperação e extração de informações, que servem de contextualização para o desenvolvimento deste trabalho. Deve-se ressaltar que os sistemas para estas áreas devem ser completos e abrangentes, oferecendo recursos eficientes para que o usuário consiga descrever corretamente a informação de que necessita. É importante treinar o usuário no funcionamento da linguagem de consulta e mostrar quais são as características de descrição da informação, ou seja, o modelo adotado pelo sistema. Procurar a informação relevante é um processo complexo que exige muito do usuário, principalmente se ele não está familiarizado com a estrutura completa da informação que precisa ou com o sistema.

Os sistemas desenvolvidos nas áreas de recuperação e extração de informações, como os mostrados nos capítulos 4, 5 e 6, apresentam bons resultados, porém sua utilização por possíveis usuários leigos em informática se torna difícil pelo fato da metodologia e linguagem não serem simples.

Com a linguagem proposta nesta dissertação, que está inserida no projeto denominado Extração Semântica de Informações, dando origem ao SES-MN, acredita-se que houve uma contribuição na busca de um sistema de recuperação e extração de informações com metodologia e linguagem não tão complexas. Os capítulos finais, onde toda linguagem é apresentada e validada, mostrou que o objetivo traçado inicialmente pode ser atingido.

Sabe-se que Extração de Informação é diferente de Recuperação de Informação. Técnicas de RI podem localizar os documentos pertinentes dentro de uma coleção, mas estas estão impossibilitadas de extrair informação dos documentos pertinentes de acordo com critérios específicos. O poder de um sistema de extração de informações comparado a um sistema de recuperação de informações está na habilidade de extrair a informação relevante dos artigos de acordo com um critério específico e a representar em estruturas, as quais sistemas de recuperação são impossibilitados de produzir [COS97]. Uma completa análise nas diferenças entre estas tecnologias pode ser vista no trabalho de Rui Scarinci [SCA2000]. Através destas colocações pode-se concluir que a linguagem proposta nesta dissertação serve tanto para recuperação de informações como para extração de informações, pois possui características das duas tecnologias. No nível de classificação estrutural – P1 há a recuperação de informações dos arquivos de uma coleção como tamanho, data de criação/modificação e estrutura. Já nos níveis de classificação por domínio – P2 e análise superficial – P3 há a extração de informações dos arquivos da coleção como assunto tratado no texto e as informações relevantes de acordo com critérios específicos.

A seguir são apresentadas algumas considerações sobre pontos específicos da linguagem de extração:

- A divisão da linguagem em fases distintas facilita a compreensão e execução de todo processo de extração visto que, ao final de cada fase, o usuário já pode visualizar resultados;
- A divisão das *stop words* em grupos propicia uma flexibilidade na utilização da linguagem de extração em diversos tipos de texto onde, em cada tipo, podem ocorrer *stop words* diferentes. Assim, o usuário pode determinar quais *stop words* serão usadas na análise do texto, de acordo com a sua classe. O benefício gerado é o tempo menor de processamento dos textos;
- De forma semelhante ao caso das *stop words*, a possibilidade do uso de diversas bases de conhecimento para a análise superficial, dependendo da classe (extensão) e categoria (assunto) do arquivo, flexibiliza a utilização da linguagem em amplos tipos de problemas;
- A linguagem é abrangente, pois permite um processamento da linguagem natural superficial com o uso de padrões, análise léxica e de contexto, métodos estatísticos, não sendo tão complexa como NLP;
- Ainda com relação a abrangência da linguagem, a possibilidade de operações lógicas (E,OU) nas regras, além de aumentar a precisão da extração, pode agrupar em uma só regra diversas situações com termos diferentes, mas que tratam do mesmo objeto;
- O uso da linguagem para extração de informações em arquivos que estão em diversos idiomas também eleva sua abrangência;
- A linguagem é de fácil compreensão, ao contrário do HPIEW, sistema apresentado no capítulo 5, que possui uma linguagem enigmática e de difícil interpretação;
- A possibilidade de criar regras com condições dentro de condições (*IF's* aninhados) confere à linguagem um potencial de refinamento nas próprias regras, pois para situações onde o que deve ser avaliado depende de outra avaliação imediatamente anterior, o aninhamento é uma opção, além de proporcionar também uma precisão maior nos resultados de extração obtidos.

Embora os resultados obtidos com a utilização da linguagem proposta tenham sido animadores, não se pode deixar de mencionar alguns problemas que podem aparecer. Mesmo que as fontes geradoras dos textos sejam confiáveis, a informação pode não o ser. Problemas podem ocorrer devido a algumas imperfeições como informação incompleta ou imprecisa. A informação incompleta ocorre quando faltam detalhes de informação, por exemplo, atributos sem valores. No caso do ETOS poderia ser um número de telefone que é esperado logo após a palavra “*phone*” mas não existe. A informação imprecisa ocorre devido à diferença de granularidade, por exemplo, datas sem o dia ou somente referenciando o ano.

Um outro problema que pode prejudicar seriamente o processo de extração são os erros ortográficos nos textos de entrada. A ocorrência destes erros certamente pode diminuir os níveis de *recall* e *precision*, distorcendo os resultados finais.

Finalmente, é de suma importância ressaltar a importância da intervenção humana em todo processo de extração de informações. O conhecimento prévio sobre o domínio influi decisivamente nos resultados finais, pois se o usuário sabe as situações particulares que podem ocorrer nos textos, poderá montar as regras das bases de conhecimento com muito mais segurança e obter resultados mais próximos da realidade. A falta de conhecimento sobre o domínio pode resultar na extração de informações que não podem ser aproveitadas e, conseqüentemente, desperdício de esforços. Uma boa maneira de obter mais conhecimento sobre o domínio é examinando os termos mais freqüentes. Isto permite ao usuário conhecer o estilo dos textos ou o escopo do conteúdo.

Terminado todo processo de extração de informações, a avaliação dos resultados pelo homem também é importante e servirá para mostrar o grau de sucesso atingido e se as informações obtidas satisfazem os objetivos traçados inicialmente.

