FREDERICO LUIZ GONÇALVES DE FREITAS

SISTEMAS MULTIAGENTES COGNITIVOS PARA A RECUPERAÇÃO, CLASSIFICAÇÃO E EXTRAÇÃO INTEGRADAS DE INFORMAÇÃO DA WEB

FLORIANÓPOLIS
2002

UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

SISTEMAS MULTIAGENTES COGNITIVOS PARA RECUPERAÇÃO, CLASSIFICAÇÃO E EXTRAÇÃO INTEGRADAS DE INFORMAÇÃO DA WEB

Tese submetida à
Universidade Federal de Santa Catarina
como parte dos requisitos para a
obtenção do grau de Doutor em Engenharia Elétrica.

FREDERICO LUIZ GONÇALVES DE FREITAS

Florianópolis, Setembro de 2002.

SISTEMAS MULTIAGENTES COGNITIVOS PARA RECUPERAÇÃO, CLASSIFICAÇÃO E EXTRAÇÃO INTEGRADAS DE INFORMAÇÃO DA WEB

Frederico Luiz Gonçalves de Freitas

Esta Tese foi julgada adequada para obtenção do Título de Doutor em Engenharia Elétrica, na Área de Concentração de *Sistemas de Informação*, e aprovada em sua forma final pelo Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Santa Catarina.

Guilherme Bittencourt, Dr. Rer. Nat.
Orientador

Edson Roberto de Pieri, Dr.
Coordenador do Programa de Pós-Graduação em Engenharia Elétrica

Banca Examinadora:

Guilherme Bittencourt, Dr. Rer. Nat.
Presidente

Luís Otávio Álvares, Dr.

Carlos Alberto Heuser, Dr. Rer. Nat.

Marcelo Stemmer, Dr. Rer. Nat.

João Bosco Mangueira Sobral, Dr.

Jorge Muniz Barreto, Dr.

"Por que trilhar caminhos já percorridos, se posso criar o meu próprio caminho?"

- Mikhail Tal (1936-1992), mais jovem campeão de xadrez russo (1957) e mundial (1960-1961) de sua época.

Dedico este trabalho a

Meu querido e amado filho ***Hugo****, que ele derrame luz e sorrisos por onde passe, como é de seu feitio,*

Guilherme*, que dosando liberdade, amizade e orientação, ajudou a tornar possível este trabalho,*

Meus pais, ***Luizinho e Darcy****, que deram-me a vida e tudo o mais de bom que lhes estava ao alcance.*

Agradecimentos

Este deveria ser o maior capítulo deste trabalho... A longa lista de amigos a quem devo, de alguma forma, gratidão ainda assim será sairá sem ordem e com omissões... Agradeço a:

. Gilson Chrestani e família, porque é maravilhoso desembarcar em terra estranha ouvindo antes de um amigo: "Aqui, você contará com todo o meu apoio". E de fato fizeram isto.

.os amigos de Recife: Kátia e Pinho, Carol, Zé Adolfo, Lipídeo, Flávio, Alemão e Patrícia, Jarbinhas, Mizinho e Silvana, Poico. Com eles vivi ótimas passagens em Recife.

.Eliane, Rui, Brivaldo, Eduardo "Ravengar" (que previu tudo!), Tito ... Ajudando-me a tirar pedras de minhas asas, prepararam-me para voar...

.meu irmão Riba, Bouwman, Sérgio, Otávio, Robertson, Karline, Gau e a turma do xadrez.

.a turma da UFSC, que tornou meu dia-a-dia mais feliz. Alguns deles:

.Tércio e Rafael, que tomam parte diretamente no trabalho e prometem muito;

.os corações generosos Vila e Lena; os grandes amigos (comparsas?) João, Sérgio, Éder, Sam, Dudu; o refinado Brandão, Antônio, Montez, Sobral, Ota, Passold, Frank...

.os que me ajudaram "no batente", além da amizade: Paulo, Lau, Luciano e Sumar;

.as flores do departamento, Jerusa, Karina, Ana Paula, Karen, Cris...;

.professores: os gente-fina Werner – às vezes co-orientador – e Bosco - que deu-me a mão, numa hora crucial, tornando-se grande amigo depois -, às boas figuras humanas de Joni, Eugênio e Edson; a Jean-Marie, pelo estímulo.

.Margarete Poll, que foi minha melhor amiga aqui,e Cláudia, companheira por longo tempo;

.minha família, que, à distância, torcia por mim: meu impagável tio Cecílio, que acreditou mesmo quando eu próprio hesitava; minhas queridas primas :** Ró, Ju, Beta e Aninha; minha vó Honorina, protetora no céu e na terra; meus irmãos Rodrigo e Marcelo; padrinho

e madrinha, sempre acolhedores; meus tios Fernando, Zé, Jorge, Armando, Carminha e Gracinha; meus primos Moisés, Mago, Claúdio, Cacá, Eduardo, Armando, Sérgio...

.Augustão e Isabel, irmãos de todas as horas em SC e na Alemanha.

.os amigos da Uni-Karlsruhe, na Alemanha: Professor Jacques, que gentilmente nos hospedou; Peter, amigo das horas fáceis e difíceis; Ulla, que se preocupou com meu lar mais como amiga do que como secretária; Clemens, que foi o companheiro diário de papo nos intervalos; passei dois natais com James e espero que isso se repita, no Brasil; Babsi (e Frank), sempre atenciosos; Werner, Andreas, Fridi, Andy, sempre próximos...

.na Alemanha: Hans (e Angelika) e Kathrin foram amigos de profunda afinidade e sinceridade; Manfred parecia meu amigo de muitos anos; Baby já era mesmo; Paulão, cumpadi de PE na Alemanha (e nas cervas!), Awale e Simone, Yves, Sylvia, Raja, Max e Corinna, Ulli (e Inga), Uwe...

.a turma da FNS deu-me muita força: Lenira, Glaine (e Geninho), Érica, Maydinha, Régi, Roxo, Well, Bel, Valéria, Grace, Rose, Lady, Tico, Marcos, Ceça e... esqueci ☺

.o grupo de desenvolvimento do Protégé, especialmente a Ray Fergerson e Natasha Noy, muito solícitos; Henrik Eriksson da Suécia, desenvolveu um *plug-in* fundamental (para o meu trabalho), deu suporte e por isso devo-lhe muito; Thomas Barnekow, recebeu-me em sua casa para tirar dúvidas, e deu-me tranquilidade.

.tantos da UFPE: Geo, Karina, Flávia, Breno, Pac, Evandro, Murilo, Rogério, Ricardo, Ana, Carlos Ferraz, Teresa...

.minha família em S. José: Lourdes e Omir, a mana Ju, Nogueira e Sônia, Paulo Miranda.

.o Prof. Luís Otávio foi um rápido relator e fez comentários importantes na qualificação que provocaram mudanças no trabalho. Os demais membros da banca pela generosidade.

.Finalmente, Daniela presenteou-me carinho e noites divertidas no último ano.

Resumo da Tese apresentada à UFSC como parte dos requisitos necessários para a obtenção do grau de Doutor em Engenharia Elétrica.

SISTEMAS MULTIAGENTES COGNITIVOS PARA RECUPERAÇÃO, CLASSIFICAÇÃO E EXTRAÇÃO INTEGRADAS DE INFORMAÇÃO DA WEB

Frederico Luiz Gonçalves de Freitas

Setembro/2002

Orientador: Guilherme Bittencourt, Dr.Rer.Nat.
Área de Concentração: Sistemas de Informação
Palavras-chave: Inteligência Artificial, Multiagentes, Recuperação de Informação, Extração de Informação, Categorização, Ontologias
Número de Páginas: 205

Existem, na Web, classes de páginas com conteúdo e estrutura similar (por exemplo, páginas de chamadas de trabalhos, referências bibliográficas, etc); algumas delas têm sido tratadas por agentes extratores. Porém, estes sistemas negligenciam o fato de que algumas destas classes inter-relacionam-se, formando grupos (por exemplo, o meio científico). Propomos uma arquitetura de sistemas multiagentes cognitivos para a recuperação, classificação e extração integradas de informação, a partir destes grupos. Para a realização destas tarefas, uma visão da Web que incorpora estas classes (visão por conteúdo) e também a funcionalidade de apresentação das informações mantidas nas páginas (i.e., se a página é uma lista, mensagem, página de uma classe ou nenhuma delas) faz-se necessária. Cada agente processa uma classe, empregando ontologias do domínio e ontologias estratégicas para reconhecer páginas e extrair delas as possíveis informações úteis, comunicando-se e cooperando com os outros agentes. As "dicas quentes" sobre páginas e *links*, trocadas entre os agentes, normalmente contêm menos lixo do que os resultados às consultas dos mecanismos de buscas tradicionais (por exemplo, AltaVista e Excite). A arquitetura de agente apresenta várias formas de reuso: código, esquema da base de dados, conhecimento e serviços dos mecanismos de busca. Resultados promissores da recuperação e classificação funcional e de conteúdo foram obtidos para agentes que processam eventos e artigos científicos, empregando uma ontologia do domínio científico criada especificamente para este fim, sugerindo que a arquitetura é factível. Foram usados a linguagem Java, o motor de inferência Jess e o editor de ontologias Protégé, para a construção dos agentes.

Abstract of Thesis presented to UFSC as a partial fulfillment of the requirements for the degree of Doctor in Electrical Engineering.

COGNITIVE MULTI-AGENT SYSTEMS FOR INTEGRATED INFORMATION RETRIEVAL, CLASSIFICATION AND EXTRACTION OVER THE WEB

Frederico Luiz Gonçalves de Freitas

September/2002

Advisor: Guilherme Bittencourt, Dr.Rer.Nat
Area of Concentration: Information Systems
Keywords: Artificial Intelligence, Multi-agents, Information Retrieval, Information Extration, Categorization, Ontologies
Number of Pages: 205

In the Web, there are classes of pages with similar structuring and contents (e.g., call for papers pages, references, etc); some of them have been processed by extractor agents. However, these systems have neglected the relevant fact that some of the classes are interrelated forming clusters (e.g., Science). We propose an architecture for cognitive multi-agent systems for information retrieval, classification and extraction from these clusters. In order to accomplish these tasks, a Web vision incorporating the classes (vision for contents) and also functionality on the presentation of information kept in the pages (i.e., whether a page is a list, message, belongs to a class or none of above) is required. Each agent processes one class employing domain and strategic ontologies to recognize pages and extract all possible useful information, also communicating and cooperating with the others agents. The "hot hints" on useful links and pages exchanged by them usually contain much less garbage than the results returned by traditional search engines (e.g., AltaVista e Excite). The agent architecture presents many sorts of reuse: all the code, DB definitions, knowledge and services of the search engines. Promising results for retrieval, contents and functional categorization were achieved by agents that process the classes of scientific events and articles, relying on an ontology specially tailored for the scientific domain. The experiments suggest that the architecture is feasible. The Java language, the Jess inference engine, and the Protégé ontology editor were used for agents' construction.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

ÍNDICE DE ABREVIATURAS E SIGLAS E TERMOS EM LATIM

$(KA)^2$	*Knowledge Annotation Initiative of the Knowledge Acquisition Community,* Iniciativa de Anotação de Conhecimento para a Comunidade de Aquisição de Conhecimento
ad hoc	particular
API	*Application Programming Interface,* interface de programação para aplicações
BIG	*Resource-Bounded Information Gathering*, manipulação de informação com recursos limitados
CFP	*Call for Papers,* chamadas de trabalho para eventos científicos
CGI	*Common Gateway Interface,* interface comum de passagem
CLIPS	*"C" Language Integrated Production System*, sistema de produção integrado á linguagem "C"
COOL	"C" *Object Oriented Language,* linguagem orientada a objetos para a linguagem "C"
CORBA	*Common Object Request Broker Architecture*, arquitetura de repassadores de pedidos de objetos comuns
DAML-O	*DARPA Agent Markup Language*, linguagem de anotação de agentes dos projetos avançados de defesa americanos
DCOM	*Distributed Component Object Model*, modelo de objetos-componentes distribuídos
e.g.	*exempli gratia,* por exemplo
FAQ	*Frequently Asked Questions,* arquivo de perguntas freqüentes
FIPA	*Foundation of Intelligent Physical Agents,* fundação para agentes físicos inteligentes
GATE	*General Architecture for Text Extraction*, arquitetura geral para extração de texto

GFP	*Generic Frame Protocol,* protocolo de *frames* genéricas
GPS	*General Problem Solver,* solucionador geral de problemas
HTML	*HyperText Markup Language*, linguagem de anotação para hipertexto
HTTP	*HyperText Transfer Protocol*, protocolo de transferência de hipertexto
i.e.	*id est,* isto é
IAD	Inteligência Artificial Distribuída
IDL	*Interface Definition Language,* linguagem de definição de interfaces
JATLite	*Java Agent Template,* modelo para agentes Java
Jess	*Java Expert System Shell*, ambiente de sistemas especialistas em Java
KIF	*Knowledge Interchange Format,* formato para troca de conhecimento
KQML	*Knowledge Query Manipulation Language,* linguagem de manipulação de consultas sobre conhecimento
KSE	*Knowledge Sharing Effort,* esforço de compartilhamento de conhecimento
LIEP	*Learning Information Extraction Patterns,* aprendizado de padrões para extração de informação
LIFE	*Logics, Inheritance, Functions and Equations*, lógica, herança, funções e equações
LISP	*List Processing*, processamento de listas
LSI	*Latent Semantic Indexing,* indexação semântica latente
ODBC	*Open Database Connectivity,* conectividade aberta para sistemas gerenciadores de bancos de dados
OIL	*Ontology Inference Layer,* camada de inferência para ontologias
OKBC	*Open Knowledge Base Connectivity,* conectividade aberta para bases de conhecimento
ORB	*Object Request Broker*, repassadores de pedidos de objetos
PDA	*Personal Digital Assistant,* assistente digital pessoal
PDF	*Portable Document File,* arquivo de documento portável

PLN	Processamento de Linguagem Natural
POP3	*Post Office Protocol 3*, protocolo de entrega de correio
PPR	agente de artigos científicos (*papers*)
RDF	*Resource Description Framework*, ambiente de descrição de recursos
RDFS	*RDF Schema*, esquema RDF
RDP	Resolução Distribuída de Problemas
RI	Recuperação de informação
RPC	*Remote Procedure Call*, chamada a procedimento remoto
SD	Sistemas Distribuídos
SGBD	Sistema Gerenciador de Bancos de Dados
SHOE	*Simple HTML Extensions*, simples extensões de HTML
SMA	Sistema Multiagente
SQL	*Structured Query Language*, linguagem de consulta estruturada
TF-IDF	*term frequency – inverse document frequency*, frequência de termos - frequência inversa de documentos
UDP	*User Datagram Protocol*, protocolo de datagrama de usuário
URL	*Universal Resource Locator*, loocalizador universal de recursos
VIE	*Vanilla Information Extraction System*, sistema de extração de informação Vanilla
W3C	*World Wide Web Consortium*, consórcio da Web
XML	*eXtensible Markup Language*, linguagem de anotação extensível

Capítulo 1

INTRODUÇÃO

1.1. Problemas de Recuperação de Informação na Internet

A popularização da Internet tem acarretado um grande aumento de recursos disponíveis através da rede, sejam eles serviços ou informações. Isto faz com que a Internet possa ser vista como um grande sistema aberto distribuído. Os sistemas abertos caracterizam-se principalmente pela capacidade de conectar redes a outras redes, tornando documentos, dados e software acessíveis remotamente por pessoas e outros sistemas, agentes e ferramentas. O surgimento dessa tecnologia acarretou uma forte mudança de paradigma na relação homem-máquina, trazida pela popularização da Internet. Os computadores pessoais, que por aproximadamente 25 anos capacitavam-se apenas a tarefas dentro de um ambiente de informações restrito, controlado e estático, transformaram-se em janelas para um mundo continuamente renovável de informações, pessoas e software. Com isso, a antiga metáfora da manipulação direta da informação – que se sustentava devido à pequena quantidade de informação manipulada – começou a declinar [Maes 97].

O resultado disto terminou por cunhar o termo "sobrecarga de informação" (do inglês "*information overload*"): uma enorme quantidade de documentos disponíveis coloca ao usuário a difícil tarefa de separar o joio do trigo na busca de informação útil. Com o intuito de minimizar este problema, os mecanismos de busca – como, por exemplo, o *Excite, Yahoo!, AltaVista* e outros - foram então projetados, com base em técnicas desenvolvidas pela área de Recuperação de Informação (RI) [Baeza-Yates & Ribeiro-Neto 99]. Eles indexam as páginas da Internet por palavras-chave, e usam métodos e estruturas de dados para recuperá-las, rapidamente devolvendo ao usuário uma lista de endereços de páginas que contém as palavras solicitadas, ordenadas por freqüência destas palavras. Dada a variedade de conteúdo da informação disponível, esta era a alternativa viável, uma vez que os dois principais pilares que dão suporte à existência da Internet, o protocolo HTTP (*HyperText Transfer Protocol*) e a linguagem HTML (*HyperText Markup Language*),

foram projetados tendo como principal intuito assegurar a apresentação e a navegação na rede. Preocupações sobre como capturar conhecimento específico, ou seja, realizar buscas semânticas, não foram prioritárias. Devido a este fato, os mecanismos de busca caracterizam-se por uma alta cobertura, porém uma significativa falta de precisão[1], muitas vezes entregando ao usuário uma grande quantidade de endereços de páginas inúteis ou irrelevantes. Utilizando algoritmos matemáticos para atribuir relevância às páginas, estes mecanismos não conseguem dotar de semântica a busca, porque possuem capacidade de representar as páginas com análises baseadas apenas no nível léxico. Alguns dos problemas que decorrem deste fato estão listados abaixo:

- O usuário mediano não conhece as linguagens de consulta dos mecanismos de busca, confundindo-se com problemas triviais como o uso de letras maiúsculas, múltiplas palavras-chave, lógica booleana (o emprego do E e do OU), e a possibilidade de modificação de consultas já efetuadas [Baeza-Yates & Ribeiro-Neto 99].

- Para achar o que procura, o usuário deve escolher as palavras mais apropriadas para encontrar a informação desejada, e, mesmo assim procedendo, não há garantias de que a informação esteja na lista de páginas retornadas pelos mecanismos de busca como resultado de consultas [Leong et al 96].

- O usuário perde tempo e paciência no "trabalho pesado" de colher a informação de que precisa na lista de endereços retornada, muitas vezes tendo que procurar nas páginas apontadas pelas páginas da lista, ou iterativamente ir refinando sua consulta com um conjunto de palavras-chave mais apropriado.

- Se o usuário mediano não conhece a lógica de indexação dos mecanismos de busca, o mesmo não ocorre com os projetistas de páginas e sítios: alguns deles mascaram a relevância das páginas que projetam, incluindo um número alto de repetições de palavras-chave muito comuns, para se colocar artificialmente no topo das listas dos resultados dos mecanismos de busca [Chekuri et al 95]. Esse fato configura o

---

[1] Medidas de performance típicas de RI: *cobertura (recall)* significa o quociente entre o total de documentos relevantes recuperados sobre o total de documentos relevantes, enquanto *precisão* significa o quociente entre o total de documentos relevantes recuperados sobre o total de documentos recuperados.

"problema de persuasão dos mecanismos de busca" [Baeza-Yates & Ribeiro-Neto 99] ou "*Web spamming*".

- Diante de uma lista de resultados muitas vezes bastante heterogênea, o usuário se vê tentado a dispersar-se, sendo vítima do chamado "fenômeno do museu de arte" [Chen et al 96][2].

- Os mecanismos de busca não conseguem resolver alguns problemas ligados à semântica inerentes aos idiomas, especialmente a polissemia (uma palavra com vários significados).

- A maneira com que os mecanismos de busca adicionam sinônimos como palavras-chave associadas ao termo buscado, apesar de melhorar a precisão, não proporciona uma real busca *por conteúdo*, semanticamente definida, no sentido que o usuário deseja [Dunkel et al 96].

- A interface genérica e simples oferecida pelos mecanismos de busca muitas vezes não permite que os usuários recuperem a informação procurada [Steele 2001] com a granularidade que desejam.

Basicamente, duas características da Internet dificultam o acesso à informação útil, específica e relevante: o volume e a falta de estrutura (e, conseqüente, falta de semântica) das informações. Por isso, torna-se difícil *agregar valor* à informação disponível, ou seja, transformá-la em informação útil e facilmente acessível, convertendo informação desestruturada ou semi-estruturada em estruturada, e permitindo ainda processos de inferência sobre a informação capturada.

A próxima geração da Web, a chamada *rede semântica* (*Semantic Web,* vide subseção 4.6.2.), visa justamente preencher a lacuna semântica deixada por HTML, de forma a prover às páginas mecanismos para definir conceitos, atributos, relações, e outras facilidades. Entretanto, ainda estão sendo estabelecidas ferramentas-padrão para definição semântica das páginas, como o RDF (*Resource Description Framework*, ou ambiente de descrição de recursos), a XML (*eXtensible Markup Language*, uma HTML extensível), e a

---

[2] O "surfar" também pode ser visto como uma excelente característica, intrínseca à Web, já que o usuário pode deparar com, ou despertar para informação útil, sintaticamente próxima ao tópico de sua busca.

padronização de conceitos para serem instanciados pelas páginas está ainda sendo discutida. Por isso, o uso destas linguagens ainda não se popularizou em páginas da Internet.

Levando-se em conta que uma representação semântica factível da rede ainda está em andamento e não atingirá todas as páginas da Web - pois isso implicaria resolver os problemas de Raciocínio de Senso Comum e de Processamento de Linguagem Natural (PLN), dois dos maiores desafios ainda não resolvidos da área de Inteligência Artificial - a presença de *contexto* se faz necessária na indexação. A negligência deste fato já produziu conseqüências nefastas, como a desastrosa tentativa do conselho de pesquisa americano em traduzir russo automaticamente nos anos 60, sem levar em conta o contexto, que provocou corte em fundos de pesquisa para PLN [Russel & Norvig 95].

Contexto pode ser definido como o conjunto de fatores relacionados a um texto que o faça ser compreendido adequadamente [Akman & Surav 96], incluindo fatos considerados verdadeiros e formas de inferência aplicáveis [Bouquet 97]. Visto sob a ótica de RI, contexto não inclui um significado semântico das páginas, mas uma visão mais ligada à categorização delas em coleções ou agrupamentos, em termos de similaridade de palavras-chaves, frases, metadados (como autor, data, tamanho, etc) e estrutura de ponteiros[3] (vide capítulo 4 para maiores detalhes). Para a área, contextos assim definidos desempenham um importante papel na construção de interfaces de visualização das páginas, que, em resposta a consultas solicitadas por usuários, mostram graficamente a freqüência de cada palavra-chave das páginas da lista retornada, ou o relacionamento dessas páginas com outras páginas ou com conjuntos de páginas, em função das palavras-chave que elas contêm. Porém, o processo de indexação perde muita informação contextual essencial à compreensão das páginas [Baeza-Yates & Ribeiro-Neto 99].

Contexto para a Web poderia ainda ser definido de uma forma muito próxima a uma rede semântica, como o conjunto de entidades, e seus atributos, relações e restrições presentes numa página. Todavia, esta definição de contextos também não poderia abarcar toda a

---

[3] Usaremos indistantemente ao longo deste trabalho as palavras âncora, *link, hyperlink* e ponteiro para denotar um apontador dentro de uma página da Web que contém o endereço de outra página.

rede: a Web reúne fatos e dados sobre assuntos cotidianos e científicos em contextos muito diferentes.

1.2. A Necessidade de Restrição de Domínios

Na realidade, talvez o foco do problema não esteja nas técnicas empregadas, e sim no escopo: será que é exeqüível de algum conjunto de ferramentas a capacidade de "compreensão" mínima de *todas* as páginas da Web, mesmo que o objetivo seja apenas recuperação?

Uma situação semelhante já preocupou cientistas de Inteligência Artificial nos anos 70, assim que surgiram os primeiros sistemas de representação de conhecimento. Após a demonstração de que estes sistemas poderiam conter conhecimento aplicável, junto com seus respectivos mecanismos de manipulação e inferência sobre esse conhecimento, acreditou-se que fazer sistemas baseados em dedução de grande porte e, inclusive, de Senso Comum, seria uma simples questão de formalizar mais conhecimento, o que redundou em rotundos fracassos. Gerou-se uma expectativa muito grande, especialmente com o projeto GPS (*General Problem Solver,* ou, em português, Solucionador Geral de Problemas) [Russel & Norvig 95], que se propunha a ser um "acumulador" de conhecimentos que poderiam ser utilizados durante processos de inferência na resolução de problemas variados. Achava-se que tudo o que pudesse ser logicamente definido poderia ser deduzido por um motor de inferência.

À época destes primeiros experimentos, os estudos sobre complexidade de problemas, e especialmente a classificação desses problemas em NP-completo, NP-difícil e outras classes, ainda não estava formalizada, e sistemas com algumas dúzias de fatos simplesmente não conseguiam alcançar seus objetivos, por, entre outros motivos, subestimar a explosão combinatorial causada pelo crescimento do número de fatos. A frustração das expectativas contribuiu inclusive para formar uma imagem negativa da área como um todo, que só veio a ser retratada com a correta aplicação das lições aprendidas com estes reveses. Após isto, os sistemas baseados em conhecimento passaram a ser desenvolvidos:

- Reduzindo o domínio de conhecimento para que o número de regras pudesse ser tratável e,

- Codificando cuidadosamente os aspectos mais importantes do domínio com o emprego de heurísticas para diminuir e acelerar a busca.

Postos estes limites, percebeu-se que eram de grande valia os sistemas com inferências sobre *domínios delimitados* para resolver problemas complexos, de difícil tratamento por sistemas convencionais, de onde se originaram os *sistemas especialistas*.

Naturalmente a idéia de domínios restritos aplica-se também aos sistemas que se propõem a tratar a gama de informações contidas na Web, pelo simples fato da rede ter sido originada a partir do senso comum. Nenhum sistema solitário conseguirá atender genericamente às necessidades de informação dos usuários, enquanto, trabalhando-se dentro de um domínio restrito, por exemplo, a *polissemia*, ou seja, uma palavra com vários significados, pode ser tratada com maior precisão.

Mesmo os pesquisadores de RI possuem intuição da necessidade de restrição de domínio, sem, no entanto, o registrarem explicitamente: muitos testes dessa área de pesquisa são realizados sobre *corpi* homogêneos, cujos textos tratam sobre um mesmo assunto, e procedem, muitas vezes, de uma mesma fonte, e não sobre conjuntos de textos tão variados em conteúdo e estilo como os disponíveis na Web. Além do mais, recentemente uma nova tendência em mecanismos de busca começa a se difundir, os *mecanismos de busca especializados* [Steele 2001], que atuam sobre áreas específicas, como notícias (*Newstracker* [Newstracker 2002] e *Moreover* [Moreover 2002]) e páginas pessoais (*HPSearch* [HPSearch 2002]), ou sobre sítios específicos, como o brasileiro *Miner* [Miner 2002], que fornece preços e disponibilidade de livros e CDs de diversas lojas, entre outros serviços.

1.3. Extração de Informações da Internet

À parte da diversidade encontrada na Web, há porções ou regiões dela mais tratáveis e menos gerais, que podem ser investigadas como um *corpus* de relativa homogeneidade, com o intuito de buscar dados específicos e objetivos. Para reforçar essa idéia, ressalte-se, ainda, o fato de que uma boa quantidade de usuários acessa a Internet com objetivos claros e específicos, mais interessados em informações relevantes, úteis, focalizadas e agregáveis do que nas páginas em que estão hospedadas. Além do mais, páginas que tratam de um

mesmo tópico (cinema, classificados, e tantas outras) costumam apresentar regularidade de formatação, estrutura e principalmente de conteúdo.

Partindo também deste conjunto de hipóteses, surgiu uma nova sub-área de Recuperação de Informações conhecida como *Extração de Informações (EI)*, cujo tema principal é a reorganização e reuso de regiões da Web [Atzeni et al 97] em bancos de dados. A construção de extratores de informação da Internet oferece vantagens: o usuário é mais bem atendido, livrando-se de processar manualmente as páginas atrás de dados, e, por isso, a rede fica com menor tráfego já que muitos ponteiros inúteis não serão listados nem carregados. Hoje, já existem várias aplicações com o objetivo de construir bancos de dados a partir de páginas bem estruturadas na Internet. Bancos de dados, diferentemente da Web, podem ser facilmente consultados, provendo ao usuário consultas semanticamente claras e precisas sobre entidades e relacionamentos entre elas, inclusive combinando e totalizando dados, tarefas que os mecanismos de busca, mesmo os especializados, não conseguem realizar.

Conseqüências interessantes advêm da existência de extratores. Em primeiro lugar, eles trazem uma noção de memória à Internet. Os mecanismos de busca atuais repetem continuamente o cômputo de relevância das páginas relativas a determinadas consultas, e não há forma de aproveitar trabalho alheio, ou seja, cômputos de consultas anteriores, e nem buscas já efetuadas por outros usuários. Além do mais, os extratores servem a dois benefícios contextuais [Akman & Surav 96]: a *implicação contextual*, em que uma nova assertiva pode ser usada junto com o conhecimento existente para gerar novas assertivas; e a *contradição* ou *eliminação*, em que uma nova assertiva pode modificar ou eliminar algumas das assertivas existentes.

1.3.1. Problema: Ausência de Extração Integrada

Contudo, os atuais sistemas de extração atuam sobre páginas de domínios muito restritos, na realidade constituindo sistemas *ad hoc*, já que visam o processamento de uma classe muito específica de páginas, como notícias sobre terrorismo do *Wall Street Journal*, obituários, classificados e outros.

Este trabalho defende a idéia de que esta não é uma abordagem apropriada para a Web, devido a várias razões. A principal delas refere-se à existência das âncoras; elas contêm

elementos indicativos muito importantes, claros e seguros de como as informações da Web estão semanticamente conectadas, e não devem ser ignoradas. Esta afirmação também traz à tona interessantes questões sobre uma possível proliferação de extratores: Como integrar suas bases de dados, permitindo aos usuários combinar suas informações? Como aproveitar as informações contidas nas âncoras com o propósito de executar extração integrada, aproveitando o conteúdo das âncoras? Como devem ser abordadas e vistas as regiões, os domínios e as classes de páginas da Web com esse objetivo?

Uma importante e negligenciada característica a ser explorada das classes de páginas processadas pelos extratores é que elas se inter-relacionam com outras classes, formando conjuntos ou *grupos* (*clusters*). Inclusive há informações pertinentes a um conjunto de dados processados por um extrator, que podem ser encontradas em páginas processadas por outro. Isto se evidencia, por exemplo, em páginas de pesquisadores. Nem sempre consta em que eventos científicos eles tomaram parte de comitês de programa, entre outras informações ausentes. Assim, um extrator pode ser mais útil se cooperar com outros extratores dentro de um modelo de domínio adequado - não se resumindo a processar informações apenas de uma classe restrita, como, por exemplo, artigos científicos - montando uma base de dados integrada sobre esse domínio a partir das informações coletadas pelos extratores.

Um contra-exemplo que ratifica o exposto acima, constituindo um caso crítico de falta de cooperação entre extratores, são os sistemas CiteSeer [Bollacker et al 98], de artigos científicos, e o DEADLINER [Kruger et al 2000] para chamadas para eventos científicos. Embora desenvolvidos pelo mesmo grupo de trabalho, e contendo páginas com *links* de uma classe para a outra, os sistemas não cooperam entre si, principalmente devido à falta de um modelo do domínio científico que representasse as ligações entre estas classes, servindo também como vocabulário de comunicação entre os sistemas.

Em outras palavras, o obstáculo encoberto que impede extratores de cooperarem uns com os outros reside na representação de conhecimento que se popularizou na área como um todo, com raras exceções, como o sistema Alembic [Villain 99]. O uso de métodos simples e rápidos como autômatos finitos e gramáticas, entre outros, priorizam a velocidade de processamento e desenvolvimento e a fácil adaptatividade, aplicando técnicas de aprendizado automático, em detrimento das possibilidades de uso do conhecimento

especificado, como os processos de inferência [Villain 99] e, por conseguinte, cooperação. Portanto, para a manipulação integrada de informação, a representação declarativa de fatos de um texto, mais de que a simples ocorrência de um padrão, revela-se de fundamental importância para ambos os processos.

1.3.2. Recuperação, Categorização e Extração como Tarefas Complementares

Entre as contribuições deste trabalho encontra-se a idéia de que recuperação, categorização e extração sobre páginas da Web constituem tarefas complementares, já que elas podem ajudar-se mutuamente, conforme explicitado na figura 1.

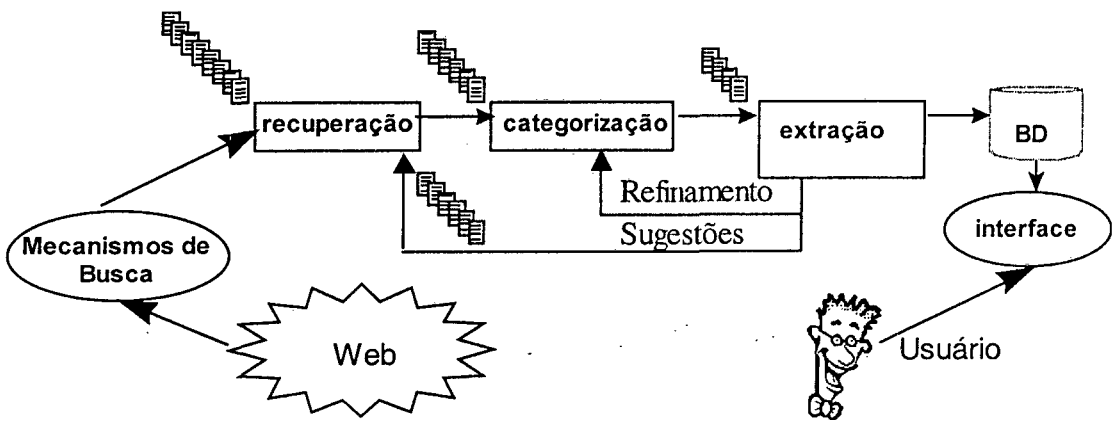

Figura 1. Esboço da arquitetura de um sistema de extração, evidenciando a complementaridade entre as tarefas de recuperação, categorização e extração.

Sistemas de recuperação podem fornecer o acesso a um conjunto inicial de páginas, que possui alta cobertura e baixa precisão. Após isto, sistemas de categorização deveriam selecionar quais páginas pertenceriam às classes a serem processadas[4], e então extratores poderiam capturar a informação requerida. Durante o processo de extração, poderiam ainda ser encontrados, dentre as âncoras das páginas processadas, endereços de outras páginas que pertenceriam a outras classes também processadas. Estas sugestões são encontradas dentro de um contexto muito mais semântico e seguro, e, assim, o processo de extração

[4] A tarefa de *filtragem* classifica páginas em dois grupos, podendo, portanto, ser considerada como categorização.

estaria auxiliando a recuperação. A extração poderia ainda refinar a categorização, uma vez que páginas que não contivessem os dados que caracterizam a classe processada seriam desprezadas.

1.4. Sistemas Multiagentes para Recuperação, Classificação e Extração Integrada de Informações da Web

A falta de mecanismos capazes de captar a semântica do conteúdo das páginas da Web criou uma forte demanda de serviços que se ajusta adequadamente à classe de serviços estudada em Inteligência Artificial, e, mais especificamente, aos agentes inteligentes. Os agentes podem caracterizar-se por *adaptatividade* ou *aprendizado*, apresentando mais robustez às diversas formas como a informação se encontra estruturada.

Por sua vez, a área de Inteligência Artificial já se ressentia dos custosos tempos de resposta que os agentes ou sistemas especialistas consumiam por não se beneficiarem da distribuição, como também da dificuldade de especificar conhecimento de áreas distintas num só agente. Minorando estes problemas, a *Inteligência Artificial Distribuída* ou *Sistemas Multiagentes* [Alvares & Sichman 97] consolidou-se como uma sub-área com vida própria, cujos objetos de estudo concentram–se sobre comunicação, cooperação e coordenação de agentes em ambientes distribuídos ou não, com objetivos comuns ou não. Com efeito, já existe tecnologia disponível para a troca de *conhecimento declarativo* entre agentes, permitindo *cooperação* entre eles na execução de suas metas comuns ou individuais, alcançando, por conseguinte, distribuição e concorrência.

1.5. Contribuições

O presente trabalho visa prover aos usuários informações estruturadas armazenadas em bases de dados, extraídas e classificadas por agentes cognitivos com capacidade de inferência e cooperação, que efetuam busca, classificação e extração de informação a partir de páginas pertencentes a estas classes de páginas, que possuem fortes conexões. Os usuários, humanos ou agentes externos ao sistema, poderão, a partir de um agente mediador, executar consultas estruturadas envolvendo várias classes de páginas, não apenas uma, como os extratores atuais.

É apresentada uma arquitetura para sistemas multiagentes concebidos para a recuperação e extração de informações de conjuntos de classes de páginas da Web. O sistema consiste de uma sociedade de agentes cognitivos, onde cada agente possui conhecimento suficiente sobre uma determinada classe de páginas. O conhecimento encontra-se estruturado em bases de conhecimento que representam os conceitos do domínio e suas relações (ontologias, vide capítulo 4), que permitem que os agentes, através de inferências, identifiquem dentre as páginas recuperadas, quais pertencem à sua classe – portanto realizando também tarefas de classificação. Esta classificação permite também que o maior número possível de informações relevantes seja extraída destas páginas, inclusive o relacionamento dela com outras páginas, que ajudará a uma recuperação mais segura e semanticamente contextualizada. As páginas são recuperadas através da conexão a vários mecanismos de buscas (atualmente *Altavista, Google* e *NorthernLight*).

Também as contribuições do trabalho inter-relacionam-se. Na realidade, todas elas se originam da forma de abordar a Web, aproveitando o conteúdo das âncoras. Assim, conforme pode ser visto na seção 2.2., a *visão por conteúdo*, explora o relacionamento semântico entre as entidades que representam as classes de páginas da Web, e este relacionamento pode estar refletido nas ligações entre âncoras das páginas destas classes.

A partir da visão por conteúdo é que surgem as primeiras contribuições do trabalho, que ajudam a equacionar uma maneira de conceber sistemas para a Web visando à integração entre as classes de páginas, e, por conseguinte, entre os agentes responsáveis por elas. As seguintes assertivas definem estas contribuições:

- *As tarefas de recuperação, categorização e extração podem e devem ser integradas, o que pode ocasionar uma melhora de performance em todas elas.*
- *A conseqüente formulação do problema da manipulação integrada de informação;*
- *É necessário que o domínio que abarca as classes de páginas inter-relacionadas, esteja representado num formalismo lógico de representação de conhecimento. As definições do domínio servirão como vocabulário de comunicação na cooperação entre agentes de classes de páginas distintas de um mesmo domínio;*

- *Os requisitos de comunicação não apenas em sistemas de manipulação integrada de informação, mas em ambientes abertos distribuídos em geral, pedem um modelo de comunicação com mais habilidades que o modelo cliente-servidor, e que só modelos de comunicação em nível de conhecimento ("*peer-to-peer*"), baseados em ontologias reusáveis como vocabulário da comunicação, são capazes de oferecer* (vide capítulo 3);

Estes fatos, junto com a existência de mecanismo de busca especializados citados na subseção 1.2., nos conduzem a outra abordagem à problemática de manipulação de informação na Internet, que também pode ser considerada uma contribuição:

- *Os mecanismos de busca tradicionais, baseados em técnicas de recuperação de informação e indexação por palavras-chave, podem constituir a base e o suporte para outros mecanismos de busca, aplicativos, agentes e extratores mais refinados, precisos e focados em domínios restritos, baseados em conhecimento explícito reusável e comunicável, e habilitados à cooperação.*

Porém, na hipótese do emprego de mecanismos de busca como componentes do sistema para iniciar a busca, apenas a visão por conteúdo é insuficiente, porque uma quantidade considerável de páginas como listas, mensagens, páginas de outras classes e mesmo páginas que nada têm a ver com a classe processada, também são devolvidas por consultas a mecanismos de busca. A solução é acrescer à visão por conteúdo uma *visão por funcionalidade,* que diz respeito à apresentação das páginas, dividida nas seguintes categorias funcionais: listas, mensagens, páginas-conteúdo, páginas auxiliares, e páginas inúteis. Surge, então, outra contribuição:

- *A formulação de uma visão da Web combinando conteúdo e funcionalidade, que facilita a resolução do problema* (vide capítulo 2);

Seguida por outra, a principal do ponto de vista prático e experimental:

- *A elaboração de uma arquitetura, e sua respectiva realização, num* framework *de sistemas multiagentes cognitivos para a solução do problema de manipulação integrada de informação, aplicável a grupos de classes identificados na Web, com a possibilidade de cooperação entre os agentes componentes e reuso massivo de*

componentes, independente do domínio tratado, conectividade a vários mecanismos de busca, e reuso de conhecimento, para a eventual construção de novos agentes (vide capítulo 5);

- *Dois agentes que buscam, filtram e classificam páginas da Web para as classes de artigos científicos e para a classe de páginas chamadas de trabalho para eventos científicos (os ditos* "Call for papers" – sobre estes dois agentes, vide capítulo 6);

- *A disponibilização pública na Web de uma ontologia relativa ao domínio científico* [Freitas 2001], *que está sendo empregada pelo sistema na manipulação integrada de dados da Web.*

1.6. Organização do Texto

O presente trabalho é assim constituído:

- O Capítulo 2 inicialmente discorre sobre as modelagens da Web, domínios existentes nela, sobre a similaridade estrutural de páginas dentro destes domínios, bem como sobre as entidades e conteúdo usualmente encontrados em determinados tipos de páginas. São caracterizadas, então, dois tipos de visões da Web: *a visão por conteúdo* - que classifica as páginas em classes, de acordo com as entidades ou informações que elas contêm, ressaltando o fato de que estas classes relacionam-se entre si em grupos fechados, chamados *grupos de classes* (*clusters*) –, e a *visão por funcionalidade* – que, objetivando a extração integrada do conteúdo de um grupo, divide as páginas em categorias funcionais de acordo com o papel que elas desempenham na localização e ligação das informações., Ao fim deste capítulo, está colocado que a abordagem ideal para a Web, visando não só a extração integrada de dados, mas um objetivo mais abrangente, a manipulação de informação semanticamente integrada da Web, requer o cruzamento entre estas duas visões. As tarefas de extração, categorização e recuperação são definidas como complementares, aproveitando o conteúdo das âncoras e a discriminância das entidades contidas nas páginas a partir desta visão da Web.

- No Capítulo 3 é justificado o uso de uma abordagem multiagente cognitiva. Os *Sistemas Multiagentes* são introduzidos em seus dois subtipos mais comuns, a Resolução Distribuída de Problemas (RDP) e os Sistemas Multiagentes (SMA), bem

como conceitos e utilidades de agentes para problemas de manipulação de informação na Internet. É explanado o modelo de comunicação que a concretiza, dito *peer-to-peer,* junto com as vantagens desse modelo em relação ao modelo Cliente-Servidor. São apresentadas, então, as duas linguagens de comunicação *peer-to-peer* mais difundidas atualmente, KQML (*Knowledge Query Manipulation Language*) [Finin et al 94] e ARCOL [O'Brien & Nicol 98], que são descritas com brevidade. A partir daí, os benefícios e facilidades do conhecimento explícito, obtidos com a declaratividade, serão enumerados. A adequação da Engenharia do Conhecimento ao problema é discutida, colocando como principal motivação à possibilidade de se especificar conhecimento *a priori* sobre as páginas a serem processadas - como localização, entidades, estrutura das classes e relacionamentos entre os *clusters* - através de conhecimento explícito, compartilhável entre os agentes. Postas estas observações, é enfatizada a adequação de Sistemas Multiagentes ao Problema, por proverem concorrência, distribuição e cooperação na manipulação de dados na Web, através de comunicação e coordenação entre os agentes participantes.

- No Capítulo 4, as ontologias reusáveis e compartilháveis são apresentadas como forma de estruturar e comunicar conhecimento, bem como os princípios que devem ser usados em sua construção, os ganhos com definições ontológicas, e tópicos abertos de pesquisa sobre ontologias. Discorre-se ainda sobre o paradigma lógico e a idéia que sempre o acompanhou: o projeto de *sistemas baseados em conhecimento* e seu histórico, que, mais tarde, evoluiriam para *ontologias reusáveis*. Os formalismos de representação de conhecimento explícito *regras de produção, redes semânticas, frames e lógica de descrição* serão apresentados com brevidade, bem como princípios de boa prática de construção de conhecimento voltado para a inferência que garantam eficiência de processamento. A partir daí são descritas ontologias, seus benefícios e ganhos e a aplicação de técnicas relacionadas a ontologias na próxima geração da Web, a chamada Web Semântica. Ferramentas para o manuseio de ontologias, como o Protégé e a Ontolingua, são explanadas a seguir. Princípios para a construção de ontologias visando o reuso constituem o tópico seguinte, e a ontologia Ciência [Freitas 2001] ilustra estes princípios, e é o ponto de partida para a especificação da arquitetura para o estudo de casos de classes de páginas do domínio científico. Ao fim do capítulo, tópicos abertos de pesquisa em ontologias são mencionados.

- No Capítulo 5, será detalhada a arquitetura para um agente do sistema multiagente proposto. Serão colocadas as hipóteses de pesquisa, examinados os componentes e os conhecimentos necessários aos agentes, as tarefas que têm de desempenhar, nominalmente, a validação de páginas, o pré-processamento delas, a categorização das páginas em grupos funcionais, o reconhecimento das classes a que pertencem dentro da ontologia do domínio e a extração de dados dessas páginas, que pode envolver a tarefa de categorização. Como decisão de projeto visando facilitar a construção de novos agentes para a sociedade, a arquitetura beneficia-se de vários tipos de reuso, que são explicados: o reuso de todo ou da maior parte do código, o reuso de serviços de robôs e mecanismos de busca já existentes, o reuso de esquemas das bases de dados, e, principalmente, o reuso do conhecimento. O capítulo é concluído com explicações sobre o que é mediação, suas funções e componentes, uma vez que, sem um mediador, o acesso às bases de dados contendo a informação integrada extraída tornar-se-ia complexo demais para o usuário mediano.

- No Capítulo 6 são analisados os estudos de casos e os resultados atingidos são apresentados e discutidos. Inicialmente, discorre-se sobre os requisitos para a construção de um sistema que instancie a arquitetura apresentada no capítulo anterior, e as respectivas ferramentas que atendem a estes reuisitos, usadas nos experimentos: a linguagem Java, o motor de inferência JESS (*Java Expert System Shell*), o pacote JATLite, que implementa a linguagem de comunicação de agentes KQML, considerado o mais popular dentre os disponíveis, além do editor de ontologias Protégé, citado no capítulo 4. O Sistema multiagente MASTER-Web é descrito, e o conhecimento dos agentes investigado com maior profundidade; são delineadas as ontologias comuns a todos eles, com exemplos ilustrativos de classes e instâncias, as ontologias reusadas, o funcionamento da herança de regras e conceitos, e as ontologias específicas de um agente. A seguir, estão descritos os passos para a construção de um agente genérico dado que já exista outro agente, tarefa bastante facilitada pelos diversos tipos de reuso citados no capítulo anterior. São relatados os experimentos com o agente "*Call for Papers*" (CFP), que reconhece e extrai dados de chamadas de trabalho para eventos quaisquer, sejam jornais, revistas, *workshops* ou congressos, e com o agente de artigos científicos (PPR), que faz o mesmo com páginas de artigos científicos, teses, dissertações, relatórios técnicos e de projetos. Os promissores resultados nas tarefas de

categorização e classificação estão dispostos, assim como resultados relativos à cooperação entre agentes. Os resultados são avaliados e discutidos pormenorizadamente. Também é explicado o funcionamento do mediador do MASTER-Web.

- No Capítulo 7 são listados os trabalhos relacionados às áreas apresentadas. Em primeiro lugar, são repassados os trabalhos que abordam a complementaridade entre as tarefas de categorização, recuperação e extração. Após isto, são sumarizadas as três abordagens à área de Recuperação de Informações: a abordagem estatística, a de aprendizado automático e a de Processamento de Linguagem Natural (PLN), sendo esta última mais detalhada, por ser também muito empregada em extração. São brevemente citados os trabalhos sobre categorização de textos e sobre Extração de Informação. Nesta última, são explicados os extratores de páginas bem-estruturadas, chamados de *wrappers,* a aplicação de aprendizado automático para a construção deles e de dicionários para extração, bem como as abordagens que utilizam PLN. Abordam-se, então, os dois tipos de extratores que correntemente fazem uso de ontologias, com uma discussão crítica sobre eles. Passa-se em revista, em seguida, sistemas similares ao sistema MASTER-Web, entre eles o CiteSeer e o DEADLINER, citados neste capítulo, traçando um paralelo entre estes sistemas. Ao fim do capítulo, enumeram-se tecnologias de sistemas multiagentes com seus respectivos empregos em problemas de recuperação de informação.

- No Capítulo 8, são levantadas as conclusões e contribuições do trabalho, e projetados os trabalhos futuros.

Capítulo 2

VISÕES DA WEB

2.1. Modelagens da Web

Por sua riqueza, variedade e falta de estrutura de seu conteúdo, um dos problemas que precedem qualquer solução relativa à Web é a forma de abordá-la ou modelá-la. O tema "modelagens da Web" costuma estar presente em muitos congressos sobre a Internet, e constitui tema de intensa pesquisa. Basicamente, existem dois tipos de modelos da Web, baseados em abordagens criadas sob o ponto de vista de Bancos de Dados [Florescu et al 98]:

- *Modelos baseados em grafos*, onde os ponteiros representam os arcos do grafo;
- *Modelos baseados em dados semi-estruturados*, em que partes da Web possuem entidades e atributos, cujo esquema não é completamente conhecido ou obedecido, ou seja, são toleradas páginas que não se adaptam completamente ao esquema.

O primeiro modelo adequa-se melhor a problemas de busca, uma vez que buscas em grafos são problemas relativamente formalizados. O segundo modelo oferece ferramentas mais acuradas de acesso aos atributos, e costuma ser empregado em tarefas como sumarização e extração de dados semi-estruturados. Por estes motivos, a modelagem da Web introduzida neste trabalho enquadra-se como um modelo baseado em dados semi-estruturados.

Os modelos baseados em dados semi-estruturados não tratam toda a Web, mas partes definidas dela, que devem obedecer minimamente a esquemas pré-definidos. A tarefa conhecida como *categorização* preocupa-se com a identificação de partes ou categorias da Web, e é empregada com esse propósito. A categorização pode ser feita de duas formas:

- Sobre categorias previamente definidas ou
- Criando-se categorias de acordo com a semelhança entre as páginas, problema este conhecido como *agrupamento (clustering)* [Sahami et al 97].

Para extração, a categorização é usualmente efetuada sobre categorias pré-definidas.

Nas próximas seções são definidos dois tipos de visões da Web introduzidas neste trabalho, que facilitam a extração integrada: a visão por conteúdo e a visão por funcionalidade. O capítulo é concluído apresentando uma forma de combiná-las de modo a otimizar a recuperação das informações a serem extraídas.

2.2. Visão por conteúdo

A visão por conteúdo tenta caracterizar partes da Web pela semântica dos dados apresentados nas páginas, identificando-os através da presença de elementos pertinentes ao contexto em que se espera que o dado se insira. Dois conjuntos de páginas, que serão descritos nas próximas subseções, relacionam-se à visão por conteúdo: as *categorias*[5] e as *classes de páginas*.

2.2.1. Categorias

Muitos mecanismos de busca adicionalmente oferecem *serviços de diretório* ou *divisões de categorias*, nos quais páginas estão separadas manualmente de acordo com o seu conteúdo. A construção e/ou manutenção destes diretórios constitui uma das principais motivações de pesquisa em classificação automática através de técnicas de aprendizado [Cohen & Singer 96], pois, ocasionalmente, o conteúdo deles torna-se obsoleto e desatualizado, faltando páginas novas e relevantes. Estas categorias freqüentemente estão organizadas em hierarquias, e possuem características bastante peculiares, vistas nas próximas subseções.

2.2.1.1.Similaridade Estrutural

Curiosamente, o primeiro fato observável nestas categorias e ignorado pelos mecanismos de busca é o estilo de composição das páginas, que denota sua estrutura em termos de aparência. Cruz et alii [Cruz et al 97] evidenciam a existência de padrões particulares de editoração das páginas pertencentes às categorias, e que podem ajudar a identificá-las, caso sejam analisados as *tags* de HTML que as constituem. Por exemplo, páginas de educação contêm mais ponteiros que as de outras categorias, páginas de artistas usam tabelas apenas

para embelezar o *lay-out*, e possuem o dobro de imagens de páginas de advogados, que, por sua vez, usam tabelas apenas para apresentar dados, e apresentam uma freqüência significativamente superior de separadores de parágrafos em relação às outras categorias.

2.2.1.2. Conteúdo

Naturalmente, na medida em que se desce na hierarquia, as categorias guardam maior especificidade, e muitas vezes apresentam itens de um mesmo tipo, como, por exemplo, dados sobre tenistas. As páginas classificadas em folhas das árvores de categorias, ou seja, páginas das categorias mais inferiores, trazem dados específicos que podem ser mapeados, como, por exemplo, dados biográficos dos tenistas, resultados de partidas de torneios, entre outras.

2.2.1.3. Entidades

Algumas destas folhas trazem páginas exclusivamente sobre apenas uma *entidade* ou *instância* de uma *classe* - isto é, sobre apenas uma pessoa, organização, empresa, artigo ou outro assunto específico – bem como seus respectivos dados ou atributos. Há ainda categorias dentro dos serviços de diretório que poderiam ser divididas com uma granularidade mais fina, de forma a comportar uma única entidade. Partes da Web podem ser vistas desta maneira, caracterizando a visão por conteúdo.

2.2.2. As Classes de Páginas

As páginas que apresentam entidades compartilham muitas características comuns entre si, tais como estilo de editoração, padrões de conexão a outras páginas, terminologia e, principalmente, o conjunto de atributos. A criação de extratores baseia-se neste fato, na existência de *classes de páginas*, que definem uma visão da Web por conteúdo. Em páginas da classe de pesquisadores, por exemplo, com certeza encontram-se dados como instituições, áreas de interesse, artigos publicados e muitos outros itens. Cada classe de página deve corresponder a um conceito na base de conhecimento estruturada (ontologia) do domínio.

---

[5] O termo *categorias* é empregado doravante neste trabalho com a sua conotação usual, não guardando nenhuma relação com a parte formal de computação, representada pela Teoria das Categorias.

Cabe ressaltar que, apesar das páginas pertencentes a uma classe localizarem-se normalmente dentro de um mesmo domínio, isto não deve ser levado em conta em categorizações, devido às muitas exceções e à existência dos *sites* espelho (*mirror sites*), que replicam as páginas em domínios diferentes, evitando gargalos de acesso a páginas muito populares.

2.2.2.1. Discriminação das Entidades: Categorização e Extração como Tarefas Complementares

A obrigatoriedade de existência de determinados atributos de uma entidade, assim como a obediência deles a determinadas regras de consistência, confere às entidades procuradas a propriedade de *discriminação*: a existência e consistência de seus atributos em páginas são um indicativo fundamental no reconhecimento de páginas consideradas como membros de uma classe. Por exemplo, páginas consideradas como de chamadas de trabalhos para eventos científicos ("*calls for papers*") devem portar pelo menos uma data ou um ponteiro para datas, e, em havendo mais de uma, a distância entre elas deve ser menor que um ano.

Assim, o emprego de regras para atributos propicia um inolvidável benefício, o uso da extração como refinador à tarefa de categorização. Na realidade, as duas tarefas podem ser consideradas como complementares na manipulação de informação da Web, uma vez que a extração depende de uma prévia categorização, ou seja, de um reconhecimento e filtragem de páginas candidatas a membros da classe de páginas que está sendo processada.

2.2.3. Grupos de Classes (Clusters)

Uma das hipóteses deste trabalho é a de que muitos ponteiros das páginas que pertencem às classes conforme definidas acima apontam para outras páginas contendo entidades pertencentes a um número reduzido de outras classes ou contendo informação referente a páginas com essas características. Por exemplo, em páginas de pesquisadores, com certeza serão encontrados ponteiros para páginas de artigos, podem ser localizados ponteiros para chamadas de trabalho de eventos científicos, e páginas de outras classes.

O conceito de "*comunidades da Web*" [Gibson et al 98] reforça essa hipótese. Comunidades são regiões da Web que agrupam um conjunto de *páginas autoritativas*[6] – as primeiras páginas ou as páginas principais de um *site,* por exemplo, *www.harvard.edu* -, sendo este conjunto topologicamente fechado, no sentido em que as páginas autoritativas são o prefixo inicial do endereço de páginas apontadas por muitas outras páginas referentes a determinados assuntos ou tópicos. Isso demonstra que determinados tópicos localizam-se dentro de partes definidas da Web. Analogamente, este trabalho supõe que um conjunto fechado de classes e seus relacionamentos reúnem conhecimento sobre entidades dentro de áreas específicas, por exemplo, o meio científico, turismo, e outras áreas. Neste trabalho, este conjunto de classes é chamado de *grupo de classes.*

As entidades, atributos e relacionamentos de um grupo de classes devem coincidir com o conhecimento usual acerca da área específica a que pertencem e, especialmente, os relacionamentos entre as entidades estão refletidos como ponteiros entre suas páginas na Web. Assim, conhecimento *a priori* sobre a área específica pode ser aplicado para encontrar, extrair e validar dados, disponibilizando as instâncias das entidades extraídas para acesso a usuários ou agentes de software. O exemplo das datas de chamadas de trabalhos, citado acima, ilustra o emprego desse conhecimento. A figura 2 evidencia parte do grupo de classes do meio científico, mostrando os relacionamentos entre elas, ressalvando-se que nem todas as classes do meio científico estão mostradas na figura. As setas indicam relacionamentos, e, setas de duas pontas denotam que páginas de ambas as classes contêm âncoras ligando uma classe à outra.

2.2.3.1. Relacionamento entre Classes: Extração e Recuperação como Tarefas Complementares

O trabalho efetuado pelos extratores atuais, com exceção do WebKb [Craven et al 99], resume-se à captura das entidades, relevando o significativo e indicador conteúdo dos *hyperlinks*, que podem ser analisados dentro do contexto onde se localizam. De fato, não só a extração e recuperação, mas qualquer tipo de manipulação de informação sobre a Internet

---

[6] Para a taxonomia na qual a visão proposta foi inspirada [Pirolli et al 95], *páginas autoritativas* são também citadas como *páginas-cabeça* (*head*).

não deve deixar de levar em conta as ligações contextuais entre as informações presentes nas páginas, apresentadas sob a forma de âncoras.

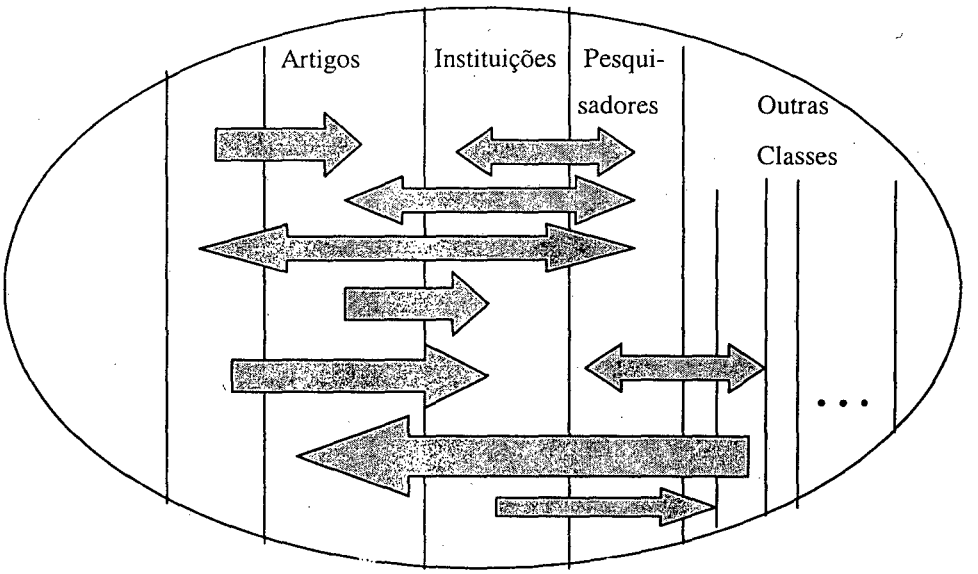

Figura 2. Grupo de classes do meio científico e seus relacionamentos.

Extratores podem fornecer com segurança endereços de páginas encontrados entre os seus ponteiros a outros extratores, se as entidades extraídas por este último tiverem relacionamento com a entidade extraída pelo primeiro. Afinal, estes endereços são obtidos dentro de um contexto confiável e bem mais seguro e com maior probabilidade de ser relevante do que as listas de resultados dos mecanismos de buscas. As listas de resultados devem, portanto, ter menor precedência de processamento de que estas "dicas" sobre endereços de páginas trocadas entre os extratores. O papel dos mecanismos de busca num acoplamento com extratores para a Web pode, dependendo da dificuldade de serem encontrados termos que prometam uma boa cobertura da classe de páginas procurada, apenas resumir-se a disparar a cooperação entre os extratores, fornecendo novas tentativas quando as sugestões de páginas enviadas pelos outros agentes tivessem se esgotado.

Destarte, a complementaridade entre a extração e a recuperação de informações está sedimentada; a extração pode otimizar a recuperação, que por outro lado inicia o processo de extração.

Digna de nota é a utilidade que esta visão que inclui relacionamentos pode trazer em termos de conhecimento da "topologia" ou dos padrões de ligação entre as classes. Às vezes, poder-se-ia supor que duas classes, como chamadas de eventos e artigos científicos,

possuem muitos *links* entre si, porém isto pode não se verificar na prática, conforme apontam os experimentos (vide capítulo 6).

2.3. Visão por funcionalidade: As Categorias Funcionais para Extração Integrada

Uma visão alternativa da Web diz respeito à funcionalidade das páginas, dividindo-as de acordo com o seu papel na ligação entre páginas e na apresentação e armazenamento de dados relevantes. Inspirada no trabalho de Pirolli et alii [Pirolli et al 95], esta visão baseia-se no exame de listas de resultados retornados pelos mecanismos de busca para o processamento de uma classe. Assim, dado este propósito e visando a extração integrada, as categorias funcionais estão assim divididas:

- *Páginas-conteúdo,* que são páginas que pertencem à classe que está sendo processada, e de onde serão extraídas a(s) entidade(s) em questão.

- Páginas *auxiliares,* que, apontadas exclusivamente por páginas-conteúdo, hospedam atributos específicos da(s) entidade(s) da página que a aponta.

- *Listas* de páginas-conteúdo, também chamadas de *diretórios* ou *índices*, de grande utilidade na localização segura e contextualizada de páginas-conteúdo, por ser composta basicamente de âncoras para páginas-conteúdo.

- *Mensagens* ou *Listas de Mensagens*, que contêm correspondências ou listas delas[7] sobre assuntos correlatos à entidade que está sendo extraída, que usualmente não possuem utilidade para extração integrada, por serem páginas muito longas, por apontar para páginas trazendo informações muito disparatadas e com relacionamentos difíceis de serem identificados.

- *Recomendações,* que são páginas-conteúdo que pertencem a outra classe, podendo ser aproveitadas quando do processamento desta outra classe, acelerando o processo de busca desta classe.

---

[7] Conhecidas como "Perguntas Freqüentes ou FAQs – *Frequently Asked Questions* – contendo perguntas e respostas informativas sobre o assunto de que tratam.

- Simplesmente *lixo*, ou seja, páginas sem qualquer utilidade para a extração, por não pertencerem a nenhum dos itens anteriores.

Cabem algumas notas a respeito destas categorias funcionais. Em primeiro lugar, convém salientar que é possível a existência de listas em páginas-conteúdo. Por exemplo, a bibliografia de um artigo científico pode constituir uma lista de *links* para outros artigos. Podem inclusive existir listas em outras páginas de outras classes; chamadas de eventos científicos freqüentemente fornecem uma lista de âncoras de pesquisadores em seus comitês de programa, por exemplo. Há classes de páginas em que cada entidade pode representar uma lista de outra classe. Uma instância de Publicação Divisível, com subclasses como livros e *proceedings,* pode conter uma lista de links para artigos científicos, que, não por acaso, é subclasse de Publicação Parte, dentro de uma taxonomia sobre o meio científico[8].

Observe-se ainda que páginas auxiliares podem ajudar a encontrar a entidade a que se referem, através de um ponteiro para a página autoritativa (como "*Home*") ou do prefixo imediatamente superior de seu endereço eletrônico.

Ressalve-se ainda que o comportamento e utilidade das categorias funcionais, e também o relacionamento entre elas para a extração integrada permanece fixo, conforme definido acima. Assim, durante a extração, a classificação de páginas resultantes de consultas a mecanismos de busca com relação a estas categorias, assim como a identificação de páginas autoritativas[9] com relação à entidade processada, proporciona um refinamento fundamental no reconhecimento de páginas contendo dados realmente pertinentes. As categorias funcionais e seus relacionamentos encontram-se detalhadas na figura 3. A área hachurada indica categorias sem utilidade para o processamento de uma classe, e as setas entre as categorias indicam relacionamentos.

---

[8] A modelagem ontológica de relações Parte-Todo é tema de intensa pesquisa nas áreas de Filosofia e Inteligência Artificial [Artale et al 96] [Staab & Mädche 2000].

[9] Neste contexto, esprestamos o termo "*página autoritativa*" para denotar a página principal de uma entidade qualquer, seja ela uma página de pesquisador ou de chamada de trabalhos, que contenha ponteiros para páginas onde estão os atributos da entidade, diferente do termo usual, que designa exclusivamente a página principal de uma instituição ou empresa, com nome de domínio (por exemplo, *http://www.ufsc.br*).

2.4. Cruzamento entre Conteúdo e Funcionalidade para Extração Integrada

Tanto para identificar precisamente as páginas que abrigam as entidades das classes processadas, como para localizá-las rapidamente, beneficiando-se dos relacionamentos entre elas refletidos nas âncoras, as duas visões devem ser usadas simultaneamente. A visão por conteúdo responsabiliza-se por trazer da Web páginas potencialmente pertencentes às classes processadas, garantindo cobertura sobre a Web, enquanto a visão por funcionalidade encarrega-se de selecionar com rigor as páginas que contêm as entidades (páginas-conteúdo), preocupando-se também em extrair o máximo de informações contextuais relevantes que ajudem a otimizar a busca de mais entidades não só da classe processada como de outras classes do grupo – os relacionamentos entre as páginas -, o que garante uma alta precisão.

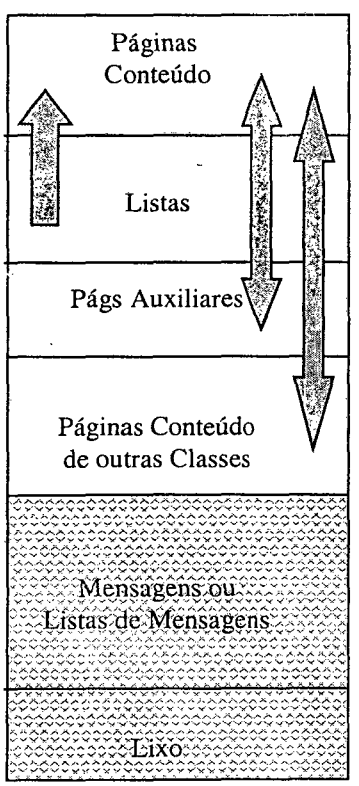

Figura 3. As categorias funcionais e seus relacionamentos.

Como será visto no capítulo 7, a combinação destas duas visões propicia uma melhora sensível na busca de páginas úteis em determinadas classes, como a de "*Call for Papers*",

em que a categoria funcional de listas acarretou numa performance otimizada em torno de 20%.

Capítulo 3

A Abordagem Multiagente Cognitiva

3.1. Agentes

A noção de agente inteligente (ou racional) há muito tempo tem servido à comunidade de IA como paradigma na construção de sistemas inteligentes [Minsky 1986]. Devido a seu alto grau de abstração, a noção de agente permite que se "descreva" um sistema em termos de ambiente, ações, dados perceptivos e objetivos, evitando considerações prematuras sobre a maneira como ele foi ou será implementado (linguagens de representação do conhecimento, mecanismos de inferência, etc). Atualmente, a tecnologia de agentes inteligentes vem extrapolando as fronteiras da Inteligência Artificial, atraindo interesse de vários ramos da Ciência da Computação, e ganhando cada vez mais notoriedade na resolução de uma vasta gama de problemas complexos, e, em especial, daqueles relacionados à Internet.

Devido a esta proliferação de tipos e áreas de aplicação, há atualmente uma certa controvérsia reinante sobre o que um agente realmente é [Franklin & Graesser 96]. Entretanto, existem algumas características consensuais. Os agentes inteligentes devem tentar atender a uma ou mais metas de forma autônoma, decidindo que ações devem tomar pelas percepções do ambiente, provocando mudanças nesse ambiente. Devem ainda apresentar robustez frente a situações não previstas de antemão (o que confronta diretamente com a forma tradicional de programação), muitas vezes possuindo uma representação dentro de si do mundo em que se encontra imerso.

Diferentemente dos sistemas tradicionais de Inteligência Artificial, que possuíam competências especializadas [Maes 94], e justificadas pela necessidade de robustez, os agentes inteligentes devem apresentar múltiplas competências integradas, especializadas ou não, que lhes garantam versatilidade. Tanto quanto possível, devem tentar aprender e melhorar sua performance através de uma avaliação de suas ações. Se o ambiente assim o

permitir, mobilidade é uma qualidade desejável, e alguns deles chegam a exibir personalidade, os chamados *agentes credíveis* (*believable agents*) [Elliott 93].
Existem três tipos de ambiente nos quais os agentes inteligentes podem atuar: o mundo físico, o mundo computacional – a Internet ou outros ambientes computacionais menores – e o mundo da realidade virtual, uma simulação de um mundo qualquer com características definidas a que o agente deve se adequar.

Reciprocamente à utilidade dos agentes na Internet, a própria Internet, enquanto mundo computacional, tornou-se um ambiente bastante propício aos agentes inteligentes, por tratar-se de um ambiente de menor complexidade que o mundo físico, mais acessível e fácil de controlar. Em robôs - agentes imersos no mundo físico - a problemática de lidar com o ambiente releva-se de complexidade muito superior, pois o robô defronta-se com situações inesperadas com muito mais freqüência.

3.2. Inteligência Artificial Distribuída (IAD)

Na medida em que mais agentes começaram a surgir, e tarefas mais complexas lhes eram ministradas, as qualidades de sociabilidade e comunicabilidade fizeram-se relevantes, inaugurando a *Inteligência Artificial Distribuída*, baseada em sociedades de agentes.

A Inteligência Artificial clássica baseava-se numa *metáfora psicológica* da inteligência, onde uma pessoa ou entidade resolvia os problemas propostos, e a inteligência era vista como *atomizada,* pois se restringia aos *micro*-aspectos de sua própria racionalidade [Wooldridge & Jennings 95]. Inspirada em áreas tão diversas como lingüística, sociologia, economia, filosofia e biologia, a Inteligência Artificial Distribuída complementa a metáfora psicológica com uma metáfora *sociológica,* concernente aos *macro* aspectos dos agentes enquanto *sociedade.* As soluções dos problemas propostos emergem de ações e interações produtivas entre os agentes membros de uma mesma sociedade. Uma das mais fortes inspirações dos sistemas multiagentes partiu de uma modelagem da própria mente humana. O livro "Sociedade da Mente" [Minsky 86] delineia mecanismos do cérebro e do pensamento. Segundo a obra, existe um grande número de agências por mente, muito variada e rica em termos de tipos de conhecimento que armazenam, metas, representações e estímulos a que estão sujeitas. As ações e soluções para a tomada de decisões emergiriam

das interações entre estas agências, através de conflitos, comunicação, hierarquias e outros mecanismos bastante similares aos sistemas multiagentes de hoje.

3.2.1. Tipos

Existem basicamente dois tipos de sistemas multiagentes, com relação à divisão de subtarefas na resolução de problemas: a Resolução Distribuída de Problemas e os Sistemas Multiagentes propriamente ditos.

3.2.1.1. Resolução Distribuída de Problemas

A Resolução Distribuída de Problemas (RDP) é aplicável quando se enxerga claramente uma divisão de tarefas entre os agentes, que interagem entre si, ajudando-se uns aos outros. O problema é dividido em subproblemas, e a partir daí são definidos detalhadamente os agentes, onde cada um deles encarregar-se-á de um ou mais destes subproblemas, como pode constatar-se observando a figura 4. Eles cooperam trocando mensagens entre si, de forma a melhorar a performance individual e coletiva e a sincronização. Normalmente, a RDP é usada para explorar o paralelismo entre agentes localizados em máquinas distribuídas que possuam interação, ou para fazer com que agentes pré-existentes possam cooperar, melhorando a performance de todos [Chaib-draa 94].

3.2.1.2. Sistemas Multiagentes

Já os Sistemas Multiagentes propriamente ditos possuem maior complexidade, uma vez que empregam a alocação dinâmica de tarefas, ou seja, não existe uma definição de antemão sobre quais tarefas serão executadas por qual agente. A maior dificuldade neste modelo consiste em produzir interações produtivas entre os agentes, com meios que garantam a *convergência* de suas ações rumo às metas. Assim, são projetados os agentes e suas formas de interação e organização, no intuito de que venham a atingir essa convergência, conforme representado na figura 5.
Os sistemas multiagentes costumam ser classificados também quanto à forma em que o conhecimento é representado nos agentes, em sistemas multiagentes cognitivos e reativos.

Os *Sistemas Multiagentes Cognitivos* são inspirados nas sociedades humanas. Os agentes possuem conhecimento explícito, codificado em um formalismo lógico de representação de conhecimento, e mecanismos de comunicação direta, baseado na Teoria de Atos de Fala

(vide próxima subseção). As ações dos agentes advêm de inferências, a partir das percepções do ambiente e das mensagens de outros agentes. Quando não se encontram organizados em RDP, as principais dificuldades destes sistemas são conseguir coordenação entre as ações dos agentes cognitivos de forma a atingir as metas, e meios para fazer com que agentes com interesses semelhantes se encontrem (*matchmaking*) num ambiente aberto. Caracterizam-se ainda por possuírem, em geral, um baixo número de agentes, da ordem de algumas dezenas [Bittencourt 98].

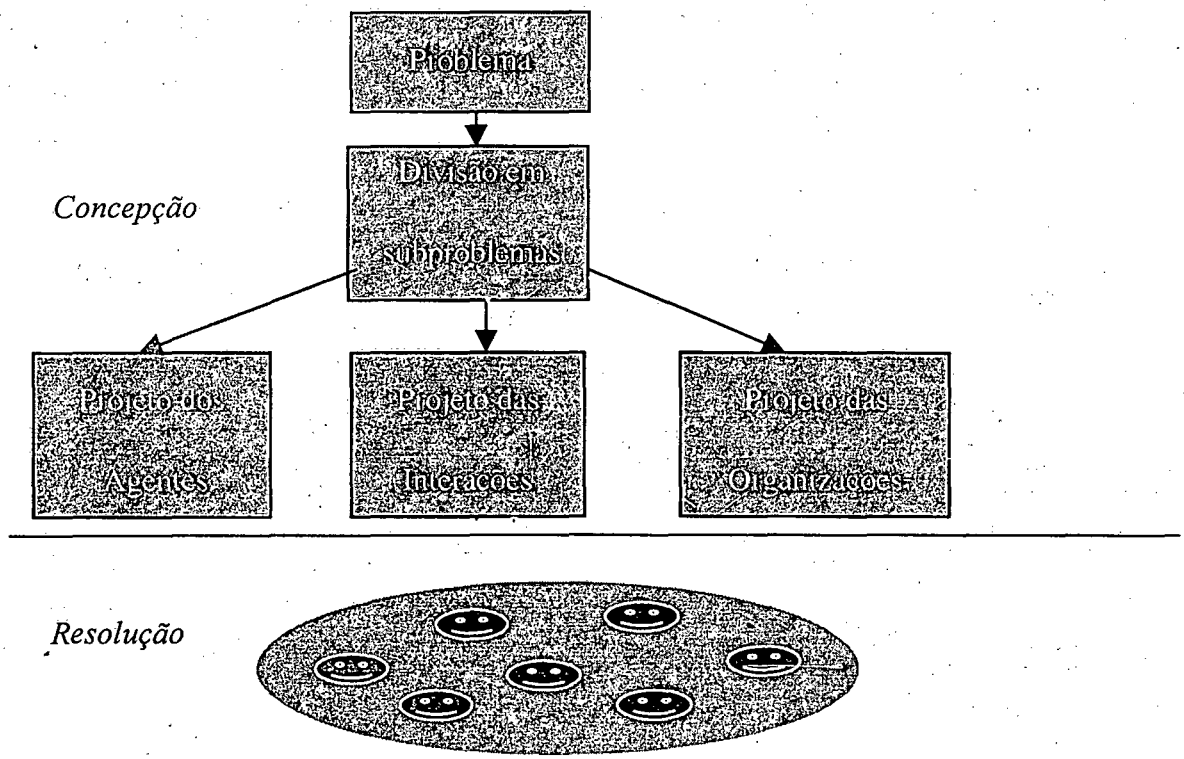

Figura 4. Resolução Distribuída de Problemas. [Alvares & Sichman 97].

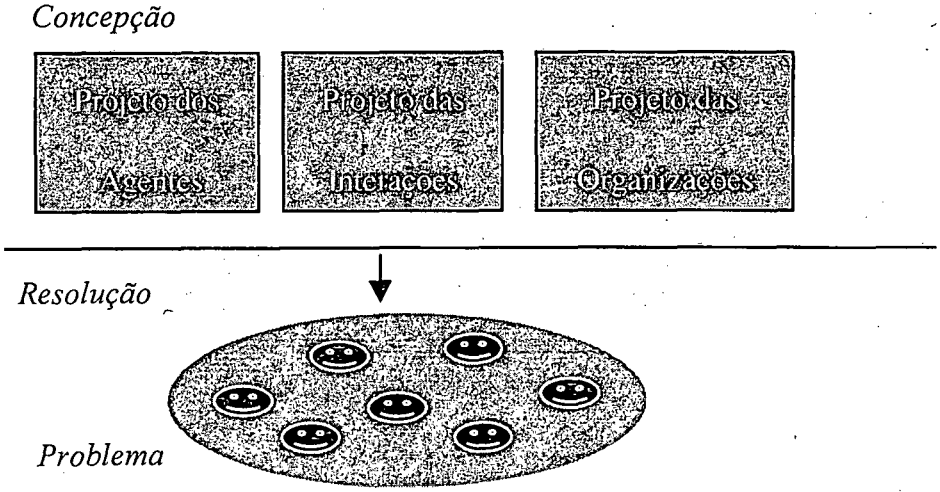

Figura 5. Sistemas Multiagentes. [Alvares & Sichman 97].

3.2.2. Comunicação entre Agentes Cognitivos[10]

A comunicação direta entre agentes abrange dois modelos: O modelo Cliente-Servidor e o modelo *peer-to-peer*.

O modelo Cliente-Servidor baseia-se em chamadas a procedimentos remotos, enquanto internamente efetua uma comunicação do tipo pedido-resposta com os parâmetros do procedimento solicitado.

Já o modelo *peer-to-peer* baseia-se na *Teoria dos Atos de Fala* [Austin 62]. A comunicação humana tem sido modelada por esta teoria, que considera que a linguagem falada tem por objetivo engendrar ações e provocar mudanças no ambiente. Os Atos de Fala, anteriormente estudados em Processamento de Linguagem Natural, uma sub-área de Inteligência Artificial Simbólica, são classificados como *assertivos* (informar), *diretivos* (pedir ou consultar), *comissivos* (prometer ou comprometer-se), *proibitivos, declarativos* (causar eventos para o próprio comunicador) e *expressivos* (emoções). A comunicação entre agentes dotados de autonomia e inferência tem seguido este modelo, que sob o ponto de vista de semântica é muito mais claro, portável, e abrangente do que o modelo Cliente-Servidor. O modelo *peer-to-peer* propõe uma comunicação proativa – ou seja, qualquer agente pode iniciá-la –, baseada em atos de fala (que expressam as intenções dos agentes) e com conteúdo preferencialmente baseado em conhecimento declarativo, sendo por isso chamada de *comunicação em nível de conhecimento*.

As condições para que conhecimento possa ser trocado e compreendido pelos agentes em comunicação em nível de conhecimento são:

- A intenção *pragmática* de cada mensagem, definida pelo ato de fala correspondente (como informar, pedir, recrutar, etc), deve fazer parte do protocolo de comunicação;
- O conhecimento contido na mensagem deve estar escrito em algum formalismo de representação de conhecimento que garanta que a *sintaxe* da mesma possa ser entendida por ambos os agentes, com a possibilidade de emprego de mecanismos de tradução entre estes formalismos;

---

[10] Toda esta subseção baseia-se em artigo publicado pelo autor [Freitas & Bittencourt 2002].

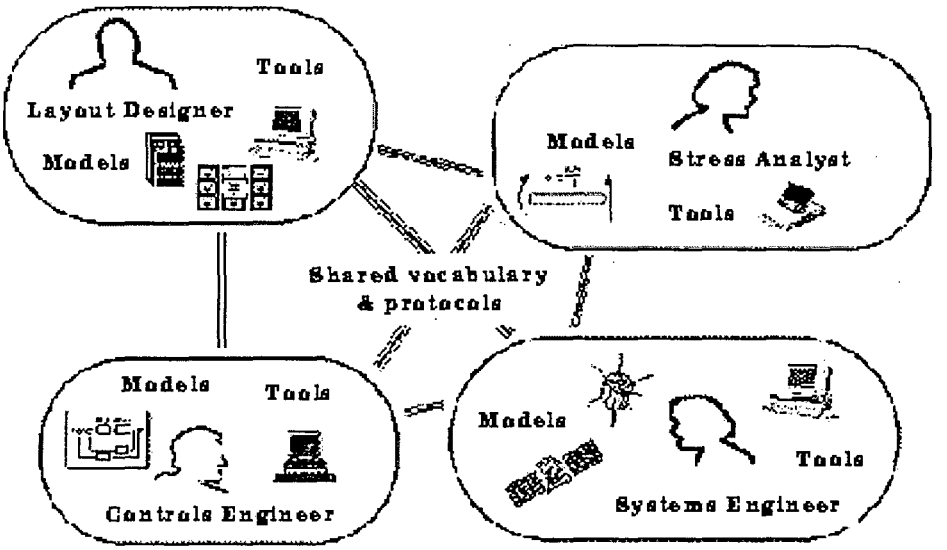

Figura 6. Comunicação em nível de conhecimento, através de protocolos e vocabulário comum, apesar de cada componente ter seu próprio agente ou sistema especialista [Fikes 98].

As mensagens devem referir-se a um contexto e vocabulário comuns, sobre o qual a troca de mensagens possa ser efetuada dentro de uma *semântica* bem definida e sem ambigüidades. Esse contexto é normalmente fornecido pelo que chamamos de *ontologia compartilhada ou reusável* [Gruber 95] (vide o capítulo seguinte). A figura 6 mostra agentes em comunicação em nível de conhecimento. Assim, já existe tecnologia disponível para a troca de *conhecimento declarativo* entre agentes, permitindo cooperação entre eles na execução de suas metas comuns ou individuais, alcançando, por conseguinte, distribuição e concorrência. A necessidade de comunicação em nível de conhecimento é tão premente na resolução de problemas complexos na Internet, que traz à tona a antiga controvérsia declarativo-procedimental [Winograd 75] (vide capítulo seguinte), dados os imensos benefícios produzidos pela comunicação em nível de conhecimento (vide próxima subseção). Com efeito, para um ambiente do porte da Internet, a inteligência na resolução de problemas reside na possibilidade de cooperação entre softwares autônomos distintos, ou *agentes inteligentes*, baseados em conhecimento explícito e com capacidade de comunicação em nível de conhecimento, o que significa a possibilidade de enviar conhecimento definido num formalismo de representação de conhecimento compreensível por outros agentes. As justificativas desta assertiva serão apresentadas na próxima seção,

onde serão apresentadas vantagens do modelo de comunicação *peer-to-peer* sobre o modelo Cliente-Servidor.

3.2.2.1. Falsos Sinônimos entre as áreas de IAD e Sistemas Distribuídos

Apresentados os conceitos de comunicação da IAD, é conveniente citar alguns termos comuns às duas áreas, que, todavia guardam significados bastante distintos.

Em primeiro lugar, na área de Sistemas Distribuídos (SD), o termo comunicação "*peer-to-peer*" designa redes que esperam que seus usuários contribuam com seus próprios arquivos, servidores ou outros recursos num projeto compartilhado, como os sítios de música no formato MP3, por exemplo. Enquanto em SD, o termo *heterogeneidade* refere-se a diferentes protocolos ou sistemas operacionais numa mesma rede, em IAD, o termo normalmente indica a existência de agentes que possuem bases de conhecimento em diferentes formalismos de representação. Em SD, são *compartilhados* recursos de rede; por sua vez, em IAD, existe a ontologia comum (ou *compartilhada*), usada como vocabulário de uma comunicação em nível de conhecimento. Também o termo *escalabilidade*, que em SD refere-se à capacidade de um servidor manter sua eficiência mesmo com o aumento do número de clientes ou de serviços, em IAD, um agente apresenta escalabilidade quando consegue receber mais bases de conhecimentos, ou aumentar significativamente uma base existente, e ainda assim realizar inferências de resultado e tempo de resposta satisfatórios. Um outro termo, desta vez importado de Engenharia de Software, é a palavra *portabilidade*: nesta área, um código executável é portável quando não precisa ser recompilado. Em IAD, considera-se uma base de conhecimento como portável quando ela está escrita num formalismo traduzível para outros formalismos, sem conter referências a objetos externos. Finalmente, a própria palavra "comunicação" em IAD ganha um sentido muito mais semântico do que técnico; a idéia de comunicação centra-se mais sobre o ato de fala, sobre o formalismo em que está escrita a mensagem, e sobre o significado da mensagem (ontologicamente contextualizado). Pode-se considerar a comunicação em nível de conhecimento como uma abstração da comunicação cliente-servidor (sobre a qual ela está montada), ou como um nível de aplicação mais elaborado.

3.2.2.2. Vantagens do modelo de comunicação "peer-to-peer" sobre o modelo Cliente-Servidor

Antes de iniciar uma comparação entre os modelos, é necessário frisar um fato, não antes citado na bibliografia existente: o modelo Cliente-Servidor deriva-se diretamente do paradigma de programação procedural, enquanto o modelo *"peer-to-peer"* origina-se do paradigma declarativo. Na verdade, o cerne das vantagens do modelo *peer-to-peer* deriva-se desta fundamental diferença de paradigma de programação. A tabela 1 apresenta um quadro comparativo entre as principais características dos dois modelos

Tópicos	***"Peer-to peer"***	**Cliente-servidor**
Paradigma	Declarativo	Procedural
Representação de Conhecimento	Teoria: conceitos, relações e restrições ("o que")	Funcionamento ("como")
Confiabilidade	Conhecimento explícito (transparente)	Descrição da ação dos métodos fornecidas pelo implementador
Legibilidade	Conhecimento explícito (transparente)	*Information Hiding*
Contexto	Atos de fala e ontologias	Parâmetros tipados
Direção	Proatividade, iniciativa e *multiálogo*	RPC unidirecional, pares
Flexibilidade	Comunicação Dinâmica	Comunicação Pré-definida
Interface	Semântica e Dinâmica	Sintática e Estática
Finalidade	Inferência, reuso, negociação ou documentação para usuários	Execução
Interoperabilidade entre Linguagens	Mapeamentos para OKBC ou tradução lógica entre formalismos	Mapeamentos para CORBA IDL ou DCOM

Tabela 1. Quadro comparativo das características dos modelos de comunicação *"peer-to-peer"* e Cliente-Servidor.

Em primeiro lugar, faz-se necessário salientar que, a exemplo dos diferentes paradigmas que os inspiraram, os dois modelos almejam objetivos distintos: o modelo *"peer-to-peer"* visa estabelecer uma comunicação entre agentes cognitivos, onde o foco é determinado

pela expressividade de comunicação e riqueza semântica das mensagens; já o modelo Cliente-Servidor tem por meta prover mecanismos de comunicação e interoperabilidade entre objetos, com focos em temas como segurança, acesso remoto aos serviços dos objetos e rapidez de comunicação, entre outros tópicos. Desta feita, a comparação tem sentido se o contexto da comunicação volta-se mais a tarefas que exigem maior expressividade que performance, caso típico, por exemplo, da comunicação entre sistemas multiagentes cognitivos. Ferber [Ferber 95] inclusive, esclarece com muita propriedade as diferenças e relacionamentos entre objetos e agentes. Agentes tentam alcançar uma meta e possuem uma função de satisfação desta meta - o que lhes proporciona autonomia -, além de possuírem um mecanismo de comunicação independente do problema a ser resolvido. Por não satisfazerem estes critérios, objetos não podem ser considerados agentes. Porém, agentes são freqüentemente implementados como objetos. Ele considera que, no contexto de IAD, agentes são definidos num prisma mais conceitual, enquanto que objetos são concebidos no nível de implementação.

Posto isto, serão comparadas as duas formas de representação de conhecimento inerente aos modelos. A representação de conhecimento do modelo "*peer-to-peer*" baseia-se na ontologia compartilhada pelos agentes em comunicação, ou seja, em conceitos (metáfora do "o que"), axiomas, relações e restrições acerca de um tópico. Uma vez que estes conceitos são representados explicitamente em algum formalismo lógico de representação de conhecimento por serem declarativos, torna-se possível empregar a Teoria dos Atos de Fala para modelar a comunicação, pois o significado das mensagens é semanticamente claro e expressivo. O modelo "*peer-to-peer*", portanto, possui a característica de *engajamento ontológico*: os agentes trocam conhecimento dentro de um determinado contexto, ou vocabulário - a ontologia compartilhada por ambos -, e as mensagens trafegam via um protocolo que porta mensagens escritas em um formalismo conhecido por eles. Por isso, correm menos riscos de usar valores inconsistentes ou recebidos em contextos impróprios, um erro comum em comunicação baseada em parâmetros.

Já o modelo Cliente-Servidor baseia-se em chamadas a procedimentos remotos, enquanto internamente efetua uma comunicação do tipo pedido-resposta com os parâmetros do método solicitado. A sua representação de conhecimento baseia-se no funcionamento

esperado dos métodos dos objetos *procedurais*[11] (metáfora do "como"). Devido à característica intrínseca de *"information hiding"* dos objetos, a implementação de seus métodos deve ficar escondida dos clientes, deixando apenas a assinatura destes métodos disponível para chamadas remotas. Portanto, um cliente *acredita* na descrição fornecida pelo implementador do objeto "servidor", disponível via CORBA ou DCOM, o que constitui um problema de *confiabilidade*. Isto não ocorre com sistemas declarativos, que mantém conhecimento explícito.

Também a *legibilidade* é prejudicada pelo binômio *"information hiding"* -parâmetros no modelo Cliente-Servidor. Um dos requisitos levados em conta na elaboração da linguagem de comunicação *"peer-to-peer"* KQML é que as mensagens pudessem ser entendidas também pelas pessoas. A própria Teoria dos Atos de Fala tem como premissa o fato de que as linguagens de comunicação devem ser expressivas, capazes de exprimir o conteúdo de uma intenção. No modelo Cliente-Servidor a intenção ou ato de fala de uma mensagem não faz sentido quando se passa um objeto ou sua referência, pois estes não possuem expressividade[12]. Se agentes cognitivos desejam trocar objetos, o correto seria lançar mão de atos de fala especialmente criados para este fim [Botelho & Ramos 2000].

Ademais, a latência e a possibilidade de falha na comunicação, inerente à abordagem RPC[13], não deve ser escondida do próprio programador [Knapik & Johnson 98], mas quando clientes executam métodos remotos, eles não sabem se estes métodos chamam outros métodos remotos.

O conhecimento resumido à assinatura de métodos no modelo Cliente-Servidor também limita o escopo de *contextos* sobre os quais o conteúdo de parâmetros de uma mensagem

---

[11] Cabe salientar que o próprio conceito de objeto aproxima-se mais de uma idéia declarativa do que procedural, face à existência de herança de classes e instanciação, que modelam as classes mais ontologicamente que funcionalmente. Poderíamos inclusive considerar os conceitos contidos nas ontologias como classes de objetos sem métodos, e cujas classes possuem relacionamentos explícitos. Na verdade, os conceitos das ontologias são freqüentemente representados por *frames* [Minsky 75], um dos precursores e inpiradores dos atuais objetos. Outra importante observação é que existem linguagens lógicas que representam objetos declarativamente, como Trinc-Prolog, evidenciando o fato de que objetos podem ser declarativos ou procedurais.

[12] Apesar disso, existem pacotes de comunicação KQML que transportam e trocam objetos, como o AgentBuilder [AgentBuilder 98].

[13] Do inglês *Remote Procedure Call* (Chamada Remota a Procedimento).

enviada numa comunicação entre agentes se insere. Exceto o tipo, formalmente nada mais se sabe sobre os parâmetros de um pedido ou reposta, ou sobre seu papel na execução, regras de restrições e críticas a que serão submetidos, entre outros dados úteis.

O modelo Cliente-Servidor significa execução remota de procedimentos com passagem de parâmetros. Procedimentos são unidirecionais e a forma de comunicação entre eles está previamente definida. Já o modelo *"peer-to-peer"* foi concebido para comunicação entre entidades autônomas. Ao invés de limitar-se ao formato pedido-resposta, o modelo *"peer-to-peer"* implementa um *multiálogo,* ou seja, várias agentes mandando e recebendo mensagens de várias outras agentes dinamicamente, sem uma ordem obrigatória (até porque respostas referenciam explicitamente as respectivas perguntas, conforme se pode constatar no exemplo de KQML da seção anterior) e de forma proativa ou voluntária – ou seja, qualquer agente pode deliberadamente iniciá-la. A extensão da autonomia dos agentes – sua característica mais básica - para *autonomia de comunicação* efetivada através do modelo *"peer-to-peer"*, acrescenta aos agentes um grau de *flexibilidade* na comunicação. Esta comunicação pode ser iniciada, por exemplo, pela descoberta ou aprendizado por um dos agentes de informação que pode ser interessante a outro(s) agente(s), já que os agentes do modelo podem possuir modelagens dos outros agentes que participam do ambiente. Isto não ocorre no modelo Cliente-Servidor porque ele foi criado sob uma perspectiva procedural, para implementar transparência de execução e não para livre comunicação entre entidades autônomas.