10.1 Trabalhos Futuros

Todo trabalho desenvolvido até aqui é apenas o início de uma proposta que deve proporcionar subsídios para a continuidade da pesquisa nesta área. A seguir destacam-se dois pontos que podem ser desenvolvidos:

- Novas funções para análise superficial dos textos: nesta dissertação foi estudado e implementado um conjunto de funções para extração de informações. Considerando importante a continuidade deste trabalho e a constante aproximação do mesmo à realidade dos usuários, os quais exigem muitas vezes a extração de informações complexas, observa-se a necessidade da expansão do grupo de funções do SES-MN. Esta expansão visa ampliar o potencial de extração e aplicabilidade do sistema em ambientes reais. Dentro desta expansão podem ser citadas:
 - a criação de uma função que não necessite a colocação de um termo a ser pesquisado, pois em alguns testes notou-se a necessidade da extração de palavras totalmente independentes da existência de termos anteriores ou posteriores;
 - a possibilidade da colocação de um termo coringa nas condições de pesquisa e verificação, ou seja, a especificação apenas parcial do termo a ser procurado;
 - adição de um parâmetro percentual nas funções de extração para aumentar a probabilidade de certeza das informações extraídas;
- Novos testes: como foi apresentado no capítulo 9, o sistema SES-MN foi testado sobre um conjunto de 259 arquivos de entrada. No entanto, os testes devem continuar, visando a correção de novas possíveis falhas, tanto na linguagem como no protótipo, bem como aprofundar a própria validação do estudo realizado. Neste sentido, novos testes permitiriam realizar um estudo quantitativo e qualitativo das informações extraídas. Estes testes poderiam ser

feitos em novos domínios como listas de discussão sobre linguagens de programação, arquivos de código-fonte de programação, chamadas para publicação de artigos e os próprios artigos.

Anexo 1 – Projeto para Validação da Linguagem

• Projeto Validação.sp

Classificação Estrutural

```

seleção de diretórios
  c:\documentos ;
  c:\etos ;
final da seleção de diretórios

seleção de arquivos
  if ((c2ext = ".doc") or (c2ext = ".txt") or
      (c2ext = ".zip")) and (c1size <= 20000) and
      (c3date >= strtodate("10/10/2000")) then
    execute("select");
final da seleção de arquivos

desdobramento de arquivos
  if c2ext = ".zip" then
    execute("expand");
final do desdobramento de arquivos

conversão de arquivos
  if c2ext = ".doc" then
    execute("convert_doc");
final da conversão de arquivos

criação das classes
  DefClasse(texto, todos arquivos texto[.txt])
  DefClasse(doc, todos arquivos documento[.doc])
final da criação das classes

```

Final da Classificação Estrutural

Classificação por Domínio

```

criação de stop words
  DefStopWord(artigos[the,a])
  DefStopWord(pronomes[i,you,he,she,it,we,they,your,our,us,
    his,her,own])
  DefStopWord(outros[to,and,or,is,are,there,have,has,by,
    been,of,in,so,that,before,after,with,against,
    even,from,among,for,without,under,on,as,
    consonant,during,means,therefore,
    consequently,why,because,since,once,just,
    well,although,same,put,less,than,if,case,
    long,unless,much,more,when,while,whenever,
    until,now])
  DefStopWord(numeros cardinais[one,two,three,four,five,
    six,seven,eight,nine,ten,eleven,twelve,
    thirteen,fourteen,fifteen,sixteen,seventeen,
    eighteen,nineteen,twenty,thirty,forty,fifty,
    sixty,seventy,eighty,ninety,hundred,thousand,
    million,billion,trillion])

```

```

DefStopWord(numeros ordinais[first,second,third,fourth,
fifth,sixth,seventh,eighth,ninth,tenth,
twentieth,thirtieth,fortieth,fiftieth,
sixtieth,seventieth,eightieth,ninetieth,
hundredth,thousandth,millionth,billionth,
trillionth])
DefStopWord(numerais multiplicativos[double,triple,
quadruple,quintuple,sextuple])
DefStopWord(numerais fracionarios[half])
DefStopWord(meses[January,February,March,April,May,June,
July,August,September,October,November,
December])
DefStopWord(semana[Sunday,Monday,Tuesday,Wednesday,
Thursday,Friday,Saturday])
DefStopWord(etos portugues[de,enviado,em,para,assunto])
DefStopWord(etos[tradepoint,european,business,eto,posted,
untpdc,electronic,commerce,post,posting,offer,
demand,visit,contact,company,information,
informations,tel,telephone,phone,fax,e-mail,
address,product,products,country,st,rd,lot,
block,fl,st,blvd,box,park,drive,homeurl,
homepage,web,site,website,url,web-site,
description,manufacturer,provide,keyword,easy,
using,form,forms,follow,following,submit,
submitted,road,import,export,floor,street,
apt])
final da criação de stop words

criação das categorias
DefIdentificadores(15)
DefCategoria(informática,arquivos sobre informática
[motherboard,0.90,motherboards,0.90,
byte,0.80,bits,0.85,computer,0.60,
computers,0.60,software,0.90,hardware,0.90,
notebook,0.90,notebooks,0.90,laptop,0.90,
laptops,0.90])
DefCategoria(celular,arquivos sobre celulares[mobile,0.85,
nokia,0.90,battery,0.40,batteries,0.60,
cellular,0.80,batt,0.30,accessories,0.50,
motorola,0.90,ericsson,0.90,samsung,0.90,
qualcomm,0.90])
DefCategoria(alimentação,arquivos sobre alimentação
[bean,0.90,food,0.65,rice,0.90,soy,0.90])
final da criação das categorias

criação dos domínios
DefDominio(texto),(artigos,pronomes,outros,
numeros cardinais,numeros ordinais,
numerais multiplicativos,numerais fracionarios,
meses,semana,etos portugues,etos),
(informática,celular)
DefDominio(doc),(artigos,pronomes,outros),(alimentação)
final da criação dos domínios

Final da Classificação por Domínio

```

Análise Superficial

```

definições
    DefExtracao(texto),(celular),(BC Celular.TXT)
    DefExtracao(texto),(informática),(BC Informática.TXT)
    DefExtracao(doc),(alimentação),(BC Alimentos.TXT)
final das definições

arquivo de saída
    C:\SES-MN\ResultadoFinal.TXT

```

Final da Análise Superficial

- **Base de Dados “Endereço”**

Address	Blv	Fl	Rd
Apt	Blvd	Floor	Road
Avenue	Box	Lot	St
Block	Drive	Park	Street

- **Base de Dados “Produto”**

acer	alcatel	aopen	asus
cd-rom	celeron	compaq	cooler
coolers	cpu	delta	dell
desk	desktop	drive	drives
dvd	epson	ericsson	ethernet
floppy drive	floppy drives	fujitsu	gb
gsm	hard disk	hard drive	hard drives
hdd	hub	hubs	ibm
kb	keyboard	keyboards	main board
main boards	mainboard	mainboards	mb
memories	memory	mitsumi	modem
monitor	monitors	mother board	mother boards
motherboard	motherboards	motorola	mouse
nokia	note book	note books	notebook
notebooks	panasonic	pentium	printer
printers	qualcomm	quantum	ram
sagem	samsung	screen	server
sony	sound card	svga	teac
toshiba	vga	windows	