Uma comunicação no modelo *"peer-to-peer"* pode causar até economia no tráfego e no tempo de processamento do sistema como um todo: módulos ou agentes não têm que seguir uma seqüência pré-determinada de mensagens, podendo pular alguns passos de processamento e comunicação de acordo com as mensagens recebidas. Além disso, o fato da comunicação ser vista como uma ação deliberada do agente, permite a implementação de outra funcionalidade, fundamental para a efetivação de comércio eletrônico inteligente: a negociação.

No contexto de IAD, a comunicação pode ter um papel mais profundo para os agentes, o de mudar o seu comportamento. Corrobora este fato a existência de um teste de sociabilidade para agentes [Huhns & Singh 97], onde um agente é considerado sociável se ele modifica o seu comportamento com a entrada de outro agente no ambiente. Neste caso, a comunicação

tem por finalidade refinar a atuação de um agente de forma a torná-lo mais útil à sociedade que pertence.

Os agentes são vistos, sob o prisma de comunicação em nível de conhecimento, não como servidores, mas como entidades de software capazes de entender e mandar mensagens referentes a determinadas ontologias, e, se projetadas adequadamente, habilitadas à alocação dinâmica de tarefas, repasse (*brokering*), recrutamento, entre outras capacidades, presentes nos atos de fala. Assim, sua *interface* é mais flexível, dinâmica e semântica; em contrapartida, os objetos disponíveis para RPC no modelo Cliente-Servidor são dotados de uma interface estática, cuja execução de métodos não pode ser modificada, e está amarrada à sintaxe correta do número dos parâmetros e seus tipos, e não à sua semântica. Isto dificulta sua aplicação em sistemas com agentes.

A própria finalidade da comunicação entre os dois modelos é diferente: no modelo *"peer-to-peer"* o conteúdo de uma mensagem pode ser empregado de várias maneiras: para disparar uma inferência ou uma negociação, para reuso - onde o agente passa a dispor de mais conhecimento -, ou simplesmente para disponibilizar o conhecimento como documentação para pessoas interessadas no conteúdo das ontologias. Na comunicação Cliente-Servidor a mensagem pode apenas ser executada no servidor; sua única função é a escolha do objeto e método a ser executado, além da passagem de parâmetros.

O único item em que o modelo Cliente-Servidor pode ser vantajoso consiste na interoperabilidade entre linguagens diferentes. A solução deste modelo resulta muito mais simples do que no modelo *"peer-to-peer"*. Objetos e seus respectivos métodos escritos em linguagens diferentes podem ser executados graças ao mapeamento de tipos entre linguagens no ORB (*Object Request Broker*). A interface CORBA representando a assinatura dos métodos garante os tipos dos parâmetros e a execução do método. No modelo *"peer-to-peer"*, agentes comunicantes podem estar escritos em linguagens de programação diferentes; o problema de interoperabilidade faz sentido quando os formalismos de representação de conhecimento de seus motores de inferência são diferentes. Neste caso, o modelo apresenta flexibilidade, fornecendo duas formas de fazer com que um agente possa utilizar o conhecimento de outro, com a desvantagem de que ambas têm de ser realizadas *off-line*:

- O mapeamento entre os formalismos de representação de conhecimento, que, à semelhança de CORBA e inspirado no componente de conectividade para sistemas gerenciadores de bancos de dados ODBC (*Open Database Connectivity*), liga dois formalismos criando uma interface comum para eles, permitindo a um agente acessar ao serviço de outro (ou o conhecimento de outro agente ou servidor) através desta interface. Por isso, o pacote gerado para implementar esta facilidade foi chamado de OKBC (*Open Knowledge Base Connectivity)* [Chaudri et al 98];

- A tradução lógica entre os diversos formalismos (vide subseção 4.5.1.) facilita o reuso e inclusão de conhecimento existente de um agente para outro, e pode vir a permitir a comunicação entre agentes em formalismos diferentes. Porém, ainda existem problemas, como, por exemplo, diferenças de expressividade entre os formalismos, que dificultam a tradução [Valente et al 99] (vide seção 4.10.). A tradução constitui uma vantagem clara das ontologias sobre sistemas procedurais, por permitir um reuso de conhecimento mesmo em formalismos de representação distintos. Ela poderia garantir a comunicação entre agentes nesta situação, todavia a tradução dinâmica ainda não é factível atualmente devido aos problemas citados, além do alto tempo de resposta que isso consumiria. Por outro lado, a tradução não foi criada como um instrumento auxiliar à comunicação, embora possa vir a ser usada com este propósito no futuro. Ela foi concebida para reuso de conhecimento, já que a parte mais cara e de difícil elaboração em qualquer agente são as bases de conhecimento.

É digno de nota enfatizar que os modelos de comunicação "*peer-to-peer*" e Cliente-servidor foram criados para servirem a propósitos distintos, como foi citado, e uma perspectiva de abordagem à tecnologia de comunicação "*peer-to-peer*" é entendê-la como uma evolução do modelo Cliente-Servidor, como mostra a Tabela 2.

No modelo Cliente-Servidor, as entidades em comunicação têm um papel definido de mestre (cliente) ou escravo (servidor). A abordagem distribuída declarativa foi montada, então, sobre o protocolo Cliente-Servidor HTTP (*HyperText Transport Protocol*), trazendo consigo características próprias e necessárias aos agentes cognitivos, que lhes proporcionam autonomia, e comunicação com conhecimento declarativo e legível. O

conhecimento transmitido pode ser ainda reusado e estendido a partir das ontologias que o agente receptor já contenha.

A partir da adição das diretivas "*client pull*" e "*server push*" ao protocolo HTTP desde a versão 3.0 – que permitem aos clientes "puxar" mensagens, e aos servidores enviá-las – pode-se considerar que já existe uma arquitetura "*peer-to-peer*" puramente declarativa, no sentido em que estas diretivas permitiram, o que a abordagem distribuída declarativa ainda não permitia: a construção dos atos de fala expressando a intenção do agente, além de habilitar ao agente a tomada de iniciativa para iniciar conversação com outro(s) agente(s). Porém, na verdade este modelo "*peer-to-peer*" constitui-se um modelo cliente-servidor adaptado. Idealmente, um protocolo "*peer-to-peer*" deveria ser derivado de uma noção forte de agente [Petrie 96, Wooldridge & Jennings 95], nativamente implementando intenções de comunicação através de atos de fala, o que, com certeza, atenderia aos requisitos e necessidades de comunicação em sistemas multiagentes cognitivos.

Tabela 2. Comparativo entre arquiteturas de comunicação distribuídas (baseado em [Huhns & Singh 98]).

Arquitetura	Entidade	Entidade	Comunicação	Característica
Cliente-Servidor	Mestre: manda.	Escravo: obedece.	RPC	Tranparência de execução
Distribuída	*Peer*: atribui tarefas (Voltado só para si).	*Peer*: satisfaz pedidos (autônomo)	Mensagens assíncronas declarativas	Autonomia, conhecimento, legibilidade, reuso e extensibilidade
Declarativa (baseada em agentes)	*Peer*: cria/ Invoca compromis-sos (Voltado só para si)	*Peer*: Mantém compromis -sos (autônomo)	Atos de Fala	Autonomia, conhecimento, legibilidade, reuso, extensibilidade e sociabilidade

Serão vistas ainda, nas próximas subseções, a linguagem de comunicação *peer-to-peer* KQML, que vem se tornando padrão em comunicação entre agentes, e um pacote que o implementa, o JATLite.

3.2.2.3. A Linguagem de Comunicação KQML (Knowledge Query Manipulation Language)

A linguagem de comunicação KQML compreende um conjunto extensível de *performativas*, que define as operações possíveis de comunicação em nível de conhecimento entre agentes. Através de mensagens escritas neste protocolo, os agentes podem trocar conhecimento, enviando sentenças lógicas, fatos e metas, com o intuito de cooperarem e/ou negociarem. Atualmente, os atos de fala implementados são os atos assertivos (informar), diretivos (pedir ou perguntar), proibitivos e declarativos (causar eventos ao comunicador).

(*ask-if* :sender **Agente1**	(*tell* :sender **Agente2**
:receiver **Agente2**	:receiver **Agente1**
:in-reply-to **id0**	:in-reply-to **id1**
:reply-with **id1**	:reply-with **id2**
:language **Prolog**	:language **Prolog**
:content "**Portuguese-cities(X,Y,Z)**")	:content "[Portuguese-cities(30,40,0), Portuguese-cities(76,10,60)]"

Tabela 3. Exemplo de troca de mensagens em KQML.

Uma mensagem KQML consiste de uma performativa, seus argumentos obrigatórios – um deles é o conteúdo da mensagem – e outros argumentos opcionais que descrevem o conteúdo, porém escritos de forma independente da sintaxe da linguagem do conteúdo. Veja-se o exemplo da Tabela 3: O Agente1 quer saber se o Agente2 sabe a localização (em termos de latitude e longitude) de alguma cidade portuguesa. O Agente1 pergunta isto ao Agente2, que retorna as respectivas coordenadas das cidades, contidas na sua base de dados. Como se pode observar, o conteúdo das mensagens foi codificado em Prolog.

3.2.2.4. A Linguagem de Comunicação ARCOL da FIPA

Visando a disseminação do uso de agentes, especialmente no comércio eletrônico, e ligada ao movimento de software livre, foi criada a FIPA (*Foundation of Intelligent Physical Agents)* [O'Brien & Nicol 98], e com ela, uma tentativa de padronização de comunicação entre agentes, e também destes com usuários e aplicativos.

Nesta abordagem, foi desenvolvida a linguagem de comunicação ARCOL, como uma extensão de KQML, porém com uma semântica mais formal. Permite ainda a composição de mais de um ato de fala numa mensagem (como, por exemplo, *:request(... :content (:register))),* e implementa atos de fala comissivos (de compromisso), através do campo *:protocol*, que especifica a estratégia de coalizão entre os agentes (como o *contract-net*, que promove um "leilão" entre os agentes na hora de alocar uma tarefa), e, por conseguinte, a seqüência de mensagens a ser seguida.

Além destas melhorias a FIPA preocupou-se também com os aspectos "de rede" dos agentes, apresentando propostas como nomes globais para os agentes, requisitos para plataformas que recebem a visita ou hospedam a comunicação entre agentes, interfaces para usuários, agentes móveis, e outras facilidades.

3.3. Adequação da Engenharia do Conhecimento ao Problema

3.3.1. Benefícios e Facilidades do Conhecimento Explícito

Como se pôde observar do capítulo 1, a abordagem tradicional de RI representa superficialmente os textos através das palavras-chave contidas neles. Adicionando melhores representações, com conhecimento, técnicas de Processamento de Linguagem Natural (PLN) e capacidade de inferência, melhores resultados são produzidos [Croft 93].

Devido à declaratividade empregada, a Engenharia do Conhecimento, cuja área de pesquisa visa o desenvolvimento de sistemas baseados em conhecimento explícito, há décadas desenvolve tecnologia complementar à tecnologia de IR, potencialmente útil na combinação de informações para problemas específicos, situações ou preferências de usuários [van de Velde 95]. Conforme sugerido no Capítulo 1, os agentes podem realizar um refinamento sucessivo sobre resultados de consultas a mecanismos de busca, de forma

a encontrar e extrair corretamente dados de uma determinada classe, empregando conhecimento *a priori* sobre o domínio tratado para inferir sobre as informações a serem extraídas e criticá-las.

A declaratividade - ou seja, a inclusão de conhecimento descritivo acerca das entidades participantes do domínio representado, normalmente localizado do lado de fora dos programas e aplicável em inferências – traz vários benefícios. Em primeiro lugar, ela empresta *engajamento ontológico* às soluções – ou seja, similaridade com o conhecimento da forma como ele é organizado, e com a terminologia empregada em sua área específica, provendo uma tradução mais direta do conhecimento e teoria do domínio. Além do mais, se ganha expressividade e flexibilidade, uma vez que o conhecimento sobre uma classe não se circunscreve a termos e palavras-chaves como em mecanismos de buscas, mas a *qualquer* fato que diga respeito às páginas, como estrutura, regiões, conceitos contidos nelas e significados de suas frases, pelo uso de técnicas de PLN. Uma interessante visão disso é que soluções declarativas não necessariamente competem com soluções de RI [Croft 93], dado que múltiplas representações advindas de RI das páginas (tais como palavras-chave e suas freqüências, centróide, ponteiros, e-mails, etc) podem tomar parte nos processos de inferência. Para este trabalho, as representações serão escolhidas de acordo com sua utilidade nas tarefas de extração e reconhecimento de membros da classe processada.

O capítulo seguinte investigará mais a fundo os benefícios de abordagens declarativas, bem como comparativos mais atuais com abordagens procedurais. As subseções seguintes descrevem alguns tipos de conhecimento que podem ser empregados em processos de inferência relativos à manipulação integrada de informação.

3.3.2. Conhecimento a priori

Ressaltando o que foi dito acima, a principal vantagem de empregar engenharia do conhecimento é a possível inclusão de conhecimento contextual ou conhecimento *a priori* sobre o domínio modelado. Este conhecimento pode provir de várias fontes.

3.3.2.1. Localização e Prefixos

Em primeiro lugar, a Internet não implementa *transparência de localização*[14], o que proporciona várias facilidades na busca e recuperação. Desta feita, pode-se eliminar páginas de antemão, como por exemplo, páginas candidatas a serem da classe de turmas universitárias que estão no domínio *.gov,* ou lojas comerciais no domínio *.edu.* Pode-se ainda, a partir da estrutura de prefixos dos diretórios, achar páginas autoritativas e conjuntos homogêneos de páginas. Por exemplo, a página http*://www.cin.ufpe.br/docentes* deve conter embaixo de si, páginas dos professores de um dos centros da UFPE. Como estes, existem muitos outros padrões, como, por exemplo, */papers/, /publications/, /CFP/* [Craven et al 98].

3.3.2.2. Classes e Grupos

Tratando-se de classes, grupos e categorias funcionais, vários fatos podem ser empregados no reconhecimento, recuperação e extração de páginas. Uma página membro de uma classe deve seguir certas características quanto ao conteúdo, como atributos da entidade da classe, ponteiros para páginas membros de outras classes do grupo, estrutura na ordem de aparecimento dos atributos ou de editoração, entre outras, já abordadas no capítulo anterior.

3.4. Adequação dos Sistemas Multiagentes ao Problema

Voltando ao problema proposto, cabe lembrar que centralizar recursos na Internet é sinônimo de baixa eficiência, devido ao congestionamento de pontos específicos da rede e a capacidade limitada de processamento dos servidores.

Face ao tamanho da Internet, qualquer aplicação que se proponha a lidar com toda a rede – seja um mecanismo de busca, um coletor de estatísticas ou outra aplicação qualquer – deve aproveitar-se de uma característica intrínseca a qualquer rede: a distribuição. A distribuição

---

[14] *Transparência de localização* significa o acesso a um recurso qualquer de uma rede, remota ou localmente, sem a preocupação ou qualquer informação sobre onde ele se localiza [Coulouris et al 94].

gera concorrência e paralelismo, e isso faz a diferença em termos de tempo de resposta[15]. Assim, um processo deve ser capaz de pedir e receber auxílio de outros processos, e assim, compartilhar os resultados de suas operações.

Entretanto, para que estes processos cooperem efetivamente, é preciso um modelo de comunicação que lhes permita trocar o conhecimento que possuem acerca da tarefa que lhes foi alocada. Uma comunidade de agentes inteligentes, ou *sistema multiagente*, como é mais comumente conhecido, se enquadra adequadamente como a solução para este problema de cooperação entre os processos. Conforme visto ao longo do capítulo, agentes podem se comunicar através da rede, pedindo e dando informações a outros agentes, em uma linguagem expressiva e semanticamente bem definida. Além disso, eles podem negociar entre si em função dos recursos disponíveis, objetivos, comuns e específicos, inclusive para a formação de equipes.

Para o problema específico aqui proposto, em que se enxerga claramente a existência de classes de páginas inter-relacionadas a serem processadas, entregar uma classe para cada agente oferece vários benefícios. Em primeiro lugar, cada agente obedece à recomendação de restrição de domínios (vide seção 1.2.), o que torna o problema tratável. Em segundo lugar, como as classes possuem ligação, os agentes podem cooperar, apressando a busca e recuperação de páginas relevantes. Além disso, a modelagem de problemas de controle e sincronização, típica de sistemas distribuídos, constitui um tema central de pesquisa em sistemas multiagentes: modelos de coordenação entre os agentes. Finalmente, o argumento da *heterogeneidade*; uma das situações em que o emprego de sistemas multiagentes cognitivos é aplicável, é quando se identifica, para a resolução do problema proposto, a necessidade do uso de conhecimentos inter-relacionados de áreas diferentes. Nestes casos, sugere-se a divisão de uma ou mais áreas de conhecimento por agente. Para capacitar-se a tratar umm grupo de classes de páginas, um agente só teria de reunir uma gama de

---

[15] Apesar do fato de os sistemas de busca serem centralizados, o processo de busca das informações para a indexação é executado não por apenas um robô de busca, mas por um conjunto deles. Ainda assim existem propostas de distribuir a base de índices [Falcão et al 97], cuja principal alegação é diminuir o tráfego na rede. Hoje, para procurar informações que geograficamente estão próximas, às vezes é necessário recorrer a indexadores distantes geograficamente. Outra interessante aplicação da distribuição consiste no próprio roteamento de mensagens. Nenhum nó centralizado contém todos os endereços; em vez disso, cada nó sabe exclusivamente para qual outro nó deve rotear os datagramas.

conhecimentos muito diversa sobre estas classes, dificultando a formalização e exigindo uma capacidade computacional bastante elevada de um nó específico da rede.

À parte do problema que formulado neste trabalho, por todo o exposto neste capítulo, conclui-se que, face à quantidade e desestruturação da informação disponível em sistemas abertos distribuídos como a Internet, inteligência significa comunicação útil, semanticamente contextualizada, e sociabilizada entre diferentes entidades de softwares, que os sistemas multiagentes podem prover adequadamente.

3.4.1. Manipulação Cooperativa de Informação[16]

Oates et alii [Oates et al 94] apontam prováveis diretrizes para a manipulação de informação baseada em conhecimento na Internet. Definindo apropriadamente manipulação de informação como a união dos processos de aquisição e recuperação de informação, o artigo propõe sistemas multiagentes, compostos de agentes independentes e cooperativos que consigam integrar, processar e construir nichos consistentes de informação relevantes. A Resolução Distribuída de Problemas (RDP) é recomendada como um meio de fazer com que os agentes negociem entre si na descoberta destes nichos, e o uso intensivo de tecnologias baseadas em inferência e conhecimento constitui uma premissa para a solução proposta pelos autores.

No artigo, os autores assumem "a disponibilidade de meios de gerar *descritores* representativos do conteúdo dos documentos em termos de um conjunto de primitivas de domínio", citando textualmente sistemas de extração como tecnologia a ser empregada. Este trabalho compartilha esta perspectiva, acrescendo-a com a percepção clara de que duas tarefas distintas são necessárias para transformar estes conceitos de manipulação cooperativa em prática: a tarefa de aquisição e a tarefa de recuperação.

Abite et alii propuseram posteriormente agentes para manipulação de informação [Abite & Knoblock 97] que coincidem exatamente com a tarefa de recuperação do modelo proposto. O sistema apresenta o seguinte funcionamento: bases de dados de uma vasta biblioteca digital são agrupadas em classes hierárquicas, onde cada classe possui seu próprio agente, que detém conhecimento explícito sobre sua classe e sobre as capacidades dos outros

agentes. Numa abordagem tipicamente de bancos de dados, os agentes constroem planos de recuperação que otimizam a eficiência da recuperação de consultas complexas. Para transportar este tipo de ferramenta para Web, seriam necessários outros sistemas multiagentes, também com agentes específicos para cada classe, que preencheriam as tabelas das bases de dados com dados extraídos de páginas da Web, a tarefa de aquisição supracitada.

A tarefa de aquisição de informação deve preceder a recuperação, e compreende as tarefas de aprendizado sobre domínios a partir de modelos destes, e de classificação e extração de dados das páginas pertencentes a estes domínios por agentes, segundo Oates et alii, pertencentes às próprias fontes de informação, os *servidores de informação*. Embora possam ser considerados servidores de informação no sentido lato do termo, no trabalho ora apresentado, como será visto no capítulo 5, os agentes não se localizam nos servidores, mas visam justamente atacar esse segundo problema, complementando a tarefa de recuperação inteligente de informação sobre bases de dados geradas por agentes encarregados da recuperação, classificação e extração otimizadas a partir de páginas de *clusters* da Web.

Assim, mesmo em *clusters* que possam sofrer perturbações de contexto, como turismo - em que as condições do tempo atuam como fator fundamental na busca de dados como hotéis, eventos e outros – o binômio formado pelos dois tipos de sistemas multiagentes, os de aquisição e os de recuperação, se capacita a prover informações úteis e focalizadas a partir de páginas de domínios da Web.

---

[16] Do inglês *Cooperative Information Gathering*.

Capítulo 4

ONTOLOGIAS

4.1. Introdução

No capítulo anterior, pôde-se constatar que a abordagem multiagente, baseada em conhecimento explícito representado por ontologias compartilháveis, constituem a ferramenta adequada para uma construção mais específica de contextos associados à busca de informações na Internet. Este capítulo visa, portanto, apresentar um pouco mais sobre os princípios, conceitos fundamentais, componentes e técnicas de construção e manutenção de ontologias.

4.2. A Controvérsia Declarativo-Procedural

As duas abordagens de elaboração de soluções computacionais, aparentemente em oposição, que surgiram logo após o advento das linguagens de alto nível - quando se iniciou a discussão sobre a melhor forma de projetar sistemas - são:

- A abordagem *declarativa*, que descreve fatos acerca de um determinado domínio (metáfora do "o quê"), fazendo uso de mecanismos de inferência, que deduzem, a partir dos fatos existentes, novos fatos, entre eles a solução ou ação a ser tomada;
- A abordagem *procedural,* que descreve o funcionamento de processos em suas nuances mais detalhadas (metáfora do "como"), sem maiores compromissos em descrever as características das entidades a serem representadas.

Os anos 70 e 80 assistiram a um debate entre qual delas seria a melhor forma de conceber e programar sistemas. Na realidade, parece existir complementaridade ao invés de competição, já que cada uma se adequa melhor a tarefas específicas.

Em termos de eficiência e controle, a programação apresentou melhor performance, tornando-se, de longe, o paradigma de programação mais empregado. A chegada da Internet e dos sistemas abertos, tornando as informações acessíveis a milhões de pessoas -

ambientes desorganizados e ricos, que requerem a existência de *agentes inteligentes* com forte orientação à cooperação e conseqüentemente à comunicação -, têm demonstrado a necessidade massiva de expressividade e de reusabilidade flexível de conhecimento, provocando uma premente revisão de conceitos. O ressurgimento com vigor de soluções declarativas baseadas em lógica acarreta inúmeros benefícios, especialmente para a classe de tarefas relacionada a estes ambientes. Recentes avanços, e, em especial, a popularização das ontologias, modificaram substancialmente as prerrogativas da controvérsia declarativo-procedural em relação à época em que foi postulada [Winograd 75].

4.3. Níveis de Especificação de Sistemas Baseados em Conhecimento

A tentativa de criar sistemas diretamente a partir do conhecimento acerca dos processos, i. e. declarativamente, tornando possível aos computadores realizar dedução automática deu origem à área chamada hoje de Inteligência Artificial Simbólica. Fundamenta-se basicamente na separação entre o conhecimento e o processo dedutivo ou inferência. A concepção de sistema dessa natureza passa por três especificações consecutivas [Newell 82]:

- O *nível de conhecimento ou epistemológico,* que consiste numa especificação abstrata do conhecimento do domínio ou problema que se deseja modelar;
- O *nível lógico*, que converte as especificações de conhecimento em sentenças de lógica formal;
- O *nível de implementação*, que codifica estas sentenças em linguagem computacional, seja numa linguagem ou num banco de dados dedutivo, dito base de conhecimento.

Marcadamente, a existência do nível de conhecimento é de suma importância para o desenvolvimento de sistemas simbólicos, principalmente por uma razão, que se confunde com a própria história da Inteligência Artificial Simbólica. A motivação principal do paradigma declarativo era *modelar sistemas num nível mais elevado.* O nível de conhecimento guarda uma relação mais próxima e direta com o conhecimento sobre o domínio a ser modelado. O modelo declarativo como um todo consiste numa forma mais flexível de se descrever um domínio, uma vez que, no nível de conhecimento ainda não há

compromisso com a implementação. A motivação de evitar escrever código e tentar abstraí-lo colocando o conhecimento do lado de fora dos sistemas é extremamente legítima: a maior parte das vantagens do paradigma declarativo – como flexibilidade, legibilidade, fidelidade semântica, engajamento ontológico, extensibilidade, reuso, abstração das compilações, portabilidade e capacidade de inferência - já era visionada desde essa época, e corroboraram essa idéia. Ainda não se enxergava, contudo, o potencial do uso do conhecimento por entidades de software, que as ontologias viriam a proporcionar, organizando o conhecimento em conceitos e domínios, e nem que elas implementariam o nível de conhecimento apropriadamente, em tese, sem refletir algum formalismo de representação de conhecimento específico. Saliente-se que esta prática traz em si a possibilidade do conhecimento ser portável, *enquanto conhecimento e não enquanto código*, e, portanto, independente de máquina.

Finalmente, cabe lembrar um último argumento favorável às soluções declarativas, que só pôde ser percebido em sua plenitude recentemente, à luz das grandes bases de conhecimento estruturadas, ou *ontologias*: *o engajamento ontológico ou de conhecimento*, ou seja, a semântica das sentenças codificadas em lógica guarda uma relação mais direta com o domínio modelado e permite o reuso de um grande volume de informação que desempenha o papel de conhecimento estruturado, disponível para reuso em larga escala por sistemas e programas.

4.4. Formalismos de Representação de Conhecimento Declarativo

Essa idéia de manter fatos e conhecimento do lado de fora dos programas começou a ser fomentada por um sistema em 1958, o Advice Taker [McCarthy 58], e sedimentou-se com o uso massivo de Lisp na representação de fatos e regras em sistemas especialistas [Winston & Horn 81]. Ao invés de colocarem-se os fatos dentro das regras, percebeu-se que as regras poderiam ser mais genéricas se os fatos estivessem separados. Com isso, o conceito de processo, ao invés de significar uma ação, passou a denotar muito mais uma decisão de ação a ser tomada – no caso de Lisp, através de casamento de padrões, um primeiro esboço de método de dedução, só que ainda sem a presença de lógica formal. LISP (*List Processing*, em português Processamento de Listas), uma linguagem funcional voltada para o manejo de listas, foi criada justamente com o propósito de representar conhecimento de forma mais abstrata.

Em meados dos anos 70, começaram a aparecer os *sistemas de inferência orientados a padrões* [Hayes-Roth *et al* 78], que incorporam a programação relacional ou lógica. Estes sistemas são compostos basicamente de três entidades:

- Um *formalismo* fundamentado em lógica matemática, no qual o código do sistema baseado em conhecimento será escrito;
- Um *método ou estratégia de resolução* para o formalismo escolhido, dito *mecanismo de inferência*;
- Estratégias de controle e escalonamento da inferência ou métodos de resolução de conflitos, implementados dentro dos mecanismos de inferência por módulos chamados *executivos* [Hayes-Roth et al 78], com a função de guiar e otimizar a inferência, que normalmente não é consistente ou completa.

4.4.1. Regras de Produção

Particularmente, o emprego de regras enquanto formalismo de representação de conhecimento tem angariado notório sucesso junto aos usuários, devido à sua simplicidade e facilidade de uso. A maior parte dos sistemas especialistas emprega este formalismo. Eis um exemplo de uma sentença lógica transformada em regra orientada ao conseqüente:

```
Animal(x)^estimação(x)^pequeno(x) => doméstico(x)
```

premissas — conseqüente

Esta regra pode ser entendida como: "Para todo x (quantificador universal implícito), se existe um fato afirmando que x é animal e outro que é de estimação e outro que é pequeno, isto implica a validade de um novo fato, o de que x é doméstico".

A partir da "aprovação" de suas premissas, várias regras podem encontrar-se em condições de serem disparadas ao mesmo tempo, não o conseguindo devido ao funcionamento seqüencial do hardware. A função da resolução de conflito consiste justamente na escolha da regra a ser disparada.

Curiosamente, apesar da popularidade das regras de produção, a literatura não reporta boas práticas de construção de regras. Durante o projeto de regras, deve-se pensar que elas são implementadas através de uma rede [Forgy 82], em que predicados iguais em regras diferentes estão representados em apenas um nó da rede para ganho de eficiência. Eis algumas práticas de construção eficiente de regras advindas da experiência deste trabalho:

- Evitar predicados com atributos muito gerais e que variam com freqüência (ex: status);
- Construir o lado das premissas como uma consulta eficiente de banco de dados, ou seja, contendo apenas predicados necessários ao disparo de regras.

4.4.1.1. Sistemas de Produção

Um dos motivos práticos que fez com que as técnicas de Inteligência Artificial não lograssem a popularidade almejada como ferramentas disponíveis na elaboração de sistemas em geral, deveu-se à postura isolacionista assumida durante a elaboração de sistemas e protótipos, sem as preocupações, comuns às outras áreas de computação, com interoperabilidade, portabilidade e reuso [Pachet 95]. À parte da demonstração do potencial dos sistemas especialistas, poucos aplicativos se transformaram em produtos, principalmente à dificuldade de integrar as primeiras linguagens padrão – LISP e Prolog – com outras linguagens [Riley 99].

Com a percepção de que a integração entre as linguagens convencionais - tais como "C" e Java - e motores de inferência poderia resultar benéfica, foram criados os sistemas de produção. Escreveram-se motores de inferência integráveis a códigos imperativos, com a vantagem de manter o conhecimento fora dos programas; assim, as regras de produção incorporaram uma separação ainda mais radical e eficiente entre dados e processo pelo uso de estruturas de dados para armazenar as regras, ditas *bases de conhecimento*, que são "compiladas" em tempo de execução, com indexação eficiente baseada em redes [Forgy 82] para o problema da resolução de conflitos, e, por isso, foi empregada maciçamente em sistemas especialistas e agentes. Com isso resolveu-se o problema da compilação, deixando o conhecimento fora dos programas, e, portanto, mais extensível e alterável.

Os sistemas de produção possuem facilidades de troca de variáveis e objetos com as linguagens hospedeiras, o que provê uma integração completa com o código imperativo, podendo inclusive executar métodos de objetos recebidos pelo motor dentro de regras. Existem ainda os chamados *sistemas imbutíveis* [Pachet 94], que tratam da integração entre objetos e conhecimento declarativo. A integração proposta por estes sistemas chega inclusive à sintaxe da declaração de regras, onde os atributos de um *frame* (mais conhecidos como *slots*) são referenciados da mesma forma que métodos de um objeto (por exemplo, *a.pai(b)* que significa que o pai de *a* é *b*).

Apesar de isso ainda não ser aceito unanimemente, a confecção de conhecimento para sistemas de produção deve empregar o mínimo de código imperativo, ou as capacidades de tradução e comunicação podem ficar comprometidas. Recentemente, num debate na lista de assinantes do motor de inferência Jess (vide subseção 6.2.3.), o autor desta tese colocaou esta posição e recebeu, como resposta, críticas de que a vantagem de um motor de inferência como esse reside na possibilidade de usar objetos da linguagem hospedeira, o que inclusive garante maior eficiência. Porém, logo depois surgiu uma contraprova: um usuário queria reaproveitar conhecimento escrito no motor de inferência CLIPS (*"C" Language Integrated Production System*, sistema de produção integrado á linguagem "C", vide susbseção 6.2.3.), e, como não tinha sido escrito sem a separação entre código e conhecimento, não poderia rodar diretamente no Jess, apesar de Jess capacitar-se a executar regras CLIPS.

4.4.2. Quadros (Frames) e Redes Semânticas

Outro formalismo bastante utilizado são os sistemas baseados em *frames* [Minsky 75]. Eles foram inspirados na forma como as pessoas resolvem problemas, trazendo da memória estruturas de padrões de situações, que tentam instanciar, e são considerados um dos precursores dos atuais objetos. Os *frames* ou quadros guardam íntima semelhança com os objetos, pois possuem:

- *Atributos* para os quadros - preenchidos com valores, ou com *facetas*, procedimentos ou funções para atribuir valores aos atributos, que não são implementadas por objetos, e
- *Herança*, que são relações entre os quadros do tipo *é-um* e *é-um-subconjunto-de*, bem semelhantes aos conceitos de instância e subclasse dos objetos.

Pode-se considerar as Redes Semânticas [Woods 75] como um sistema baseado em *frames*, se os arcos da rede forem atributos do frame correspondente. Ressalve-se, porém, o fato de que as relações de herança e os conceitos de classe constituem o cerne de um *frame*, e se a hierarquia de conceitos for mais importante na representação do domínio tratado que os atributos, a representação em frames ganha em organização. Quando os atributos são mais importantes, as Redes Semânticas são mais adequadas. Por sua expressividade nesses casos, elas têm sido largamente aplicadas em Processamento de Linguagem Natural.

A semelhança entre *frames* e objetos é patente, exceto pelas facetas e pela ausência de encapsulamento, desnecessária para a declaratividade. Por exemplo, a linguagem *LIFE*

(Logics, Inheritance, Functions and Equations) [Ait-Kaci & Podelski 93], uma linguagem lógica e funcional com tipos e herança, representa redes semânticas e facetas, porém não código em forma de objetos. Um exemplo de rede semântica aparece na figura 7, precedido de seu código e de uma faceta em LIFE.

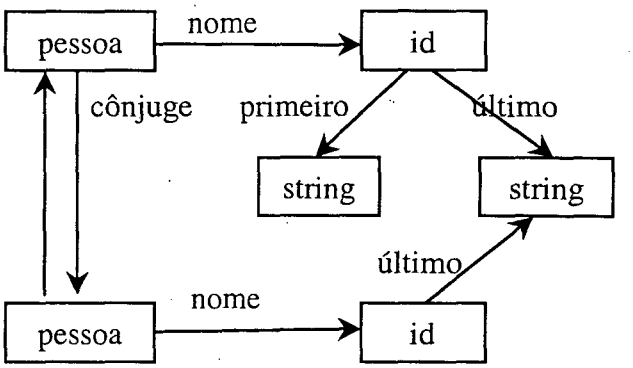

Figura 7. Representação gráfica de uma rede semântica, descrita abaixo.

```
X: pessoa ( nome => id ( primeiro => string, último => Y: string ),
        Cônjuge => pessoa ( nome => id ( último => Y),Cônjuge => X))
```

Exemplo de uma faceta: `idade(pessoa(aniversário=>data(ano=>X)),`
`EsteAno:integer)=>EsteAno-X.`

Ao fim deste capítulo, encontram-se exemplos de *frames*, atributos, heranças e relacionamentos, onde pode constatar-se a similaridade com redes semânticas. Vale ainda lembrar que nada impede que *frames* e regras possam ser combinadas; o próprio CLIPS, para ganhar expressividade, foi projetado para disparar regras sobre *frames,* implementados com todas as suas características [Giarratano & Riley 98].

4.4.3. Lógica de Descrição

Inicialmente chamada de *lógica terminológica* [Brachman & Schmolze 85], este formalismo propõe a definição de conceitos por meio de descrições, fornecendo nativamente atributos bastante expressivos, como [Fensel 2001]:

- *Herança Múltipla*: e.g., (PRIMITIVE (AND CAR EXPENSIVE-THING) sports-car), denotando que uma instância do conceito carro esporte (*sports-car*) herda as descrições de CAR e EXPENSIVE-THING. Esta não pode ser considerada a única forma de herança para esta lógica, uma vez que existem outras formas de aproveitar descrições já existentes, como pode ser observado em outros atributos logo abaixo.

- *Papéis*: e.g., (FILLS thing-driven Corvette), significa que um Corvette desempenha o papel de algo dirigível;
- *Restrição de valores:* e.g., (ALL thing-driven CAR), onde tudo que é dirigível dentro deste domínio deve ser da classe CAR;
- *Restrição de limites:* e.g., (AT-LEAST 3 wheel), descrevendo qualquer objeto relacionado a 3 outros objetos distintos que "desempenham o papel" de roda (*wheel*).
- Co-referência: e.g., (SAME-AS (driver) (insurance-payer)), atribuindo a todos os indivíduos que atuam como motoristas (*driver*) o papel de segurados (*insurance-payer*).

Exemplo de sistemas que implementam este formalismo são BACK, CLASSIC, CRACK, FLEX, K-REP, KL-ONE, KRIS, LOOM e YAK [Classic 2002].

4.4.4. O Elemento de Representação Comum: Frames

Até o surgimento do projeto KSE *(Knowledge Sharing Effort* – esforço de compartilhamento de conhecimento), os sistemas baseados em *frames* eram aplicados quase que exclusivamente na especificação do nível lógico de agentes e sistemas especialistas, devido à inexistência de boas ferramentas de apoio para o nível de implementação, tanto de ambientes de programação quanto de edição do conhecimento. O KSE modificou esse quadro substancialmente, provendo uma excelente ferramenta de edição de ontologias, o Editor da *Ontolingua* [Farquhar 96], e também tradução para formalismos diversos de representação de conhecimento. A motivação era, principalmente, aproveitar conhecimento existente codificado em grandes bases de conhecimento, como o Cyc [Lenat 90], o WordNet [Miller 95] e outras. Os formalismos de representação destas bases coincidiam apenas na existência de *frames*, em sua forma original, como em CLIPS, de forma mais expressiva, como em lógica de descrição ou redes semânticas, ou de forma indireta, com linguagens que pudessem facilmente representar *frames*, como Lisp e Prolog.

Vários benefícios imprevistos foram gerados a partir do advento do KSE. Eles ajudaram a inspirar e impulsionar:

- A "indústria de ontologias", ou seja, despertou-se o interesse pela codificação de conhecimento reusável e comunicável, em oposição a código imperativo,

- A capacidade de comunicação em nível de conhecimento, já que o conhecimento se transformou em *dados* utilizáveis por um programa com motor de inferência,
- E, indiretamente, a capacidade de tradução, por juntar os fatores de conhecimento fora do código, organizado num formalismo lógico, com semântica claramente definida e independente de linguagem.

Provavelmente, num futuro próximo, muitas das páginas da Internet se farão disponíveis através de formalismos lógicos (vide subseção relativa à Web Semântica, 4.6.2.), permitindo a agentes cognitivos tratarem as informações com maior precisão, sem tropeçar tanto em sintaxe e polissemia, como os mecanismos de busca atuais.

4.5. Ontologias Reusáveis e Compartilháveis

Os formalismos de representação de conhecimento proporcionaram a preciosa oportunidade de se estruturar conhecimento em ontologias, com conjuntos de conceitos sobre determinados domínios, definindo as propriedades e relações destes conceitos como axiomas. Apesar da palavra "ontologia" denotar uma teoria sobre a natureza do ser ou existência, em Inteligência Artificial ela pode ser interpretada como o conjunto de entidades com suas relações, definições, axiomas e vocabulário. Uma ontologia é capaz de definir um domínio, ou, mais formalmente, especificar uma conceitualização acerca dele [Gruber 95]. Normalmente, uma ontologia é organizada em hierarquias de conceitos ou taxonomias. Pelo fato de, idealmente, não refletirem nenhum formalismo específico, e de representarem com freqüência uma interface de vocabulário comum entre usuários e sistemas [Clark 99], pode-se considerar as ontologias como a materialização do nível de conhecimento [Newell 82].

Atémeados do anos 90, o conhecimento de um sistema especialista não podia ser reusado ou compartilhado, organizando-se em bases de conhecimento monolíticas e isoladas, sem interfaces de acoplamento, e, portanto, sem interoperabilidade. A construção de ontologias pode ser vista como um passo importante de evolução na especificação de conhecimento. A decomposição desse conhecimento em módulos de construção com forte engajamento ontológico – ou seja, similaridade com o conhecimento da forma como ele é organizado, e, especialmente com a terminologia empregada numa área específica – propiciou a criação de ontologias reusáveis, com conceitos instanciáveis e especializáveis.

Ontologias pré-construídas sobre domínios restritos têm sido bastante reutilizadas e podem vir a representar um papel fundamental como fornecedoras de conhecimento para a inferência dinâmica realizada por agentes inteligentes. Fora isso, as ontologias garantem, através de um vocabulário comum de termos e seus respectivos conceitos, definições, relações, axiomas e restrições, a comunicação em nível de conhecimento entre agentes, como ilustra a figura 8.

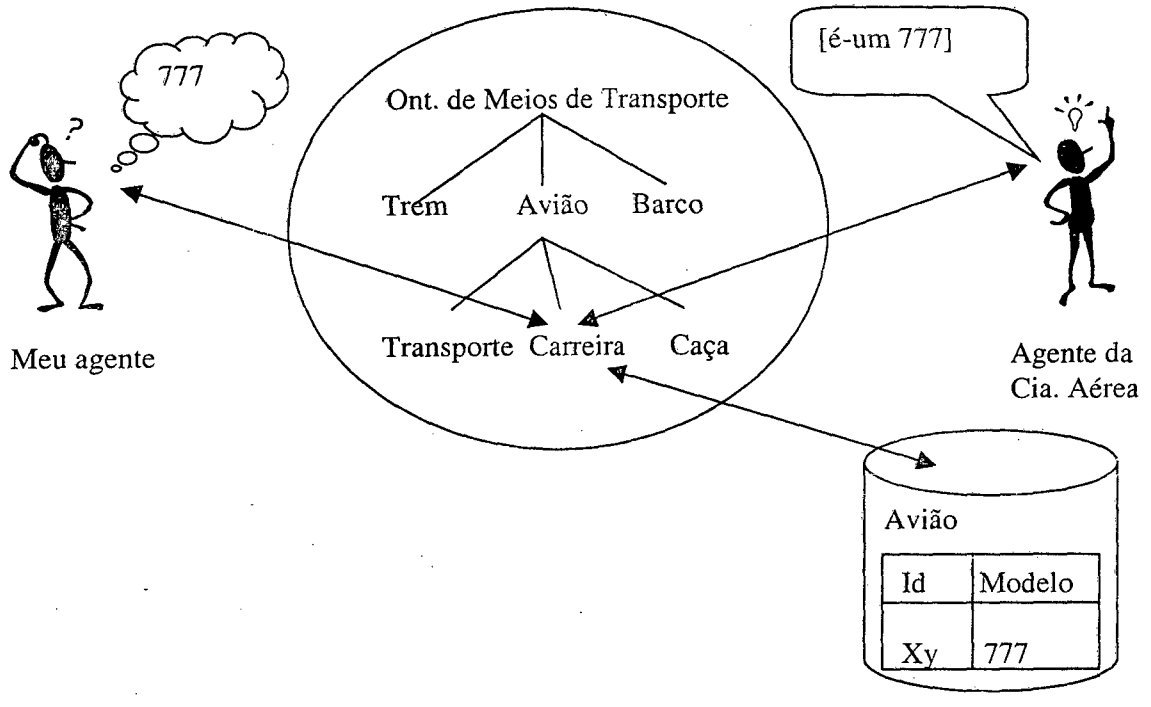

Figura 8. Agentes em cooperação através de ontologias [Huhns & Singh 97a].

Esse compromisso com o vocabulário do domínio desempenha, ainda, um importante papel na identificação de significadosdistintos de uma mesma palavra ou termo, o que pode acarretar uma significativa melhoria de precisão em sistemas de manipulação de informação na Web. Quanto mais específica é uma ontologia, menos ambigüidades e interpretações tornam-se possíveis [Uschold & Jasper 99]. Para o usuário que está à busca de dados específicos, isso pode significar a economia de horas de trabalho [Price 95].

4.5.1. Benefícios das Ontologias

Desde que foi criado o projeto KSE (*Knowledge Sharing Effort*) [Farquhar 96], estão sendo criadas e mantidas ontologias extensíveis, abrangentes, gerais e muito detalhadas, por grupos de pesquisa, abarcando toda a pesquisa da área cujo conhecimento se deseja representar. Esta orientação ontológica trouxe muitos benefícios, alguns dos quais não

previstos, e que só vieram frutificar à época de sua implementação. De início, o KSE e suas ontologias contribuíram para uma maior cooperação entre os grupos de pesquisa responsáveis por manter as ontologias, da mesma forma como mantêm conhecimento, o que, tornando-se uma tendência, pode vir a provocar uma mudança cultural. Outros benefícios são:

- A oportunidade para os desenvolvedores de reusar ontologias e bases de conhecimento, mesmo com adaptações e extensões. Sem dúvida, o impacto sobre sistemas baseados em conhecimento é enorme: a construção de bases de conhecimento redunda na tarefa mais cara e demorada de um projeto de sistemas especialistas e/ou agentes. Às ontologias, permite-se ainda aos usuários efetuarem consultas, comparações, integração e checagens de consistência (vide figura 9).

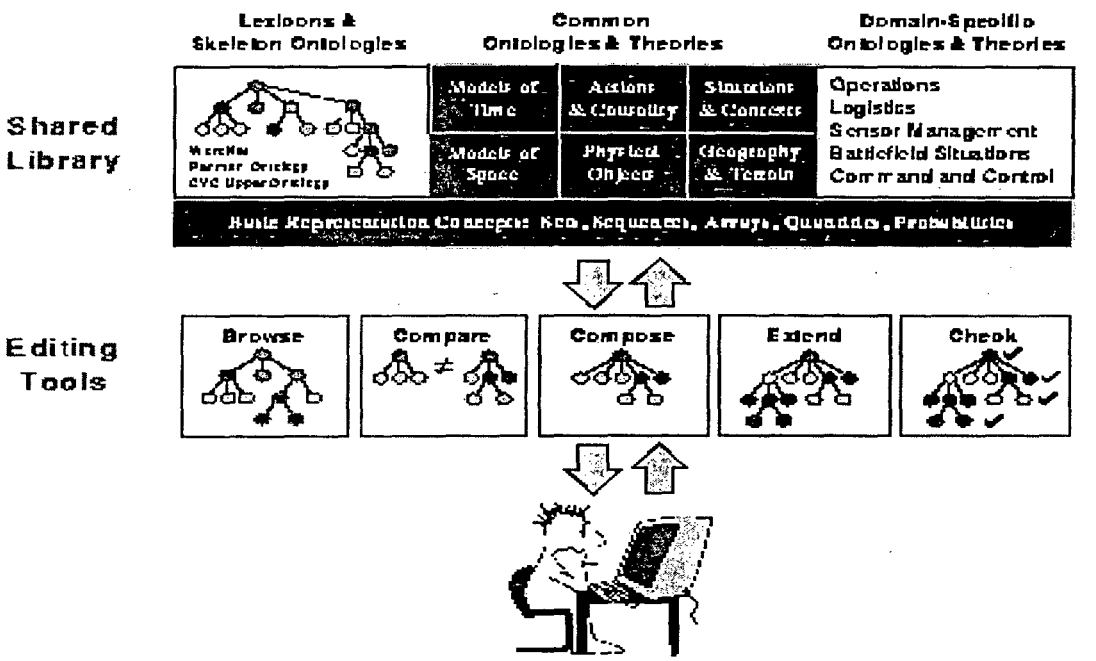

Figura 9. Possibilidades de criação, manutenção e reuso de ontologias através de editores [Fikes 98].

- A disponibilização de uma vasta gama de "ontologias de prateleira", prontas para uso, reuso e comunicação por pessoas e agentes. Hoje as ontologias mais maduras, algumas com mais de 2.000 definições, incluem metadados de imagens de satélites e para integração de bases de dados de genoma, catálogos de produtos, osciloscópios, robótica, semicondutores, terminologia médica, o padrão IEEE para interconexões entre ferramentas, e muitas outras [Farquhar 96].

- A possibilidade de tradução entre os diversos formalismos através de uma *interlingua*, normalmente KIF (*Knowledge Interchange Format*). Esta tradução concretiza um ideal perseguido por gerações de pesquisadores de Inteligência Artificial. A tradução via uma *interlingua* economiza a construção de tradutores *ad hoc*, sendo apenas necessários os tradutores para a *interlingua*. Assim, se são considerados *n* formalismos, o número de tradutores cai de $(n-1)^2$ para $2n$ [Baalen & Fikes 93]. A tradução facilita o reuso de conhecimento, e pode vir a permitir comunicação entre agentes em formalismos diferentes, uma vez que este serviço encontra-se disponível um número cada vez maior de formalismos de representação de conhecimento (atualmente LOOM, CLIPS, Prolog, Epikit, EXPRESS, KIF e a IDL do CORBA - vide Figura 10). Maiores detalhes sobre as ferramentas da Ontolingua e outras são encontrados na seção 4.7 e sobre problemas na tradução na seção 4.10.

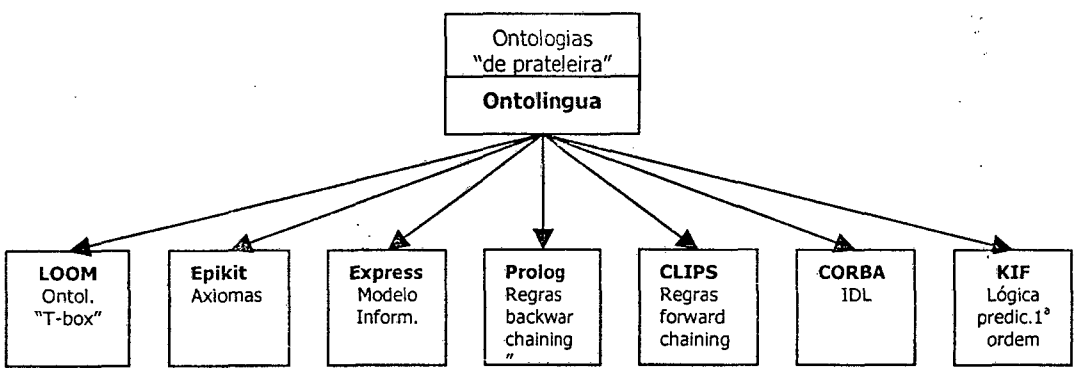

Figura 10. A Ontolingua e os formalismos para os quais podem ser traduzidas as ontologias.

4.6. Ontologias e a Internet

A necessidade de melhores ferramentas de busca para a Web terminou por disseminar o termo "agente" para praticamente qualquer aplicação que se propusesse a manipular ou procurar informação na rede. Nesta trilha, o novo termo da moda, que vem herdando essa popularidade, face às promessas de melhoria destas atividades, é o termo ontologia, uma vez que elas estão presentes em muitos sistemas, ferramentas e produtos de manipulação de informação e comércio eletrônico, representadas como hierarquias de palavras-chave, conceitos, e muitas outras formas, chegando a haver, inclusive, vagas no mercado para desenvolvedores de ontologias, ou "ontologistas" [Clark 99]. Esta seção apresentará um dos três estágios da aplicação de ontologias: e as propostas de linguagens de *markup* com

referências ontológicas, que documentam as páginas com informação semanticamente contextualizada. Os outros estágios são desempenhados pelos editores de ontologias, que ajudam a construí-las (explanados na seção seguinte), e pelas ontologias para a comunicação e cooperação entre agentes (vide capítulo 3).

4.6.1. Ferramentas de Recuperação de Informação para a Web Envolvendo Ontologias

Conforme dito anteriormente, os sistemas abertos, as WANs (*Wide Area Networks*, ou redes de vastas áreas), a Internet, o crescimento da capacidade de conectividade, e a impossibilidade de estruturação de redes cada vez maiores e seus respectivos documentos propiciaram uma revisita às técnicas de Inteligência Artificial, como a alternativa mais factível para um melhor tratamento aos problemas destes ambientes. Basicamente, dois tipos de solução foram propostos, que não são mutuamente exclusivas:

- Dotar os sistemas de inteligência e autonomia para percorrer e selecionar informação relevante na imensidão da rede, o que veio a originar os chamados *agentes inteligentes* (vide seção 3.1.), e,

- Dotar as redes e a própria Internet de inteligência, fazendo com que as páginas possuam uma semântica mais clara e definida, já que as ferramentas que lhe dão suporte, a linguagem HTML *(HyperText Markup Language)* e o protocolo HTTP *(HyperText Transfer Protocol)* preocupam-se, em seu estágio atual, com apresentação e navegação, e por isso só podem ser aproveitados em buscas léxicas e mensuração de proximidade de texto, e não em buscas semânticas com contexto delimitado.

Vale a pena salientar que as ontologias representam um papel fundamental em ambos os tipos: no primeiro como elemento de comunicação de agentes, organização, reuso e disseminação de conhecimento, e no segundo como parte intrínseca das linguagens propostas para a definição de páginas com semântica, como será visto logo a seguir. A arquitetura a ser apresentada neste trabalho enquadra-se no primeiro tipo; esta decisão será justificada, após ser explanado o segundo tipo.

*4.6.2. A Rede Semântica (*Semantic Web*) e suas Propostas*

A visão de uma Internet semântica tem sido apregoada por ninguém menos que um dos idealizadores da Internet, Tim Bernes-Lee. A revista PC Week [PC Week 2000], apresentou uma retrospectiva da Internet, que finaliza com este possível advento. Segundo a reportagem, a primeira geração da rede, a Internet, permitia apenas a troca de dados entre *máquinas* distintas. A segunda geração, a *World Wide Web* (teia de alcance mundial), ou simplesmente Web, provocou uma revolução por disponibilizar uma vasta gama de aplicativos e informação para as *pessoas*, tornando possível, ainda, o comércio eletrônico entre clientes e empresas (no jargão de negócios, *business-to-clients* ou *b2c*). Porém, como as páginas só possuem informação léxica, mesmo os agentes e/ou robôs dotados de inferência encontram um ambiente hostil para a realização de suas tarefas, porque tanto o conteúdo das páginas como o relacionamento entre elas é difícil de ser compreendido por entidades de software, por encontrarem-se, em sua maioria, em linguagem natural. Mesmo as pessoas sofrem com essa falta de semântica: por vezes os usuários possuem dados parciais sobre a informação que procuram e não podem utilizá-los para localizar esta informação.

A próxima geração da rede está sendo batizada de *rede semântica (*ou *Semantic Web)*. Sua maior motivação é transformar os dados e aplicativos em elementos úteis, legíveis e compreensíveis para o *software*, ou, mais exatamente, para os *agentes inteligentes*, de forma a facilitar-lhes a comunicação dinâmica, a cooperação e o comércio eletrônico entre empresas (*business-to-business* ou *b2b*). Segundo Bernes-Lee, "a rede semântica globaliza o hipertexto *conceitual* e a representação de conhecimento" [PC Week 2000]. O Consórcio da Web ou W3C *(World-Wide Web Consortium)* está debruçado sobre um anteprojeto, do qual novas linguagens de navegação, baseadas em conhecimento estruturado em ontologias, serão propostas, a serem definidas com as características mostradas na figura 11.

A exemplo da Ontolingua, as novas linguagens contemplarão vários tipos de lógicas, com mecanismos de intercâmbio (tradução) entre elas, tornando-as compreensíveis aos agentes que raciocinam em formalismos diferentes. São linguagens que estendem a atual HTML, como o SHOE (*Simple HTML Extensions*, ou simples extensões de HTML), e a XML (*eXtensible Markup Language*, uma HTML extensível), e padrões como o RDF (*Resource*

Description Framework, ou ambiente de descrição de recursos), que amarram definições XML a formalismos lógicos, transformando-as realmente em definições de classes, atributos e instâncias de ontologias. Existem ainda ferramentas similares auxiliares, que permitem a definição de ontologias traduzindo-as para definições RDF sobre XML, como a linguagem DAML-O (*DARPA Agent Markup Language* – linguagem de *markup* de agentes dos projetos avançados de defesa americanos) e a camada ontológica OIL (*Ontology Inference Layer*) [Fensel 2000].

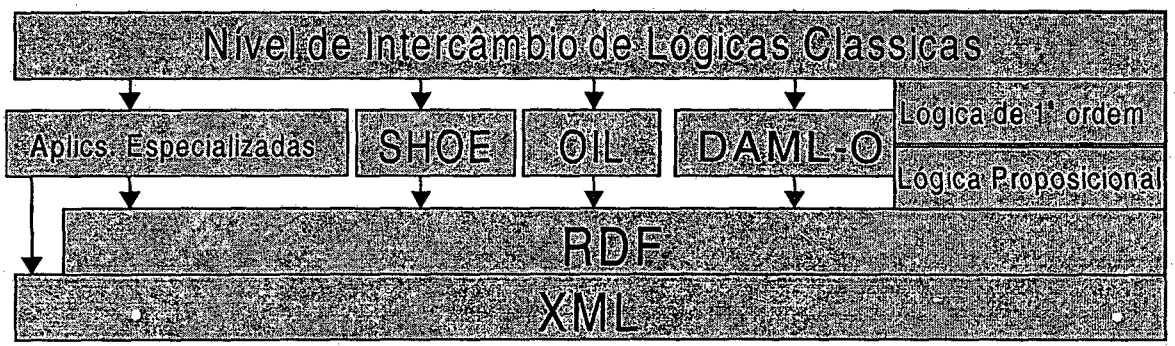

Figura 11. Visão parcial do anteprojeto proposto pelo W3C para uma linguagem de confecção de páginas que inclui semântica.

Para ilustrar os benefícios de uma linguagem para construção de páginas que envolvem ontologias, observe-se a mais simples destas ferramentas, a linguagem SHOE [Luke et al 96], precursora das atuais RDF e XML.

Para descrever as páginas, a linguagem HTML provê *tags* opcionais como título, descrição, sumário e palavras-chave. Com linguagens extensíveis, como a SHOE, as *tags* podem referir-se a contextos descritos por ontologias, e basear-se não apenas em palavras-chave, mas num conjunto de atributos e relações descritos pela ontologia a que se refere a página. Um exemplo de definição de ontologia em SHOE é apresentado a seguir [Luke et al 96]:

```
<ONTOLOGY "our" VERSION="1.0>
      <ONTOLOGY-EXTENDS "organization" VERSION="2.1" PREFIX="org"
            URL="http://www.ont.org/orgont.html">
      <ONTDEF CATEGORY="Person" ISA="org.Thing">
      <ONTDEF RELATION="lastName"ARGS="Person STRING">
      <ONTDEF RELATION="firstName"ARGS="Person STRING">
      <ONTDEF RELATION="marriedTo"ARGS="Person Person">
      <ONTDEF RELATION="employee" ARGS=" org.Organization Person">
</ONTOLOGY>
```

Trata-se de uma ontologia, de nome "*our*",definida por um usuário, que estende a ontologia pré-existente "*organization*", especificando que uma pessoa *(Person)* é uma subclasse da classe de coisas (*Thing*), e que uma pessoa possui quatro atributos: o primeiro e último nomes, que são *strings*, o atributo "*casado com*", cujo conteúdo é preenchido com uma referência à outra pessoa, e o atributo empregador, preenchido com uma referência a uma organização. Como pôde constatar-se, SHOE permite definições de ontologias específicas e gerais em HTML, com heranças entre ontologias, hierarquia de classes do tipo 'é-um', e outros recursos que facilitam a inferência, como estrutura para reflexividade, fechamento transitivo e outras propriedades.

Agora, com a definição de uma pessoa à mão, com seu nome, cônjuge e empregador, são especificados estes atributos na *home-page* de George Cook, cuja esposa, de nome Helena, trabalha na mesma instituição que ele. Em primeiro lugar, a página precisa referenciar a ontologia "*our*" e seu endereço na Web. Isso é realizado através do seguinte código, posto logo no início da *home-page*:

```
<HEAD>
<META HTTP-EQUIV="Instance-Key" CONTENT="http://www.cs.umd.edu/~george">
<USE-ONTOLOGY "our" VERSION="1.0" PREFIX="our"
URL="http://ont.org/our.html">
<BODY>
<CATEGORY="our.Person">
      <RELATION="our.firstName" TO="George">
      <RELATION="our.lastName" TO="Cook">
      <RELATION="our.marriedTo" TO="http://www.cs.umd.edu/~helena">
      <RELATION="our.employee" FROM="http://www.cs.umd.edu/">
</CATEGORY>
```

Logo após a referência, pode-se perceber outro trecho de código que diz que se encontra na página uma instância da classe pessoa, cujo primeiro nome é George, último nome é Cook, e cuja esposa e empregador são especificados por URLs. A instância pode também ocorrer durante o texto, como abaixo:

```
My name is <ATTRIBUTE "our.firstName">
George</ATRIBUTE> <ATTRIBUTE "our.lastName"> Cook </ATRIBUTE>. I live ...
```

A página pode conter dados que não estão nela, como os atributos referentes à esposa de George:

```
<INSTANCE "http://www.cs.umd.edu/~george#HELENA">
      <CATEGORY="our.Person" >
            <RELATION="our.firstName" TO="Helena">
            <RELATION="our.lastName" TO="Cook">
            <RELATION="our.marriedTo" TO="http://www.cs.umd.edu/~George">
</INSTANCE>
```

O conjunto de ferramentas gráficas SHOE abrange editores de ontologias e de páginas, e também o Robô Exposé, que assessora o usuário, através de uma interface gráfica, a construir uma consulta complexa. Uma consulta com dados parciais pode ser montada com muita especificidade empregando conceitos, entidades, atributos e relacionamentos. Uma consulta montada pela interface do robô, procurando páginas de um dos cônjuges de sobrenome "Cook", sabendo que eles trabalham na mesma empresa, é apresentada abaixo:

```
(query (:and
      (#!instanceOf ?X #!Person)
      (#!instanceOf ?Y #!Person)
      (#!instanceOf ?Z #!Organization)
      (#lastName ?X "Cook")
      (#lastName ?Y "Cook")
      (#employee ?Z ?X)
      (#employee ?Z ?Y)
      (#marriedTo ?X ?Y)))
```

Usando um componente de RDF, o RDFS (*RDF Schema*), ontologias podem ser representadas, e assim, pode ser efetuada inferência sobre páginas e consultas mais complexas ainda. Contudo, este trabalho adota uma estratégia baseada em agentes, levando em conta que organizar toda a Internet ontologicamente esbarra em problemas de várias naturezas:

- O usuário comum, que navega na rede e publica páginas, mesmo com interfaces gráficas teria dificuldade de formular consultas que envolvessem regras de lógica e ontologias, e também de lidar com as complexidades em especificar ontologias ou instanciá-las em suas próprias páginas, ainda mais em padrões que se sobrepõem em várias camadas, como o trio OIL-RDF-XML;
- Novos problemas surgiriam, relacionados à veracidade e corretude das especificações contidas nas páginas;

- As próprias páginas possuem, às vezes, conteúdo ambíguo e vago, e isso pode fazer parte da própria informação, seja por seu conteúdo (como, por exemplo, em poesia), ou pela localidade do vocabulário empregado, mantendo o problema da linguagem natural;
- Dificilmente um padrão ontológico para a codificação de páginas, tanto no que se refere à linguagem de *markup*, como de ontologias a referenciar, será adotado pela "rede da liberdade" num espaço curto de tempo;
- Ainda que o fosse, problemas de escalabilidade tanto das ontologias como da indexação ontológica se fariam presentes;
- A adaptação da arquitetura desta proposta frente a uma Internet semântica não invalida e até confirma e facilita a confirmação das principais hipóteses levantadas, como a de que as tarefas de recuperação, categorização e extração devem ser integradas e direcionadas a grupos de classes de páginas, conforme descritos no capítulo 2;
- Além do mais, a maior parte das páginas que já estão publicadas dificilmente será alterada, ou o será dentro de um tempo suficiente para manter um mercado propício exclusivamente aos agentes que processam ontologias dentro de páginas.

Um projeto semelhante ao SHOE, chamado de Ontobroker [Benjamins et al 98], está sendo testado na Universidade de Karlsruhe, na Alemanha. Para especificar os *metadados* de uma página, isto é, as entidades, atributos e relacionamentos contidos nela, foi sugerido pelo projeto o uso de robôs ou agentes com técnicas de aprendizado automático para fazer o trabalho (pesado) de anotar as páginas [Erdmann et al 2000], classificando-as e extraindo delas os metadados que iriam compor a anotação ontológica das páginas na linguagem de *markup* XML.

4.7. Ferramentas para Manuseio de Ontologias

A construção de ontologias para a comunicação em nível de conhecimento entre agentes cognitivos tem se tornado alvo de estudos e propiciou a elaboração de ferramentas com o objetivo de dar suporte a esta atividade. Assim, vários editores de ontologias estão surgindo, com o objetivo não só de facilitar a sua construção, como também de disponibilizar ontologias públicas para reuso e extensão, integrando diferentes grupos de pesquisa, amiúde distantes geograficamente, que pesquisam sobre as mesmas áreas ou

áreas afins. O Ontosaurus [Neches 91], por exemplo, congrega a linguagem de representação de conhecimento LOOM, com uma interface gráfica via um navegador, permitindo a colaboração de diferentes pesquisadores no desenvolvimento de conhecimento para, por exemplo, planejamento de campanhas militares aéreas [Valente et al 99].

Alguns requisitos tornam uma ferramenta popular: facilidades de acesso e conexão a repositórios, portabilidade e ferramentas de apoio que facilitem o desenvolvimento.

4.7.1. O Editor da Ontolingua

A Ontolingua converteu-se no primeiro editor de ontologias a angariar popularidade, por dispor de um leque de funcionalidades amplo. A ferramenta disponibiliza uma interface de acesso através de navegadores, possibilitando a qualquer usuário da Web a visualização das ontologias criadas e disponíveis, a criação de novas e o trabalho colaborativo. A arquitetura do servidor Ontolingua está representada na Figura 12. Os colaboradores remotos lêem e contribuem com novos conhecimentos para a biblioteca digital de ontologias em texto, mantida em HTML e acessível através de navegadores da Internet. As aplicações podem usar as ontologias de duas maneiras: fazendo consultas ou atualizações usando o *NGFP*, uma API *(Application Programming Interface)* que extende o *Protocolo de Frames Genéricas (GFP),* que, por sua vez, implementa a troca de conhecimento em redes expresso em *frames* de forma otimizada [Karp et al 95], ou mandando e recebendo conhecimento das bases através da linguagem KIF (*Knowledge Interchange Format*).

Além das funcionalidades - como um gerador gráfico de ontologias -, a Ontolingua destaca-se pelos critérios cuidadosos com que foi projetada. Por exemplo, a Ontolingua distingue com bastante propriedade os níveis de conhecimento (ou epistemológico), o nível lógico e o de implementação, criando ainda o nível de apresentação. No nível de conhecimento está o conhecimento propriamente dito em formato texto; o nível lógico pode ser visto como o KIF, pois representa o conhecimento num formalismo, sem preocupação como ele será implementado computacionalmente para os processos de inferência; no nível de implementação, a ontologia é representada num determinado formalismo, como CLIPS ou Prolog; e o nível de apresentação resolve conflitos de conceitos homônimos de áreas distintas, que podem causar problemas quando há

interconexões entre estas áreas. O nível de apresentação mantém isto transparente ao usuário, dando a real impressão que o sistema "entende" a diferença conceitual entre eles [Farquhar 96]. Residem nesta característica as raízes do *engajamento ontológico* (vide subseção 4.3.); na realidade, a forma ideal de interação com os sistemas deve ocorrer no nível de conhecimento e não no de implementação.

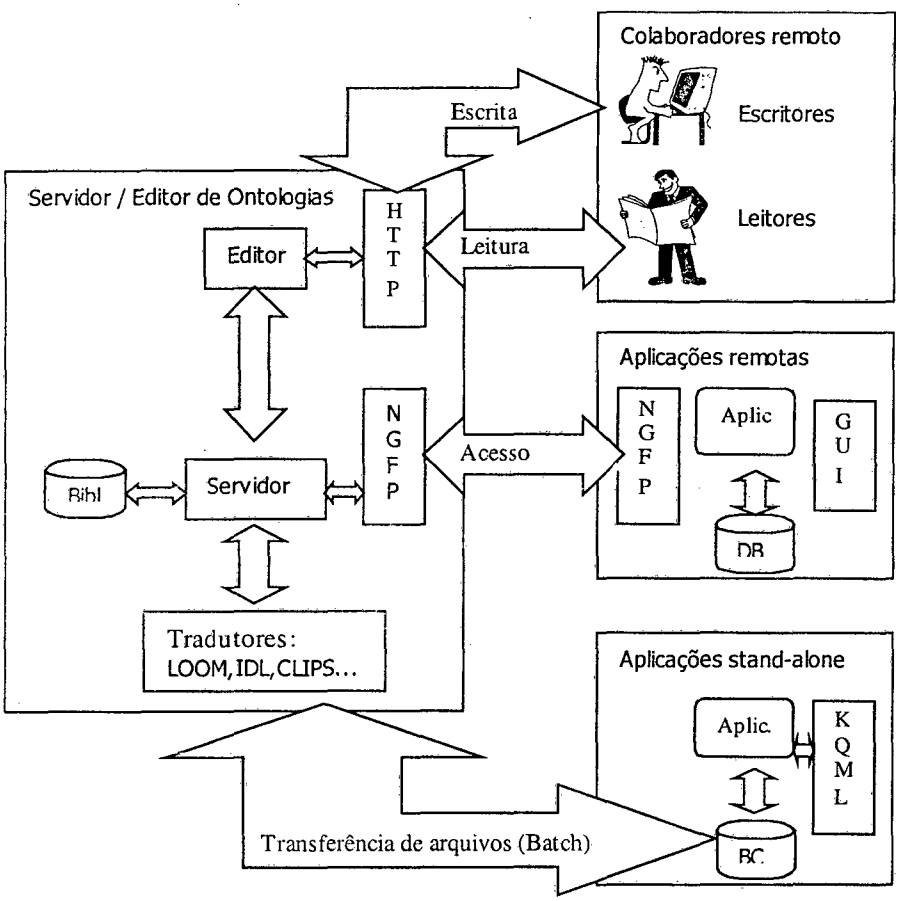

Figura 12. Arquitetura geral do servidor Ontolingua [Farquhar 96].

Mesmo fornecendo boas interfaces gráficas para navegadores, falta à Ontolingua uma boa interface para estações de trabalho e computadores, que permitam a usuários, entender, reusar, manusear e verificar ontologias, sem estar necessariamente acessando a Internet. Para permitir definições muito completas, qualquer ontologia criada na Ontolingua deve incluir e referenciar uma ontologia complexa para lidar com *frames*, a *Frame-Ontology* [Gruber 95], e o código gerado ou traduzido pode tornar-se complexo devido às constantes referências às classes desta ontologia. Este erro de requisitos no projeto (a falta de interfaces simples), aliados á complexidade das definições geradas, impediu que a

Ontolingua se difundisse e abriu mercado para o surgimento de ferramentas que sanassem este problema.

4.7.2. O Ambiente Protégé-2000

Na esteira destes acontecimentos, o ambiente Protégé, que vinha sendo desenvolvido pelo Departamento de Informática Médica da Universidade de Stanford face às necessidade de ontologias médicas desde os anos 80, ocupou o nicho de mercado aberto pela Ontolingua. O Protégé foi projetado de forma a maximizar a sua adaptação aos mais diversos usos. Suas características principais são [Noy et al 2000]:

- Um modelo de conhecimento extensível: é possível redefinir declarativamente as classes primitivas de um sistema de representação (e.g., um *slot* de um *frame* é uma classe do sistema, ou *metaclasse*, as facetas são atributos da classe, e a metaclasse :SLOT é empregada nas definições de outras classes). A linguagem axiomática PAL (*Protégé Axiomatic Language*) foi elaborada desta forma, e permite a inserção de restrições e axiomas que incidem sobre as classes e instâncias de uma ou mais ontologias.

- Gera arquivos de saída alteráveis, permitindo que sejam implementados componentes para traduzir o conhecimento para outros formalismos. Atualmente podem ser criados classes e instâncias em CLIPS - a base de conhecimento é gerada nativamente para esse motor de inferência, o mais popular quando se iniciou a construção do Protégé, nos anos 80 -, Jess, F-Logic, Prolog, RDF, OIL, XML, Topic Maps (linguagem padrão ISO para descrever estruturas de conhecimento em páginas da Web, nos mesmos moldes de RDF e OIL) – todos estes últimos através de componentes específicos. A saída pode ser armazenada ainda em outros formatos, como bancos de dados relacionais[17].

- Uma excelente interface para entrada de conhecimento, pois inclui um gerador automático de formulários para as classes definidas (vide figura 13), admitindo ainda a reposição da interface original por componentes mais adequados à aplicações

---

[17] As tabelas geradas não são normalizadas, já que *frames* ferem já a 1a. forma normal, pois *slots*, diferentemente de campos de tabelas, podem ser multivalorados [Bertino et al 2001].

específicas. Esta interface facilita sobremaneira o gerenciamento de conhecimento de uma ou mais ontologias.

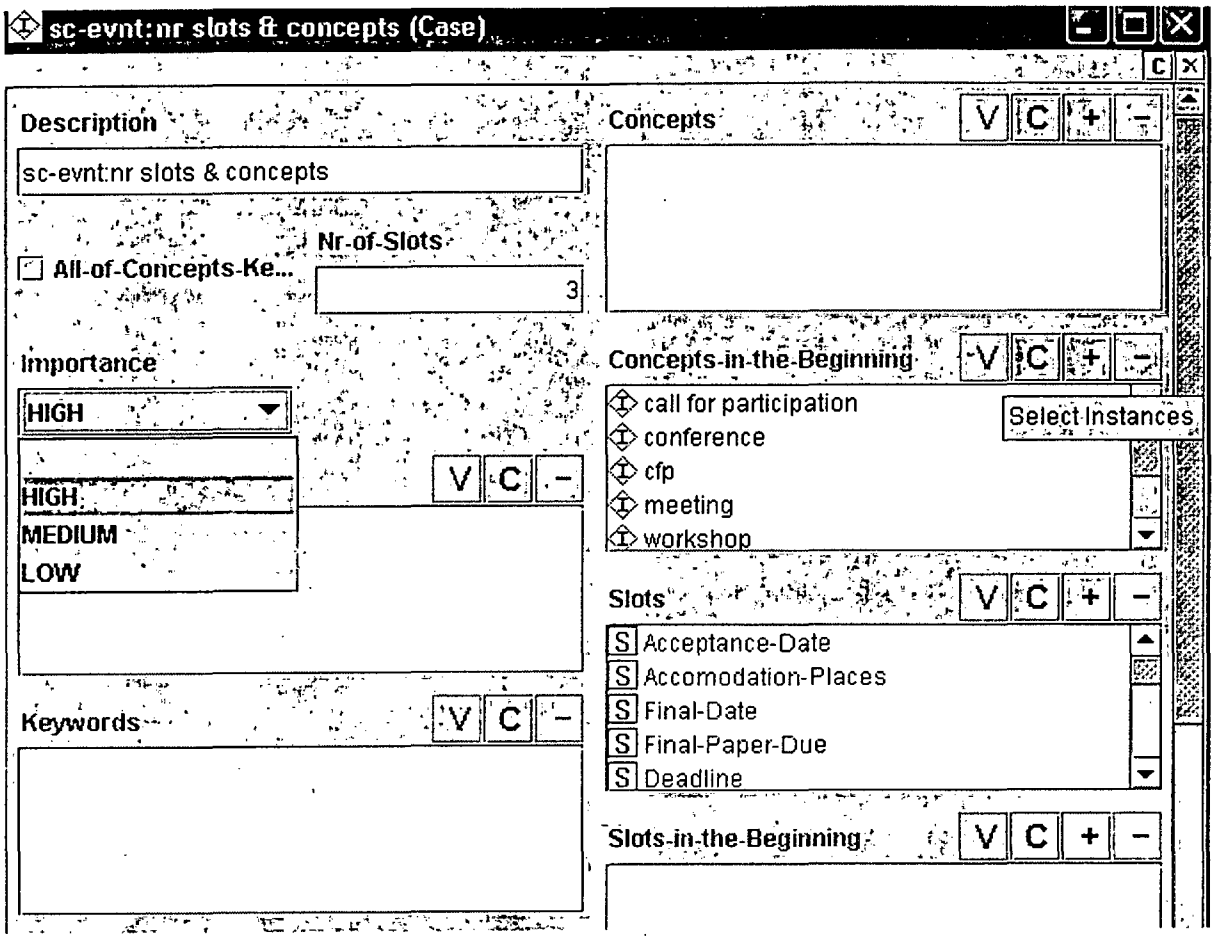

Figura 13. Formulário automático para entrada de dados gerado pelo Protégé.

- Uma arquitetura integrável a diversas aplicações, via componentes que podem ser conectados ao sistema. Este requisito, a adaptabilidade à integração de novos módulos ao sistema, trouxe muitos benefícios em termos de engenharia de software. Como conseqüência de sua difusão, componentes de vários matizes, elaborados por grupos de pesquisa usuários do ambiente, puderam ser adicionados ao sistema, sem necessitar o redesenvolvimento, entre os quais o Jambalaya, um utilitário com animação e vários outros recursos em visualização de dados, e o Ontoviz, um componente que faz com que o gerador de gráficos Graphviz da AT &T produza gráficos com instâncias, heranças e outros tipos de relacionamento. Alguns exemplos de gráficos encontram-se nas próximas páginas.

Do ponto de vista de integração de ontologias, o Protégé oferece várias alternativas. Existem componentes que integram grandes ontologias como o WordNet – um dicionário semântico da língua inglesa -, o GATE (*General Architecture for Text Extraction*, em português, arquitetura geral para extração de texto, um conjunto de ferramentas de linguagem natural para o inglês confeccionadas pela Universidade de Sheffield, Inglaterra), o Cyc e a UMLS (uma volumosa ontologia médica). Está disponível, também, um repositório de ontologias[18] com ontologias médicas, a ontologia de metadados Dublin Core que visa definir ontologias comuns para a criação de páginas da Internet. Por fim, através de um componente-cliente que conecta servidores OKBC, todas as ontologias da Ontolingua podem ser acessadas e baixadas em todo ou em parte diretamente para o ambiente Protégé da máquina-cliente.

4.8. Princípios para a Construção de Ontologias Reusáveis

A concepção de ontologias deve ser conduzida como qualquer outro projeto de software, no sentido de serem tomadas decisões de projeto que determinam sua qualidade baseada em critérios como eficiência, legibilidade, portabilidade, extensibilidade, interoperabilidade e reuso. Por isso, deve basear-se em seu futuro emprego, e não em aspectos filosóficos do conhecimento acerca do domínio representado. Alguns princípios, se usados com precisão, garantem sua qualidade:

- *Clareza*: Os programas usam diferentes modelos e abstrações na resolução de seus problemas. Na definição do conhecimento, deve-se ter a objetividade de definir apenas o que se presume ser útil na resolução da classe de problemas a ser atingida. Definições completas, com condições necessárias e suficientes, devem ter precedência sobre definições parciais.

- *Legibilidade*: As definições devem guardar correspondência com as definições correntes e informais. A ontologia deve usar um vocabulário compartilhável – normalmente o jargão e terminologia usados por especialistas do domínio.

- *Coerência*: As inferências derivadas da ontologia definida devem ser corretas e consistentes do ponto de vista formal e informal com as definições.

---

[18] Criado a partir de sugestão do autor desta tese, este repositório teve como primeira ontologia a ontologia Ciência, explanada na seção seguinte.

- *Extensibilidade*: A ontologia deve permitir extensões e especializações monotonicamente e com coerência, sem a necessidade de *revisão de teoria*, que consiste na revisão lógica automática de uma base de conhecimento em busca de contradições.

- *Mínima codificação*: Devem ser especificados conceitos genéricos – essa genericidade limitada pela clareza – independente de padrões estabelecidos para mensuração, notação e codificação, garantindo a extensibilidade.

- *Mínimo compromisso ontológico*: Para maximizar a reusabilidade, apenas o conhecimento essencial deve ser incluído, gerando a menor teoria possível acerca de cada conceito, permitindo a criação de conceitos novos e mais especializados ou estendidos.

Esses conceitos podem ser evidenciados através do seguinte exemplo da ontologia para o domíno científico [Freitas 2001], gerada neste trabalho a partir do reuso com vários refinamentos da ontologia do projeto europeu $(KA)^2$ (*Knowledge Annotation Initiative of the Knowledge Acquisition Community* – Iniciativa de Anotação de Conhecimento da Comunidade de Aquisição de Conhecimento) [Benjamins et al 98] (vide seção seguinte). Este projeto teve por objetivo criar uma organização virtual de pesquisadores, universidades, projetos e publicações, entre outros itens, envolvidos com a subárea de Inteligência Artificial conhecida como Aquisição de Conhecimento. Nesta ontologia pode-se verificar o cuidado com que os autores conceberam os conceitos de forma a representar apenas os essenciais, guardando uma correspondência direta - engajamento ontológico - com o domínio e terminologia existentes, satisfazendo os requisitos *clareza* e *legibilidade*.

Pela definição da classe Documento Científico abaixo, na figura 14[19], pode-se perceber que não há restrições desnecessárias, ou seja, especificou-se exclusivamente o conhecimento do conceito sem definir prematuramente certas decisões (e.g., definir que subclasse da classe Pessoa pode ser autor de artigos, muito embora a maioria deles seja da subclasse Pesquisador), garantindo *mínimo compromisso ontológico*.

Por outro lado, a ontologia do $(KA)^2$ foi alterada para fazê-la ganhar *extensibilidade*, de forma a que novas classes pudessem ser definidas a partir das já existentes. Como mostra a

---

[19] Nessa e nas figuras 17 e 18, o uso do asterisco (*) após a definição do tipo de um atributo assinala que ele pode assumir múltiplos valores, e não um só como os outros atributos.

figura 15, foram criadas as classes abstratas (sem instâncias) Evento Científico ao Vivo (*Live-Scientific-Event*) - com subclasses (não mostradas na figura) Conferência e Workshop – e Evento de Publicação Científica (*Scientific-Publication-Event*), com subclasses Jornal, Revista (*Magazine*) e Evento de Publicação Científica Genérica (*Generic-Scientific-Publication-Event*).

Scientific-Document		
Publication-Year	Instance	Year
Authors	Instance*	Person
Keywords	String*	
URL-of-Publication	String	
Publication-Title	String	
...		

Figura 14. Parte dos atributos da Classe Documento Científico.

Em relação à *coerência*, foi percebido que a relação meronímica (ou parte-todo) entre artigos de um *proceedings*, ou entre capítulos de um livro, não estava explicitada. Dado que seria importante diferenciar artigos e capítulos e, por outro lado, evidenciar que um determinado conjunto deles pertence a um mesmo livro ou *proceedings*, foram criadas as classes Publicação-Divisível (*Dividable-Publication*) - com as subclasses Livro (*Book*), *Proceedings* e Evento de Publicação Científica (*Scientific-Publication-Event*) - e Publicação-Parte, com subclasses (não mostradas na figura) como Artigo de Workshop, Artigo de Conferência, Capítulo de Livro, etc. Aliás, a definição de relações parte-todo requer cuidados, contituindo-se tópico de pesquisa ativa na área de ontologias [Staab & Maedche 2000] [Artale et al 95], pois existem vários tipos de partições que devem gerar inferências distintas, como componente-objeto (ramo/árvore), membro-coleção (árvore/floresta), porção-massa (fatia/bolo), matéria-objeto (alumínio/avião), característica-atividade (pagar/comprar), lugar-área (cidade/estado), fase-processo (adolescente/crescimento) [Noy 97].

Observe-se a precisão semântica causada pelo uso da herança múltipla na classe Evento de Publicação Científica: Jornais e Revistas ao mesmo tempo em que são Publicações Divisíveis, guardam também características (e, portanto, herdam atributos) de eventos, como comitês de programa, datas limite, e outros atributos, embora não ocorram ao vivo.

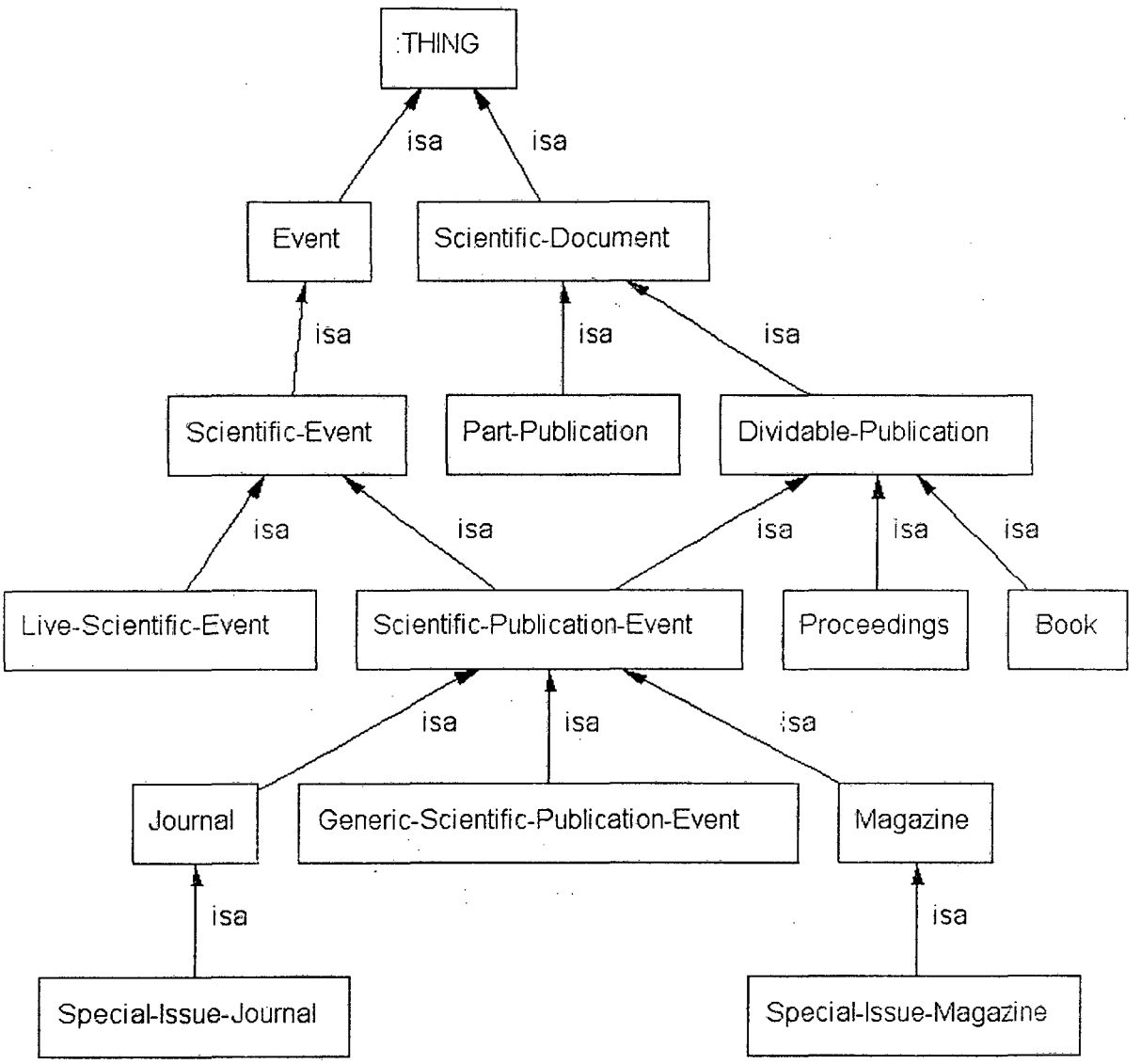

Figura 15. Parte da ontologia Ciência [Freitas 2001], salientando as classes Evento e Publicação, suas subclasses e heranças.

4.8.1. Ganhos com definições ontológicas

O reuso de conhecimento estruturado, com a aplicação de vocabulário e terminologia empregados na área de conhecimento da ontologia, com certeza, terá forte impacto também sobre Engenharia de Software. As técnicas convencionais de projeto de sistemas de informação pouco atacaram problemas de conversão de dados (datas, unidades de medida, etc), e de semântica da informação (Alberto Caeiro e Fernando Pessoa são autores diferentes para sistemas de biblioteca, quando na realidade trata-se da mesma pessoa).

O primeiro problema deve-se a uma desnecessária codificação dos dados (uma data ou unidade não é um simples número, mas um conceito conversível para diversas unidades),

que provoca uma redundante construção de rotinas de validação e conversão, além de tornar os programas inflexíveis e sem robustez com relação a unidades de mensuração existentes e não previstas pelos programas, o que é crítico em sistemas para o mercado globalizado. No fundo, o que causa ambos os problemas é a falta de *engajamento ontológico*, ou seja, os sistemas não aproveitam o conhecimento existente, e, se o fizessem, deveriam mantê-lo de forma declarativa, ao invés de procedural, para, portanto, tornarem-se flexíveis aos avanços científicos e às novas padronizações. As ontologias, se expressas de acordo com os princípios enumerados acima, não padecem desses problemas e abrangem cada vez mais áreas de conhecimento (vide próxima seção).

À parte disso, o código declarativo gerado a partir de uma ontologia, por ter correspondência direta com a teoria e terminologia do domínio, possui clareza e legibilidade. A relação custo-benefício da construção de ontologias é muito alta: trata-se de conhecimento, não código, utilizável até pelos usuários. Ademais, a construção de ontologias estimula a confecção de mediadores e facilitadores que seriam os responsáveis por prover os serviços relativos ao conhecimento a outras entidades de software, fornecendo a semântica e funcionando como um especialista daquele conhecimento, respondendo a consultas utilizando a inferência sobre o conhecimento de que dispõe.

4.9. Ontologia do domínio Ciência

A ontologia do domíno científico empregada neste trabalho foi reusada a partir da ontologia do projeto europeu $(KA)^2$. Esta ontologia estava disponível no sítio espelho da Ontolingua em Madri[20], porém não podia ser acessada diretamente pelo Protégé porque o sítio não dispunha de um servidor OKBC. Hoje ela poderia ser diretamente reusada, mas, à época, foi necessário redigitar a ontologia no Protégé-2000, constando ainda as alterações descritas na seção anterior. A ontologia Ciência [Freitas 2001] atualmente encontra-se disponível no repositório de ontologias do Protégé, já tendo inclusive sido reusada por analistas da Mercedes-Benz de Stuttgart, Alemanha.

---

[20] Trata-se de ontologias públicas, definidas pelos grupos de pesquisa de Sistemas Baseados em Conhecimento das Universidades de Karlsruhe (Alemanha), Madri (Espanha), Amsterdã (Holanda), e Stanford (Estados Unidos), e que podem ser folheadas no endereço eletrônico *http://www-ksl-svc-lia.dia.fi.upm.es:5915*, sob o login "*ontologias-ka2*" e senha "*adieu007*" [Benjamins et al 98].

A figura 16 mostra as subclasses e atributos da classe Organização da ontologia Ciência, mostrando os relacionamentos entre elas, neste caso, apenas do tipo herança (como o fato de que a classe *Research-Institute* ou Instituto de Pesquisa é uma subclasse da classe Organização) ou parte-todo (como o fato de que uma empresa ou universidade têm como partes departamentos). Após o nome da classe na primeira linha de cada retângulo que representa uma classe, as linhas seguintes listam os atributos de cada classeque sâo instâncias de outras classes, citadas logo a seguir.

Já a figura 17 apresenta uma gama maior de relacionamentos, mostrando como se entrelaçam as classes principais da ontologia, incluindo também classes da ontologia auxiliar de tempo (vide subseção seguinte), como Mês (*Month*) e Ano (*Year*). Constata-se, por exemplo, que os membros de um comitê de programa da classe Evento são instâncias da classe *Researcher* (Pesquisador), através do atributo *member-Of-Program-Committee.*

Ambas as figuras foram geradas com o auxílio do componente Ontoviz do sistema Protégé.

4.9.1. Ontologias Auxiliares

A ontologia Ciência converteu-se na ontologia principal do sistema, comum a todos os agentes, e necessitou de ontologias auxiliares de tempo, locais e turismo, esta última englobando definições de moedas correntes de países, tipos de acomodações, enfim, definições úteis para a extração de atruibutos da classe Evento ao Vivo. Foi feita a opção de criar ontologias auxiliares simples, mesmo porque o componente OKBC para o Protégé ainda não havia sido disponibilizado, impedindo o reuso de ontologias de tempo e lugar disponíveis no repositório da Ontolingua. Outras ontologias estratégicas ou operacionais, relativos ao processo de recuperação, classificação e extração, serão apresentadas nos dois próximos capítulos.

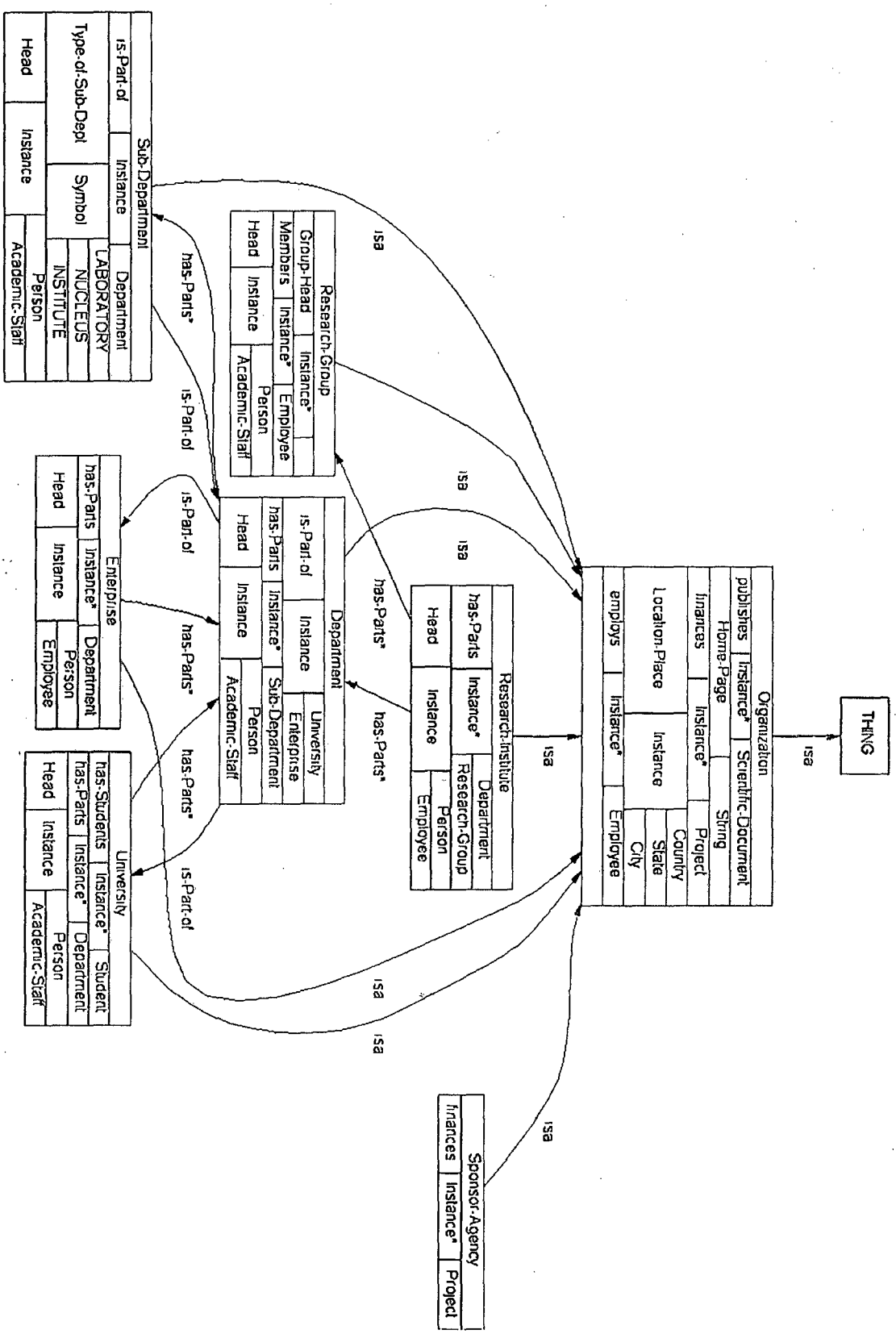

Figura 16. Classe Organização, da ontologia Ciência, e suas subclasses, mostrando atributos e relacionamentos.