- **Base de Dados “Países”**

Ar	au	br	chile
china	cl	cn	dakar
eg	egypt	gb	guangdong
il	in	india	ir
iran	israel	jo	korea
kr	lk	malta	md
moldova	mt	new york	ng
nigeria	pakistan	pk	ru
sa	seoul	sg	singapore
sn	taiwan	tr	turkey
tw	united states	united states of america	us
usa	zimbabwe	zw	

- **Base de Dados Auxiliar “Separadores”**

Espaço em branco	,	-
.	/	:
;		

- **Base de Dados Auxiliar “Meses”**

Janeiro	January	Jan	Feb
Fevereiro	February	Fev	Apr
Março	March	Mar	Aug
Abril	April	Abr	Sep
Maio	May	Mai	Oct
Junho	June	Jun	Dec
Julho	July	Jul	
Agosto	August	Ago	
Setembro	September	Set	
Outubro	October	Out	
Novembro	November	Nov	
Dezembro	December	Dez	

- **Base de Dados Auxiliar “Dias da Semana”**

Domingo	Sunday	Dom	Sun
Segunda-feira	Monday	Seg	Mon
Terça-feira	Tuesday	Ter	Tue
Quarta-feira	Wednesday	Qua	Wed
Quinta-feira	Thursday	Qui	Thu
Sexta-feira	Friday	Sex	Fri
Sábado	Saturday	Sab	Sat

- **Base de Conhecimento “BC Alimentos.TXT”**

{Aqui é um comentário - Validação para a Dissertação}

```
if PesqString(De,CxLivre,1,5) or PesqString(From,CxLivre,1,5)
  and VerifCaracter(:,CxLivre,1,1,Palavra,DEP) and
  VerifCaracter(@,CxLivre,1,2,Palavra,DEP) then
  Copia(1,Linha,DEP,NALT,EMAIL);
```

```
if PesqString(Enviado,CxLivre,1,5) then
begin
  if PesqString(em,CxLivre,1,5) and
    VerifCaracter(:,CxLivre,1,1,Palavra,DEP) and
    verifdata(1,1,Linha,DEP) then
    Copia(1,Linha,DEP,NALT,DATAENVIO)
end;
```

```
{Aqui é um comentário}
if PesqString(Subject,CxLivre,1,5) or
  PesqString(Assunto,CxLivre,1,5) and
  VerifCaracter(:,CxLivre,1,1,Palavra,DEP) then
begin
  {Aqui é um comentário}
  Copia(1,Palavra,DEP,NALT,TIPO)
  Desloca(1,Palavra,DEP)
  Copia(1,Palavra,DEP,NALT,SIGLAPAIS)
  Desloca(1,Palavra,DEP)
  Copia(1,Linha,DEP,NALT,ASSUNTO)
end;
```

```
if PesqBase(produto,CxLivre,5,100) then
  Copia(1,Frase,DEP,NALT,PRODUTO);
```

```
if PesqString(looking,CxLivre,5,100) then
begin
  if PesqString(for,CxLivre,5,100) then
    Copia(1,Frase,DEP,NALT,PRODUTO)
end;
```

```
if PesqString(want,CxLivre,5,100) then
begin
  if PesqString(to,CxLivre,5,100) then
  begin
    if PesqString(buy,CxLivre,5,100) then
      Copia(1,Frase,DEP,NALT,PRODUTO)
    end
  end
end;
```

```
if PesqString(purchase,CxLivre,5,100) then
begin
  if PesqString(of,CxLivre,5,100) then
    Copia(1,Frase,DEP,NALT,PRODUTO)
end;
```

```

if PesqString(fax,CxLivre,8,100) or
  PesqString(telefax,CxLivre,8,100) and
  VerifCaracter(:,CxLivre,1,1,Palavra,DEP) then
  Copia(1,Linha,DEP,NALT,FAX);

```

- **Base de Conhecimento “BC Celular.TXT”**

```

{Aqui é um comentário - Validação para a Dissertação}

```

```

if PesqString(De,CxLivre,1,5) or PesqString(From,CxLivre,1,5)
  and VerifCaracter(:,CxLivre,1,1,Palavra,DEP) and
  VerifCaracter(@,CxLivre,1,2,Palavra,DEP) then
begin
  Copia(1,Linha,DEP,NALT,EMAIL)
  Desloca(1,Linha,DEP)
end;

```

```

if PesqString(Enviado,CxLivre,1,5) then
begin
  if PesqString(em,CxLivre,1,5) and
    VerifCaracter(:,CxLivre,1,1,Palavra,DEP) and
    verifdata(1,1,Linha,DEP) then
    begin
      Copia(1,Linha,DEP,NALT,DATAENVIO)
      Desloca(2,Linha,DEP)
    end
end;

```

```

{Aqui é um comentário}
if PesqString(Subject,CxLivre,1,5) or
  PesqString(Assunto,CxLivre,1,5) and
  VerifCaracter(:,CxLivre,1,1,Palavra,DEP) then
begin
  {Aqui é um comentário}
  Copia(1,Palavra,DEP,ALT,TIPO)
  Copia(1,Palavra,DEP,ALT,SIGLAPAIIS)
  Copia(1,Linha,DEP,ALT,ASSUNTO)
  Desloca(1,Linha,DEP)
end;

```

```

if PesqBase(produto,CxLivre,5,500) then
begin
  Copia(1,Linha,TOT,NALT,PRODUTO1)
  Desloca(1,Linha,DEP)
end;

```

```

if PesqString(desktop,CxLivre,5,500) or
  PesqString(desk,CxLivre,5,500) and
  VerifString(ram,CxLivre,1,2,Linha,DEP) and
  VerifString(hdd,CxLivre,1,2,Linha,DEP) and
  VerifString(modem,CxLivre,1,2,Linha,DEP) then
  Copia(2,Linha,TOT,NALT,PRODUTO2);

```

```

if PesqString(ready,CxLivre,5,500) then
begin
  if PesqString(to,CxLivre,5,500) then
  begin
    if PesqString(buy,CxLivre,5,500) then
    begin
      Copia(1,Frase,DEP,NALT,PRODUTO3)
      Desloca(1,Frase,DEP)
    end
  end
end;

if PesqString(interested,CxLivre,5,500) then
begin
  if PesqString(to,CxLivre,5,500) then
  begin
    if PesqString(import,CxLivre,5,500) then
    begin
      Copia(1,Frase,DEP,NALT,PRODUTO4)
      Desloca(1,Frase,DEP)
    end
  end
end;

if PesqString(to,CxLivre,5,500) then
begin
  if PesqString(purchase,CxLivre,5,500) then
  begin
    Copia(1,Frase,DEP,NALT,PRODUTO5)
    Desloca(1,Frase,DEP)
  end
end;

if PesqString(looking,CxLivre,5,500) then
begin
  if PesqString(for,CxLivre,5,500) then
  begin
    Copia(1,Frase,DEP,NALT,PRODUTO6)
    Desloca(1,Frase,DEP)
  end
end;

if PesqBase(endereco,CxLivre,8,100) and
VerifBase(países,CxLivre,1,4,Linha,DEP) then
begin
  Desloca(1,Linha,ANT)
  CopiaAte(Base,países,1,CxLivre,4,Linha,DEPINC,NALT,
    ENDERECO_BASE)
  Desloca(1,Linha,DEP)
end;

if {Aqui é um comentário} PesqBase(endereco,CxLivre,8,100) then
begin
  Copia(3,Linha,TOT,NALT,ENDERECO)
  Desloca(1,Linha,DEP)
end;

```

```

if PesqString(tel,CxLivre,8,100) or
  PesqString(PHONE,CxLivre,8,100) or
  PesqString(telephone,CxLivre,8,100) and
  VerifCaracter(:,CxLivre,1,1,Palavra,DEP) then
  Copia(1,Linha,DEP,NALT,TELEFONE);

{***** OUTRAS FUNÇÕES PARA TESTE *****)}

if PesqData(5,100) then
  CopiaCond(OUTRADATA);

if PesqString(macaco,CxLivre,5,100) then
begin
  DeslocaAte(String,velho,3,CxLivre,1,Linha,DEP)
  Copia(1,Linha,DEP,NALT,ANIMAL)
end;

if Pesq(String,total,"",CxLivre,5,100) then
  CopiaAte(String,Reais,1,CxEsp,1,Linha,DEP,NALT,VALOR);

if PesqString(casa,CxLivre,5,100) then
begin
  if PesqString(amarela,CxLivre,5,100) then
  begin
    Copia(1,Frase,DEP,NALT,MORADIA)
  end
  else
  begin
    Copia(3,Palavra,ANT,NALT,NAOMORADIA)
  end
end;

if Pesq(String,CL,[,],CxEsp,5,100) then
  CopiaTexto(Chile,PAIS);

```

- **Base de Conhecimento “BC Informática.TXT”**

```

{Aqui é um comentário - Validação para a Dissertação}

if PesqString(De,CxLivre,1,5) or PesqString(From,CxLivre,1,5)
  and VerifCaracter(:,CxLivre,1,1,Palavra,DEP) and
  VerifCaracter(@,CxLivre,1,2,Palavra,DEP) then
  Copia(1,Linha,DEP,NALT,EMAIL);

if PesqString(Enviado,CxLivre,1,5) then
begin
  if PesqString(em,CxLivre,1,5) and
    VerifCaracter(:,CxLivre,1,1,Palavra,DEP) and
    verifdata(1,1,Linha,DEP) then
    Copia(1,Linha,DEP,NALT,DATAENVIO)
end;

```



```

{Aqui é um comentário}
if PesqString(Subject,CxLivre,1,5) or
  PesqString(Assunto,CxLivre,1,5) and
  VerifCaracter(:,CxLivre,1,1,Palavra,DEP) then
begin
  {Aqui é um comentário}
  Copia(1,Palavra,DEP,NALT,TIPO)
  Desloca(1,Palavra,DEP)
  Copia(1,Palavra,DEP,NALT,SIGLAPAIS)
  Desloca(1,Palavra,DEP)
  Copia(1,Linha,DEP,NALT,ASSUNTO)
end;

if PesqBase(produto,CxLivre,5,500) then
begin
  Copia(1,Linha,TOT,NALT,PRODUTO1)
  Desloca(1,Linha,DEP)
end;

if PesqString(desktop,CxLivre,5,500) or
  PesqString(desk,CxLivre,5,500) and
  VerifString(ram,CxLivre,1,2,Linha,DEP) and
  VerifString(hdd,CxLivre,1,2,Linha,DEP) and
  VerifString(modem,CxLivre,1,2,Linha,DEP) then
  Copia(2,Linha,TOT,NALT,PRODUTO2);

if PesqString(ready,CxLivre,5,500) then
begin
  if PesqString(to,CxLivre,5,500) then
  begin
    if PesqString(buy,CxLivre,5,500) then
    begin
      Copia(1,Frase,DEP,NALT,PRODUTO3)
      Desloca(1,Frase,DEP)
    end
  end
end;

if PesqString(interested,CxLivre,5,500) then
begin
  if PesqString(to,CxLivre,5,500) then
  begin
    if PesqString(import,CxLivre,5,500) then
    begin
      Copia(1,Frase,DEP,NALT,PRODUTO4)
      Desloca(1,Frase,DEP)
    end
  end
end;

if PesqString(to,CxLivre,5,500) then
begin
  if PesqString(purchase,CxLivre,5,500) then
  begin
    Copia(1,Frase,TOT,ALT,PRODUTO5)
    Desloca(1,Frase,DEP)
  end
end;