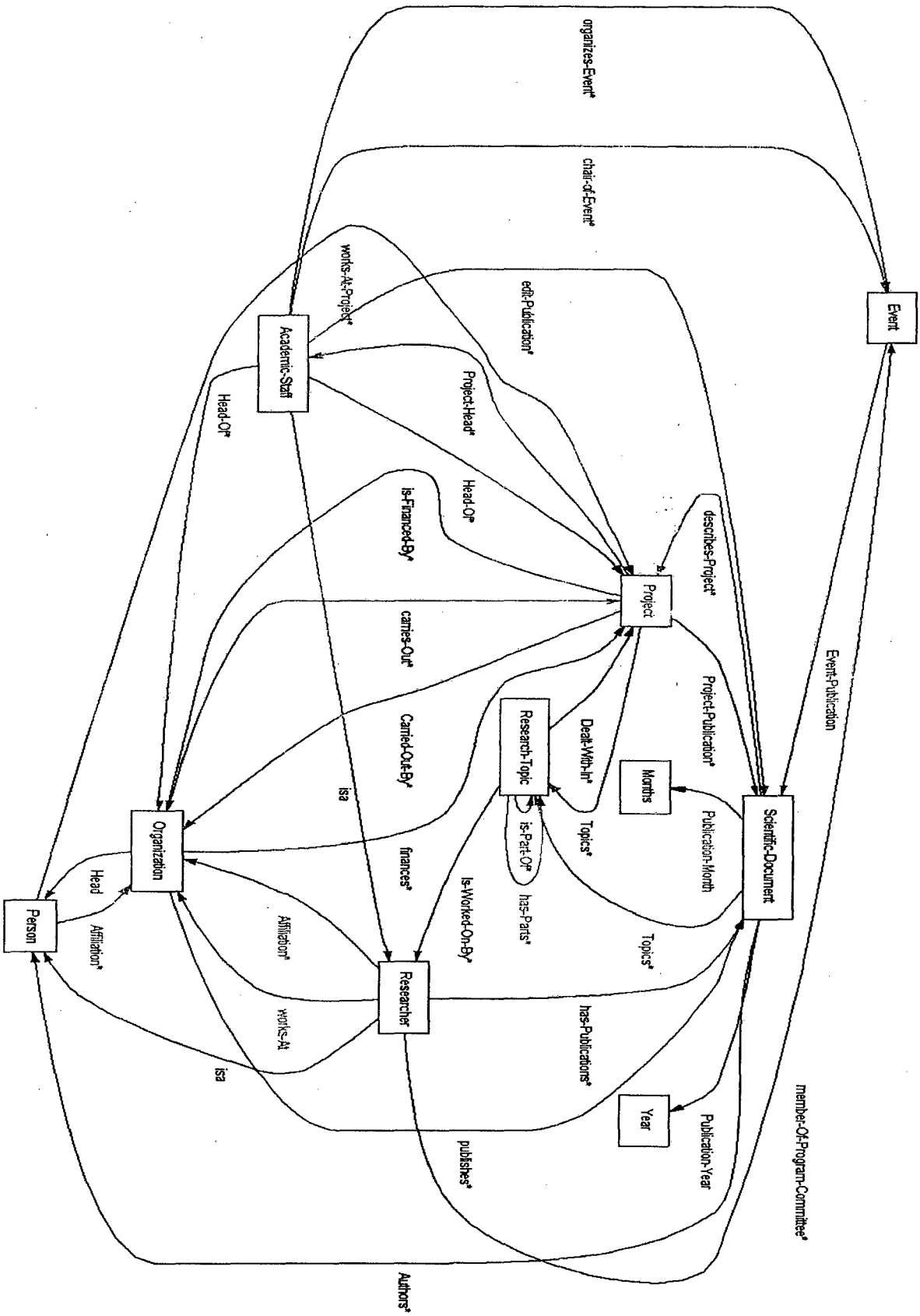

Figura 17. Relacionamentos entre as principais classes da ontologia Ciência.

4.10. Tópicos Abertos de Pesquisa em Ontologias

Devido à sua popularização, o crescimento de interesse, e conseqüente pesquisa sobre ontologias é patente; por tratar-se de uma área relativamente nova em informática, inúmeros tópicos ainda aguardam solução.

Apesar dos princípios de construção enumerados nesta seção, ainda não existem ferramentas de *metodologia*[21] que guiem uma elaboração padronizada; em conseqüência disso, não há métodos de validação, verificação, desenvolvimento, [Gómez-Pérez 94] e mesmo documentação de ontologias, que muito facilitariam seu reuso e aplicação. Os primeiros protótipos para comparação e união de ontologias começam a despontar [McGuinness et al 2000].

Ligada a estes problemas relativos à concepção de ontologias, uma questão parece ser crucial, e que soa como um paradoxo: os princípios de construção apresentam uma contradição, pois afirmam que uma ontologia deve ser projetada visando sua aplicação, e, ao mesmo tempo, ser extensível, representando as entidades dos domínios com forte engajamento ontológico. Na verdade, esse paradoxo revela-se presente na maior parte dos tópicos abertos de pesquisa sobre ontologias, e diz respeito à tensão entre sua aplicabilidade funcional (que deve restringi-las) e extensibilidade (que deve expandi-las, incluindo o máximo de conhecimento acerca do domínio). Outras facetas deste problema:

- Até que ponto as ontologias devem refletir as peculiaridades dos métodos de inferência sobre os quais será usado o conhecimento contido nelas? Vale lembrar que mesmo a Ontolingua possui uma ontologia de topo sobre a qual estão definidas propriedades, relacionamentos e outras características.
- Os projetos de ontologias devem abordá-las de acordo com suas entidades (visão real) [Guarino 97] ou sob o prisma dos serviços que as entidades disponibilizam (visão funcional) [Visser & Cui 98]?

---

[21] Para esta observação, é fundamental ter em mente que "as definições contidas em ontologias são mais gerais do que as de uma base de conhecimento", já que "ontologias devem ser escritas numa linguagem expressiva, declarativa, portável, independente de domínio, semanticamente bem-definida e processável" [Gómez-Pérez 94]. Existem metodologias para elaboração de bases de conhecimento, especialmente o conhecimento estratégico – o conhecimento responsável pelas inferências em direção à solução de um problema -, entre as quais a que mais se disseminou é conhecida como CommonKADS [Screiber et al 94].

Persistem, também, problemas relativos à tradução. Existem diferenças de expressividade entre os formalismos, que dificultam a tradução [Baalen & Fikes 93], por exemplo, F-logic [Kifer et el 95] possui restrições para facetas de cardinalidade máxima e mínima [Corcho & Gómez-Pérez 99]. Também os diferentes tipos de inferências e premissas das ontologias "de topo" de muitos sistemas (vide figura 18) normalmente estão refletidas no conhecimento definido, de forma que essas ontologias "de topo" deveriam ser mapeadas entre si para permitir uma tradução e reuso mais seguros [Valente et al 99].

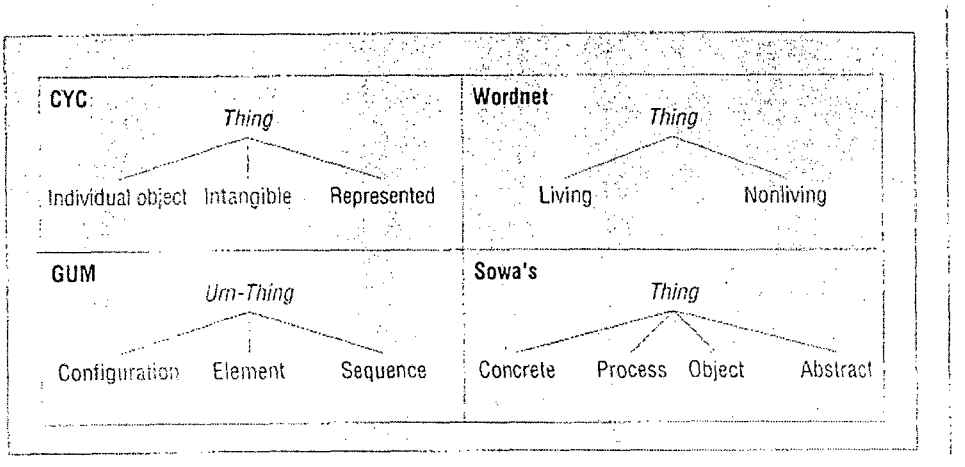

Figura 18. Diferentes estruturas de ontologias "de topo". [Chandrasekaram 99]

4.11. Conclusões

Foram levantados argumentos para o uso de estruturas declarativas, como maior poder de abstração, engajamento ontológico – que confere semântica ao conhecimento -, e o problema da linguagem – lógica, em oposição a código de uma linguagem imperativa, que é uma forma mais direta e natural de se especificar conhecimento.

Saliente-se o advento das *ontologias reusáveis*, que podem ser traduzidas para outros formalismos de representação de conhecimento e transportadas em nível de conhecimento, bem como os princípios para a construção dessas ontologias de forma a proporcionar forte extensibilidade e reuso. Uma ontologia do domínio científico foi reusada e refinada, atualmente disponível para reuso no repositório do Protégé, e com isso, um sistema multiagente começa a ser delineado a partir de seu vocabulário de comunicação.

As ontologias já começam a desempenhar o papel de conhecimento estruturado disponível para reuso em larga escala por sistemas e programas, um acontecimento sem paralelo na

história da ciência da computação. Isso representa para a história da informática o que, para a história humana, representou a criação de escolas, enciclopédias e bibliotecas. Enfim, o armazenamento de conhecimento, já que o conhecimento agora trafega entre computadores e sistemas que podem lançar mão deles, manipulá-los e aplicá-los no cumprimento de suas funções. As ontologias também têm servido, em certas áreas, para unificar o conhecimento e para formar consensos sobre certos conceitos, causando uma integração de grupos de pesquisas e sendo utilizadas, inclusive, com propósitos educativos.

Capítulo 5

ARQUITETURA DE SISTEMAS MULTIAGENTES PARA MANIPULAÇÃO INTEGRADA DE INFORMAÇÃO DA WEB

5.1. Visão Geral

Um dos propósitos principais deste trabalho consiste em promover a cooperação, tanto entre as tarefas de extração, categorização e recuperação, como entre agentes de um grupo de classes de páginas, de forma a que a cooperação resulte compensatória para o processamento das classes de páginas. Em ambos os casos, o fator precisão determinará a consistência da cooperação, se os relacionamentos entre as classes se mostrarem úteis - ou, em outras palavras, se as recomendações dos agentes a outros agentes exibirem uma precisão mais alta do que os resultados das consultas aos mecanismos de busca, contendo menos lixo e facilitando a procura de páginas-membro das classes dos grupos procurados. Também a hipótese dos grupos funcionais pode revelar-se preciosa nesse sentido: a classe das listas, por exemplo, se identificada apropriadamente, conduz diretamente a um grande número de páginas-membro das classes, corroborando as hipóteses da abordagem de extração integrada e de complementaridade entre as tarefas de extração, categorização e recuperação e a de cooperação entre agentes distintos dentro de um mesmo grupo.

Será apresentada a seguir uma arquitetura de Sistemas Multiagentes Cognitivos para resolver o problema da extração integrada de entidades pertencentes às classes que integram um grupo de classes de páginas ou *cluster* [Freitas & Bittencourt 2000]. A motivação principal para o emprego de sistemas multiagentes é beneficiar-se dos relacionamentos entre as classes. A concepção da arquitetura teve como pilar o princípio de torná-la o mais reusável possível, tanto a nível *macro*, permitindo a sua aplicação em outras regiões e domínios da Web, além do científico, quanto a nível *micro*, habilitando os agentes aos vários reusos descritos na seção 5.6. Em relação ao nível *macro*, a idéia é que

qualquer grupo ou região formada por classes de páginas inter-relacionadas possa beneficiar-se do trabalho. A visão geral da arquitetura está ilustrada na figura 19.

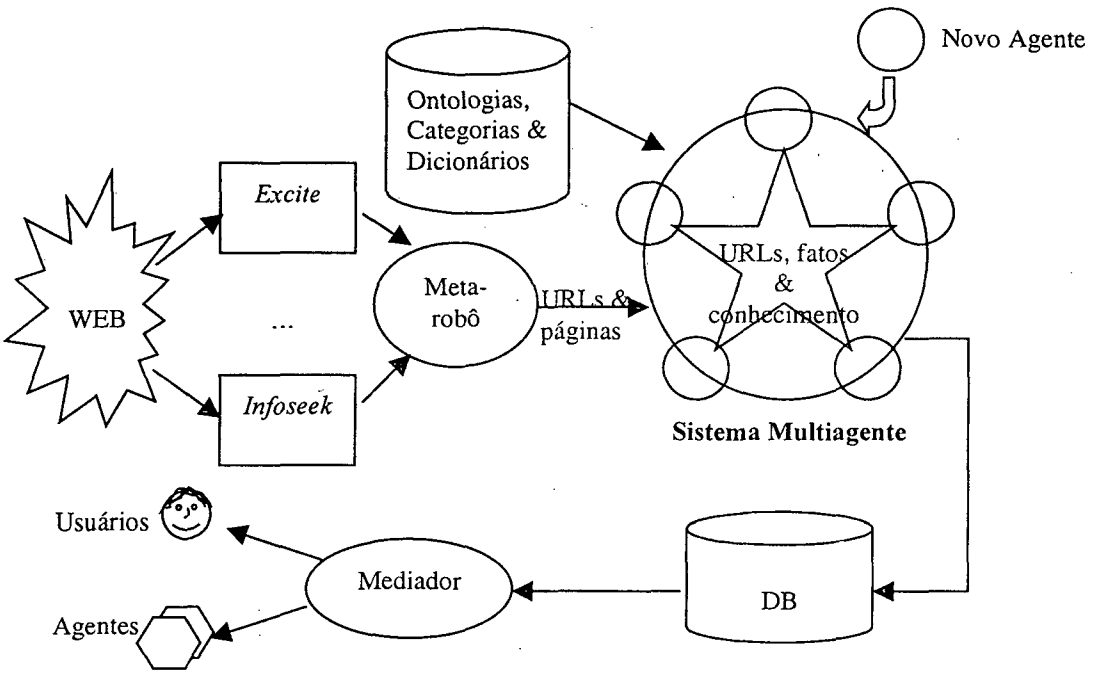

Figura 19. Arquitetura de um sistema multiagente cognitivo para extração integrada de dados da Internet.

Cada agente é um especialista no reconhecimento de páginas que correspondem a instâncias da classe que ele processa (por exemplo, páginas de pesquisadores, chamadas de eventos científicos - "*Call for Papers*" -, artigos e outras do grupo científico) e na extração de atributos dessa entidade (por exemplo, áreas de pesquisa e instituição de pesquisadores), procurando também identificar páginas e ponteiros úteis a outros agentes. Uma vez que os agentes possuirão responsabilidades distintas, praticamente sem interseção, cooperando uns com os outros pela troca de sugestões de endereços de páginas candidatas a membros das classes que processam, a arquitetura baseia-se na abordagem de Resolução Distribuída de Problemas (RDP), como apregoado por Oates et alii [Oates et al 94]. A estrela nos multiagentes indica troca de mensagens contendo regras de reconhecimento e fatos (conhecimento dos agentes), além das URLs (dados).

A granularidade da classe de páginas a ser processada por um agente, ou seja, se um agente deve processar a classe Documento-Científico ou Publicação-Parte, depende da similaridade entre os padrões de suas classes. Se os padrões diferem muito, com certeza uma solução mais eficaz é separar as subclasses entre agentes distintos.

A qualquer momento, um novo agente pode ser introduzido no sistema, e os agentes já existentes trocarão mensagens com ele, de forma a estabelecerem uma cooperação no desempenho de suas tarefas individuais (maiores detalhes na seção seguinte). Usuários e agentes externos podem acessar os dados extraídos pelos agentes através de um mediador [Freitas et al 99]. As subseções seguintes descrevem os elementos periféricos dos agentes delineados na figura 19, ou seja, o meta-robô - que consulta vários mecanismos de busca procurando *links* úteis -, as fontes de informação (ontologias, categorias e dicionários) e o mediador.

5.1.1. Meta-robô

Uma questão relevante no desenvolvimento de sistemas para a Internet consiste na construção de robôs coletores. Todos os mecanismos de busca usam robôs para coletarem os documentos e os representarem com palavras-chave e suas respectivas freqüências em bases de índices, que servem para responder aos usuários quais documentos são mais relevantes às suas consultas. A proliferação destes robôs ameaçava inviabilizar o tráfego na rede e sobrecarregar os servidores durante a coleta das páginas para os índices. Para sanar o problema, foram criadas convenções para robôs "bem-comportados" [Koster 93], que, entre outras características:

- Aproveitam índices de outros mecanismos de busca e serviços de outros robôs, evitando redundância de esforços,
- Alternam vários servidores, evitando sobrecarga,
- Processam apenas os dados que interessam (tipos de arquivos, data dos arquivos, etc),
- Sabem fugir de *loops* na procura de ponteiros, e de ponteiros repetidos,
- Possam ser controlados interativamente,
- Mantenham um *log* ou base de dados pública com estatísticas de sucesso, facilitando ao usuário a escolha do robô ou mecanismo de busca adequado à sua aplicação.

Apesar de parecer na figura que há apenas um meta-robô na arquitetura, na verdade cada agente possui o seu meta-robô. Um meta-robô é um robô que pode conectar-se a múltiplos mecanismos de busca - como *Altavista, Excite, Infoseek* e outros – aproveitando os seus

índices, uma vez que não é necessário, para a recuperação de página de uma classe, indexar ou percorrer toda a Web. O meta-robô proposto segue todas as diretrizes enumeradas acima.

Ele funciona da seguinte forma: efetuam-se consultas aos mecanismos de busca com palavras-chave que garantam cobertura das páginas retornadas em relação à classe de páginas processada pelo agente. Os termos *'call for papers'* e *'call for participation'* asseguram a cobertura das páginas de chamadas a eventos científicos para o agente *'Call for Papers'* (doravante CFP), por exemplo. Devido à falta de precisão, o conjunto de páginas resultante das consultas recai em vários grupos funcionais além do grupo de páginas-conteúdo associado ao meta-robô, apresentando muitas listas, mensagens, páginas-conteúdo de outras classes, e lixo. O uso do meta-robô pode ser encarado como o disparo de atividade de um agente, já que os agentes devem priorizar o processamento das "dicas quentes" enviadas pelos outros agentes, páginas que, presumivelmente, por advirem dos relacionamentos entre as classes ou de outras páginas reconhecidas pelo próprio agente como listas, supostamente devem apresentar uma precisão significativamente maior em relação à classe processada. Portanto, cada agente continuamente acessará duas filas de URLs: o conjunto sugerido pelo meta-robô é colocado numa fila de baixa prioridade de processamento, enquanto os "palpites" de outros agentes e listas ficam numa fila de alta prioridade. Em resumo, o robô de busca executa o seguinte:

```
. Com cada mecanismo de busca
   . Com cada consulta
      . Com cada página retornada pelo mecanismo de busca
       . Acha os ponteiros ou retorna condição de erro
       . Entrega-os ao Processador de Páginas
       . Grava estatísticas da consulta
  . Até estourar o tempo fixado na criação do robô.
```

Vale salientar que o meta-robô foi projetado de forma parametrizada, permitindo a inclusão de novos mecanismos de busca como registros numa tabela do banco de dados, abstraindo alterações de código ou recompilações. A sintaxe aproximadamente padronizada como os

mecanismos de busca chamam as suas consultas através de um comando CGI[22] e dispõem as URLs na página foi descoberta no melhor estilo RUDE[23], e propiciou a criação do primeiro extrator do sistema, retirando das páginas de resposta dos mecanismos de busca apenas as URLs resultantes.

Em princípio, o uso de meta-robôs satisfaz aos agentes, exceto quando não existem palavras-chave representativas de uma boa cobertura, e.g., para um agente de pesquisadores. Neste caso, heurísticas a respeito de localização (vide subseção 3.3.2.1.) com certeza serão mais efetivas para a recuperação das páginas da classe. Também cabe lembrar que o meta-robô só deve ser posto em ação caso não haja URLs na fila de alta prioridade a serem processadas.

5.1.2. Categorias e Dicionários

Três conjuntos de informações dão suporte ainda às tarefas de reconhecimento de páginas das classes e extração, os dicionários, as categorias e as ontologias.

Em qualquer sistema de Recuperação de Informação, e especialmente nos da subtarefa de categorização, a construção de dicionários de palavras-chave consideradas inúteis para o processamento (chamadas de *stop-lists* [Baeza-Yates & Ribeiro-Neto 99]) e também de dicionários de termos que identificam ou rejeitam páginas para uma dada classe, redunda numa tarefa importante na elaboração de um sistema, muitas vezes resolvida automaticamente, com técnicas de aprendizado [Riloff 94]. Este problema foge ao escopo deste trabalho; os dicionários dos estudos de caso foram preenchidos manualmente, embora, no futuro, seja factível o uso de ferramentas públicas de indução de regras, como o RIPPER [Cohen 96] ou o W4F [Sahuguet & Azavant 99] (vide subseção 7.2.1.1.).

---

[22].Do inglês *Common Gateway Interface.* O termo denota um padrão usado em programas de interface com servidores quaisquer através da Web, que habilitam a forma mais simples de passagem de parâmetros para a execução de serviços destes servidores

[23] A sigla em inglês, *Run, Understand, Debug and Edit,* traduz um estilo de programação comum para sistemas de Inteligência Artifical Simbólica. Justificado pelos riscos, incertezas e tentativas em enfrentar problemas complexos, especifica-se o conhecimento, e com iterações e desenvolvimento incremental, onde novos conhecimentos vão sendo descobertos e adicionados, até o sistema atingir um comportamento satisfatório.

O segundo conjunto é o de categorias, armazenadas num banco de dados, e possuem tabelas gerais a quaisquer grupos, como países, estados e outros, e específicas, como as áreas de pesquisa e as hierarquias de cargos em centros de pesquisa para o grupo científico.

O terceiro conjunto engloba as ontologias ou o conhecimento dos agentes sobre o domínio a que o *grupo* se refere, sobre suas tarefas e também sobre os outros agentes. A seção 5.4 discorrerá mais sobre esse conhecimento, e a seção 5.5, sobre o funcionamento dos agentes e suas tarefas.

5.1.3. Mediador

Cada agente pode gerar mais de uma tabela, devido à normalização. Para prover consultas sobre o meio científico, as tabelas estarão integradas numa só base de dados; por exemplo, se o usuário deseja consultas sobre uma dada área de pesquisa, interessam-lhe os pesquisadores, artigos, congressos, etc. Como a base de dados gerada pelos agentes é normalizada, construir consultas corretas para o acesso aos dados contidos nela, espalhados em várias tabelas diferentes, pode vir a tornar-se uma tarefa complicada demais para o usuário mediano [Freitas et al 99]. Além disto, outros agentes que não tomam parte na extração devem interessar-se em obter estes dados através de consultas, o que só será possível com o conhecimento do esquema da base de dados. Um mediador deverá, então, estar disponível, com a função de ajudar às consultas, provendo visões não-normalizadas, mais básicas da base de dados, e permitindo a qualquer usuário ou agente beneficiar-se do acesso aos dados extraídos. Questões relativas à mediação serão revisitadas ao final deste capítulo.

5.2. Modelo de Interação

Huhns e Singh [Huhns & Singh 97b] propuseram um teste para agentes que, na realidade, funciona como uma avaliação de sua sociabilidade. Segundo o teste, um agente deve modificar o seu comportamento quando um novo agente é introduzido na sociedade. Este novo agente deve assemelhar-se aos da sociedade em funcionalidade e estrutura, mas não em conhecimento, exceto da ontologia usada na comunicação entre eles. O referido artigo, inclusive, indaga se os agentes de Recuperação de Informação e os assistentes digitais pessoais (PDAs, ou *Personal Digital Assistants*), que buscam páginas de acordo com um

modelo de usuário montado dinamicamente, podem ser considerados agentes, uma vez que, usualmente, são incapazes de reconhecer outros agentes que estejam no ambiente em que operam, às vezes incumbidos das mesmas tarefas.

O modelo de interação dos agentes da arquitetura assemelha-se ao implementado em vários sistemas multiagentes, como o InfoSleuth [Bayardo Jr et al 96]. Ao entrar no sistema, os agentes registram-se e anunciam-se aos outros agentes, mandando as regras gerais e específicas (para um agente específico) de reconhecimento de páginas e ponteiros úteis a si próprio, que serão empregadas pelos outros para lhe indicarem sugestões de páginas a processar. Além de recebê-las dos outros agentes, o novo agente receberá também, reciprocamente, as regras para identificar páginas e ponteiros úteis a eles. Assim, quando um agente acha informação que dispara alguma dessas regras referentes aos outros, este agente repassa a informação (ponteiro ou página e o conjunto de regras e conceitos usados para encontrá-la) ao agente que lhe enviou a(s) regra(s) disparada(s).

Este modelo de interação obedece ao teste de sociabilidade em seus dois critérios. No momento em que um novo agente entra no sistema, os outros agentes modificarão seu comportamento, tentando identificar informação útil para ser entregue a ele. Como será visto na seção 5.6., os agentes possuem ainda a mesma estrutura e código, o que acarreta várias vantagens de reuso e flexibilidade. Torna-se, inclusive, fácil modificar o comportamento de um agente ou de todo o sistema; pode-se incluir no ambiente um agente que anuncie novas regras para os que outros identifiquem páginas de suas próprias classes, para melhorias de performance ou inclusão de novos atributos a serem extraídos.

5.3. Aprendizado

A partir do modelo de cooperação descrito acima, uma possível extensão da arquitetura, sugerida como trabalho futuro (vide seção 8.4.) consiste na inclusão de técnicas de aprendizado, implementando o nível reativo de uma arquitetura em três níveis (reativo, instintivo e cognitivo) [Bittencourt 97]. A inclusão de aprendizado pode auxiliar a, dinamicamente, avaliar a eficiência das regras de sugestão de páginas a processar (vide apêndice A.1., para um exemplo), trocadas entre os agentes, bem como das próprias regras de reconhecimento.

Ao receber uma sugestão, um agente avalia a página correspondente com suas próprias regras, verificando se, de fato, ela pertence à classe processsada pelo agente. As regras com as quais o agente efetua esta avaliação são, naturalmente, mais refinadas do que as que ele enviou ao agente que sugeriu a sugestão da página. Este resultado pode ser empregado para determinar a adequação das regras de sugestão de páginas, bem como sua adaptação a um determinado agente, o que pode levar a um aprimoramento das regras de sugestão, através da possível aplicação de métodos automáticos de aprendizado. Este aprimoramento pode, ainda, levar em conta resultados estatísticos sobre a eficácia das próprias regras do agente, no sentido de detectar regras candidatas a serem escolhidas como regras de sugestão.

Portanto, a utilização de aprendizado automático pode fornecer três serviços à arquitetura:

- Na fase de aquisição de conhecimento, se devidamente anotadas as páginas, o(s) algoritmo(s) de aprendizado pode(m) encontrar as regras de reconhecimento e extração para um ou mais agentes;
- Durante o processamento, um agente avalia a eficácia das regras sugeridas e, em caso de desempenho insatisfatório, pedir aos agentes que não as usem mais;
- Neste caso, o agente pode sugerir outras regras comprovadamente capazes de reconhecer com mais precisão as páginas de sua classe.

Contudo, estes serviços e sua respectiva pesquisa não se incluem no elenco de atividades desta tese, face às complexidades inerentes ao problema, como a definição de características, tratabilidade, escolha de algoritmos adequados, etc. A incorporação de aprendizado na arquitetura deverá constituir o tema de uma dissertação à parte.

5.4. Conhecimento dos Agentes

O conhecimento dos agentes estará codificado em cinco ontologias, mais as bases de regras e funções. As ontologias encontram-se descritas logo abaixo:

1) *Ontologia do domínio tratado*, nos estudos de caso deste trabalho, uma ontologia do meio científico (vide seção 4.9.).
2) *Ontologia da Web*, contendo, no mínimo, definições de *hyperlink*, termo e freqüência, e de página da Web em suas várias representações e atributos - como listas de palavras-

chaves e suas freqüências, ponteiros, e-mails e outros (vide subseção 5.5.1.). Esta ontologia pode adicionalmente conter definições e instâncias de protocolos, tipos de arquivos e outros conceitos relativos à Internet.

3) *Ontologia de manipulação integrada de informação*, contendo classes com as seguintes definições:

   a) *Templates* reconhecedores das diversas categorias funcionais (lixo, mensagens, listas, sugestões e páginas-conteúdo),

   b) *Templates* classificadores de páginas-conteúdo (em eventos científicos Conferência, Workshop, Jornal, etc),

   c) *Templates* extratores de atributos,

   d) Definições auxiliares a estes *templates*, como definições de conceitos e seus sinônimos e palavras-chave, e definições de casos identificadores de atributos, classes de páginas e categorias funcionais.

   e) Definições relativas à cooperação, como a classe Recomendação e a classe Agente, instanciada cada primeira vez em que um agente avisa que está ativo e qual classe de páginas-conteúdo ele se propõe a processar.

4) *Ontologias contendo definições de entidades úteis* tanto na conceitualização do domínio quanto em inferências dos agentes. Podem ser divididas em dois subtipos:

   a) As que contêm conceitos sobre outras áreas de conhecimento, como ontologias de tempo, locais, etc.

   b) Ontologias lingüísticas, como o WordNet [Miller 95] e o GATE (*General Architecture for Text Extraction)*, da Universidade de Sheffield, Inglaterra. O emprego de uma ontologia deste gênero provê uma tratamento de textos mais preciso e rico, e provocaria mudanças em todas as outras ontologias, exceto a do domínio e as de outras

5) *Ontologias particulares de cada agente*, que podem subdividir-se em:

a) Ontologias de outras áreas de conhecimento, porém de interesse específico do agente. Nem todo agente possui ontologias deste tipo.

b) Ontologia principal do agente, que inclui todas as outras ontologias, usando-as tanto para como vocabulário de comunicação, como também para o desempenho de suas funções. Esta ontologia não possui classes, mas apenas instâncias de conceitos, casos, extratores e reconhecedores de classes e categorias funcionais.

O conjunto de regras pode ser dividido em dois grupos:

- *Regras de reconhecimento e extração*, sendo que um subconjunto mais simples das de reconhecimento será enviado aos outros agentes.
- *Regras de recomendações,* que identificam páginas e ponteiros a serem sugeridos aos outros agentes, e que, em princípio, foram recebidas deles.

As ontologias e regras só são carregadas uma vez a cada execução do agente, por motivos de eficiência. Exemplos serão apresentados no capítulo 6.

5.5. Tarefas dos Agentes

Como pode ser observado na figura 20, cada agente desempenha quatro tarefas consecutivas no processamento de cada página: validação, pré-processamento, reconhecimento e extração. Naturalmente, as três últimas tarefas dependem dos resultados das tarefas anteriores; por exemplo, páginas inválidas (que não foram aprovadas na validação) não são processadas, assim como páginas reconhecidas como páginas de mensagens não passarão pela fase de extração.

Cada agente manipula três filas de URLs durante sua execução: duas são processadas pelo próprio agente, uma de baixa e outra de alta prioridade, e a terceira contém sugestões de *links* para serem enviadas aos outros agentes.

O controle de chaveamento entre as fases de processamento é efetuado por uma classe da ontologia de manipulação integrada, o *monitor de processamento*. Esta classe contém o atributo *Page-Status,* que vai sendo modificado à medida que transcorre o processamento da página, assumindo sucessivamente os estados de *ACESSADA, VÁLIDA, ARMAZENADA, RECONHECIDA, CLASSIFICADA, RECLASSIFICÁVEL e DADOS*

EXTRAÍDOS. Ao final das inferências, caso a página não consiga ser reconhecida e classificada, ela estará categorizada funcionalmente como *INVÁLIDA, INACESSÍVEL, MENSAGEM, LISTA, FRAME* (tratada como uma lista, onde cada frame é uma nova URL), *RECOMENDAÇÃO* (página reconhecida como pertencente à outra classe de páginas) e *REJEITADA*. As próximas subseções apresentam descrições detalhadas das quatro fases de processamento.

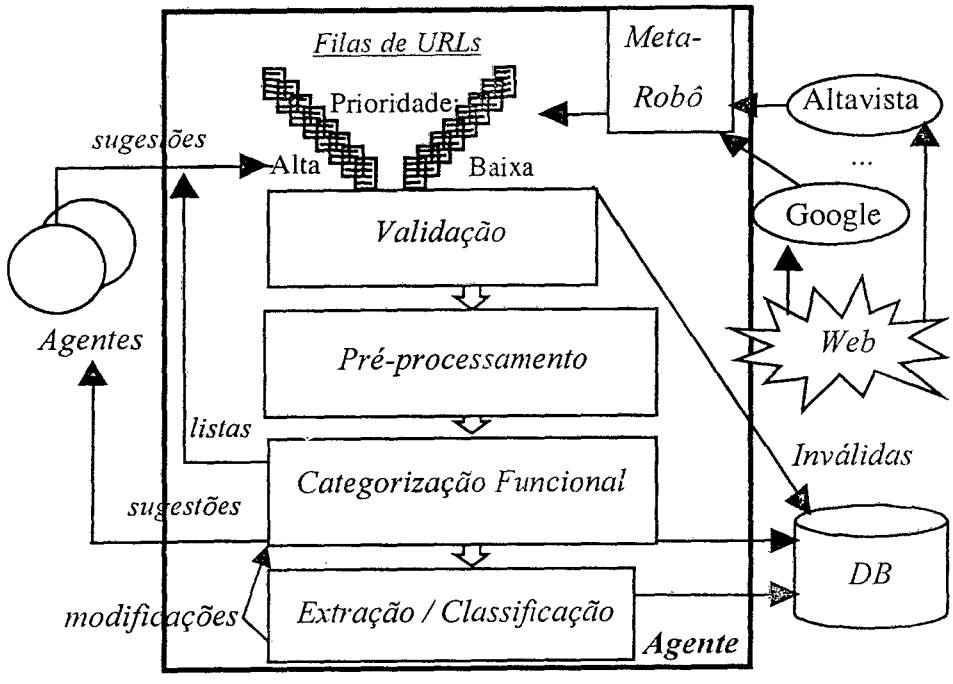

Figura 20. Detalhe de um agente, evidenciando as tarefas a serem realizadas.

5.5.1. Validação

Nesta fase, concretiza-se a recomendação para os robôs de só recuperarem páginas escritas em formatos que os agentes possam processar. Com efeito, a validação elimina trabalho redundante, verificando se a página já está presente no banco de dados, seja ela válida ou não, indicando se já foram processadas, se está acessível, se o seu formato e protocolo são compatíveis, e outras regras. Mesmo as páginas inválidas ficam armazenadas no banco de dados, porque o robô freqüentemente depara-se com ponteiros repetidos, só as recuperando novamente se sua data de última modificação foi modificada. A validação acelera todo o processamento, além de evitar trabalho redundante, economizam banda passante e tempo de processamento, gastos com a recuperação de páginas inúteis.

A maior parte das regras de validação é reusável por todos os agentes, mas regras específicas podem ser implementadas, referenciando outros campos, como data e tamanho, ou mesmo tratando outros protocolos.

O exemplo apresentado a seguir, foi escrito na linguagem do motor de inferência Jess (descrito na subseção 6.2.3.):

```
(defrule v_311_content-type "validity"
      ?f <- (object (Page-Status ACCESSED) (is-a Processing-Monitor))
      (object (Content-Type ?x&:(eq FALSE (str-index "TEXT/HTML" ?x)))
              (is-a Web-Page))
      =>
      (modify-instance ?f (Page-Status INVALID) (Recognition-Rule 311)))
```

Esta regra testa se a página está escrita em HTML. Ela quer dizer que, se a página conseguiu ser acessada - ou seja, antes outra regra pôde ler através da conexão dados, como protocolo, tipo de conteúdo, etc - e seu tipo de conteúdo não for HTML, ela será considerada inválida. A partir disso, seu processamento termina, sendo o seu endereço, data e estado armazenados no banco de dados. A representação de uma página pelo motor será vista na próxima subseção.

5.5.2. Pré-processamento

A fase de pré-processamento tem por meta representar as páginas de diversas maneiras, com dados extraídos delas, aplicando, se necessário, técnicas de recuperação de informação e processamento de linguagem natural, como *stop-lists*, *stemming* e etiquetagem (*tagging*) [Appelt & Israel 99]. Estes dados são repassados para o motor de inferência para que possam auxiliar as fases posteriores (de reconhecimento e extração), e podem vir a ser enriquecidos com outros que venham a provar-se úteis. Eles estão reproduzidos na figura 21, que apresenta a classe Web-Page e seus atributos e facetas, que estão na ontologia Web.

Os primeiros quatro itens a serem obtidos, através da conexão à URL, são *Protocol, Content-Type, Page-Date* e *Length*. Os outros itens são preenchidos durante o pré-processamento, depois da validação.

5.5.3. Categorização Funcional

Durante esta fase, um agente classifica as páginas em grupos funcionais, como lista, mensagem, lixo ou reconhecida como membro da classe processada pelo agente ou da classe de outro agente do sistema. Esta tarefa precisa ser realizada com precisão, sob pena de perda de informação relevante - se as páginas membros não são reconhecidas -, perda de tempo tentando extrair atributos a partir de páginas inúteis, e até mesmo de perda de consistência dos dados extraídos, se os atributos forem extraídos destas páginas irrelevantes. Particularmente, o reconhecimento de listas deve apresentar boa precisão, caso contrário uma infinidade de endereços de importância duvidosa, advindos de falsas listas, serão inseridos na fila de processamento de alta prioridade.

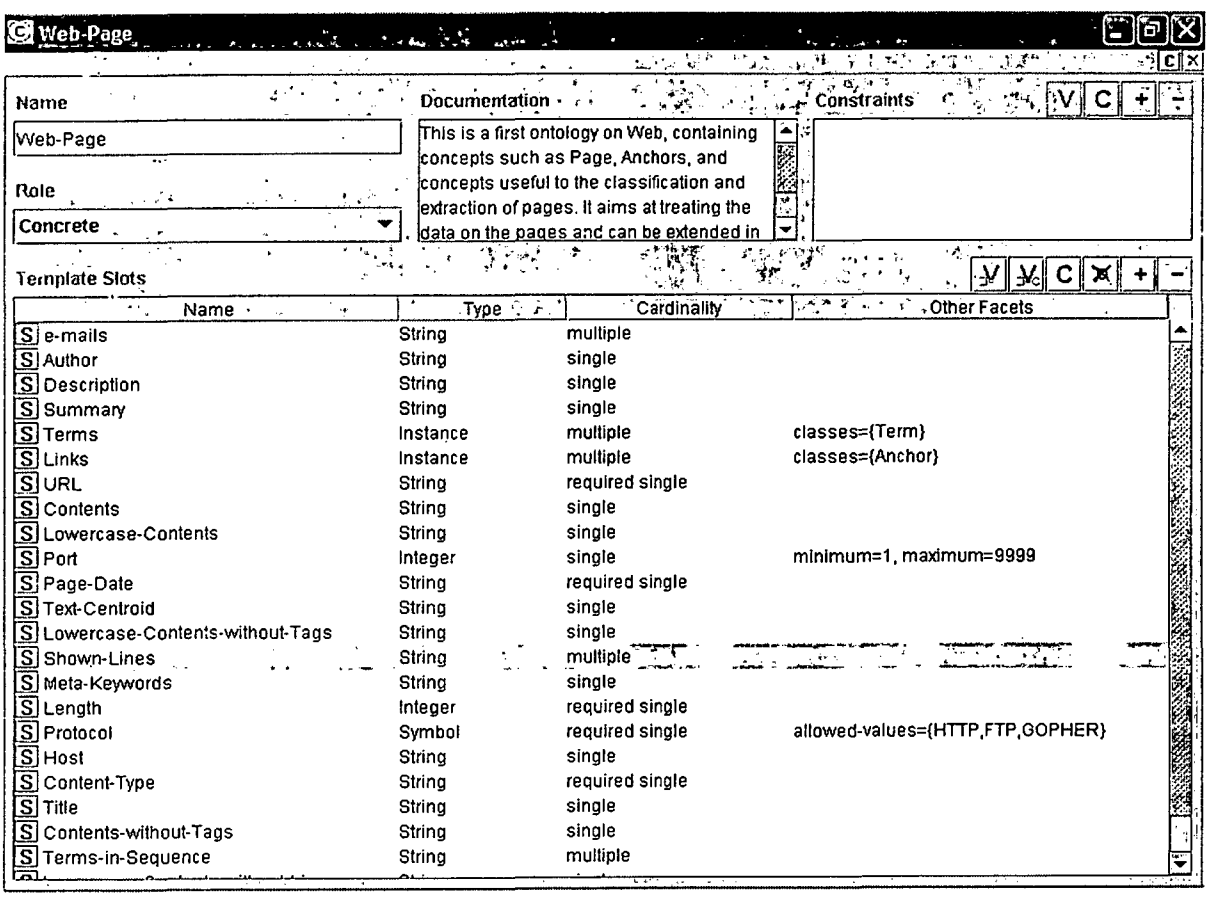

Figura 21. Classe *Web-Page* e alguns de seus atributos e facetas. A classe encontra-se definida na ontologia Web, que, por sua vez, foi definida dentro do editor de ontologias Protégé [Noy et al 2000].

Para esta fase e as próximas, emprestou-se o conceito de *casos* do sistema de extração AutoSlog [Riloff 94]. Neste sistema, os casos são adquiridos por algoritmos estatísticos de

aprendizado para posterior filtragem, se um dos casos ocorre no texto. O uso de casos, na arquitetura aqui proposta, será usado tanto para categorização funcional quanto para classificação, promove genericidade e expressividade na especificação de condições em que uma página deva ser categorizada ou classificada. Além do mais, como a maior parte das regras de reconhecimento instancia casos, elas podem ser incluídas em todos os agentes, sendo completamente reusáveis por instanciação (vide exemplo na subseção 6.5.2.). Porém, regras específicas de um agente, a partir de casos peculiares, podem possuir uma precedência maior, por se beneficiarem de informação contextual e conhecimento *a priori*.

Segue abaixo um exemplo de um caso particular muito comum em chamadas para eventos científicos ao vivo, como conferências e *workshops*, em que uma página apresenta no seu início os atributos (*slots*, no código) data inicial do evento e localização (país ou estado americano em que irá ocorrer o evento) e algum termo relacionado a um evento ao vivo, como as expressões *"call for papers"* ou *"call for participation"*, ou um termo sobre o tipo do evento, como *"conference"* ou *"workshop"*.

```
([Science_00583] of Case
      (Importance HIGH)
      (Description "Date-time-cfp in beggining")
      (All-of-Concepts-Keywords FALSE)
      (Slots-in-the-Beginning
            [Initial-Date]
            [takes-Place-at])
      (Absent-Concepts
            [listing-of-call-for-papers])
      (Concepts-in-the-Beginning
            [conference]
            [workshop]
            [cfp]
            [Call-for-participation]
            [meeting]))
```

O caso é considerado de importância alta e assinala que nem todos os conceitos ou palavras-chave mencionados (o caso não definiu nenhuma palavra-chave) precisam estar presentes. Define ainda que termos relacionados a listas de chamadas de eventos, contidas no conceito *listing-of-call-for-papers* deverão estar ausentes para a página ser reconhecida. Com efeito, além da importância, que determinará a precedência do caso no disparo das

regras, os casos podem especificar conceitos, atributos e palavras-chave presentes e/ou ausentes, na página ou no início dela, e número de atributos encontrados dentre um conjunto definido no caso. Mais detalhes sobre casos e regras, relativos à implementação podem ser encontrados na subseção 6.5.2.

5.5.4. Extração e Classificação

O trabalho executado nesta fase é particular a cada agente, não permitindo qualquer tipo de reuso. De uma página reconhecida como pertencente à classe do agente (pesquisadores, "*call for papers*", etc); são extraídos os atributos, armazenados na base de dados, ou, pela inconsistência ou inexistência destes, corrigida a classificação de página em relação aos grupos funcionais. Para as páginas válidas rejeitadas, procuram ser identificados ponteiros úteis, interessantes para outros agentes, baseados nos textos e nos prefixos das hierarquias das âncoras (vide subseção 3.3.2.1.), ou mesmo no próprio contexto.

A extração de atributos é efetuada através de regras que mencionam os diversos campos (ou representações), e pode ser feita de três maneiras: diretamente da página, através de inferência ou através de categorização. Devido à falta de estrutura das páginas, nem sempre será possível a extração das informações. Após a extração, os atributos podem, adicionalmente, ser formatados para o armazenamento (como, por exemplo, com datas). Para realizar a extração por categorização, palavras-chave da página ou de uma determinada região dela são conferidas contra uma lista de termos dos dicionários, associados à respectiva tabela de categorias (sobre esta tabela, vide seção 5.1.2.). Se uma das palavras-chave está contida num dos termos que designa a categoria, testa-se se o termo inteiro está na página ou região. Se isto ocorrer, isto indica que o atributo pertence àquela categoria, e, se for possível, múltiplas categorias (por exemplo, áreas de interesse de um pesquisador), o processo continua até a última palavra da região ou página, caso contrário considera-se findo o processamento do atributo para a página.

5.6. Reuso

Seguindo o teste para agentes descrito na subseção 5.2., a arquitetura apresenta vários tipos de reuso, facilitando e acelerando a construção de novos agentes, que compartilham a mesma estrutura em termos de código, esquema da base de dados, serviços dos

mecanismos de busca, e até mesmo de boa parte do conhecimento de que dispõem, tanto em termos de ontologias e instâncias, como de regras. Estes benefícios põem de relevo as vantagens da aplicação de abordagens baseadas em conhecimento para problemas como o apresentado neste trabalho, uma vez que, se fosse empregada uma abordagem procedural, muitas alterações teriam de ser realizadas para a criação de um novo agente, sem contar que, para a elaboração de um novo sistema, boa parte do código teria de ser modificado. A abordagem procedural, neste caso, constituiria uma solução rígida e de difícil manutenção.