```

```

if PesqString(looking,CxLivre,5,500) then
begin
  if PesqString(for,CxLivre,5,500) then
  begin
    Copia(1,Frase,DEP,NALT,PRODUTO6)
    Desloca(1,Frase,DEP)
  end
end;

if PesqBase(endereco,CxLivre,8,100) and
  VerifBase(países,CxLivre,1,4,Linha,DEP) then
begin
  Desloca(1,Linha,ANT)
  CopiaAte(Base,países,1,CxLivre,4,Linha,DEPINC,NALT,
    ENDERECO_BASE)
  Desloca(1,Linha,DEP)
end;

if PesqBase(endereco,CxLivre,8,100) then
begin
  Copia(3,Linha,TOT,NALT,ENDERECO)
  Desloca(1,Linha,DEP)
end;

if PesqString(tel,CxLivre,8,100) or
  PesqString(PHONE,CxLivre,8,100) or
  PesqString(telephone,CxLivre,8,100) and
  VerifCaracter(:,CxLivre,1,1,Palavra,DEP) then
  Copia(1,Linha,DEP,NALT,TELEFONE);

if PesqString(web,CxLivre,8,100) or
  PesqString(homeurl,CxLivre,8,100) or
  PesqString(homepage,CxLivre,8,100) or
  PesqString(site,CxLivre,8,100) or
  PesqString(website,CxLivre,8,100) or
  PesqString(url,CxLivre,8,100) and
  VerifCaracter(:,CxLivre,1,1,Palavra,DEP) then
  Copia(1,Linha,DEP,NALT,WEBSITE);

if PesqString(valid,CxLivre,5,100) or
  PesqString(validity,CxLivre,5,100) then
begin
  if PesqString(until,CxLivre,5,100) or
    PesqString(of,CxLivre,5,100) and
    VerifCaracter(:,CxLivre,1,4,Palavra,DEP) then
    Copia(1,Linha,DEP,NALT,VALIDADE)
end;

```

Anexo 2 – *E-mail's* Utilizados na Validação

- **Alguns *e-mail's* do ETOS utilizados na validação**

DEMAND CL REFURBISHED CELLULAR PHONES .txt

De: bmchile@yahoo.es
 Enviado em: terça-feira, 30 de janeiro de 2001 23:11
 Para: demand-mail@heuristic.untfdc.org
 Assunto: DEMAND: [CL] REFURBISHED CELLULAR PHONES

The following ETO was submitted by
 M. Rojas (bmchile@yahoo.es)
 on Wednesday, January 31, 2001 at 11:11:27

 Meu amigo Pedro mora em Guaporé em uma casa amarela muito bonita. Ela tem dois andares.

December 31, 2001

Assunto: DEMAND: [CL] REFURBISHED CELLULAR PHONES

João e Maria são os pais de Adriana que casa no dia 11/04/2002, na Igreja Matriz.

22/10/2002

Contact_name: M. Rojas
 Email: bmchile@yahoo.es
 Company: B.M. & Otro , Telephony Div.
 Address: P.O.Box 828,
 Argomedeo 125 ,
 Curico - Chile .

"TOTAL" : 5.000,00 Reais para todas as obras necessárias.

33-17-1234

O macaco velho é muito velho mesmo, mais velho do que cem anos.

ETO:
 Looking for refurbished , used , working conditions , home power electricity 220 V , 50 Hz , battery and charger included .

We preffer digital cellular phones , 900Mhz ; and analogs depending prices .

Ericsson DH 618 , DF 688 , Nokia 2160 , Nokia 5120 , etc.
 Other analogs like as : Samsung SH 800 , Motorola Elite 5000 , etc .

Regards ,

Easy ETO posting using our web forms at <http://eto.untpdc.org/>

REMOTE_HOST: 164.77.245.218
HTTP_USER_AGENT: Mozilla/4.0 (compatible; MSIE 5.0; Windows 98; DigExt)

DEMAND IL COMPUTER KEYBOARD COVERS .txt

De: yeultecs@zahav.net.il
Enviado em: domingo, 14 de janeiro de 2001 22:52
Para: demand-mail@heuristic.untpdc.org
Assunto: DEMAND: [IL] COMPUTER & KEYBOARD COVERS

=====
=====
This ETO has been posted to UNTPDC
by the EC21(Electronic Commerce 21) <http://www.ec21.com>
To post your own offer, visit <http://www.ec21.com>
=====
=====

Product Description:

Dear sir
we need sets of coveres for computer systems
a/ keyboard standard
b/ cpu mini tower
c/ screen 17-19"

quantity - 1000 as a start + 1000 pc's from time to time
including 1 color printing of logo.

we will need a sample (not printed) befor we make our order

looking forward to your offers

Eli Zamet
Yeul Tec Support
Israel
Valid Until : 2001. 01. 30.

[Contact & Company Information]

Contact : Eli ZAmet
Company : Yeul Tec Support
 p.o.b. 711 karkur, Israel
 37106
 Israel
Tel : 972-52-452937-0
Fax : 972-151-54-452937
E-mail : yeultecs@zahav.net.il
Homeurl : <http://www.ec21.com/yeultecs>

General Information :

```
=====
=====
This ETO has been posted to UNTPDC
by the EC21(Electronic Commerce 21) http://www.ec21.com
To post your own offer, visit http://www.ec21.com
=====
=====
```

DEMAND JO NOKIA .txt

De: accutime@globalone.com.jo
 Enviado em: sábado, 13 de janeiro de 2001 20:34
 Para: demand-mail@heuristic.untpdc.org
 Assunto: DEMAND: [JO] NOKIA

```
-----
      Product Informations
-----
-Product   : GSM Phones
-Description
```

Looking for supply of new Nokia 9210 Communicator & Nokia 3310.

Serious offers only, PLEASE

```
-----
      Company Informations
-----
-Company Name : AccuTime Ltd
-Address      : Sweifiyeh Amman
-Country      : JO
-TEL          : 962-6-585-4871
-FAX          : 962-6-585-4872
-Contact Name : Fady Shakaa
```

```
* REMOTE_HOST : 217.23.33.3
* USER_AGENT  : Mozilla/4.0 (compatible; MSIE 5.5; Windows NT
5.0)
```

```
=====
      Tpage.com (http://www.Tpage.com)
      Trade Search Engine / Auto Multi-Posting
=====
```

DEMAND TR USED COMPUTERS .txt

De: aakyildiz@kobim.com
 Enviado em: sexta-feira, 2 de fevereiro de 2001 13:47
 Para: demand-mail@heuristic.untpdc.org
 Assunto: DEMAND: [TR] USED COMPUTERS

* BUY offer to buy used computers [Turkey]

Offer Dated: Friday, 2 February 2001 15:46:51 GMT

We want to buy used computers, printers and fax machines.
Please let us know urgently if you can help.

Contact: aakyildiz@kobim.com

armagan akyildiz (import export)
kobim import export
cumhuriyet blv. no:36/510
izmir 35250, Turkey ()
Tel: 90 232 4893959
Fax: 90 232 4893983
URL: <http://www.kobim.com>

This ETO has been posted to UNTPDC by:

OPTIMA EUROPEAN TRADE BOARD
since 1995

Official European bulletin board for posting offers, business
opportunities, and responses to them. Posting is free.