5.6.1. Código

Se elaborado de forma parametrizada, o código pode ser inteiramente reusado independentemente do domínio tratado. O funcionamento dos agentes é igual em todos os aspectos; a única diferença entre eles reside no conhecimento sobre a classe específica de páginas que cada um trata. Apesar desta característica, se o código for programado com orientação a objetos, os agentes têm ainda a alternativa de implementar métodos particulares que podem se sobrepor aos métodos já herdados.

5.6.2. Esquema da Base de Dados

Como um corolário à subseção anterior, os agentes manipulam tabelas da base de dados com a mesma estrutura: as tabelas de páginas não reconhecidas, códigos de erro, categorias, mecanismos de busca e consultas aos mecanismos. As únicas tabelas particulares de um agente são justamente as que guardam as entidades e atributos extraídos por ele (e.g., artigos, para o agente de artigos, etc). A arquitetura pode abstrair o esquema desta tabela e inserir-lhe registros, pelo acesso às estruturas das tabelas, através de meta-dados, disponíveis através de componentes de conectividade aos sistemas gerenciadores de bancos de dados relacionais, como os componentes ODBC (*Open Data Base Connectivity*) ou JDBC (*Java Data Base Connectivity*) [JDBC 99].

5.6.3. Robôs e Mecanismos de Busca

Além dos motivos já apresentados na subseção 5.1.1. sobre reaproveitamento de robôs e mecanismos de busca, como uma deliberada decisão de projeto e abordagem à problemática de manipulação de informação na Internet, a concepção da arquitetura foi motivada pela visão de que os mecanismos de busca tradicionais, baseados em técnicas de

recuperação de informação e indexação por palavras-chave, podem constituir a base e o suporte para outros mecanismos de busca, aplicativos, agentes e extratores mais refinados, precisos e focados em domínios restritos, baseados em conhecimento explícito reusável e comunicável, e habilitados à cooperação.

5.6.4. Conhecimento

Pode-se afirmar que foi a busca por reuso do conhecimento que reavivou as abordagens baseadas em conhecimento, a partir da implementação de ontologias (vide capítulo 4). Na arquitetura apresentada, três tipos de reuso do conhecimento dos agentes são possíveis:

- Reuso das ontologias "de prateleira" (vide subseção 4.5.1.), já publicamente disponíveis, como as ontologias sobre tempo, documentos e turismo, e mesmo sobre o domínio científico.

- Reuso das ontologias usadas na comunicação entre os agentes, ou seja, o vocabulário a respeito de páginas da Web enquanto documentos (vide seção 5.5.1.), acrescido de conhecimento considerado necessário a todos os agentes, por exemplo, sobre as categorias.

- Reuso da ontologia sobre os grupos funcionais, incluindo a maior parte das regras de reconhecimento. Para alguns destes grupos, as regras são gerais (mensagens é o melhor exemplo), contudo as regras instanciáveis (vide seção 5.5.3.) são em maior número; pela definição do relacionamento entre o agente e o grupo funcional, é que a regra poderá opcionalmente ser instanciada com casos e conceitos de dicionários (vide subseção 5.1.2.).

Como é perceptível, a definição de novos conceitos e ontologias será mínima; ao invés disso, o conhecimento dos agentes instanciará conceitos pré-definidos. Portanto, o desenvolvimento de um novo agente torna-se rápido, exceto pela tarefa de aquisição de conhecimento necessário à categorização (ou reconhecimento) das páginas em grupos funcionais [Freitas & Bittencourt 2000], realizável de duas maneiras: através da engenharia de conhecimento, examinando exemplos de páginas para entender seus padrões, ou através de técnicas de aprendizado automático, com algoritmos estatísticos que geram regras a

partir de *corpi* anotados [Appelt & Israel 99]. A escolha vai depender de fatores como a complexidade das regras, número de páginas a anotar e heterogeneidade das páginas.

A seção seguinte disserta sobre mediação, suas funcionalidades e componentes, com o objetivo de situar o leitor sobre esta tecnologia, empregada neste trabalho.

5.7. Mediação

Na medida em que os sistemas e servidores crescem, uma gama diversa e heterogênea de bases de dados e sistemas passa a incorporar um acervo, com dados e funcionalidades potencialmente acessíveis aos usuários (de agora em diante, usuário designa pessoas, agentes ou outras entidades de software). Tanto quanto possível, os usuários devem ser poupados do conhecimento de detalhes de acesso e funcionamento, como linguagens de bancos de dados, parâmetros, entre outros. Um mediador propõe-se justamente a prover uma fácil interação entre os usuários com os serviços de um sistema ou servidor, atuando como relações públicas dos serviços junto aos usuários. Sua existência pode ser justificada de várias maneiras [Lopez 98]:

- Apenas parte do conjunto de informações e funcionalidades é relevante para os usuários. Devido à normalização, a estrutura de bancos de dados, por exemplo, tornaria as consultas difíceis e complexas para o usuário mediano;
- Os dados em bancos de dados distintos muitas vezes possuem formatos distintos, dificultando sua integração;
- Há ainda serviços e sistemas que possuem funções similares, e seus usuários, para lançar mão deles, têm dificuldade de distinguir qual deles adequar-se-á melhor à sua tarefa, e também de formular pedidos com o grau de granularidade correto;
- Integrar cada novo servidor consome muito tempo, e não provê a flexibilidade e legibilidade desejadas;
- Um requisito fundamental para facilitar o uso automático destes sistemas ou agentes é que os sistemas ou agentes usuários possam acessar facilmente informações relativas à semântica dos bancos de dados e dos serviços disponíveis, necessitando, portanto, de

entidades de software ou agentes que desempenhem adequadamente as tarefas de disponibilizar esta semântica e proceder à comunicação sobre elas com segurança.

Atualmente, pesquisas em duas classes de sistemas suportam os serviços de mediação:

- *Facilitadores*, que implementam a transparência de localização em programação baseada em componentes ou agentes, i. e., um serviço de nomes e roteamento de mensagens e;
- *Mediadores*, que podem ser vistos como camadas de *middleware* que provêem serviços intermediários, tentando tratar a mensagem recebida, entender e processar o pedido.

5.7.1. Funções

A *mediação* pode reunir entidades múltiplas e heterogêneas, como sistemas, agentes, componentes, bases de dados, bases de conhecimento, *sites*, documentos, bibliotecas digitais e sistemas geográficos entre outros. Especialmente em bancos de dados, pode proporcionar uma semântica ao conteúdo das informações contidas nos esquemas, já que esses esquemas trazem apenas uma descrição lógica das informações que guardam, e não conhecimento sobre elas, sem a possibilidade de inferência ou raciocínio. Existem pesquisas sobre bancos de dados dedutivos, mas ainda está longe a existência de um padrão de sintaxe para SQL que incorpore essa facilidade. São tarefas típicas de mediação ao usuário [Lopez 98] [Wiederhold & Genesereth 97]:

- Filtragem e extração de informação, entregando ao usuário apenas o que lhe interessa;
- Transformação da informação para um formato compreensível ao usuário – este formato pode ser da estrutura física da informação ou, no caso de agentes, de uma ontologia para outra [Wiederhold & Genesereth 97];
- Combinação ou integração de informação;
- Repasse *(brokering)*, quando mediadores começam a conversar com outros mediadores, o que, aliás, confere transparência de localização à informação;
- Notificação, isto é, aviso ao usuário que há informação que lhe pode ser útil, entregando-a sob sua concordância.

A existência de mediadores *adiciona valor* aos serviços fornecidos, atuando como uma recepção e interface amigável e interoperável dentro de uma arquitetura que reúne esses serviços, refinando os pedidos para o grau de granularidade adequado, traduzindo-os e otimizando-os [Arens et al 96]. Os mediadores podem ainda executar tarefas ligadas à segurança e permissões, modelar o cliente de forma a identificar recursos relevantes e interessantes, anunciar serviços numa rede para assinantes e potenciais usuários, e até planejar a execução otimizada de funções típicas de agentes de informação, como busca, recuperação de informação e outras.

Qualquer área de aplicação que necessite integrar informação heterogênea pode servir-se de mediadores. Entre as áreas já testadas com sucesso, incluem-se comércio eletrônico, sistemas hospitalares, *data warehousing*, transporte aéreo [Arens et al 96], hotelaria, tráfego de estradas e sistemas corporativos [Lopez 98].

5.7.2. Componentes dos Mediadores

Os mediadores para sistemas abertos são compostos dos seguintes elementos:

- A *visão* ou *visões* com que o usuário irá enxergar o repositório de dados ou funcionalidades que o mediador provê. Esta visão, um conceito herdado de bancos de dados, almeja facilitar a interação lógica do usuário com os dados, sem descer ao (baixo) nível de normalização, implementação. Normalmente, isso é feito de duas formas:
  - Através de conhecimento estruturado em ontologias – que são bases de conhecimento organizadas em conceitos, sobre uma determinada área de aplicação, e que têm as vantagens de determinar restrições e relações entre as entidades, além de permitir inferências e especializações;
  - Através de um esquema global construído sobre diversos esquemas locais, sobre o qual o usuário construirá consultas. A linguagem de consultas SQL é um padrão de fato [Wiederhold & Genesereth 97, Lopez 98], face à sua popularidade.
- *Adaptadores* [Lopez 98, Arens *et al* 96], mecanismos bidirecionais para tradução ou mapeamento de mensagens recebidas e enviadas. Se o conhecimento estiver estruturado em ontologias, deve-se usar uma linguagem intermediária, mais conhecida como

interlíngua, para a tradução entre formalismos diferentes, economizando a construção de tradutores *ad hoc*. No caso de um esquema global, o adaptador deverá converter o pedido original num *script* de operações de bancos de dados, envolvendo consultas, junções, etc. Os adaptadores são componentes fundamentais para a arquitetura, no sentido de esconder e proteger as diversas fontes de informação, sejam elas bancos de dados, outras ontologias, arquivos estruturados, planilhas, e até mesmo mecanismos de busca para a Web;

- Um *mecanismo de comunicação* entre uma entidade de software e o mediador, que pode ser:
  - Um *protocolo de comunicação em nível de conhecimento*, i.e., capaz de portar mensagens num formalismo, ou melhor, ainda, em qualquer formato. O protocolo deve possuir mecanismos de registro e roteamento de mensagens;
  - Uma *especificação CORBA* [CORBA 98], que conterá as assinaturas dos métodos, contendo parâmetros do tipo *string* que serão tratados como um pedido de acesso às fontes de informação;
  - Um *applet*, com acesso às bases de dados via JDBC [JDBC 99].

Na seção 6.10. será apresentada a implementação do mediador do estudo de caso, desenvolvido em Delphi, HTML e Java usando KQML, que possui a novidade de empregar visões tradicionais de bancos de dados, ao invés de construir adaptadores, para traduzir as consultas, economizando tempo de desenvolvimento ao custo de espaço em disco [Freitas et al 99].

Capítulo 6

ESTUDOS DE CASO E RESULTADOS

6.1. Introdução

Neste capítulo, será apresentado o sistema multiagente MASTER-Web (do inglês *Multi-Agent Sytem for Text Extraction and Retrieval over the Web,* ou sistema multiagente para recuperação e extração de textos da Web), que instancia a arquitetura do capítulo anterior com as restrições descritas na subseção 6.3.1. Na seção 6.2., serão listados os requisitos necessários ao desenvolvimento do sistema, bem como as ferramentas que foram empregadas por atenderem a estes requisitos. A seção seguinte apresenta o sistema, discriminando o conhecimento que deve ser comum a todos os agentes do conhecimento particular de um agente. A seção 6.4. lista que itens precisam ser definidos para a construção de um novo agente, enquanto a seção 6.5. discorre sobre detalhes de funcionamento e especificação, ilustrando com um exemplo o funcionamento de um agente. A seção 6.6. especifica as técnicas básicas de concorrência, justificando o uso da sincronização de objetos como mecanismo de consistência da execução. A seção 6.7. descreve os experimentos realizados, e a seção 6.8., os resultados de cada um dos dois agentes , com tabelas totalizadoras para cada uma das classificações realizadas por cada agente em cada teste. Nesta seção, será mostrado que num dos agentes, o uso de listas incrementou a precisão da recuperação de páginas-conteúdo, justificando o uso de categorias funcionais para o processo de recuperação. Na seção 6.9., os resultados são discutidos, enquanto a seção 6.10. apresenta o protótipo do mediador implementado numa versão anterior do MASTER-Web.

6.2. Ferramentas de Desenvolvimento

6.2.1. Requisitos da Arquitetura

O desenvolvimento de um sistema de acordo com a arquitetura proposta no capítulo anterior deve atender aos seguintes requisitos:

- Facilidades de acesso aos recursos de rede da Internet;
- Conexão a um sistema gerenciador de banco de dados;
- Um motor de inferência que permita o uso de ontologias;
- Um editor de ontologias cujas bases de conhecimento geradas sejam acopláveis ao motor de inferência escolhido;
- Uma linguagem de comunicação entre agentes (*ACL – Agent Communication Language).*

Para atender a estes requisitos, foram escolhidos a linguagem Java [Sun 98a], o motor de inferência Jess [Friedman-Hill 2000], o editor de ontologias Protégé (vide seção 4.7.2.) e o pacote JATLite [JATlite 99] para comunicação entre agentes. Estas três ferramentas de desenvolvimento serão apresentadas a seguir, junto com as respectivas características e justificativas de sua escolha.

6.2.2. A Linguagem Java

Atualmente, a linguagem Java [Sun 98a] é a única que consegue reunir todos os requisitos acima citados implementados nativamente, ou seja, sem necessidade de conexão com outras linguagens. Java foi criada com a tecnologia de orientação a objetos como uma evolução da linguagem "C++" em 1991, e lançada ao mercado em 1995. Nenhuma linguagem adequa-se mais ao desenvolvimento de aplicativos para a Internet, pelos seguintes motivos:

- Java é a única que garante portabilidade total de código entre plataformas distintas (atualmente Windows e Unix), sem necessidade de recompilação, pelo uso de máquinas virtuais:
- A demanda de acesso à Internet ou Intranets por dispositivos mais simples, como impressoras e aparelhos domésticos, pode ser absorvida por Java através da tecnologia Jini [Jini 2000], um conjunto de programas pequenos que implementa protocolos e facilidades de registro e acesso a recursos de rede disponíveis;

- Inclui um pacote nativo específico (*.net*), com classes para acesso aos diversos campos do protocolo HTTP, recursos de rede (como soquetes, portas e outros), e páginas (URLs);
- Inclui um outro pacote nativo (*.jdbc*) para conectividade a gerenciadores de bancos de dados, diretamente ou através de pontes (*bridges*) para o ODBC (*Open DataBase Connectivity*);
- Caracteriza-se pela capacidade de processamento concorrente com múltiplas linhas de execução[24], de simples implementação, inclusive processando primitivas de entrada e saída sem espera ou bloqueio de execução;
- Possui inúmeros pacotes desenvolvidos, *freeware* e *shareware,* para os mais diferentes propósitos, incluindo motores de inferência e linguagens de comunicação entre agentes.

6.2.3. O Motor de Inferência Jess

A seleção de um motor de inferência revelou-se a decisão mais capciosa dentre a escolha dos componentes do MASTER-Web. Para representar ontologias, faz-se necessário o uso de um formalismo de representação de conhecimento capaz de raciocinar sobre o formalismo *frames*. Outro requisito era o de que a linguagem empregada pelo motor se incluísse entre as traduzíveis pelos servidores de ontologia, como a Ontolingua, ou o editor conectável a ela, Protégé.

O motor de inferência com encadeamento para frente Jess (*Java Embedded Expert System Shell*) [Friedmann-Hill 97] uma das mais populares ferramentas de desenvolvimento de agentes e sistemas especialistas da história, possuindo alguns milhares de usuários, principalmente devido à boa integração entre objetos Java e os formalismos regras de produção e *frames* (vide subseções 4.4.1. a 4.4.3.), tornou-se um candidato natural. O Jess implementa a maior parte das funcionalidades do sistema de produção CLIPS (*"C" Language Integrated Production System*) [Riley 99] para Java, já que o CLIPS integrava-se à linguagem "C". Se o conhecimento codificado for mantido separado do código, conforme recomendado na subseção 4.4.1.1., bases de conhecimento em CLIPS e Jess são

---

[24] Do termo em inglês *multithreading*.

compatíveis. Ambos fazem uso de uma linguagem que, à semelhança do LISP, emprega notação polonesa, com operador seguido de operandos. Apesar das máquinas virtuais de Java, o Jess, em certos problemas, consegue apresentar melhor performance que o próprio CLIPS, por adicionar otimizações ao famoso algoritmo de resolução de conflitos Rete [Forgy 82].

Contudo, a sutil e fundamental diferença de representação entre Jess e CLIPS reside na forma em que as classes são representadas nos dois motores. CLIPS inclui uma linguagem interna, COOL ("C" *Object Oriented Language*) [Giarratano & Riley 98], que representa classes como *frames*. Já Jess usa *Java beans* - componentes em Java que provêem reflexão – mas, como foi projetado para servir à comunidade de orientação a objetos, são incapazes de representar declarativamente listas estruturadas, e muito menos *frames*. Em Jess, um atributo não pode ser instância de uma classe declarativa, característica imperiosa em *frames* e ontologias.

A outra opção disponível, o motor de inferência JEOPS (*Java Embedded Object Production System*) [Figueira Filho & Ramalho 2000], visa uma aproximação sintática entre a linguagem hospedeira - no caso, Java - e o motor de inferência, com o objetivo de melhorar a integração entre objetos e regras. Todavia, para ontologias, sintaxe não é relevante, uma vez que qualquer definição que não se acomode num modelo orientado a objeto – e especialmente em Java, que não possui herança múltipla -, ainda assim pode ser expressa em lógica de 1ª ordem [Farquhar 97]. Além do mais, a priorização de objetos a declaratividade, mola mestra deste motor de inferência, afeta a capacidade de comunicação intencional dos agentes, já que objetos não têm expressividade (vide subseção 3.2.2.2.). Portanto, o JEOPS foi projetado para a orientação a objetos, não se adequando à representação de ontologias.

Felizmente, a popularidade de Jess e Java terminou por gerar uma solução para o problema de representação de *frames*. Um componente integrando as ferramentas Jess e Protégé chamado JessTab [Eriksson 2000] foi desenvolvido a partir de iniciativa da equipe do Protégé, habilitando o Jess a reusar, definir e refinar ontologias através do Protégé. Este componente substitui as definições de classes declarativas do Jess de forma a implementar a mesma expressividade para *frames* do CLIPS, incluindo muitas funções relativas às facetas dos *slots*, que não teriam sentido em Jess. Uma lição aprendida deste caso é que a

capacidade expressiva de representar *frames* constitui um requisito mínimo para motores de inferência quando se almeja reusar ontologias.

O Jess conta ainda com um ambiente de desenvolvimento com interface gráfica, que possibilita ao usuário desenvolver sistemas, *applets* e verificar a eficiência da rede de regras compilada. Possui ainda capacidade de encadeamento para trás, além de muitas outras facilidades foram implementadas pelos usuários e adicionadas ao pacote, como a manipulação de bancos de dados como bases de fatos, lógica difusa [Bittencourt 98], entre outras.

Entre outras qualidades do Jess, podem ser citadas a capacidade de raciocinar e manipular diretamente sobre objetos, métodos e variáveis Java dentro de regras Jess, a possibilidade de criação de regras e fatos Jess dentro de código Java, a passagem de objetos nos dois sentidos entre Java e Jess.

Hoje, o editor Protégé já conta com pelo menos duas outras opções, os componentes integradores para a linguagem Prolog e o formalismo F-Logic [Kifer 95]

Exemplos de definição de classes e regras Jess encontram-se nas subseções 5.5.2. e 6.5.2.

6.2.4. O Pacote JATLite para Comunicação entre Agentes

O pacote JATLite [JATLite 99] permite a seus usuários criar agentes que podem se comunicar na Internet através da linguagem de comunicação KQML (vide subseção 3.2.2.3.). O pacote provê um *template* com classes pré-definidas em Java, que facilitam a construção de agentes. As classes estão separadas em camadas, de forma que os desenvolvedores podem escolher que camadas seus agentes implementarão.

Adicionalmente, o pacote JATLite provê uma arquitetura básica de serviços de rede para a comunicação e coordenação entre agentes, através de agentes chamados de *facilitadores.* São providos os seguintes serviços:

- Registro do serviço de nomes dos agentes [Coulouris et al 94].
- Roteamento de mensagens entre os agentes.

- Repasse (*brokering*) de informação entre fornecedores e consumidores, quando ambos se anunciam.

- Recrutamento de fornecedores para lidar diretamente com consumidores que se anunciam.

- Notificação de informação (via *multicast*) para agentes interessados.

- Para a entrada ou saída de um agente, ele precisa anunciar esta decisão ao serviço de registro do seu facilitador local. É através dos facilitadores que os agentes podem enviar mensagens aos outros agentes que ignoram sua localização. Os roteadores usam os serviços de nomes dos facilitadores para achar os *hosts* dos agentes, o que confere *transparência de localização* (vide subseção 3.3.2.1.) aos agentes, e escalabilidade da comunicação em relação ao número de agentes. Normalmente, existe um facilitador para cada grupo de agentes. A comunicação entre os facilitadores locais é efetuada via facilitador central, conforme mostrado na figura 22. Dependendo da complexidade do sistema e do número de agentes, pode haver muitos níveis de facilitadores.

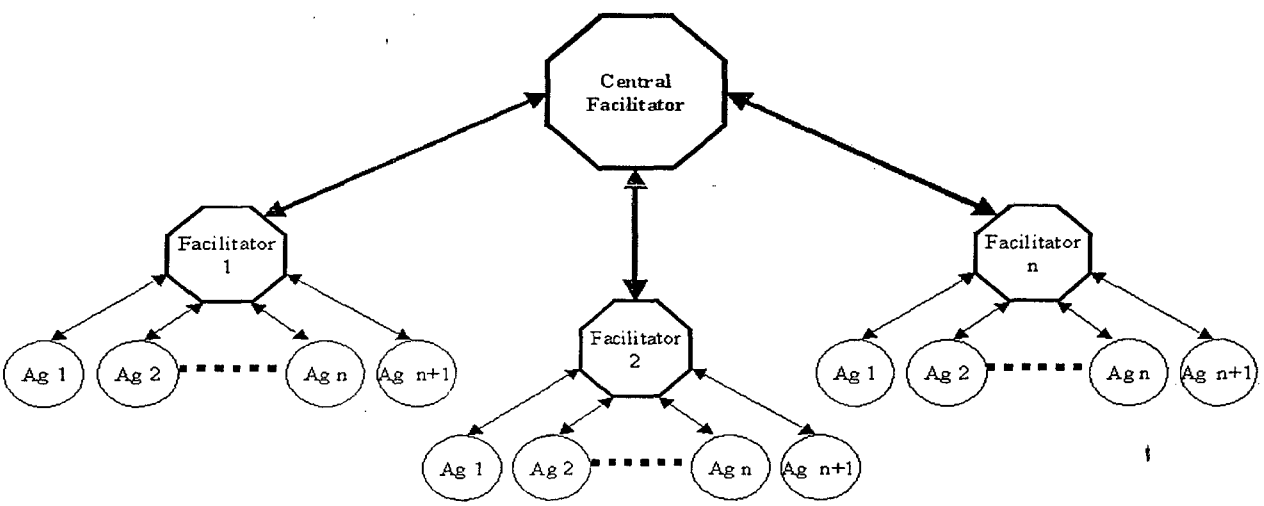

Figura 22. Exemplo de uma hierarquia de facilitadores.

As camadas providas pelo pacote, junto com suas respectivas funcionalidades, estão listadas abaixo:

- A Camada *Abstrata* disponibiliza uma coleção de classes abstratas necessárias a qualquer implementação com o JATLite. Embora o JATlite assuma o uso de TCP/IP

pelos agentes, é possível, pelo emprego e herança de classes desta camada, implementar diferentes protocolos, como o UDP, por exemplo;

- A *Camada Básica* provê funcionalidades básicas de comunicação relacionadas ao protocolo TCP/IP, usado em toda a Internet;
- A *Camada KQML* efetua o armazenamento e *parsing* de mensagens KQML, bem como os serviços de registro, conexão e desconexão;
- A *Camada Roteadora* responsabiliza-se pelo serviço de nomes e pelo roteamento e enfileiramento de mensagens. Os agentes só podem enviar e receber mensagens através do roteador, que repassa as mensagens a seus respectivos destinatários, através dos endereços contidos no serviço de nomes;
- A *Camada* de *Protocolo* dá suporte a diversos serviços da Internet como o SMTP, FTP, POP3, HTTP, tanto para aplicativos específicos como para *applets*.

Para o desenvolvimento dos estudos de caso, o sistema MASTER-Web, apresentado logo abaixo, foi usada apenas a camada KQML dentro dos agentes, e uma instância de roteador foi criada na rede em que estavam os agentes, para que eles pudessem trocar mensagens através dele.

6.3. O Sistema Multiagente MASTER-Web

Dois agentes e um protótipo do mediador do sistema MASTER-Web (do inglês *Multi-Agent Sytem for Text Extraction and Retrieval over the Web)* foram desenvolvidos usando as ferramentas acima e aplicados sobre o grupo científico da Web, conforme definido no capítulo 2. A ontologia Ciência, que serve aos agentes como vocabulário de comunicação, base para a cooperação e restrições na extração encontra-se descrita com brevidade na seção 4.9.

Os agentes e o mediador rodam em ambiente Windows, mas, como foram escritos em Java, podem ser portados para ambientes Unix. As ontologias do sistema foram escritas no editor de ontologias Protégé e convertidas dinamicamente para Jess. Os dados resultantes dos agentes foram armazenados pelo sistema gerenciador de bancos de dados (SGBD) SQL Server da Microsoft.

Um agente recupera, reconhece e classifica páginas de chamadas de trabalhos ("Call *for papers*") de eventos científicos, revistas, jornais, etc, no passado ou futuro, sendo, por isso, chamado de agente CFP. Diferentemente do sistema DEALINER [Kruger et al 2000], que, restritivamente, requer que uma chamada de evento possua título descrevendo o evento, lista de tópicos, comitê de programa, *deadline* e informações de submissão, o agente CFP caracteriza-se por mais generalidade, aceitando quaisquer páginas que contenham os principais dados que identificam um só evento científico (vide exemplos dos dois agentes no Apêndice A).

O outro agente desempenha as mesmas funções para a classe de páginas de documentos científicos. Cada agente possui um arquivo de configuração, determinando onde estão os servidores de bancos de dados e de páginas (se o teste estiver sendo realizado sobre um *corpus*), número de páginas do *corpus*, *driver* JDBC a ser empregado, usuário e senha para o banco de dados, classe processada pelo agente e ontologia do domínio. Logo, tanto os servidores quanto os agentes podem estar distribuídos pela Internet.

O mediador foi especificado para responder mensagens em KQML, tanto de agentes usuários como de agentes do sistema, e oferece, adicionalmente uma página de acesso para usuários da Web. A página de interface do mediador foi codificada em HTML com CGIs em Delphi. O mediador é apresentado na seção 6.10. O código foi desenvolvido independentemente da tarefa de agentes específicos, ou mesmo de *grupos*, apresentando os reusos citados na seção 5.6.; nominalmente, reuso completo de código, de esquemas da base de dados, de serviços de robôs e mecanismos de busca pré-existentes, e parcial do conhecimento usado para comunicação e inferência.

O meta-robô de cada agente habilitou-se, nestes experimentos, a conectar o *Google*, o *NorthernLight* e o *Altavista*, porém outros mecanismos de busca podem ser facilmente incluídos (vide subseção 5.1.1.). Também o *Excite* e o *Infoseek* já foram conectados em versões anteriores.

O sistema MASTER-Web será visto nas próximas seções enquanto instanciação da arquitetura proposta no capítulo anterior, ou seja, serão justificadas as decisões de projeto para a construção dos agentes.

6.3.1. Simplificações da Arquitetura

O MASTER-Web instancia a arquitetura com algumas simplificações. Em primeiro lugar, as definições de entidades úteis tanto na conceitualização do domínio quanto em inferências dos agentes (item 4 da seção 5.4.) não incluem ontologias lingüísticas, indisponíveis à época do início dos experimentos. Estas definições são basicamente instâncias da classe Conceitos *(Concepts)*, que possuem os atributos Sinônimos e Palavras-Chave. Por exemplo, veja-se a definição dos conceitos *cfp* e *listing-of-call-for-papers,* instâncias que, posteriormente, serão úteis nas subseções seguintes para esclarecer o funcionamento de regras e casos:

```
([Call-for-papers] of Concept
      (name "cfp")
      (Synonyms
            "call for papers"
            "call for presentation"
            "call 4 papers"
            "call4papers"))

([listing-of-call-for-papers] of Concept
      (name "listing of call for papers")
      (Synonyms
            "calls for papers"
            "confs"
            "cfps"
             "call for papers service"))
```

A falta de ontologias lingüísticas trouxe conseqüências para a extração, que basear-se-á apenas em delimitadores e funções para formatação e classificação do atributo a ser extraído. A segunda simplificação incide também sobre a extração. Como o objetivo dos experimentos consiste em provar que a arquitetura capacita-se a, cooperativamente, encontrar páginas pertencentes a classes de páginas, as duas classificações – a categorização funcional e a classificação por conteúdo nas subclasses processadas por cada agente – foram consideradas como tarefas primordiais. Como a tarefa de extração requer certa complexidade, como formatação, mapeamento, consistência tanto do dado extraído propriamente dito (como, por exemplo, rejeitar a data 30 de fevereiro) como em relação a outros dados (um evento científico não pode durar um mês), inferências sobre novos dados e classificação (verificar se algum dado de uma tabela está presente no texto, direta ou

indiretamente), foi decidido, para os experimentos, que seria necessário apenas identificar a existência de atributos, sem, no entanto, extraí-los. Com isso, as duas classificações não ficam prejudicadas e os agentes continuam a seguir a premissa de reconhecer páginas de classes a partir de seu conteúdo.Também não foi tratado o problema de eventos iguais em páginas diferentes, que consta entre os cuidados associados à extração.

Outra simplificação diz respeito à concorrência do processamento de páginas. Embora o sistema pudesse ter sido confeccionado de modo a processar várias páginas ao mesmo tempo, foi percebido que, dada a latência causada pelo pré-processamento na geração das representações de uma página, como retirada de palavras das *stop-lists*, freqüência e ordenamento de termos entre outros, os resultados demorariam mais, aparecendo apenas ao fim do processamento do conjunto de páginas simultaneamente tratadas. Para apressar os testes, então, foi processada uma página por vez.

Uma última simplificação refere-se aos tipos de arquivos tratados. Os agentes não estão aptos a processar páginas cujos conteúdos não estejam codificados em HTML, embora que, para o agente de artigos científicos, entre outros futuros agentes do sistema, seria desejável o *parsing* de arquivos de outros tipos como *.ps* e *.pdf*.

6.3.2. Conhecimento Comum a Todos os Agentes

As ontologias comuns aos agentes representam não só o domínio tratado, mas também as classes necessárias à realização das tarefas dos agentes. Os conceitos do domínio científico localizam-se na ontologia Ciência, que também inclui outras ontologias simples, como Tempo e Locais, com classes e instâncias necessárias à definição das entidades do meio científico, como cidades e datas de eventos. Estas ontologias são mencionadas na subseção 4.9.1.

Cada agente possui uma ontologia que contém os conceitos a serem usados apenas pelo próprio agente (item 6(a) da seção 5.4.). Todavia, boa parte das instâncias relativas ao domínio, como a instância *Call-for-papers* definida na subseção anterior, podem ser úteis e reusadas por outros agentes.

6.3.3. Conhecimento Específico

O conhecimento específico de um agente restringe-se ao que ele necessita para o cumprimento de suas tarefas particulares, compreendidos por:

- Ontologias de seu interesse exclusivo que lhe permitam melhores inferências nas tarefas de reconhecimento e extração (o agente CFP prescindiu de uma ontologia de Turismo, com classes comumente citadas em chamadas de eventos, como Hotéis, Moeda Corrente de um país, etc);
- Relacionamentos com os grupos funcionais (casos), instanciando as regras reusáveis;
- Regras para serem enviadas aos outros agentes, para que eles identifiquem ponteiros e páginas para o agente;
- Regras de sugestão recebidas de outros agentes, para o envio a eles de páginas e ponteiros;
- Regras específicas de reconhecimento, extração e, opcionalmente, de validação e correção do reconhecimento.

6.4. Construção de Agentes

Um modelo de agente foi desenvolvido atendendo os requisitos de reuso mencionados na seção 5.6. Uma vez pronto, o modelo foi instanciado na criação do agente CFP para eventos científicos, que ainda não realiza extração. Para a confecção de qualquer agente do sistema, nenhum código precisa ser reescrito; apenas os seguintes itens serão definidos:

- A tabela da base de dados relativa à entidade cujos dados serão extraídos;
- Termos para as consultas aos mecanismos de busca;
- Entradas (palavras-chave) nos dicionários, que estão nos conceitos, usadas nas fases de categorização funcional, classificação e extração;
- Instâncias de relacionamentos entre o agente, os grupos funcionais nos quais as páginas serão classificadas (reconhecidas, listas, mensagens, rejeitadas, etc) e palavras-chave;
- Regras de extração e, opcionalmente, de recomendação para outros agentes;

- Um número pequeno de regras específicas, não reusáveis, de reconhecimento;
- Indicação de quais conhecimentos (regras, e, se necessário, conceitos) serão anunciados aos outros agentes para que eles lhe indiquem um ponteiro ou página. Esse conhecimento pode incluir regras específicas e/ou partes da instância de relacionamento entre o agente e a sua classe de páginas reconhecidas.

6.5. Detalhes de Funcionamento e Especificação

6.5.1. Instâncias e relacionamentos

A entidade extraída das páginas gerará uma instância de um dos conceitos, como eventos, organizações, pesquisadores e publicações, cujos atributos são armazenados num banco de dados. A entidade e a página, neste caso, mantém um relacionamento, ou seja, as situações em que uma página contém uma entidade. Estes relacionamentos estão generalizados como casos incluídos em instâncias do reconhecedor de classes no agente que trata esta classe. Se uma ou mais destas situações acontece, isto indica que o relacionamento existe, e a página passa a pertencer a uma subclasse de páginas relativas à entidade, passando a ser, por exemplo, uma página de artigo. No fundo, neste caso, está sendo realizada uma categorização, ou seja, uma página-conteúdo está sendo reconhecida. Outros relacionamentos e instâncias podem ser enumerados:

- O relacionamento entre páginas relativas a conceitos distintos, indicando que uma contém âncoras apontando para outra (por exemplo, páginas de pesquisadores apontam para artigos), ou que, na busca de uma, foram encontradas páginas da outra (isso ocorreu com freqüência no agente CFP, aonde vinham páginas de organizações como resultado de consultas aos mecanismos de busca).
- O relacionamento entre páginas e grupos funcionais, determinando, por exemplo, quando considerar uma página como mensagem. Para esse grupo funcional específico, o relacionamento vale para todos os agentes, contudo o normal é que cada agente empregue um relacionamento particular, ligando casos a regras reusáveis, para o reconhecimento de páginas de grupos funcionais, contidos em instâncias de reconhecedores de listas, de mensagens, de páginas inúteis e de páginas e *links* a serem sugeridos para outros agentes. Cabe frisar que estes relacionamentos dizem respeito ao

contexto da busca de cada agente, ou seja, do padrão de páginas que retorna dos mecanismos de busca em resposta às consultas requisitadas pelo meta-robô, que devem garantir cobertura sobre a Web, não sendo aplicáveis por outros agentes, que naturalmente, possuem outros contextos. Por exemplo, na busca de páginas de chamadas de trabalho, os mecanismos de busca são consultados com os termos *"call for papers"* e *"call for participation"*. No contexto do conjunto de páginas que retornam dos mecanismos é que as palavras-chave usadas nas regras reusáveis de categorização devem possuir eficácia; nos resultados das consultas de outros agentes os casos não funcionam.

6.5.2. Regras e Reuso

Esta subseção mostrará, na prática, como funciona um relacionamento (um caso) e uma regra reusável. O código abaixo evidencia um caso do agente de artigos científicos, que reconhece uma página, se ela contém no seu início o conceito *abstract* e os atributos *(slots) First-Name* (primeiro nome de um dos autores), *name* (nome da organização a que está afiliado) e *Location-Place* (país ou estado americano onde se localiza a organização).

```
([ppr_00356] of Case
      (Description "aff,1st,loc")
      (Importance HIGH)
      (Absent-Concepts [thesis])
      (Concepts-in-the-Beginning [abstract])
      (Slots-in-the-Beginning
            [First-Name]
            [name]
            [Location-Place]))
```

Por questões de clareza, seguem abaixo os conceitos *abstract* e *thesis,* seguidos da definição do reconhecedor da classe abstrata *Part-Publication*[25] que pode abarcar vários casos, inclusive o caso acima (*ppr_00536*).

```
([abstract] of Concept
      (name "abstract")
      (Synonyms "summary"))
```

---

[25] Para os *frames* de CLIPS, uma classe abstrata não pode possuir instâncias [Giarratano & Riley 98]. *Part-Publication* denota um documento científico, como um artigo ou um capítulo de livro, que constitui parte de uma Publicação Divisível (*Dividable-Publication*, na ontologia Ciência[Freitas 2001]).

```
([thesis] of Concept
      (name "thesis")
      (Keywords
            "partial fulfillment"
            "partial fullfilment"
            "partial fulfilment"))

([ppr_00528] of Class-Recognizer
      (Default-Subclass [Generic-Part-Publication])
      (Cases
            [ppr_00536]
            [ppr_00356])
      (Class [Part-Publication]))
```

O relacionamento é completado por regras reusáveis por instanciação, muitas delas comuns a vários agentes, como a descrita no próximo código.

```
(defrule r_900_slots_hi_funct
      (object (Page-Status STORED)
            (is-a Processing-Monitor))
      (object (Name-of-List "Slot-Found")
            (Instance-Values $?iv&:(> (length$ $?iv) 0)))
      ?f <- (object (is-a Case)
            (Importance HIGH)
            (Slots-in-the-Beginning $?s&:(> (length$ $?s) 0))
            (Concepts-in-the-Beginning $?cb&:(> (length$ $?cb) 0))
            (Absent-Concepts $?ac&:(> (length$ $?ac) 0)))
      (object (is-a Class-Recognizer)
            (Class ?c&:(class-abstractp ?c))
            (Cases $?cs))
      (test (and
            (integerp (member$ (instance-address ?f) $?cs))
            (if-occur (words-of-concepts $?cb)
                  (beginning (slot-get [PAGE] Lowercase-Contents)))
            (not (if-occur (words-of-concepts $?ac)
                  (beginning (slot-get [PAGE] Lowercase-Contents)))
            (subsetp $?s (slot-get [SLOTS-FOUND] Instance-Values))))
      =>
      (modify-instance [PAGE] (Page-Status RECOGNIZED)
            (Classified-Class ?c)
            (Recognition-Rule 900)))
```

A regra pode ser traduzida como:

```
Regra r_900_slots_hi_funct

      Se tenho uma instância do de página em estado ARMAZENADA
      E a lista de atributos que foram encontrados na página
      E um caso de importância ALTA que tem
            Uma lista de atributos que devem estar no começo da página
            E uma lista de conceitos que devem estar no começo da página
            E uma lista de conceitos ausentes da página
      E um reconhecedor de uma classe abstrata com uma lista de casos
      Teste
            Se o caso especificado está na lista de casos
            E se alguma palavra dos conceitos do começo da página está lá
            E se nenhuma palavra dos conceitos ausentes está lá
            E se os atributos do caso estão na
                  lista de atributos encontrados
      Então
      A página passa para o estado de RECONHECIDA como sendo da mesma
            classe abstrata do caso especificado
```

Esta prática economiza a construção de regras *ad hoc* para cada agente, além de estruturarem melhor o conhecimento, já que um caso pode fornecer a base para muitas regras reusáveis.

6.5.2.1. Exemplo de Extração de Atributo

As regras de extração de atributos também são reusáveis. Para instanciar uma destas regras, foi definida uma instância de extrator (classe *Slot-Recognizer*) para o atributo *name* (o nome da organização a que os autores do artigo são filiados), que pode ser vista abaixo:

```
([ppr_00352] of Slot-Recognizer
      (In-the-Beginning TRUE)
      (Concepts-in-the-Beginning
            [corporation]
            [university]
            [institute])
      (Slots-of-Web-Page [Lowercase-Contents-without-Lines])
      (Importance HIGH)
      (Class [Organization])
      (Slot-in-Process [name]))
```

O atributo é de importância alta, será procurado no início da página usando a representação com o conteúdo minúsculo sem linhas da página, e será identificado pelos conceitos *corporation, university* e *institute.* O fato mais interessante e preciso semanticamente desta

definição é a classe a que pertence o atributo, a classe Organização. Aliás, como pode-se constatar abaixo, nenhum atributo dentre os assinalados integra a classe abstrata que foi encontrada, *Part-Publication,* ou alguma de suas subclasses concretas. Porém, isto está bem colocado porque o artigo é elaborado por autores que podem ou não pertencer a uma organização.

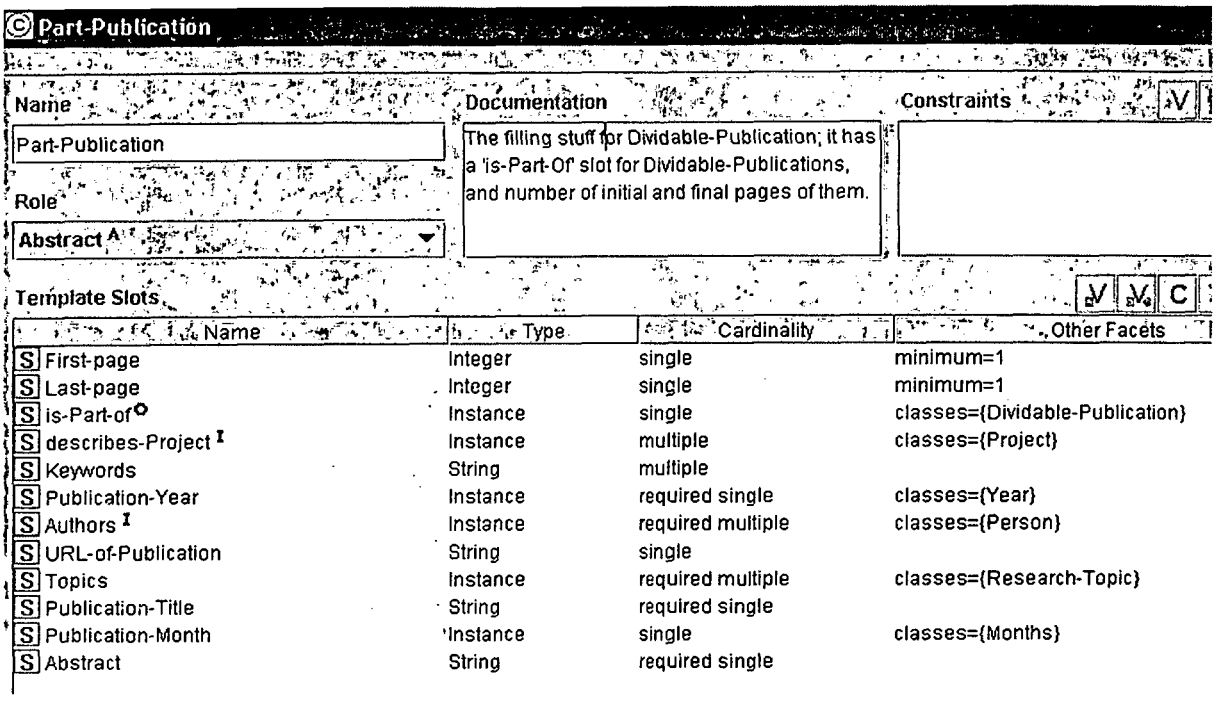

Figura 23. Definição da classe *Part-Publication*.

Assim, o atributo *name* pertencerá indiretamente a uma instância de *Part-Publication*, percorrendo o caminho:

Part-Publication

↳ *Author* (atributo de *Part-Publication*, instância da classe Pessoa)

↳ Filiação (se a instância de Pessoa acima for da classe Pesquisador, atributo da classe Pesquisador, instância da classe Organização)

↳ *name* (atributo da classe Organização)

Cabe a observação de que nem sempre os atributos desejados são "estrangeiros". No agente CFP, os atributos pertencem diretamente às classes procuradas.

6.5.3. Regras Específicas e de Correção de Categorização

Apesar da combinação regras reusáveis-relacionamentos produzir uma boa performance, podem ser necessárias regras específicas para um só agente, relativas a qualquer das tarefas, que dizem respeito ao contexto das encontradas. No agente CFP, uma regra relevante reconhece como lista uma página que tenha mais de quatro períodos (um período é detectado por duas datas próximas temporalmente, separadas por um delimitador, como o caracter "-"). No agente de artigos, um par de regras específicas captura a existência de um nome próprio num artigo antes da primeira linha que contenha um dado sobre a filiação dos autores.

Podem existir, ainda, regras de correção do reconhecimento durante a extração, que se enquadram em dois tipos: as que apontam erros e as reclassificadoras. As primeiras procuram fatos contraditórios e inverossímeis, e.g., a presença de datas com mais de um ano de intervalo entre si denunciam uma página erroneamente classificada como chamada de trabalhos. As reclassificadoras são disparadas após as que apontam erros, e têm uma sintaxe do tipo: "dado que a página *não* pertence à categoria funcional *x* e <condições> , então reclassifique-a como sendo da categoria funcional *y*".

6.5.4. Exemplo do Agente de Artigos Científicos

Agora, será visto o caso em execução, reconhecendo e classificando a página cujo início está ilustrado na figura 24, seguida da listagem do processamento da mesma pelo agente de artigos científicos.

Durante esta execução, o código Java do agente verificou as três filas de URLs, tirou o endereço da página da fila de baixa prioridade, e entregou o controle ao motor de inferência. Foram acessados os dados básicos de conexão (disparo da regra 1), validou-se a página (regra 2), executaram-se corretamente os métodos da fase de pré-processamento (regra 3) coletando os resultados (regra 4), foram achados os atributos (regras 5 a 7), foi reconhecida a página como *Part-Publication* (regra 8) pela regra 900 que instancia o caso *ppr_00536*, mostrados há pouco, a página foi classificada como uma Parte de Publicação Genérica (regra 9), sugestões de *links* foram procuradas para outros agentes entre os *links* e os textos dos *links* da página (regras 10 e 11) e foram devolvidos os resultados e o controle ao código Java.

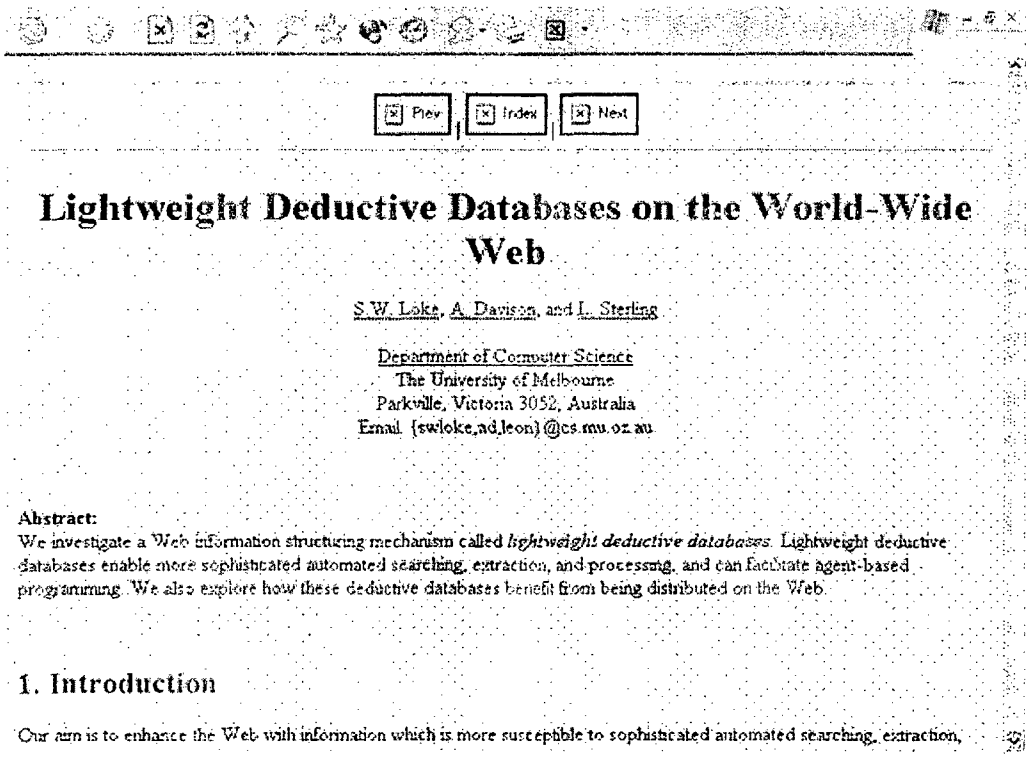

Figura 24. Página reconhecida e classificada pelo agente de artigos científicos.

```
hi Prio (8) Row's length:0
low Prio(5) Row's length:17
to Send Row's length:0
Processing page from row with priority 5
Page : http://brown.lcmi.ufsc.br/ppr/b553.html
FIRE 1 MAIN::i_901_start f-411
FIRE 2 MAIN::v_314_valid f-869, f-867
FIRE 3 MAIN::i_905_filling f-870
FIRE 4 MAIN::i_907_fill-fields f-871
Found country ("Australia")
FIRE 5 MAIN::r_450_slots_hi_funct f-894, f-450,,
SLOT FOUND : Location-Place
FIRE 6 MAIN::r_450_slots_hi_funct f-894, f-68,,
SLOT FOUND : First-Name
FIRE 7 MAIN::r_430_slots_hi_ccpt_bgn f-894, f-593,,
SLOT FOUND : name
FIRE 8 MAIN::r_900_slots_hi_funct f-894, f-957, f-377, f-301,
FIRE 9 MAIN::c_600_recognized_default f-960, f-301
CLASS Generic-Part-Publication
FIRE 10 MAIN::e_203_links f-963, f-748, f-964,, f-790, f-533,
FIRE 11 MAIN::e_203_links f-963, f-378, f-964,, f-845, f-526,
FIRE 12 MAIN::s_002_result f-963, f-971, f-966, f-964
```

```
fact : CLASSIFIED
Inserting as recognized...
CLASS Generic-Part-Publication
Saving Statistics...
hi Prio (8) Row's length:0
low Prio(5) Row's length:16
to Send Row's length:0
```

O reconhecimento em geral recai primeiramente sobre uma classe abstrata - como *Part-Publication* e *Tese* no agente de artigos, e ainda *Evento Científico ao Vivo* ou *Evento de Publicação* no agente CFP – mas há exceções, como as classes concretas *Relatório Técnico* e *Relatório de Projeto* no agente de artigos e a classe Jornal no agente CFP, que têm características peculiares, tornando-as fáceis de serem classificadas diretamente. A partir do reconhecimento da classe abstrata, regras de classificação incluindo casos tentam classificar a página entre as subclasses dela. Caso isso não ocorra, existe uma classe genérica à qual a classe será classificada, como *Parte de Publicação Genérica*, no agente de artigos e *Evento Científico Genérico ao Vivo* ou *Evento de Publicação Genérica* (vide seções 4.81. e 4.9.).

6.6. Concorrência

Um agente deve executar cinco processos concomitantemente. Ao *recebimento* e *envio de mensagens* são atribuídas as mais altas prioridades de execução, pois a troca de mensagens pode trazer "dicas" muito importantes de páginas a processar, que serão tratadas com uma prioridade maior que as páginas advindas do meta-robô.

Os dois *processadores de páginas*, um para a fila de URLs *de alta prioridade* e outro para a de *baixa*, processam uma página por vez, recebendo prioridade de execução menor que a troca de mensagens. O *meta-robô de busca* só deve ser executado se as filas esvaziaram-se.

Em Java, estão disponíveis quatro técnicas básicas de execução concorrente para máquinas dotadas de apenas um processador [Oaks & Wong 97]:

- A *concorrência por prioridade,* em que o processo controlador de linhas de execução distribui o tempo de execução do processador entre os processos candidatos, proporcionalmente à prioridade de cada um;

- A *sincronização de métodos,* em que métodos de um objeto são considerados como *transações atômicas*, ou seja, que uma vez ganhando o controle do processador, o método deve mantê-lo até o fim de sua execução;

- A *sincronização de objetos,* na qual vários trechos de código compartilham um objeto com operações que, se executadas simultaneamente, quebrariam sua integridade (o exemplo clássico é uma conta corrente de banco, em que duas operações efetuadas em caixas distintos, devem ser seqüenciadas, para que o saldo fique correto).

- Os *semáforos*, em que trechos de código enviam e recebem sinais sobre quando ganhar o controle de processamento. Esta técnica, embutida na anterior, consiste na forma mais segura e complexa de realizar concorrência, quando especificada corretamente.

Nos agentes, os processos compartilham entre si as filas de execução e mensagens e o motor de inferência, o que inviabiliza o uso das duas primeiras técnicas, que causariam impasses na execução (*deadlocks*). Determinou-se, ainda, como requisito, que a inferência sobre uma página deveria ser uma transação atômica, para que, do início ao fim da inferência, não se misturassem mensagens do robô e de outros agentes ao processamento de uma página. Assim, optou-se por sincronizar os objetos em questão, elevando também a inferência de páginas à mais alta prioridade, enquanto ela estiver em funcionamento.

6.7. Descrição dos Experimentos

Com cada agente foram realizados cinco testes, cada um processando de 150 a 250 páginas. Para os dois primeiros testes, lançou-se mão de um *corpus* de 400 páginas recuperadas previamente pelo meta-robô rodando à parte do agente. As páginas resultaram de consultas com as expressões *"call for papers" e "call for participation"* sobre os mecanismos de busca *Altavista* e *Google,* no agente CFP, enquanto no agente de artigos científicos foram usadas palavras-chave bastante comuns em artigos: *"abstract", "keywords", "introduction", "motivation", "related work", "overview", "scenario", "results", "discussion", "conclusion", "future work", "references", "bibliography"* e *"acknowledgements"*. Estas mesmas consultas seriam usadas durante a coleta dinâmica de páginas pelo meta-robô.

A repetição de páginas ocorreu em todos os testes, tanto dentro do conjunto de páginas retornadas pelo mesmo mecanismo de busca, como em mecanismos de busca diferentes, em um percentual aproximado de 30%.

O primeiro teste tinha o propósito de adquirir conhecimento, ou seja, examinar cuidadosamente as páginas, com o intuito de identificar e representar padrões que pudessem inferir com eficácia, tanto as categorias funcionais quanto a extração de atributos e a classificação nas quais deveriam ser enquadradas as páginas.

O segundo teste, com um corpus não acessado anteriormente, serviu para a validação antes do agente acessar a Web. Ele produziu resultados com poucas discrepâncias em relação ao primeiro, sugerindo que a aquisição de conhecimento refletiu o conjunto de páginas analisada com fidelidade.

Os testes que se seguiram tinham os agentes conectados diretamente à Web, recuperando páginas a partir de consultas com as mesmas palavras-chave, porém com outros mecanismos de busca, ou com uma diferença de meses entre os testes, para que as páginas não fossem as mesmas. No terceiro teste, os agentes não cooperavam e também não se beneficiavam das listas de páginas-conteúdo que encontravam. No quarto teste, os agentes continuavam separados, mas aproveitavam os *links* contidos nas listas, para tentar melhorar sua performance, ou seja, processar menos páginas inúteis para o agente – as classes lixo, mensagens, páginas auxiliares e páginas de outras classes - e mais páginas das categorias conteúdo, listas e *frames*. Estas duas últimas categorias são consideradas úteis por produzirem diretamente um conjunto de páginas úteis.

Finalmente, o último teste usava as listas e os agentes trocavam mensagens de sugestão de páginas entre si, visando um aumento de eficiência no funcionamento de ambos. Ensejava-se que páginas de eventos contivessem *links* para listas de publicações, e que artigos publicados fornecessem *links* – em menor número – para o agente CFP.

Nos testes na Web e durante a recuperação dos *corpi* para os testes *off line*, regras de validação permitiram que muitas páginas inválidas fossem descartadas antes de sua recuperação, economizando banda passante e tempo de processamento. Nos agentes, notou-se que páginas com tamanho maior que 100 KB deveriam ser e foram rejeitadas,

pois, além de serem inúteis em sua grande maioria, atrasavam muito todo o processo, por seus tempos de recuperação e processamento.

6.8. Resultados

Em ambos os agentes, os resultados de todas as tarefas apresentaram regularidade, como pode ser confirmado pelos percentuais de desempenho apresentados nas tabelas 4 e 5 das próximas páginas. As particularidades de cada agente e seus desempenhos nos testes serão comentadas nas próximas subseções. Cada um dos dados das tabelas desta seção pode ser checado contra as tabelas detalhadas de performance da categorização funcional e da classificação por conteúdo de cada teste dispostas no apêndice B.

6.8.1. Agente CFP

O agente teve como base de sua performance as tentativas de identificar no texto indicadores de 21 atributos das várias subclasses de eventos científicos, garantindo um reconhecimento baseado no conteúdo da página. Durante este processo, 28 casos de classificação por conteúdo, asseguravam o reconhecimento de 8 classes de páginas e de outras categorias funcionais.

As performances do agente de chamadas a eventos científicos de cada tarefa em cada teste estão na tabela 4. As poucas falhas nas tarefas obedeceram a padrões.

Na categorização funcional, os falsos positivos da categoria de reconhecidos (páginas-conteúdo) se caracterizavam por serem páginas muito longas, geralmente sobre um assunto ou de uma comunidade que não pôde ser identificada como uma recomendação. Estas páginas contêm muitos dos termos associados a atributos de eventos, como patrocinadores, tópicos de pesquisa, e outros, e amiúde contêm links para eventos, o que termina por confundir o agente. Por sua vez, os falsos negativos advêm de chamadas de eventos que não empregam o jargão comum a eventos científicos, como os termos *"deadline" e "acceptance date"*, entre tantos outros. Chamadas com poucos dados, mesmo corretos, também se incluíram entre as falsas negativas. Listas foram identificadas pela presença de um número factível de âncoras citando um evento - identificado por heurísticas para siglas e/ou palavras-chave de conceitos como *conferência* e *jornal* – ou por uma quantidade razoável de intervalos de tempo (e.g., 1-4 de dezembro), mas algumas delas foram

rejeitadas. As listas devem ser reconhecidas com precisão, sob pena de fazer o agente processar com alta prioridade um conjunto grande de páginas inúteis.

Agente CFP	Corpus de Aquisição	Corpus Desconhecido	Web sem Listas	Web com Listas
Reconhecimento	97,1	93,9	96,1	96,3
Categorização Funcional	93,8	93,9	93,8	95,7
Classificação	94,9	93,3	92,9	91,7
Páginas Processadas	244	147	129	188

Tabela 4. Percentuais de acerto do agente CFP em suas tarefas, nos vários testes a que foi submetido.

No teste na Web usando os *links* das listas, foram percebidos padrões radicalmente diferentes sob vários aspectos na classificação funcional. Páginas de categorias inúteis sumiram, como as recomendações e mensagens, outras menos úteis diminuíram sensivelmente, como as listas, para dar lugar a páginas-conteúdo, ou seja, as reconhecidas e as páginas *frames* de HTML, essa em número bastante elevado, acima de um quarto das páginas válidas processadas. A figura 25 delineia um gráfico com a distribuição de categorias funcionais entre os testes do agente, mostrando a mudança de cenário na categorização funcional.

Nos testes anteriores, apenas um *frame* tinha sido encontrado. Esta categoria possui boa precisão: dos 48 *frames* testados, apenas 8 não se referiam a eventos científicos. Esta diferença compõe a quinta coluna do gráfico acima.

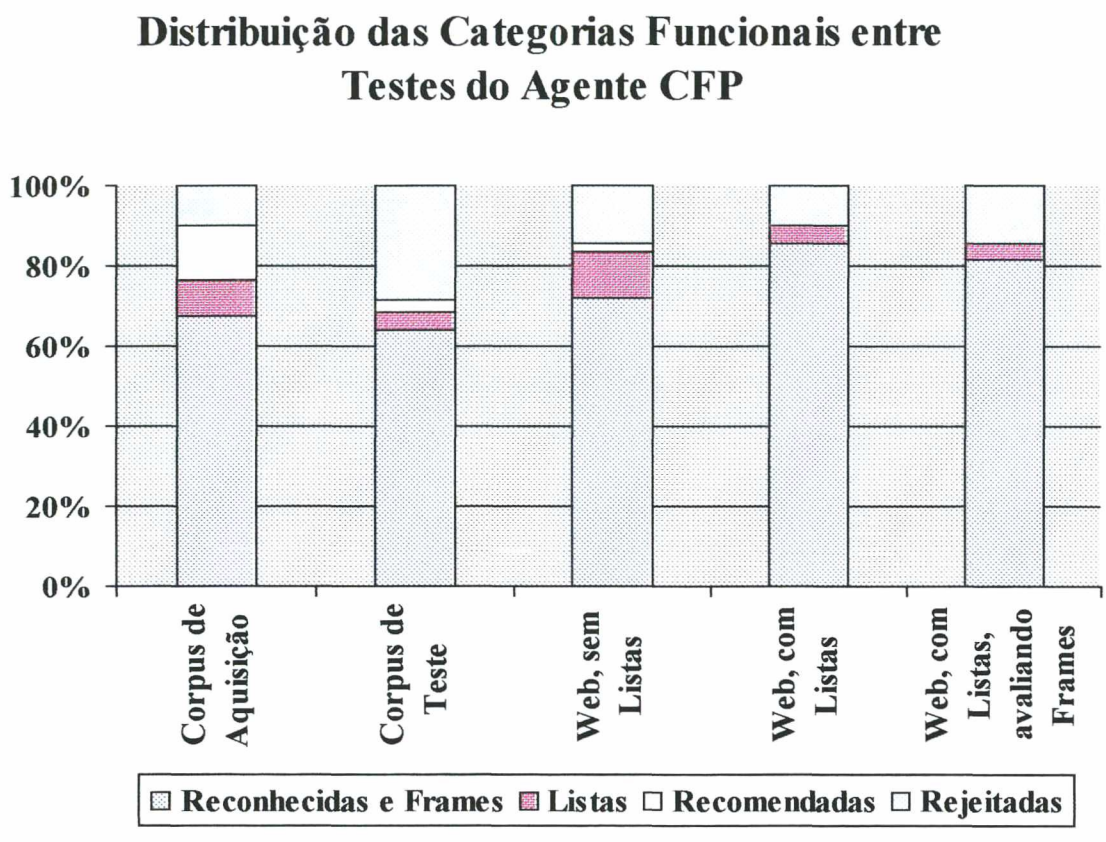

Figura 25. Gráfico comparativo entre os testes do agente CFP, evidenciando o ganho de desempenho com o uso de listas.

Os *links* das várias páginas que compõem um *frame* entram na lista de alta prioridade, porém a abordagem ideal consistiria em representar em conhecimento para inferência o fato de estas páginas se referirem ao mesmo evento.

Outro fato digno de nota destaca o novo padrão entre as páginas rejeitadas. Trata-se de chamadas de eventos, com código HTML que não pode ser tratado, como ponteiros para imagens com texto sobre eventos, páginas iniciais ou de animações de eventos, etc.

Passando agora à classificação por conteúdo, os erros vieram basicamente de quatro fontes, sem alteração de comportamento do agente ao longo dos testes:

- Trocas entre conferências e workshops, presentes em conferências com *links* para *workshops* no início do código HTML da página, ou de *workshops* que, no seu início, dizem que são uma grande conferência ou citações similares;

- Eventos ao vivo com sigla foram classificados como genéricos, quando podiam tê-lo sido como conferências ou *workshops*;
- Eventos ao vivo com atributos típicos de eventos de publicação, como editor e publicação, e vice-versa, como atributos de data inicial e lugar do evento.
- Chamadas de eventos desconhecidos, como projetos, não-previstos na ontologia, que citam eventos ao vivo. Apesar disso, o agente logrou sucesso em eventos de competições científicas, classificadas adequadamente como *Eventos Genéricos ao Vivo*.

O agente deparou-se, ainda, com páginas de eventos conjuntos, em que a classificação em qualquer das classes foi aceita como correta. A extração deve lidar com essa complexidade adicional.

6.8.2. Agente de artigos científicos (PPR)

Para a construção deste agente, houve uma dificuldade adicional: encontrar mecanismos de busca capazes de aceitar um número alto de palavras-chave em suas consultas ou de permitir acesso aos resultados por meio de programas que não fossem navegadores. Foram testados 36 sistemas desta natureza, dos quais puderam ser aproveitados apenas três: o *AltaVista*, o *NorthernLight* e o *Google*. Este último foi acessado emulando um navegador através de campos do protocolo HTTP. Apesar de muito preciso, a quase totalidade de resultados às consultas compunha-se de arquivos de formato *.ps* ou *.pdf*, que não são tratados pelo sistema e formam uma fonte de fornecimento segura de artigos científicos. Os mecanismos de busca traziam muitas páginas grandes e inúteis, que atrasavam o processamento dos experimentos. Mesmo para coletar páginas para os *corpi* dos primeiros testes, estas páginas eram muito freqüentes; triando estas páginas durante a coleta para o corpus de aquisição, de 380 páginas sobraram apenas 167, e para o de testes, 300 ficaram 129.

Foram realizados três testes do agente, uma vez que listas de artigos contidos na seção de bibliografia dos próprios só causaram declínio de performance durante um pré-teste. Outra fonte de listas de artigos plausível para o futuro está nas páginas pertencentes à classe de *Publicações Divisíveis,* como jornais e revistas *on-line*.

Apenas 8 reconhecedores de atributos no começo do artigo, como autor, afiliação, local dela, e-mail, etc, foram suficientes para instanciar 52 casos, sendo 16 de classificação e a maior parte dos outros para rejeitar páginas. Os resultados das tarefas do agente estão sintetizados na tabela 5. O agente caracterizou-se por um comportamento homogêneo nos diferentes testes, que não permite, nesta avaliação, a disposição de um gráfico como o da figura 25.

Agente PPR	Corpus de Aquisição	Corpus Desconhecido	Web sem Listas
Reconhecimento	93,1	82,7	87,8
Categorização Funcional	96,8	94	95,1
Classificação	97	93	81,4
Páginas Processadas	190	150	184

Tabela 5. Percentuais de acerto do agente PPR em suas tarefas, nos vários testes a que foi submetido.

Os erros no reconhecimento devem-se a artigos com poucos atributos, com os atributos numa mesma linha (caso não presente no conjunto de aquisição), com atributos difíceis de identificar, - como a afiliação a uma empresa desconhecida - ou com atributos no fim do artigo. Um artigo deve possuir um conjunto mínimo destes atributos; foram encontrados em todos os testes artigos apenas com o nome do autor, rejeitados muito embora pertençam à classe procurada. Números sobre estes artigos "impossíveis" de serem reconhecidos aparecem na seção B.2. dos apêndices.

6.8.3. Cooperação

O último teste dos dois agentes consistiu no teste integrado, em que eles trocavam mensagens de anúncio de entrada no sistema e suas capacidades (que classe o agente processa), pedidos de páginas de acordo com regras enviadas e a troca de sugestões de páginas úteis, a serem colocadas na fila de URLs de alta prioridade, conforme colocado

anteriormente. Apesar de ter funcionado, apenas três páginas foram sugeridas pelo agente de artigos ao agente CFP e nenhuma página foi sugerida erradamente. Um exemplo detalhado de toda a troca de mensagens e reconhecimento e envio de uma sugestão pode ser seguido na seção A.2. dos apêndices.

Como o número de páginas sugeridas entre os dois agentes foi baixo, optou-se por evidenciar que a cooperação é possível fazendo o agente CFP enviar sugestões de páginas ao futuro agente de pesquisadores, a partir da lista de âncoras encontrada com o auxílio de heurísticas após a localização do atributo *Comitê de Programa*. O trecho de execução apresentado abaixo mostra a eficácia destas sugestões:

```
FIRE 10 MAIN::c_603_recognized_concepts-begin f-2067, f-756, f-139,
CLASS Workshop
FIRE 11 MAIN::d_60_x f-0,, f-2070,
FIRE 12 MAIN::d_51_lists_committee f-2072, f-2071, f-930, f-1760,
Patrick Cousot recommended
FIRE 13 MAIN::d_51_lists_committee f-2072, f-2071, f-927, f-1760,
Vladimiro Sassone recommended
FIRE 14 MAIN::d_51_lists_committee f-2072, f-2071, f-924, f-1760,
Lisbeth Fajstrup recommended
FIRE 15 MAIN::d_51_lists_committee f-2072, f-2071, f-921, f-1760,
Martin Raussen recommended
FIRE 16 MAIN::d_51_lists_committee f-2072, f-2071, f-918, f-1760,
Maurice Herlihy recommended
FIRE 17 MAIN::d_51_lists_committee f-2072, f-2071, f-915, f-1760,
Eric Goubault recommended
FIRE 18 MAIN::e_203_links f-2070, f-411, f-2071,, f-620, f-191,
```

Nos testes, o agente sugeriu 30 *links* corretos e 7 errados. Dado o fato de que não foi empregado nenhum dicionário de nomes próprios, como o faz o sistema DEADLINER [Kruger et al 2000] ou técnica de extração de nomes, como no sistema Nymble [Bikel et al 98], pode-se considerar encorajador o potencial sugestivo dos agentes.

6.9. Discussão dos Resultados

Os resultados da seção anterior mostraram alguns fatos relevantes. A aquisição de conhecimento em ambos os agentes foi suficientemente cuidadosa, assegurando regularidade à categorização funcional e à classificação.

A utilidade das categorias funcionais, encarregadas da tarefa de separar as páginas úteis das inúteis, revelou-se preciosa. Em especial, a utilidade das listas - páginas de *frames* e sugestões de páginas do próprio agente fizeram crescer o percentual de páginas úteis entre 13 a 22% no agente CFP - corrobora a impressão de que a visão por funcionalidade reveste-se de importância fundamental para a manipulação integrada de informação, podendo até localizar páginas-conteúdo não recuperáveis pelos mecanismos de busca. As listas causaram uma mudança significativa no padrão de páginas do agente CFP, tornando-o mais focado: sumiram as páginas de outras classes, e mesmo as páginas que o agente não reconheceu se deveram não à falta de precisão da recuperação e categorização, mas a páginas iniciais de eventos científicos que só tinham gráficos e pouco ou nenhum texto.

Do fato da cooperação entre os agentes ter gerado poucas sugestões de endereços de páginas, tiram-se certas lições. Em primeiro lugar, uma possível função do sistema seria a de *mapear os padrões de conexão entre as classes de páginas*. Antes dos experimentos, a suposição motivadora era de que freqüentemente chamadas de trabalho continham âncoras para listas de publicações, e também que, em contrapartida, um número menor de páginas de artigos possuíam *links* apontando para páginas dos eventos onde os artigos foram publicados. Nenhum dos fatos se confirmou na prática. A compensação, no entanto, surgiu na fácil e hipotética cooperação entre o agente CFP e o agente de pesquisadores, que gerou sugestões de *links* com uma precisão superior a 75%. Portanto, a cooperação pode converter-se numa forma muito útil de buscar páginas de classes com pouca estrutura, como páginas pessoais de pesquisadores.

Na medida em que mais agentes forem sendo produzidos, a cooperação poderá ocorrer com mais freqüência, fazendo com que os resultados tendam a melhorar. É comum, por exemplo, que páginas de pesquisadores incluam listas de publicações, que melhorariam substancialmente a performance do agente de artigos científicos. Aliás, é bastante provável que esta cooperação possa se tornar a principal fonte de páginas do agente PPR.

6.10. O Mediador do Sistema MASTER-Web

Prevendo a construção de outros agentes do sistema, foi implementado um mediador para o sistema MASTER-Web, que facilitará consultas efetuadas por agentes de software ou usuários. A figura 26 mostra como o mediador centralizará as consultas sobre as bases de

dados montadas pelos agentes extratores do sistema. A figura evidencia ainda, mais uma vez, o agente CFP como um agente extrator genérico, que usa o meta-robô para acessar a Web através de vários mecanismos de busca. O processamento ou extração das entidades das páginas gera registros para as bases de dados extraídos.A escolha de componentes para o mediador baseou-se no aspecto popularidade: como visão foi escolheu-se usar um esquema global em SQL (vide subseção 5.7.2.), e como mecanismo de comunicação, o protocolo KQML, ideal para comunicação entre agentes.

O primeiro protótipo do mediador trata comandos SQL encapsulados em mensagens KQML. O código foi projetado de maneira que outras linguagens, como o KIF ou o Jess, também possam futuramente ser tratadas. Um agente usuário, antes de utilizar o mediador, deve registrar-se em um roteador KQML responsável por enviar suas mensagens ao mediador. Ao receber a mensagem de um agente através do roteador, o mediador armazena a mensagem numa fila de espera. Quando chegar ao momento de ser processada, a mensagem é imediatamente retirada da fila e o *parser* para KQML, disponibilizado pelo JATLite, é acionado para separar as variáveis KQML a serem utilizadas durante o processamento (linguagem, ontologia, identificação do agente remetente, etc, vide subseção 3.2.2.3.).

As operações permitidas restringem-se a consultas à base de dados ou a operações relativas à modelagem do agente usuário, com a criação e manipulação de um *login* com senha e as preferências do usuário, no caso do agente CFP, as áreas de pesquisa que lhe interessam. Por ora, a modelagem do usuário baseia-se numa representação por *estereótipo* [Rich 89], em que um conjunto de atributos (no caso, áreas de pesquisa) pode assumir um conjunto fechado de valores.

Para validar as operações, foi feito um pequeno *parser* com o intuito de separar *logins* e senhas do usuário, e saber se ele está apto a concluir as operações requisitadas que alteram o banco de dados (apagar ou atualizar registros sobre o seu modelo ou perfil de usuário). Caso a validação seja negativa, uma mensagem de erro encapsulada em KQML é retornada ao agente que a enviou. Caso seja positiva, o comando SQL é entregue ao JDBC que se encarrega de executá-lo ou de gerar exceções, caso ocorra algum erro. Depois que todo o processamento é realizado pelo JDBC, a função do mediador fica restrita a encapsular a resposta numa mensagem KQML e a enviar para o agente que a requisitou. A mensagem

de retorno pode conter a resposta para o comando SQL do agente, ou uma exceção gerada pelo JDBC.

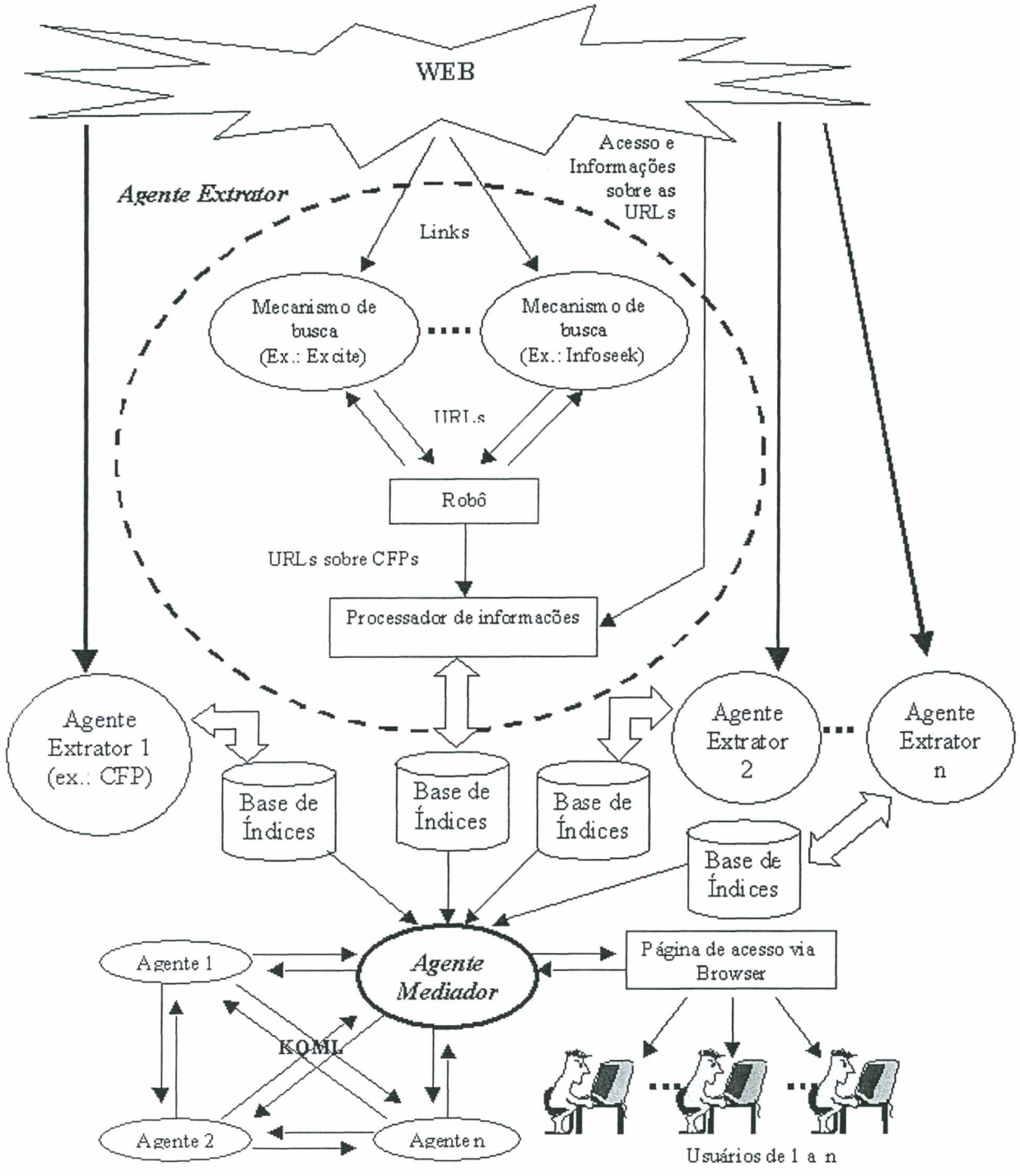

Figura 26. Arquitetura do Sistema MASTER-Web, mostrando interações do Agente Mediador com os usuários e agentes externos e com agentes extratores. No detalhe, a busca e processamento das páginas por um agente extrator [Freitas et al 99].

Outra forma de interagir com o mediador é utilizando sua página de acesso na Internet, disponível para agentes e usuários. Através de um navegador, o usuário obtém as visões das bases de índices que lhe interessam, e assim, pode realizar suas consultas. As páginas de inclusão de usuário e suas preferências, bem como de uma resposta do mediador a uma consulta do usuário encontram-se na figura 27.

6.10.1. Visões de Bancos de Dados como Adaptadores

Este trabalho inova na modelagem de mediadores [Freitas et al 99]. Baseando-se na existência de arquiteturas de conectividade como ODBC (*Open Database Connectivity*) [ODBC 98] e JDBC (*Java Database Connectivity*) [JDBC 98], que possuem SQL como linguagem de interação, foi proposta e implementada uma arquitetura de mediadores que troca os adaptadores por visões de bancos de dados, por economia no desenvolvimento em vários aspectos.

O *parser* que trata as consultas preocupa-se exclusivamente com *segurança*, ou seja, não permite que os usuários consultem tabelas indevidas, ou que possam realmente alterar informações além das que se refiram ao seu perfil de usuário. Assim, o desenvolvedor não precisa se preocupar com erros de sintaxe em SQL na consulta, referências a tabelas, campos ou relacionamentos inexistentes, tipos de dados e mais uma gama de situações que poderiam causar erro. O tratamento destas situações fica a cargo do mecanismo de conectividade (JDBC ou ODBC) ou do próprio Sistema Gerenciador de Banco de Dados (SGBD), e a mensagem com erro, será repassada de volta ao usuário. Obviamente, o trabalho de modelar uma visão simplificada para os usuários tem que ser realizado. As respostas a consultas corretas são geradas automaticamente pelo JDBC ou ODBC ou mesmo pelo SGBD, e também repassadas ao usuário no formato em que são recebidas pelo mediador. Não é necessária a montagem de mensagens de respostas instanciando campos e traduzindo formatos, entre outras tarefas. Portanto, evita-se a complexidade do desenvolvimento de adaptadores, com mecanismos de tradução e verificação, e de resolver o problema adicional de qual é a melhor consulta equivalente à consulta solicitada. Esse problema já foi atacado através de planejamento simbólico [Arens et al 96], uma sub-área de Inteligência Artificial Simbólica.

A arquitetura de mediadores proposta apresenta escalabilidade, uma vez que instâncias de mediadores podem ser distribuídas pela rede ou Internet para evitar gargalos, a exemplo dos roteadores KQML (vide subseção 3.2.2.3.). Unicamente no caso em que houver restrições de espaço em disco e as bases de dados forem muito volumosas, o modelo seria desaconselhável, devido ao espaço em disco que uma visão pode vir a ocupar.

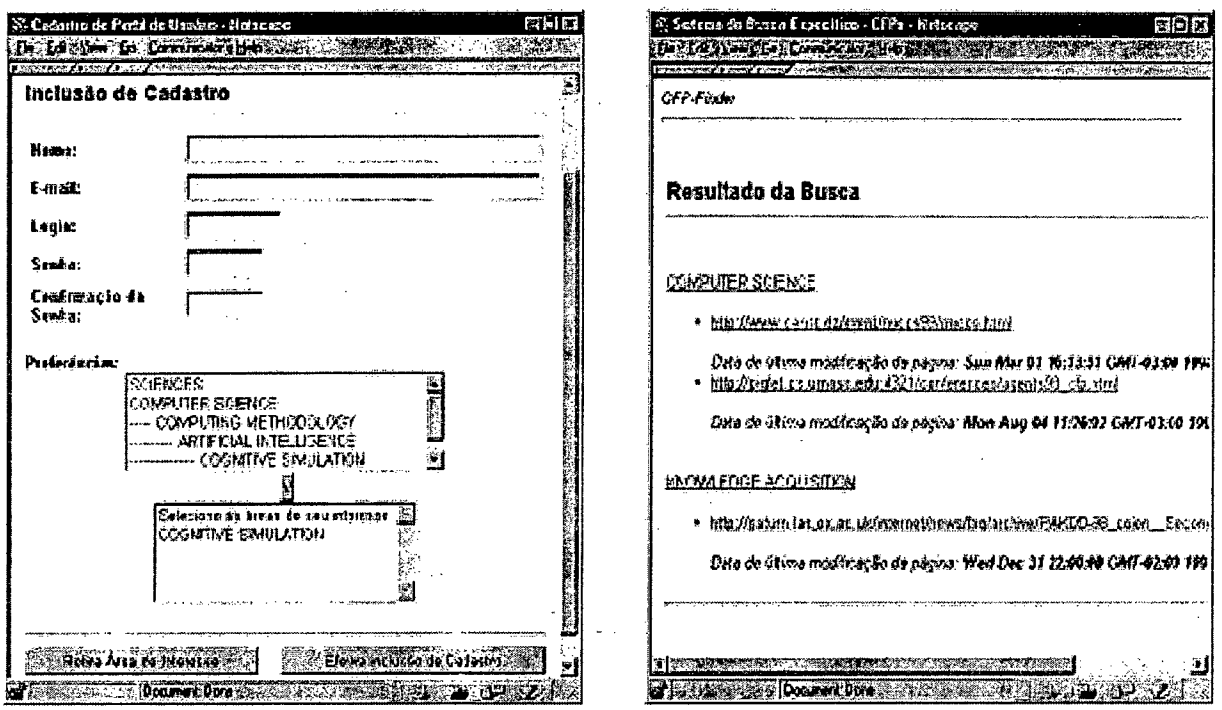

Figura 27. Páginas de interface do mediador para usuários. A primeira cadastra o usuário e as áreas de sua preferência, enquanto a segunda apresenta os resultados de uma consulta.

Capítulo 7

TRABALHOS RELACIONADOS

7.1. Recuperação de Informação

7.1.1. Tarefas

A chegada da Internet e seu manancial de informação desestruturada deram margem ao nascimento de outras tarefas ligadas à busca de informação relevante e útil, além da recuperação de informação. Surgiram a tarefa de extração - citada na seção 1.3. - e várias outras decorrentes da tarefa de classificação (ou categorização), a saber:

- *Filtragem*: No fundo, pode-se considerá-la como uma categorização de documentos em duas classes; a dos documentos selecionados ou considerados relevantes, e a dos desprezados ou irrelevantes.

- *Agrupamento (Clustering)*: Consiste em separar os documentos em grupos por similaridade, medindo-os e comparando-os matematicamente (vide subseção 7.2.1.) por um limiar. Neste caso, as classes ou grupos nos quais os documentos serão organizados não são pré-definidas. Esta tarefa possui várias utilidades; funciona, por exemplo, como base para a expansão de consultas para a busca de documentos similares e potencialmente relevantes em sistemas de recuperação [Baeza-Yates & Ribeiro-Neto 99]. Serve também em sistemas de recomendação de itens a usuários, e ainda para a criação de hierarquias de classificação [Chen et al 96a].

- *Roteamento*: A tarefa de notificação - ou seja, o aviso a um usuário específico sobre uma nova informação - originou esta tarefa, que perfaz uma classificação de documentos em classes pré-definidas para que eles possam ser encaminhados aos usuários interessados nestas diversas classes, de acordo com perfis de usuário preenchidos por eles de antemão.

Após serem discutidos trabalhos relacionados à integração entre as tarefas básicas do problema, as tarefas de extração, categorização e recuperação, serão descritas abordagens atuais que suportam soluções para estas três tarefas.

7.1.2. Complementaridade entre Extração, Categorização e Recuperação

A idéia de integrar as três tarefas possui antecedentes, e inspira-se na necessidade de um processamento adicional após a recuperação de documentos, para determinar a real relevância de seus conteúdos. O acoplamento entre recuperação e extração já havia sido proposto em alguns trabalhos[26]. O uso de recuperação como uma entrada para a extração é aplicado em vários sistemas, como o VIE (*Vanilla Information Extraction System* [Gatzauskas & Robertson 97]) e o Academia [Magnanelli et al 97]. Existem, ainda, experimentos em que foi constatada uma melhora de precisão em sistemas de recuperação através de uma posterior extração, como os experimentos com chamadas de trabalhos (*Call for Papers)*, realizados em Glasgow [Lazarinis 97], e com o sistema de extração FASTUS [Bear et al 98], avaliado como filtro de documentos relevantes após a recuperação.

Outro interessante projeto, o sistema BIG [Lesser et al 97] (do inglês *resource-Bounded Information Gathering*, ou manipulação de informação com recursos limitados), conjuga as duas tarefas, fundamentado numa premissa completamente pioneira. Para seus idealizadores, o usuário objetivo possui necessidades de informação, atendidas por páginas que estão espalhadas em conjuntos de páginas de diferentes tipos; por exemplo, para comprar um artigo, além de observar preços, os usuários verificam dados técnicos de desempenho, segurança, prazos, critérios, comparativos, etc. Assim, para realizar este trabalho, um sistema multiagente auxilia o usuário a planejar e escalonar a recuperação e extração dos dados, com intensivo emprego de raciocínio e aprendizado para um possível aumento de performance.

Há também sistemas que unem as tarefas de categorização e extração para recomendar documentos - que com freqüência representam itens, como livros, CDs e mesmo páginas - de interesse dos usuários, como o Libra [Mooney et al 98], que sugere livros. A

categorização baseia-se no comportamento do usuário, e os documentos ou itens são agrupados por similaridade, o que, na verdade, constitui uma tarefa de *clustering,* e não de categorização.

Também a pesquisa em mecanismos de busca específicos encontra suporte em serviços já disponíveis na Internet. O Miner [Miner 2000] é um sistema nacional de metabusca, ou seja, que, para a tarefa de recuperação, utiliza resultados filtrados de outros mecanismos de busca. Os sistemas de busca específica Cora [McCallum et al 1999] e CiteSeer [Bollacker et al 99], para artigos científicos em informática, incorporam as três tarefas: buscam novos artigos, categorizam-nos automaticamente por técnicas de aprendizado de acordo com as áreas em que se enquadram, e extraem entidades, como as áreas dos artigos, autor, título e referências.

Conforme visto nos dois últimos exemplos, o meio científico eletrônico forma uma excelente massa de teste para vários tipos de experimentos concernentes ao problema apresentado, por sua razoável estrutura, o que confirma a escolha dele como estudo de caso. No entanto, com exceção destas últimas ferramentas, nenhuma das abordagens acima se beneficia da complementaridade entre as tarefas de recuperação e extração, ou seja, além da recuperação seguida de extração, como alguns efetuam, a exploração e extração de âncoras relevantes como base para uma recuperação mais segura e contextualizada. Neste trabalho, ao invés de focalizar apenas uma classe, como os artigos científicos do CiteSeer e do Cora, deseja-se também extrair os ponteiros úteis ao grupo de classes inter-relacionadas, procurando, portanto, aproveitar o contexto das páginas da forma mais completa possível. Para esse intuito, a melhor tecnologia disponível, senão a única, reside nas ontologias, que podem definir apropriadamente o grupo de classes tratado. O sistema WebKb [Craven et al 99] encontra dados científicos através de aprendizado, empregando ontologias para a categorização e extraindo atributos para as entidades definidas, porém sem realizar busca da forma usual. Ao invés disso, o sistema processa *corpi* de faculdades, uma por vez. Um comparativo entre o MASTER-Web e este sistema encontra-se na subseção 7.4.1.2.

---

[26] A imensa quantidade de artigos e abordagens relativos à area de recuperação de informação e suas tarefas correlatas, além do trabalho de outras áreas envolvidas por este trabalho, como bancos de dados, ontologias e multiagentes, impede uma revisão extensiva de todos os trabalhos da área. Assim, possivelmente, durante este capítulo, deixaremos de citar trabalhos relevantes que também guardem relação com os itens mencionados.

7.2. Abordagens em Recuperação de Informação

Considerando as três tarefas (recuperação, extração e categorização), as diversas propostas de solução para problemas na área de recuperação de informação foram separadas em dois tipos de abordagens: as abordagens estatísticas e as baseadas em processamento de linguagem natural. As abordagens estão descritas logo abaixo.

7.2.1. Abordagens Estatísticas

São aquelas que seguem sempre as seguintes premissas básicas:

- Os documentos (ou partes deles[27]) são representados em vetores de palavras-chave e freqüências, podendo ainda conter outras peculiaridades, como, por exemplo, ordem das palavras, que ajuda a identificar a presença de *n*-gramas (termos com *n* palavras-chave em seqüência, como "redes neurais sem peso").

- A representação não inclui palavras muito freqüentes, que atrapalhariam a indexação, e são especificadas num dicionário próprio pelo qual são reconhecidas, chamado de *stop-list*.

- As palavras são reduzidas a seus morfemas por algoritmos de *stemming* [Baeza-Yates & Ribeiro-Neto 99]; por exemplo, *engineer* e *engineering* são reduzidas para *engine*.

- A tarefa a ser realizada (recuperação, categorização e/ou extração) buscará palavras-chave nos vetores que representam os documentos, ordenando os documentos por freqüência destas palavras-chave, opcionalmente utilizando outras palavras-chave - cuja probabilidade de estarem presentes nos documentos é alta - e/ou seus respectivos pesos ajustados pela relevância destas novas palavras-chave.

Portanto, a tarefa a ser realizada é efetuada através de um algoritmo matemático que atua sobre os vetores que representam os documentos, indicando como solução o conjunto ordenado de documentos que melhor atende à tarefa solicitada. A heurística matemática mais comum em sistemas de representação é conhecida como algoritmo TF-IDF (do inglês

---

[27] Existem, ainda, tarefas ligadas à divisão de documentos, como a *sumarização*, que pretende encontrar uma parte, região ou conjunto de palavras-chave dentro de um documento que melhor o represente, e a *segmentação*, cuja proposta é dividir o texto em partes de acordo com o assumto que tratam.

term frequency – inverse document frequency). O algoritmo atribui pesos a palavras-chave na representação de um documento de acordo com duas medidas:

- A freqüência da palavra-chave, chamada na fórmula abaixo de *TF(p),* onde *p* é a palavra-chave,
- A freqüência em documentos da palavra-chave, chamada sob a mesma convenção de *DF(p),* que significa o número de documentos no qual a palavra-chave *p* está presente.

A fórmula que representa o peso de uma palavra-chave num documento será dada por

$$TFIDF(p) = TF(p) * log(d / DF(p)),$$

onde *d* é o total de documentos do conjunto procurado. O algoritmo visa salientar palavras pouco freqüentes em outros documentos, que, portanto, são mais representativas do documento representado, ou, conforme o jargão da área, são palavras *discriminantes.* Para uma consulta de busca com várias palavras-chave, por exemplo, são totalizados os pesos de cada uma em cada documento, e os documentos com peso maior que zero são ordenados de forma decrescente e apresentados como o conjunto resposta.

Os mais diversos tipos de algoritmos matemáticos, probabilísticos e de aprendizado automático já foram testados em RI, com o fito de conferir aos sistemas vantagens que os peculiarizem, como adaptatividade, usadas por algoritmos que implementam aprendizado, e busca por conceito, como o algoritmo LSI (do inglês *Latent Semantic Indexing).* Com efeito, são encontrados trabalhos com abordagens as mais diversas, com algoritmos envolvendo trigonometria para verificação de similaridade de documentos, probabilidades de documentos serem relevantes em relação a consultas, lógica difusa, algoritmos genéticos, redes neurais, redes de inferência, redes de crença e redes de inferência [Baeza-Yates & Ribeiro-Neto 99].

7.2.1.1. Aprendizado de Máquina

As técnicas de aprendizado de máquina prestam valioso serviço às três tarefas de RI citadas, especialmente à tarefa de categorização, promovendo flexibilidade à representação dos documentos em relação a mudanças lingüísticas e conceituais. Esta utilidade deve-se aos seguintes fatores:

- A renovação vocabular de uma língua, que, dinamicamente, cria, exclui, populariza e condena ao ostracismo determinados termos, tornando-os comuns ou raros;
- A expansão do conhecimento humano, gerando novas teorias e conseqüentes terminologias;
- A existência de "vocabulários regionais", fator que não se restringe às gírias e expressões de uma região, mas também, conforme observam Steier e Belew [Steier & Belew 93], ao vocabulário de uma instituição, ou, segundo a terminologia dos autores, os seus "conjuntos de subcontextos lingüísticos socialmente definidos", que diferem significativamente de outra instituição, mesmo em meios de pesquisa, e que, com freqüência, migram entre os meios até tornarem-se populares, perdendo sua "opacidade".