<http://www.trade-board.com/>

OFFER CN COMPUTER MONITOR .txt

De: www70_98@yahoo.com
Enviado em: terça-feira, 28 de novembro de 2000 09:22
Para: offer-mail@heuristic.untppdc.org
Assunto: OFFER: [CN] COMPUTER MONITOR

The following ETO was submitted by
James Lee (www70_98@yahoo.com)
on Tuesday, November 28, 2000 at 21:21:53

Contact_name: James Lee
Email: www70_98@yahoo.com
Company: Ningbo Universal Micro-Electronics Co., LTD.
Telephone: 8657477-8672
Fax: 865747708672
Address: No.144 Sijia Xiang Ningbo 315000 P.R.China

ETO:
We produce 15 inch computer color monitor, FOB Ningbo China
USD102/pc. SPEC:1280x1024 100KHZ DDC1/DD2B

Easy ETO posting using our web forms at <http://eto.untppdc.org/>

REMOTE_HOST: 61.130.141.225
HTTP_USER_AGENT: Mozilla/4.0 (compatible; MSIE 5.0; Windows 98;
DigExt)

- **Conteúdo do arquivo “ResultadoFinal.TXT”**

```

c:\etos\demand cl refurbished cellular phones .txt;
    EMAIL;bmchile@yahoo.es;
c:\etos\demand cl refurbished cellular phones .txt;
    DATAENVIO;terça-feira, 30 de janeiro de 2001 23:11;
c:\etos\demand cl refurbished cellular phones .txt;
    TIPO;DEMAND;;
c:\etos\demand cl refurbished cellular phones .txt;
    SIGLAPAIS;[CL];
c:\etos\demand cl refurbished cellular phones .txt;
    ASSUNTO;REFURBISHED CELLULAR PHONES;
c:\etos\demand cl refurbished cellular phones .txt;
    OUTRADATA;Wednesday, January 31, 2001;
c:\etos\demand cl refurbished cellular phones .txt;
    MORADIA;muito bonita.;
c:\etos\demand cl refurbished cellular phones .txt;
    OUTRADATA;December 31, 2001;
c:\etos\demand cl refurbished cellular phones .txt;
    PAIS;Chile;
c:\etos\demand cl refurbished cellular phones .txt;
    NAOMORADIA;de Adriana que;
c:\etos\demand cl refurbished cellular phones .txt;
    OUTRADATA;11/04/2002;;
c:\etos\demand cl refurbished cellular phones .txt;
    OUTRADATA;22/10/2002;
c:\etos\demand cl refurbished cellular phones .txt;
    ENDERECO_BASE;Address: P.O.Box 828, Argomedeo 125 , Curico
    - Chile;
c:\etos\demand cl refurbished cellular phones .txt;
    VALOR;5.000,00 Reais;
c:\etos\demand cl refurbished cellular phones .txt;
    ANIMAL;do que cem anos.;
c:\etos\demand cl refurbished cellular phones .txt;
    PRODUTO6;refurbished , used , working conditions , home
    power electricity 220 V , 50 Hz , battery and charger
    included .;
c:\etos\demand cl refurbished cellular phones .txt;
    PRODUTO1;Ericsson DH 618 , DF 688 , Nokia 2160 , Nokia 5120
    , etc.;
c:\etos\demand cl refurbished cellular phones .txt;
    PRODUTO1;Other analogs like as : Samsung SH 800 , Motorola
    Elite 5000 , etc .;
c:\etos\demand cl refurbished cellular phones .txt;
    PRODUTO1;HTTP_USER_AGENT: Mozilla/4.0 (compatible; MSIE
    5.0; Windows 98; DigExt);
c:\etos\demand il computer keyboard covers .txt;
    EMAIL;yeultecs@zahav.net.il;
c:\etos\demand il computer keyboard covers .txt;
    DATAENVIO;domingo, 14 de janeiro de 2001 22:52;
c:\etos\demand il computer keyboard covers .txt;
    TIPO;DEMAND;;
c:\etos\demand il computer keyboard covers .txt;
    SIGLAPAIS;[IL];
c:\etos\demand il computer keyboard covers .txt;
    ASSUNTO;COMPUTER & KEYBOARD COVERS;

```

```

c:\etos\demand il computer keyboard covers .txt;
    PRODUTO1;a/ keyboard standard;
c:\etos\demand il computer keyboard covers .txt;
    PRODUTO1;b/ cpu mini tower;
c:\etos\demand il computer keyboard covers .txt;
    PRODUTO1;c/ screen 17-19";
c:\etos\demand il computer keyboard covers .txt;
    VALIDADE;2001. 01. 30.;
c:\etos\demand il computer keyboard covers .txt;
    TELEFONE;972-52-452937-0;
c:\etos\demand il computer keyboard covers .txt;
    WEBSITE;http://www.ec21.com/yeultecs;
c:\etos\demand jo nokia .txt;
    EMAIL;accutime@globalone.com.jo;
c:\etos\demand jo nokia .txt;
    DATAENVIO;sábado, 13 de janeiro de 2001 20:34;
c:\etos\demand jo nokia .txt;
    TIPO;DEMAND;;
c:\etos\demand jo nokia .txt;
    SIGLAPAIS;[JO];
c:\etos\demand jo nokia .txt;
    ASSUNTO;NOKIA;
c:\etos\demand jo nokia .txt;
    PRODUTO1;Product : GSM Phones;
c:\etos\demand jo nokia .txt;
    PRODUTO6;supply of new Nokia 9210 Communicator & Nokia
    3310.;
c:\etos\demand jo nokia .txt;
    ENDERECO_BASE;-Address : Sweifiyeh Amman -Country
    : JO;
c:\etos\demand jo nokia .txt;
    TELEFONE;962-6-585-4871;
c:\etos\demand jo nokia .txt;
    PRODUTO1;USER_AGENT : Mozilla/4.0 (compatible; MSIE 5.5;
    Windows NT 5.0);
c:\etos\demand tr used computers .txt;
    EMAIL;aakyildiz@kobim.com;
c:\etos\demand tr used computers .txt;
    DATAENVIO;sexta-feira, 2 de fevereiro de 2001 13:47;
c:\etos\demand tr used computers .txt;
    TIPO;DEMAND;;
c:\etos\demand tr used computers .txt;
    SIGLAPAIS;[TR];
c:\etos\demand tr used computers .txt;
    ASSUNTO;USED COMPUTERS;
c:\etos\demand tr used computers .txt;
    PRODUTO1;We want to buy used computers, printers and fax
    machines.Please;
c:\etos\demand tr used computers .txt;
    ENDERECO_BASE;cumhuriyet blv. no:36/510 izmir
    35250, Turkey;
c:\etos\demand tr used computers .txt;
    TELEFONE;90 232 4893959;
c:\etos\demand tr used computers .txt;
    WEBSITE;http://www.kobim.com;
c:\etos\offer cn computer monitor .txt;
    EMAIL;www70_98@yahoo.com;

```



```
c:\etos\offer cn computer monitor .txt;  
    DATAENVIO;terça-feira, 28 de novembro de 2000 09:22;  
c:\etos\offer cn computer monitor .txt;  
    TIPO;OFFER;  
c:\etos\offer cn computer monitor .txt;  
    SIGLAPAIS;[CN];  
c:\etos\offer cn computer monitor .txt;  
    ASSUNTO;COMPUTER MONITOR;  
c:\etos\offer cn computer monitor .txt;  
    TELEFONE;8657477-8672;  
c:\etos\offer cn computer monitor .txt;  
    ENDERECO_BASE;Address: No.144 Sijia Xiang Ningbo 315000  
    P.R.China;  
c:\etos\offer cn computer monitor .txt;  
    PRODUTO1;We produce 15 inch computer color monitor, FOB  
    Ningbo China USD102/pc. SPEC:1280x1024 100KHZ DDC1/DD2B;  
c:\etos\offer cn computer monitor .txt;  
    PRODUTO1;HTTP_USER_AGENT: Mozilla/4.0 (compatible; MSIE  
    5.0; Windows 98; DigExt);
```

Bibliografia

- [BEC97] BECKER, C. C.; DEITOS, H. R. **SIRI – Sistema Inteligente de Recuperação de Informações**. Projeto de Diplomação (Bacharelado em Ciência da Computação) – Instituto de Informática, Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre.
- [BOR97] BORLAND INTERNATIONAL. **Borland Delphi 3 Object Pascal Language Guide**. Scotts Valley, CA, USA, 1997.
- [BOR97a] BORLAND INTERNATIONAL. **Borland Delphi 3 User's Guide**. Scotts Valley, CA, USA, 1997.
- [BOR97b] BORLAND INTERNATIONAL. **Borland Delphi 3 Visual Component Library Reference**. Scotts Valley, CA, USA, 1997. v. 1 e 2.
- [CAN96] CANTÙ, M. **Dominando o Delphi – “A Bíblia”**. São Paulo: Makron Books, 1996.
- [CAR2000] CARUANA, R. et al. High Precision Information Extraction. In: ACM SIGKDD INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, 2000. **Papers**. Boston: [s.n.], 2000.
- [COS96] COSTANTINO, M. et al. **Financial information extraction using pre-defined and user-definable templates in the LOLITA System**. Disponível em: <http://www.advanced-finance.co.uk/marco.html>>. Acesso em: 11 jan. 2001.
- [COS97] COSTANTINO, M. et. al. **Natural Language Processing and Information Extraction: Qualitative Analysis of Financial News Articles**. Disponível em: <http://www.advanced-finance.co.uk/marco.html>>. Acesso em: 11 jan. 2001.
- [COW96] COWIE, J.; LEHNERT, W. Information Extraction. **Communications of the ACM**, New York, v. 39, n. 1, Jan. 1996.
- [CRO82] CROCKER, D. H. **Standard for the Format of Arpa Internet Text Messages**. RFC #822. Newark, DE: Dept. of Electrical Engineering University of Delaware, 1982. Disponível em: <http://www.ietf.org/rfc/rfc0822.txt?number=822>>. Acesso em: 24 jan. 2001.
- [ELE2000] ELECTRONIC TRADING OPPORTUNITIES (ETO) SYSTEM, developed by the United Nations Trade Point Development Center, UNTPDC. Disponível em: <http://www.unicc.org/untpdc/welcome.html>>. Acesso em: 11 dez. 2000.

- [FAY96] FAYYAD, U. M.; PIATETSKY-SHAPIO, G.; SMYTH, P. **From Data Mining to Knowledge Discovery: An Overview**. Menlo Park, CA: AAAI Press/The MIT Press, 1996.
- [FEL97] FELDENS, M. A. **Engenharia da Descoberta de Conhecimento em Bases de Dados: estudo e aplicação na área de saúde**. 1997. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.
- [FER97] FERREIRA, S. N. **Mecanismos para Classificação e Recuperação de Informações em Correio Eletrônico**. 1997. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre.
- [FER97a] FERREIRA, S. N. A Query Language for Retrieving Information in Electronic Mail Environments. In: SIMPÓSIO BRASILEIRO DE BANCO DE DADOS, SBBD, 12., 1997, Fortaleza. **Anais...** Fortaleza: UFC, 1997.
- [GLO97] GLOOR, P. A. **Short Introduction to Information Retrieval**. Elements of Hypermedia Design – Techniques for Navigation and Visualization in Cyberspace. Birkhäuser – Scientific and Technical Publishing. Switzerland, 1997. Disponível em: <<http://www.birkhauser.com/hypermedia/hypermedia.html>>. Acesso em: 7 nov. 2000.
- [GOL92] GOLDBERG, D. et al. Using Collaborative Filtering to Wave an Information Tapestry. **Communications of the ACM**, New York, v. 35, n. 12, Dec. 1992.
- [GRI94] GRISHMAN, R. et al. Complex Syntax: Building a Computational Lexicon. In: INTERNATIONAL CONFERENCE ON COMPUTATIONAL LINGUISTICS, COLING, 15., 1994, Kyoto, Japan. **Proceedings...** [S.l.: s.n.], 1994.
- [GRI95] GRISHMAN, R. Design of the MUC-6 Evaluation. In: MESSAGE UNDERSTANDING CONFERENCE, 6., 1995. **Proceedings...** San Francisco: Morgan Kaufmann, 1995.
- [GRI96] GRISHMAN, R.; SUNDHEIM, B. Message Understanding Conference – 6: A Brief History. In: INTERNATIONAL CONFERENCE ON COMPUTATIONAL LINGUISTICS, COLING, 16., 1996, Copenhagen, Denmark. **Proceedings...** [S.l.: s.n.], 1996.
- [GRI97] GRISHMAN, R. Information Extraction: techniques and challenges. In: INTERNATIONAL SUMMER SCHOOL ON INFORMATION EXTRACTION, SCIE, 1997, Frascati, It. **Information Extraction: a multidisciplinary approach to an emerging information technology**. Berlin: Springer-Verlag, 1997. (Lecture Notes in Artificial Intelligence, v. 1299).

- [HAN94] HAN, J.; FU, Y. **Discovery of Multiple-Level Association Rules from Large Databases.** Disponível em: <http://db.cs.sfu.ca/sections/publication/kdd/kdd.html>>. Acesso em: 20 nov. 2000.
- [HAN95] HAN, J. **Mining Knowledge at Multiple Concept Levels.** Disponível em: <http://db.cs.sfu.ca/sections/publication/kdd/kdd.html>>. Acesso em: 20 nov. 2000.
- [HIL85] HILTZ, S. R.; TUROFF, M. Structuring Computer-Mediated Communication Systems to Avoid Information Overload. **Communications of the ACM**, New York, v. 28, n. 7, Jul. 1985.
- [HOB96] HOBBS, J. R. et al. **FASTUS: A Cascaded Finite-State Transducer for Extracting Information from Natural-Language Text.** Disponível em: <http://www.ai.sri.com/pubs/papers/Hobb96:FASTUS/document.html>>. Acesso em: 12 jan. 2001.
- [LAQ93] LAQUEY, T.; RYER, J. R. **The Whole Internet User's Guide & Catalog.** New York: Addison-Wesley, 1993.
- [LEW95] LEWIS, D. D. **Active by Accident: Relevance Feedback in Information Retrieval.** Disponível em: <http://cora.whizbang.com>>. Acesso em: 20 nov. 2000.
- [MAC88] MACKAY, W. Diversity in the Use of Electronic Mail: A Preliminary Inquiry. **ACM Transactions on Office Information Systems**, New York, v. 6, n. 4, Apr. 1988.
- [MCC85] McCUNE, B. P. et al. RUBRIC: A System for Rule-Based Information Retrieval. **IEEE Transactions on Software Engineering**, New York, v. SE-11, n. 9, Sept. 1985.
- [PRO2000] THE PROTEUS PROJECT. Disponível em: <http://cs.nyu.edu/cs/projects/proteus>>. Acesso em: 10 ago. 2000.
- [RIL93] RILOFF, E. Automatically Constructing a Dictionary for Information Extraction Tasks. In: NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE, AAAI, 11., 1993. **Proceedings...** Disponível em: <http://cora.whizbang.com>>. Acesso em: 20 nov. 2000.
- [RIL94] RILOFF, E.; LEHNERT, W. **Information Extraction as a Basis for Portable Text Classification System.** Disponível em: <http://cora.whizbang.com>>. Acesso em: 20 nov. 2000.
- [RIL94a] RILOFF, E.; LEHNERT, W. Information Extraction as a Basis for High-Precision Text Classification. **ACM Transactions on Information Systems**, New York, v. 12, n. 3, July 1994. Disponível em: <http://cora.whizbang.com>>. Acesso em: 20 nov. 2000.

- [RIL99] RILOFF, E.; JONES, R. Learning Dictionaries for Information Extraction by Multi-Level Bootstrapping. In: NATIONAL CONFERENCE ON ARTIFICIAL INTELIGENCE, AAAI, 16., 1999, Orlando. **Proceedings...** Menlo Park: AAAI Press, 1999.
- [ROB91] ROBINSON, M. Through a Lens Smartly. **Byte Magazine**, May 1991.
- [ROB97] ROBERTSON, S. E. The Probability Ranking Principle in IR. In: SPARCK-JONES, Karen; WILLET, Peter (Ed.). **Readings in Information Retrieval**. San Francisco: Morgan Kaufmann, 1997.
- [SAL68] SALTON, G. **Automatic Information Organization and Retrieval**. New York: McGraw-Hill, 1968.
- [SAL83] SALTON, G. **Introduction to Modern Information Retrieval**. New York: McGraw-Hill, 1983.
- [SAL87] SALTON, G.; BUCKLEY, C. **Term Weighting Approaches in Automatic Text Retrieval**. New York: Cornell University, 1987. 21 p. (Technical Report 87-881).
- [SCA97] SCARINCI, R. G. **SES – Sistema de Extração Semântica de Informações**. 1997. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.
- [SCA97a] SCARINCI, R. G.; OLIVEIRA, J. P. M. SES – Sistema de Extração Semântica de Informações. In: SIMPÓSIO BRASILEIRO DE BANCO DE DADOS, SBBD, 12., 1997, Fortaleza. **Anais...** Fortaleza: UFC, 1997.
- [SCA2000] SCARINCI, R. G. **Extração de Informação**. 2000. Exame de Qualificação (Doutorado em Ciência da Computação) – Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.
- [SCA2001] SCARINCI, R. G. **Uma Metodologia para Tratamento e Manipulação de Informações de E-Commerce**. 2001. Proposta de Tese (Doutorado em Ciência da Computação) – Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.
- [SPA97] SPARCK-JONES, K.; WILLET, P. **Readings in Information Retrieval**. San Francisco: Morgan Kaufmannm, 1997.
- [TRE2000] TREIN, D. S. **Uma Arquitetura em Múltiplos Níveis para Extração de Informações**. Projeto de Diplomação (Bacharelado em Ciência da Computação) – Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.

- [TUR91] TURTLE, H.; CROFT, B. W. Evaluation of an Inference Network Based Retrieval Model. **ACM Transactions on Information Systems**, New York, v. 9, n. 3, July 1991.
- [WEI88] WEISS, S. M.; KULIKOWSKI, C. A. **Guia Prático para Projetar Sistemas Especialistas**. Rio de Janeiro: LTC – Livros Técnicos e Científicos, 1988.
- [WIV97] WIVES, L. K. **Um Estudo sobre Técnicas de Recuperação de Informações com Ênfase em Informações Textuais**. 1997. Trabalho Individual (Mestrado em Ciência da Computação) – Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.
- [WIV99] WIVES, L. K. **Um Estudo sobre Agrupamento de Documentos Textuais em Processamento de Informações Não Estruturadas usando Técnicas de “Clustering”**. 1999. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.
- [ZAI97] ZAIANE, O. R. **Resource and Knowledge Discovery on the Internet: Paving the Information Super-Highway**. Disponível em: <http://faz.sfu.ca/cs/people/GradStudents/zaiane/personal/htmltxt/abstractEng.html>. Acesso em: 20 nov. 2000.