Os benefícios do emprego de aprendizado para a área de recuperação de informação são inúmeros. Para agrupamento, por exemplo, o reconhecimento automático de novas classes resolveria um problema extremamente complexo, a criação de espaços conceituais [Chen et al 96]. A tarefa de categorização em classes pré-definidas quase sempre se baseia em alguma forma de aprendizado, seja ela neural, estatística ou simbólica. Os algoritmos de aprendizado simbólicos demonstraram uma melhor performance numa comparação com algoritmos estatísticos e neurais [Lamounier 95], todavia seu maior benefício continua sendo a capacidade de explicitar o que foi aprendido em regras. Esta característica revela-se fundamental para a arquitetura, que se baseia em conhecimento explícito, aproveitando as regras aprendidas não só para comunicação e cooperação entre agentes, como também para reuso de conhecimento. Uma possível extensão da arquitetura reside justamente na inclusão de algoritmos de aprendizado explícito, como o RIPPER [Cohen 96], para a classificação das páginas em categorias funcionais e para a extração de atributos das entidades destas páginas.

7.2.2. Abordagens baseadas em Processamento de Linguagem Natural

As técnicas de Processamento de Linguagem Natural (PLN) são capazes de processar textos escritos num subconjunto de um idioma. Têm por propriedade um alto custo computacional, principalmente porque as várias fases empregam representações diferentes; devido a isto, soluções de PLN costumam ser empregadas em tarefas de *background*, como

categorização e extração. Contudo, existem sistemas que as empregam de forma superficial, sem aplicar todas as tarefas e técnicas possíveis, visando otimizar a precisão de sistemas de recuperação com um tempo de resposta aceitável. Este tipo de solução foi aplicado com sucesso em bibliotecas digitais, onde o fato de os documentos guardarem semelhança estilística e temática constitui um fator preponderante.

O sistema FAQ-finder [Burke et al 97], por exemplo, recebe como entrada uma pergunta em linguagem natural, e procura num conjunto de arquivos de perguntas freqüentes (*Frequently Asked Questions ou FAQs)*, representadas como redes semânticas, as que mais se adequam ao que foi indagado, apresentando, naturalmente as respostas. Partes específicas do dicionário semântico WordNet [Miller 95], que define as palavras também através de redes semânticas são empregadas, e os experimentos comprovaram uma maior precisão do que as técnicas baseadas apenas em buscas estatísticas.

A seguir, serão brevemente descritas as fases consecutivas de processamento de linguagem natural. Ressalte-se que, conforme visto no exemplo do FAQ-Finder, nem todas essas fases são obrigatórias, e que, estas fases podem também ser separadas em dois grandes grupos [Grishman 97]: a *análise do texto*, que compreende desde a separação de palavras até a análise semântica, e a *análise do discurso*, que envolve as fases de análise de discurso propriamente dita e análise pragmática.

7.2.2.1. Separação ou Tokenização

Esta é uma fase muito simples, pois nela é efetuada apenas a separação de palavras ou *tokens*; nas línguas ocidentais, isso constitui uma tarefa banal. Porém em línguas orientais, as palavras e sinais de pontuação encontram-se misturados, exigindo, para seu processamento um módulo de segmentação de palavras [Appelt & Israel 99].

7.2.2.2. Análise Léxica e Morfológica

Aqui, cada palavra é desflexionada, se necessário, e uma classe morfológica é designada a cada uma delas (substantivo, adjetivo, etc), bem como seu gênero, número e tempo, para o caso dos verbos [Barros & Robin 96]. Consultas a um dicionário conhecido como léxico fazem parte do processo de identificação morfológica, reconhecendo o sentido em que ela é empregada, de acordo com sua posição no texto. Os léxicos não precisam conter todas as

palavras do idioma, mas devem conter as palavras mais comumente usadas, e, principalmente, as palavras relativas ao domínio a que se destina o sistema.

7.2.2.3. Análise Sintática

Nesta fase, as palavras são vistas como constituintes de frases, e o papel de cada uma delas é determinado através de bases de conhecimentos portáveis, independentes de domínio, as *gramáticas*. À medida que o processamento vai ocorrendo nesta fase e na seguinte, etiquetas são colocadas em palavras e partes do texto, denotando os papéis delas e também das orações e das partes das orações.

7.2.2.4. Análise Semântica

Se as gramáticas são reusáveis, o mesmo não se aplica à base de conhecimento que dá apoio à análise semântica, conhecida como *modelo do domínio*. Para que um sistema de PLN interprete corretamente o conteúdo das frases e seja capaz de inferir fatos triviais ou complexos a partir delas, um conhecimento de "senso comum" sobre o domínio modelado, enumerando os *papéis temáticos* [Barros & Robin 96] das entidades é aplicado sobre o texto, gerando instâncias representadas em dois tipos de formalismos: os fracos, que definem a forma de representação, como redes semânticas e *frames* (vide subseção 4.4.2.), e os fortes, como scripts, gramáticas de caso e *templates* [Appelt & Israel 99], que trazem parte do conhecimento do domínio. Por exemplo, exceto no caso da voz passivo, o papel temático de agente é realizado pelo sujeito; existem ainda co-agentes, beneficiários e temas, entre outros papéis.

7.2.2.5. Análise do Discurso

Após a análise semântica, o significado de cada frase está representado, mas não o seu conjunto. A inferência citada na fase anterior só poderá ser efetuada com a ligação de entidades em frases distintas que representam uma mesma entidade. Por exemplo, no texto:

"O policial perseguiu a ladra. Ele estava motorizado".

o sujeito da segunda frase refere-se a uma entidade já existente na primeira. Este problema, conhecido em PLN como *anáfora,* e em extração como *co-referência*, provoca enormes dificuldades de robustez, e demanda extensa pesquisa, onde técnicas baseadas em

conhecimento e em aprendizado são utilizadas, porém ainda sem solução geral. No sistema Alembic, a representação explícita trouxe ao menos dois benefícios para a inferência [Villain 99]: a detecção de formas sintáticas diferentes que redundam numa mesma interpretação semântica e a resolução de co-referência, mesmo a aposicional, como em:

"Rivaldo marcou o segundo gol. O jogador pernambucano fez sua melhor partida na temporada."

Outra técnica baseada em conhecimento consiste no algoritmo lógico conhecido como *unificação* [Ait-Kaci & Podelski 93], que auxilia o processo de ligação entre frases e entidades – já que dados estáticos sobre uma mesma entidade podem encontrar-se espalhados num texto – lançando mão de casamento de padrões sobre dados incompletos – ditos *parciais* - para identificar as mesmas entidades. Ele aparece embutido no interpretador das linguagens voltadas para PLN, como Prolog e LIFE, empregando estruturas de dados que otimizam sua eficiência, de forma a permitir inferências bastante complexas.

7.2.2.6. Análise Pragmática

Com todas as frases – em tese - interpretadas e contextualizadas entre si, o texto pode ser compreendido; porém a mensagem intencional, ou o ato de fala [Austin 62] pode permanecer oculto. A análise pragmática intenta revelar o(s) tópico(s) a que se refere(m) um texto, e envolve tarefas como segmentação do texto em *focos*, percebendo quando o texto muda de um foco para outro, agrupamento de focos, formando *segmentos*, e reconhecimento nos segmentos de um *propósito* ou intenção [Barros & Robin 96]. Este tipo de análise encontra-se presente em poucos sistemas, devido às dificuldades inerentes à tarefa, tais como tempo de processamento, especificação do conhecimento e falta de padronização das técnicas.

7.3. Extração de Informação

Segundo Kushmerick [Kushmerick 99a], extração de informação "é a tarefa de identificar os fragmentos específicos de um documento que constituem o núcleo de seu conteúdo semântico". A exemplo dos sistemas de PLN, e como qualquer aplicação de Inteligência Artificial, aqui também é obedecido o requisito de restrição de domínios; ou seja, os

extratores atuam apenas sobre classes de documentos que guardam entre si regularidade de formatação, estrutura e conteúdo. Os extratores processam textos em geral pequenos e ricos em dados [Embley et al 98], com exceção do WebKb (vide subseção 7.4.1.) e da arquitetura aqui apresentada neste trabalho. Ambos almejam extrair dados de páginas na Web, que não apresentam uma formatação pré-estabelecida.

Existem basicamente dois tipos de sistemas de extração: os sistemas baseados em processamento de linguagem natural e os *wrappers*. Ambos os tipos beneficiam-se de aprendizado automático para garantir escalabilidade e portabilidade, proporcionando um desenvolvimento rápido de novos extratores pela automatização parcial ou total do processo de aquisição de conhecimento. Após serem detalhados os dois tipos, serão abordados os papéis, benefícios, vantagens e abordagens da aplicação de ontologias em sistemas de extração.

7.3.1. Processamento de Linguagem Natural

Os extratores baseados em processamento de linguagem natural trabalham sobre textos escritos sobre um determinado assunto, como terrorismo, micro-eletrônica [Riloff 94] e sucessão de cargos em empresas [Gaziauskas & Robertson 97]. Os textos possuem regularidade de estrutura e conteúdo, mas de não formatação, e muitos deles provém de uma mesma fonte. Integrantes de um mesmo domínio, os textos usados em extração baseada em PLN são classificados entre as classes de semi-estruturados e desestruturados.

7.3.1.1. Premissas e Características

Nem todos os sistemas de extração abarcam todas as fases de PLN descritas na seção anterior; em alguns deles, uma análise sintática superficial é suficiente para uma boa performance [Appelt & Israel 99], e a análise pragmática não tem utilidade em extração. Extratores de nomes próprios, por exemplo, perfazem apenas até a fase de análise léxica e morfológica [Appelt & Israel 99].

Um grande problema para qualquer sistema de PLN é o fato de que um fato pode ser expresso de várias maneiras [Gaziauskas & Robertson 97]. Por isso, a modelagem do domínio resulta na tarefa mais cuidadosa destes sistemas. A análise semântica, para a qual servirá esta modelagem, irá demonstrar resultados consistentes na medida em que:

- Forem escolhidas as partes adequadas de um dicionário semântico, como o WordNet, que se adequem ao domínio, e
- Sejam criados *templates* suficientes para instanciar as diversas formas de expressão dos fatos que se deseja extrair.

Após a análise semântica, a análise do discurso ajuda a inferir a extração de atributos, e também a minimizar problemas de co-referência, eliminando instâncias de *templates*, que contém fatos inverossímeis. Extratores baseados em PLN representam o conhecimento explicitamente, facilitando qualquer processo de inferência que se siga durante ou após a extração.

7.3.1.2. Exemplos de Sistemas

Os extratores baseados em PLN empregam regras aprendidas para reconhecer características sintáticas e semânticas no texto. O sistema LIEP (*Learning Information Extraction Patterns*) [Huffman 95] aprende regras com ambas as características, reconhecendo relacionamentos que envolvem os constituintes das frases. As regras do LIEP necessariamente incluem mais de um elemento do texto, identificando o trecho exato onde ocorre o dado extraído [Muslea 99].

Acrescente-se que estes sistemas apresentam um modelo do domínio com papéis lingüísticos e semânticos que lembram os conceitos e relacionamentos das ontologias, porém sem um rigoroso engajamento ontológico. O sistema AutoSlog [Riloff 94] constrói dicionários extração pela análise de *corpi* previamente anotados, definindo nós conceituais declarativos que representam os papéis citados. Por exemplo, a frase:

"The Parliament was bombed by the guerrillas."

gerou o seguinte nó conceitual [Muslea 99]:

```
Name:                   target-subject-passive-verb-bombed
Trigger:                bombed
Variable Slots:         (target (*SUBJECT* 1))
Constraints:            (class phys-target *SUBJECT*)
Constant Slots:         (type bombing)
Enabling Conditions:    ((passive))
```

A palavra "*bombed*" disparou o reconhecimento dessa frase como sendo do domínio de terrorismo, do tipo bombardeio (*bombing*) e a frase, por seu padrão sintático de voz passiva, reconhece que o alvo (*target*) do bombardeio é o sujeito e agente da passiva da frase (*The Parliament*).

A melhor contribuição deste sistema consiste em evidenciar a fácil portabilidade de sistemas de extração baseados em PLN, uma vez que os testes foram realizados com resultados satisfatórios em pelo menos três domínios (terrorismo, micro-eletrônica e *joint-ventures*), diminuindo o processo de aquisição de conhecimento de, em média, 1500 homens-hora com engenharia de conhecimento, para apenas cinco homens-hora usando aprendizado automático, obtendo resultados 98% iguais ao da engenharia do conhecimento [Riloff 94]. Estudos mostram, porém, que a engenharia do conhecimento ainda supera em performance as abordagens baseadas em aprendizado automático, especialmente em relação aos problemas de co-referência, onde, ademais, o processo de anotação revela-se muito capcioso [Appelt & Israel 99].

7.3.2. Wrappers

Outro tipo de extratores, os *wrappers*, foi projetado com o pressuposto de que existem documentos com muita informação útil, especialmente na Internet, estruturados ou semi-estruturados em forma de tabelas, planilhas ou registros, e que apresentam forte regularidade de formatação, estrutura e conteúdo. Os *wrappers* visam, portanto, preencher bases de dados com informações extraídas a partir de seus delimitadores, que podem ser sinais de pontuação (vide exemplo na subseção 7.3.3.1.) ou *tags* em HTML. Algumas técnicas foram sugeridas e testadas: existem *wrappers* que usam gramáticas de compilação [Ashish & Knoblock 97; Atzeni et al 97], autômatos finitos [Appelt & Israel 99] e mesmo simples formatação [Kushmerick 99]. Exemplos de domínios em que foram aplicados os *wrappers* incluem dados geográficos [Ashish & Knoblock 97], previsão do tempo e programação de cinemas [Kushmerick 99].

Naturalmente, este tipo de extrator requer uma estrutura das páginas até certo ponto rígida, não conseguindo capturar o contexto com a flexibilidade desejada. Algoritmos de aprendizado também se revelaram úteis aos *wrappers,* e são empregados no intuito de resgatar, compreender e explicitar a formatação das páginas. O aprendizado ou *indução* de

wrappers [Kushmerick 99] possui a vantagem de adaptar-se imediatamente a diferentes formatos de documentos, exigindo apenas a anotação das páginas que servirão como exemplos para o algoritmo de aprendizado.

7.3.3. Ontologias em Extraçào

Apesar de já ter sido aplicada para extração, como será visto nesta seção, uma hipótese deste trablho é que o potencial das ontologias ainda não foi completamente percebido, particularmente para o novo problema aqui apresentado, a extração integrada. As ontologias podem reunir e combinar sobre um mesmo ambiente os três tipos de conhecimento necessários aos processos de extração: o conhecimento sobre reconhecimento de estruturas sintáticas e semânticas do texto - podendo-se inclusive incluir PLN -, o conhecimento sobre reconhecimento de diferentes formatos em textos processados - que não empregam PLN, como nos *wrappers* -, e o conhecimento para a realização de inferências com engajamento ontológico, ou seja, empregando fatos conhecidos a respeito das entidades extraídas. Esta singular situação pode acarretar o crescimento dos ramos de aplicação de sistemas de extração, de páginas e *sites*, como atualmente, para grupos de classes de páginas, conforme delineados no capítulo 2.

Serão abordadas duas formas já exploradas de empregar ontologias para extração, com seus respectivos exemplos, vantagens e desvantagens. A primeira fundamenta-se numa visão típica da área de banco de dados, enquanto a segunda baseia-se em aprendizado automático e será descrita na seção seguinte.

7.3.3.1. Extração com Ontologias Baseadas em Bancos de Dados

A exemplo dos *wrappers*, a abordagem baseada em bancos de dados volta-se também ao processamento de uma classe de textos pequenos e ricos em dados, como anúncios de classificados e obituários. Observe-se um exemplo com um classificado de jornal com o anúncio de um carro à venda [Embley et al 98]:

```
'97 CHEV Cavalier, Red, 5 spd, only 7,000 miles on her.
Previous owner heart broken!  Asking only $11,995.  #1415.
JERRY SEINER MIDVALE, 566-3800
```

Do anúncio, o sistema proposto por Embley et alii [Embley et al 98] retirou os seguintes atributos, mostrados logo abaixo: ano, fabricante, modelo, características (no caso, cor e

quantidade de marchas), rodagem, preço, dono e telefone. O primeiro dado é o valor atribuído ao atributo; os dois seguintes a posição de início e término de onde ele se localiza no texto.

```
Descriptor/String/Position(start/end)
Year|97|1|3
Make|CHEV|5|8
Model|Cavalier|10|17
Feature|Red|20|22
Feature|5 spd|25|29
Mileage|7,000|37|41
KEYWORD(Mileage)|miles|43|47
Price|11,995|108|114
Owner|JERRY SEINER MIDVALE|124|143
PhoneNr|556-3800|146|153
```

O protótipo provê ferramentas de definição de ontologias, a partir das quais são geradas automaticamente as palavras-chave, constantes, relacionamentos, regras de extração e o esquema normalizado do banco de dados. Apesar da economia de desenvolvimento, essa abordagem traz muitas limitações:

- As ontologias especificam apenas o conhecimento estritamente necessário à extração, e, portanto, não possuem engajamento ontológico (vide seção 4.3.), inclusive tornando o conhecimento não-reusável;
- As ontologias não estão expressas num formalismo lógico de representação de conhecimento, o que inibe inferências e comunicação em nível de conhecimento entre diferentes extratores;
- A falta de comunicação entre extratores impede a cooperação entre eles, e, em conseqüência disso, uma possível extração integrada.

A próxima seção é dedicada a sistemas similares ao MASTERWeb, traçando um paralelo entre este trabalho e outros sistemas existentes.

7.4. Sistemas Similares

Esta seção descreverá sistemas que realizam tarefas similares às do MASTERWeb. O primeiro, o WebKB, usa ontologias numa abordagem baseada em aprendizado para as

tarefas de classificação e extração, enquanto o segundo - na realidade dois sistemas, o CiteSeer e o DEADLINER – mostra a busca e extração de dados de artigos e eventos científicos, respectivamente. Após descritos os sistemas, será apresentado um resumo das diferenças, com vantagens e desvantagens entre eles.

7.4.1. WebKB: Classificação e Extração Baseadas em Aprendizado usando Ontologias

O sistema WebKb [Craven et al 98] aprende automaticamente regras de categorização e extração de páginas na Web, empregando uma ontologia do domínio que possui entidades e relacionamentos, definida num formalismo de representação que pode permitir inferência. As páginas da Web são representadas, enquanto sistema de recuperação de informação, com título, palavras-chave, freqüências e ponteiros, e o sistema executa extração integrada.

7.4.1.1. Vantagens e Desvantagens

A decisão de aplicar aprendizado automático para qualquer tarefa de manipulação de documentos depende de uma comparação entre os custos de anotação de *corpi* e o trabalho de inspeção das páginas e aquisição do conhecimento [Appelt & Israel 99]. Existem vantagens em usar algoritmos de aprendizado, como velocidade, adaptabilidade e flexibilidade da solução a novas estruturas de formatação, e desvantagens como legibilidade, engajamento ontológico das regras aprendidas – que tendem a ser muito específicas -, e dificuldades de aproveitar conhecimento *a priori* e de fazer com que a solução capture regras de razoável complexidade – a não ser pagando o tributo de um alto custo computacional e de anotação de *corpi* de ser utilizada uma porção de características para o aprendizado. Um exemplo de uma regra aprendida pelo WebKB coloca a relevo estes fatores. A regra abaixo reconhece páginas de alunos de ciência da computação de faculdades americanas:

```
student(A):-
not(has_comment(A)),link_to(B,A),has_james(B),has_paul(B),
not(has_mail(B))
```

Traduzindo: uma página (A) é reconhecida como sendo de estudante de computação se não contiver comentários e for apontada por um ponteiro de outra página (B) que não contenha

a palavra "*mail*", mas contenha uma das palavras "*james*" ou "*paul*". A regra considera que páginas que instanciem (B) como páginas de turmas de ciência da computação americanas.

Se por um lado, fatos interessantes são aproveitados, como a descoberta de que páginas pessoais não constam comentários e de que páginas de turmas não possuem endereços eletrônicos explícitos, recomendar que todas as turmas tenham ao menos um aluno com o nome de James ou Paul, e não mencionar nada referente ao curso de computação trata-se de uma abordagem de generalização precária ou, no mínimo, sem engajamento ontológico.

O sistema apregoa o uso de aprendizado para a Web, mas pelos próprios experimentos, verifica-se que sua aplicação adequa-se mais a sítios extensos, como os de universidades americanas, do que a regiões ou domínios da Web.

7.4.1.2. Comparativo com o Sistema MASTERWeb

Como o sistema WebKB se propõe a processar sítios de faculdades, uma ontologia que representa este domínio, com, basicamente, três entidades é empregada. São elas: atividades - como projetos e cursos -, pessoas - em seus variados papéis, como estudante, professor, membro do *staff*, e outros -, e departamentos. Relações também estão presentes, como instrutores de cursos, membros de projeto, orientadores, etc. Esta ontologia naturalmente precisa ser mais simples que a ontologia Ciência do MASTERWeb, porque além do domínio ser menor, diminuindo-se o número de características dos algoritmos de aprendizado, a captura dos padrões para as tarefas de classificação e extração se torna mais fácil.

O WebKB corresponde aproximadamente ao trabalho de três futuros agentes do sistema MASTER-Web - o de pesquisadores, o de projetos e o de organizações -, com interseções e exclusões de lado a lado (agentes do MASTERWeb não processam cursos, agentes do WebKB não processam relações da entidade pessoa ligadas ao domínio científico, como publicações, eventos dirigidos, etc). O MASTERWeb visa tratar o domínio científico sob o prisma de pesquisa, enquanto o WebKB processa sítios de faculdades.

O MASTERWeb é ontologicamente mais rico, por abarcar o meio de pesquisa como um todo, com relações mais complexas e capacidade de inferência já durante a classificação e extração. Por isso, ele exige mais esforço na confecção de agentes para cada classe de

páginas. O WebKB adapta-se rapidamente a novos domínios e usa heurísticas estatísticas de padrões de conexão entre páginas e de palavras-chave, não processando expressões, enquanto o MASTERWeb baseia-se em palavras-chave e expressões associadas a conceitos, contidas em *links* para sugeri-los a outros agentes, se estes processam classes de páginas que se relacionam semanticamente com os conceitos referidos.

Com relação a performances, o MASTERWeb tem clara preponderância. Os autores do WebKB avaliam a classificação apenas através dos falsos positivos, reportando percentagens entre 73 e 83 %, exceto para as classes *membro do Staff* e *outros* (correspondente à categoria funcional de rejeitadas). Contudo, se computados apenas os falsos negativos, a classe *outros* tem boa performance (93,6%), a classe estudante tem 43% e as outras seis classes comportam-se abaixo de 27%, algumas inclusive abaixo de 10%, baixando a média de acerto para apenas cerca de 50%. A extração obtém entre 65 e 78% de performance e não foi testada ainda no MASTERWeb.

No MASTERWeb, foram computados tanto falsos positivos como negativos para a categorização funcional e para a classificação (vide apêndice B). A categorização funcional obteve mais que 93% de performance nos dois agentes e o reconhecimento, onde se somam apenas as páginas reconhecidas da classe e as não reconhecidas, obteve 82,8 e 87,8% nos testes com *corpus* desconhecido e na Web do agente de artigos científicos PPR e mais de 93% em todos os outros testes. A classificação esteve sempre acima dos 91%, exceto no teste na Web do agente PPR (81,4%).

Um outro fato relevante da classificação realizada pelo WebKB, não mencionado pelos autores, reside na grande incidência da classe *outros*: esta classe recebeu aproximadamente três vezes mais instâncias que a soma de todas as outras classes, 65% delas erradas. Esta classe inclusive piorou significativamente o desempenho da classificação, pois 90% dos erros eram causados por instâncias desta classe. Porém, mesmo que estivessem corretas, seria estranho o fato de que quase três quartos das páginas não conseguissem ser classificadas numa das classes da ontologia, restando apenas a opção de classificá-las como *outros*. Até certo ponto, isso coloca em xeque a abordagem dos autores, levantando várias hipóteses:

- A ontologia empregada no WebKB não foi suficientemente abrangente e completa. Note-se que a ontologia de Ciência usada pelo MASTERWeb não padece deste problema: apesar de algumas classes, como projetos, produtos e outras, não terem sido usadas, mesmo assim elas integram a ontologia, pois é possível que os agentes necessitem destes conceitos para o desempenho de suas funções, além de abrir a possibilidade de cooperação com futuros agentes que tratem delas.

- Os algoritmos de aprendizado não conseguem obter uma generalização tão grande, o que ratifica o emprego de multiagentes ao problema. Talvez um sistema multiagente com apenas um agente por classe atingisse melhores resultados.

- Será a idéia de processar sítios inteiros de universidades factível e compensatória, mesmo com uma boa ontologia? Em sítios grandes como estes, é comum serem encontradas páginas muito específicas ou com informações estritamente pessoais, que não têm nenhuma importância ou relação com uma ontologia geral acadêmica ou científica. É possível que um agente filtrando mecanismos de busca possa oferecer melhor precisão e rapidez, processando menso páginas desnecessárias. Só uma análise destes sítios pode esclarecer esta hipótese.

Os trabalhos futuros de ambas as abordagens do WebKB e do MASTERWeb direciona-se à integração entre representação de conhecimento e aprendizado. O WebKB relata a necessidade de explorar melhor estruturas lingüísticas semelhantes aos processos de extração; neste trabalho, pretende-se usar aprendizado para extração, classificação e cooperação, conforme especificado na subseção 5.3.

7.4.2. Os Sistemas CiteSeer e DEADLINER

Estes sistemas foram desenvolvidos em épocas diferentes pelo grupo de pesquisa da Universidade de Austin, Texas, que depois se mudou para o instituto de pesquisas da multinacional de telefonia NEC. Ambos consistem em sistemas muito bem concebidos sob o ponto de vista de classificação e extração, usando métodos estatísticos e de aprendizado automático típicos da área de recuperação de informação, e conhecimento *a priori* acerca da informação desejada, especialmente na busca de eventos científicos.

7.4.2.1. O CiteSeer

O CiteSeer [Bollacker et al 98] é um sistema bastante completo para busca de artigos científicos em computação. Como o MASTERWeb, ele emprega um meta-robô para localizar bons pontos de partida para a procura de artigos na Web, procurando em *links* a partir destes pontos e repetindo o processo nesses *links*. As páginas encontradas passam também por uma fase de pré-processamento, principalmente para converter arquivos do formato Postcript e PDF em texto, aproveitando software produzido para este fim. O CiteSeer baseia-se, principalmente, nas citações de cada artigo, inspirando-se em trabalhos anteriores que sugeriam indexar trabalhos científicos pelas citações, classificando sua relevância de acordo com o número de citações em outros artigos.

Um artigo científico é reconhecido pela existência de uma seção de referências ou bibliografia. Constatou-se, porém, que artigos desconsiderados pelo agente de artigos PPR do MASTERWeb por falta de dados dos autores no seu início também não foram encontrados ou classificados pelo CiteSeer. De qualquer forma, a heurística de existência de bibliografia é duvidosa, pois, nos testes realizados neste trabalho, deparou-se com uma grande quantidade de listas de bibliografias, que seriam erroneamente classificadas como artigos, nesse caso.

Os artigos e citações da base de dados mantida pelo CiteSeer são acessados através de palavras-chave procuradas nos campos de título, autor e corpo do artigo, de acordo com a escolha do usuário. Este software disponibiliza 200.000 artigos e dois milhões de citações, podendo ser obtido gratuitamente para fins não-comerciais.

7.4.2.2. DEADLINER

O DEADLINER [Kruger et al 2000] monitora a Web, *newsgroups* e e-mails enviados a listas em busca de anúncios de conferências, das quais são extraídos os atributos data inicial, final e limite, comitê de programa, afiliação de cada membro do comitê, temas, nome do evento e país. Uma página de conferência precisa conter um título descrevendo o evento, uma lista de tópicos de pesquisa, um comitê de programa, datas limite e informações para submissão de artigos. Neste aspecto, o agente CFP do MASTERWeb oferece mais cobertura e flexibilidade: são aceitas quaisquer páginas de chamadas de trabalho, de jornais, capítulos de livros, *workshops*, revistas, competições, mostras, e

outras, sem quaisquer exigências diretas a respeito da presença de atributos, exceto as codificadas dentro dos casos, que possuem muita variedade, não se restrigindo a algum atributo em especial. O agente CFP foi projetado desta forma tendo em mente que a comunidade de futuros usuários se interessa por quaisquer chamadas de eventos, desde que incluam os tópicos de pesquisa de interesses dos usuários.

Um ponto interessante do sistema DEADLINER é que ele só busca informação atual, e a busca parte de *newsgroups*, universidades e organizações de profissionais. É empregado um robô com um complexo algoritmo de navegação focalizada, que lança mão de grafos de contexto [Diligenti et al 2000], um eficiente algoritmo de busca de páginas que promete encontrar páginas relevantes percorrendo menos páginas irrelevantes. O meta-robô do agente CFP do sistema MASTERWeb reusa serviços dos mecanismos de busca existentes, evitando o trabalho de navegação e aproveitando uma base de índices já ordenada por relevância, e sua performance cresceu ainda mais com o uso de listas. Possivelmente o agente CFP acessa menos páginas irrelevantes que o DEADLINER, porém o artigo não traz dados a respeito da performance de recuperação das páginas.

Cada atributo no DEADLINER é extraído através de um grande número de filtros de extração posicionais, similares aos dos *wrappers*, e a integração entre estes filtros é efetuada usando técnicas de aprendizado. A performance de extração ultrapassa os 70% em todos os atributos extraídos, e uma atenção especial foi dada ao atributo *data limite*, que será provavelmente um dos mais empregados no acesso aos registros de eventos. A data limite estava certa em 86% dos casos. Para a extração dos pesquisadores que integram o comitê de programa, o sistema se beneficia da base de dados pesquisadores gerada pelo CiteSeer, sendo esta a única maneira do DEADLINER aproveitar o trabalho do Cite Seer, e vice-versa.

A performance de reconhecimento do DEADLINER é muito alta, acima de 95%, todavia deve-se levar em consideração o fato de que o agente CFP alcança uma percentagem similar processando páginas muito mais variadas e com menor padronização que o DEADLINER.

À parte de sua performance, o modelo empregado pelo DEADLINER possui menos generalidade do que, por exemplo, o sistema AutoSlog [Rillof 94]. No DEADLINER, um

documento é aceito numa classe unicamente se possui um conjunto de atributos fixo. O AutoSlog, e também os agentes do MASTERWeb aqui apresentados, generaliza a aceitação de documentos através de casos, que possuem uma estrutura mais flexível, como pode ser visto nas subseções 5.5.3., 6.5.2 e na seção A.2. dos apêndices.

O maior defeito destes dois sistemas reside no fato de que, mesmo sendo confeccionados pelo mesmo grupo de pesquisa, ambos deparam com *links* que interessam ao outro, e, não os repassam. Abordando esta questão sob o prisma de uma possível multiplicação de extratores pela Internet, pode-se considerar este fato como um problema de representação de conhecimento: os dois sistemas (e outros que podem vir a surgir) não podem expressar intenções variadas como pedir páginas ou sugerir *links,* porque não possuem ontologias do domínio ou de páginas da Web. Além do mais, o conhecimento destes sistemas está escondido dentro de algoritmos, não sendo possível o compartilhamento deles para outros sistemas, nem a especificação de contextos em que seriam úteis.

7.4.3. Quadro Comparativo entre os Sistemas

A tabela 6 ilustra uma comparação das capacidades e qualidades dos sistemas discorridos nesta seção, junto com o protótipo deste trabalho, o sistema MASTERWeb.

Sistemas de MI baseados em ontologias	Web	Abrangência	Cobertura	Precisão	Flexibilidade da Represntação	Aprendizado e Extração	Cooperação e Integração	Dicionários
WebKB	Sítios	v	Sítios		v	v		
CiteSeer	v	v	v	v		v		v
DEADLINER	v			v		v		v
MASTERWeb	v	v	Futuro	v	v	Futuro	v	Futuro

Tabela 6. Quadro comparativo das capacidades e qualidades dos sistemas de manipulação de informação (MI) WebKB, CiteSeer, DEADLINER e MASTERWeb.

O WebKB tem como única vantagem o uso de aprendizado, não tratando páginas da Web, e sim sítios de universidades ou empresas (mencionada como trabalho futuro). A abrangência, definida como a capacidade do sistema de processar páginas sem exigir a presença de certos itens. O DEADLINER não possui abrangência por exigir, com muita rigidez, a presença de vários itens de dados.

A cobertura sobre a Web garante a viabilidade de um eventual produto comercial. O CiteSeer é o único com esta característica, podendo o MASTERWeb, no futuro, alcançá-la. Apenas o WebKB não atinge uma precisão satisfatória e as grandes vantagens do MASTERWeb residem na flexibilidade da representação de conhecimento, por aceitar novas classes a serem processadas sem necessitar de alteração de programas, e a cooperação entre agentes que expressam intenção. A integração entre as bases de dados a serem extraídas é garantida apenas no MASTERWeb, porque elas baseiam-se numa ontologia de domínio comum aos agentes. Por outro lado, os sistemas CiteSeer e DEADLINER já geram e consultam dicionários que já ajudam na extração, como o de pesquisadores e cidades, o que só poderá ser feito no futuro no MASTERWeb.

Ainda há muito a ser feito no MASTERWeb, como o uso de aprendizado e a extração dos atributos, mas a flexibilidade de seu projeto permite que estas capacidades sejam adicionadas ao sistema num futuro próximo.

7.5. Sistemas Multiagentes

As tecnologias para sistemas multiagentes já começam a demonstrar um grau de maturação sem, contudo, que isso esteja refletido em sistemas ou produtos comerciais em uso no cotidiano de usuários, muito embora se identifique claramente uma gama de aplicações em que eles se adequam, como sistemas de telecomunicações, transportes, reservas e outros. Existem, ainda, áreas em que eles despontam como grandes promessas. Talvez, a maior delas seja a resolução de problemas ligados à Internet ou a grandes redes corporativas, relacionados ao tratamento de um grande volume de informações, como mineração de dados, indexação de informação, gerenciamento de conhecimento em organizações e, principalmente, comércio eletrônico, onde o advento das ontologias oferece grandes perspectivas.

Estas tecnologias em fase de maturação podem ser divididas em dois grandes grupos. O primeiro é constituído pelas propostas de modelos de sociedades de agentes inteligentes, onde são estudadas estratégias de interação, cooperação e negociação, onde redes de contratos, leilões, dependências e outras teorias sociais, além de questões ligadas a segurança, tais como formas de garantir confiança e estabelecer e gerenciar reputação de agentes. O outro grupo é formado pelos papéis sociais dos agentes numa sociedade, tratando de temas como papéis organizacionais, comportamento dinâmico e convergência, e cujas principais técnicas são a corretagem (*brokering*) e o encontro de agentes com necessidades complementares (*matchmaking*) [Decker et al 96], sem contar as já mencionadas mediação e facilitação.

Tais técnicas, envolvendo muita complexidade, serão úteis sem dúvida quando a disseminação de agentes tornar-se uma realidade. Ao desenvolver este trabalho, ensejou-se estimular a transformação desta teoria abstrata em prática concreta: foram desenvolvidos uma arquitetura e um protótipo de sistema multiagente cognitivo, de modelo muito simples, mas que permite aos agentes comunicação em nível de conhecimento e cooperação, e que pode vir a ser empregada em ambientes dinâmicos, com agentes se anunciando, enviando e recebendo pedidos e sugestões.

7.5.1. Sistemas Multiagentes para Recuperação de Informação

Trabalhos relacionados a sistemas multiagentes para recuperação de informação enquadram-se tipicamente na abordagem de Resolução Distribuída de Problemas (vide subseção 3.2.1.1.), onde os agentes ou grupos deles responsabilizam-se por tarefas bastante distintas dentro do sistema, como monitoração, planejamento, etc. Na arquitetura proposta neste trabalho, os agentes não estão estruturados desta forma. Eles apresentam estrutura e funcionalidades iguais, obedecendo à filosofia de que agentes devem ter capacidades múltiplas, em oposição à visão anterior de uma só capacidade muito especializada, típica de sistemas especialistas [Maes 97]. Uma das vantagens do MASTER-Web reside no reuso massivo na construção de agentes (vide seção 5.6.), deixando aos desenvolvedores praticamente apenas a aquisição e codificação do conhecimento específico de cada agente (vide subseção 6.3.3.). Em contra-partida, alguns destes sistemas com agentes diferentes possuem uma estrutura mais complexa, que prometem resultados com maior grau de refinamento, ainda que com menor abrangência. Serão apresentadas, a seguir, uma

variedade de sistemas multiagentes, com tipos, metas e abordagens distintas, para que se possa ter uma idéia da aplicação de sistemas multiagentes com relação ao problema de manipulação de informação na Web.

Em primeiro lugar, cade frisar que a abordagem multiagente não havia, ainda, sido empregada para extração. Alguns projetos incluem extração como refinamento para a recuperação, como o projeto BIG, já citado na subseção 7.1.2., que visa a recuperação de informação através de planejamento colaborativo com o usuário, com a possível execução de extração sobre documentos relevantes à consecução das etapas planejadas. Além de agentes extratores e planejadores, o BIG inclui ainda um agente para assessorar o usuário na formulação do planejamento da recuperação, outro para fornecer e comparar custos relativos às alternativas de solução das etapas durante o planejamento, e um agente escalonador. As tarefas ou etapas são divididas em subtarefas entregues dinamicamente aos agentes que estiverem disponíveis e estejam aptos a resolvê-las, através de um modelo de comunicação em "quadro-negro", onde os agentes informarão numa área de acesso comum a todos, o andamento e resultados de cada etapa, na medida em que elas forem sendo solucionadas.

O sistema InfoSleuth [Bayardo Jr et al 96] aplica a tecnologia de multiagentes para responder a consultas de usuários ou agentes sobre volumosos bancos de dados federados, distribuídos numa ambiente aberto. O sistema lança mão de ontologias que provêem a semântica necessária às consultas, aplicando técnicas de repasse (*brokering*) e planejamento para atender às consultas, uma vez que alguns recursos do ambiente podem estar indisponíveis no instante em que são solicitados. A integração de novas bases de dados é também facilitada pelo uso de ontologias, e o sistema como um todo poderia ser classificado como um mediador multiagente. Diferentemente da arquitetura aqui proposta, onde cada agente exerce várias funções, porém com bases de conhecimento diferentes, as tarefas específicas do sistema InfoSleuth estão entregues a cada agente. O sistema é composto de um Agente de Repasse, um Agente de Recursos, um Agente Monitor, além de outros agentes. Uma vez que os agentes dependem uns dos outros para a execução e sincronização de suas tarefas, o sistema inclui agentes com a função específica de monitorar a interação dos agentes e a evolução do cumprimento de suas tarefas.

Já o sistema multiagente para recuperação de informação Oyster [Müller 99] segue um modelo típico de aprendizado automático, para realizar, em tempo de execução, busca de páginas do meio científico, classificadas automaticamente de acordo com seu tipo. Como esse processamento teria um *throughput* muito alto para os usuários, o sistema multiagente visa apressar e paralelizar os processos que o envolvem, enviando, num tempo definido pelo usuário a resposta à sua consulta via e-mail. Cada pedido de um usuário é dividido em subtarefas que serão alocadas dinamicamente por agentes capazes de desempenhá-las, também através de um modelo de comunicação em "quadro-negro". No sistema, estão presentes agentes de busca, de interface, de modelagem individual dos usuários, *wrappers* (com a função de extrair os dados para a classificação) e agentes de *daemons* (para iniciar processos).

Finalmente, o sistema IMPS [Crow & Shadbolt 98] é um sistema multiagente para a construção de ontologias para auxílio à busca de informação na Web. As ontologias são extraídas a partir de textos pré-selecionados, inclusive para funcionar sobre um método de inferência previamente definido, considerado adequado para o domínio. Existem, no sistema, dois tipos de agentes: Agentes de Extração de Conhecimento e Agentes de Construção de Ontologias. O sistema emprega ainda dois conjuntos de arquivos de código, um para extração de conhecimento e outro que fornece informação suplementar sobre as tarefas a serem desempenhadas, além da base de dados e do dicionário semântico da língua inglesa WordNet. Uma curiosidade sobre esse sistema é que ele corrobora a escolha de ferramentas para a confecção do MASTER-Web, usando, segundo os autores, as ferramentas-padrão emergentes em compartilhamento de conhecimento: a linguagem Java, o motor de inferência Jess, a comunicação em linguagem KQML através do pacote JAT (que, posteriormente, evoluiu para JATLite), porém realiza comunicação em KIF (*Knowledge Interchange Format* – vide subseção 4.5.1.), e não em JessTab (Jess acrescido dos *frames* do Protégé), como nos estudos de caso descritos neste trabalho.

Capítulo 8

CONCLUSÕES E TRABALHOS FUTUROS

8.1. Contribuições

A problemática referente á manipulação de informação em grandes redes como a Internet, colocou questões de difícil solução para tornar fácil o acesso ao grande manancial de informação disponível pelos usuários. Áreas como recuperação de informação, agentes inteligentes, ontologias, classificação e extração, e modelagens da Web subitamente incluíram-se entre os temas de maior pesquisa no campo da informática. A pesquisa nestas áreas têm, continuamente, tentado prover soluções adequadas, mas ainda encontram-se em fase de maturação, e longe de soluções gerais, de alta performance.

O presente trabalho foi iniciado partindo da premissa de que sistemas inteligentes em geral, e baseados em conhecimento em especial, consistem numa alternativa de solução mais flexível e promissora do que as abordagens procedurais tradicionais. O advento das ontologias, em particular, propicia a estruturação de conhecimento necessária para a especificação do conhecimento necessário às tarefas propostas.

Isso posto, serão revistas as contribuições do trabalho citadas na seção 1.5., acrescidas de comentários que elucidam como os problemas para a sua efetivação foram atacados nos experimentos e argumentados no texto.

- *As tarefas de recuperação, categorização e extração podem e devem ser integradas, o que pode ocasionar uma melhora de performance em todas elas.*

Pelos resultados dos experimentos, verificou-se que a categorização funcional elimina a informação inútil, não deixando de aproveitar *links*, encontrados dentro de contextos específicos, que contêm informação que possa ser processada por um dos agentes do sistema. Portanto, a categorização aliada à extração melhora a performance da recuperação; e para que a categorização e a extração sejam efetuadas sobre a Internet, prescindem da recuperação. Ademais, a união destas tarefas pode criar uma base de dados bastante

completa em um só passo de execução; se a extração for bem realizada, a partir de um modelo de domínio coerente, tão completo quanto possível, e bem definido do ponto de vista semântico, a grande maioria das informações úteis deverá estar presente na base de dados resultante. A necessidade de integração destas tarefas leva, portanto, à:

- *Formulação do problema da manipulação integrada de informação.*

Para resolver o problema da diversidade de informação, devem ser levados em conta tanto o conteúdo das páginas quanto a funcionalidade, no que tange a apresentação da informação dentro das páginas. Isso, por sua vez, leva à:

- *Formulação de uma visão da Web combinando conteúdo e funcionalidade, que facilita a resolução do problema* (vide capítulo 2).

Com uma visão da Web adequada para a resolução do problema de manipulação integrada, será justificado agora o tipo de solução escolhido. A pesquisa pioneira em manipulação cooperativa de informação, que inaugurou esta nova sub-área, [Oates et al 94] (vide subseção 3.4.1.) aponta o uso de sistemas multiagentes cognitivos para a o processamento e integração de nichos *(clusters)* de informação de alta qualidade. Para os autores, a área de manipulação de informação requer o emprego intensivo de conhecimento explícito. Corroboram esta assertiva os experimentos com aprendizado usando apenas um classificador, que não obteve bons resultados, o que pode sugerir que o emprego de multiagentes é adequado ao problema. Os autores recomendam a Resolução Distribuída de Problemas (vide subseção 3.2.1.1.) como forma de organização dos agentes dentro do sistema, uma vez que os subproblemas são independentes e interagem entre si, melhorando tanto o desempenho local de cada agente como do sistema como um todo. Segue, então, que, para que o sistema multiagente entre em funcionamento, e os agentes cooperem efetivamente:

- *É necessário que o domínio que abarca as classes de páginas inter-relacionadas, esteja representado num formalismo lógico de representação de conhecimento. As definições do domínio servirão como vocabulário de comunicação na cooperação entre agentes de classes de páginas distintas de um mesmo domínio.*

A subseção 3.2.2.2. traz um comparativo entre os modelos de comunicação em nível de conhecimento (ou *peer-to-peer*) e cliente-servidor, concluindo que:

- *Os requisitos de comunicação não apenas em sistemas de manipulação integrada de informação, mas em ambientes abertos distribuídos em geral, pedem um modelo de comunicação com mais habilidades que o modelo cliente-servidor, e que só modelos de comunicação em nível de conhecimento ("*peer-to-peer*"), baseados em ontologias reusáveis como vocabulário da comunicação, são capazes de oferecer.*

Estes fatos, junto com a existência de mecanismo de busca especializados citados na subseção 1.2., nos conduzem a outra abordagem à problemática de manipulação de informação na Internet, possivelmente mais promissora que as empregadas atualmente:

- *Os mecanismos de busca tradicionais, baseados em técnicas de recuperação de informação e indexação por palavras-chave, podem constituir a base e o suporte para outros mecanismos de busca, aplicativos, agentes e extratores mais refinados, precisos e focados em domínios restritos, baseados em conhecimento explícito reusável e comunicável, e habilitados à cooperação.*

Estas contribuições teóricas foram materializadas com protótipos e experimentos, que, obtiveram resultados promissores. As contribuições práticas do trabalho estão enumeradas a seguir:

- *A elaboração de uma arquitetura, e sua respectiva realização, num* framework *de sistemas multiagentes cognitivos para a solução do problema de manipulação integrada de informação, aplicável a grupos de classes identificados na Web, com a possibilidade de cooperação entre os agentes componentes e reuso massivo de componentes, independente do domínio tratado, conectividade a vários mecanismos de busca, e reuso de conhecimento, para a eventual construção de novos agentes* (vide capítulo 5);

Ressalte-se que a arquitetura apresentada utiliza conhecimento acerca do domínio extraído, de forma a integrar as tarefas de extração, recuperação e categorização, aplicando-o a grupos de classes da Web. Portanto, o domínio de aplicação da arquitetura não estará restrito a páginas ou sítios, mas a grupo de classes da Web. Foram implementados:

- *Dois agentes que buscam, filtram e classificam páginas da Web para as classes de artigos científicos e para a classe de páginas chamadas de trabalho para eventos científicos (os ditos* "Call for papers" – sobre estes dois agentes e seus resultados, vide capítulo 6).

Os agentes, através de técnicas de recuperação de informação, como contagem de freqüências de palavras-chave e *stop-lists* [Baeza-Yates & Ribeiro-Neto 99] pré-processam as páginas, enviam e recebem mensagens com intenções diferentes, como pedir páginas e *links* de acordo com um determinado padrão ou sugerir uma página específica, e efetuam a categorização funcional e a classificação por conteúdo. Eles se comunicam e cooperam usando a ontologia Ciência, descrita nas seções 4.9. e 4.10., como vocabulário de comunicação, constituindo assim, mais uma contribuição do trabalho.

- *A disponibilização pública na Web de uma ontologia relativa ao domínio científico* [Freitas 2001], *que está sendo empregada pelo sistema MASTERWeb na manipulação integrada de dados da Web.*

8.2. Trabalhos futuros

Naturalmente, para propor uma abordagem inteiramente nova, baseada em conhecimento, para um conjunto de áreas vastas e recentes e que ainda estão se maturando, como recuperação, classificação e extração de informação da Web ligadas a ontologias e agentes inteligentes, o projeto dos protótipos foi simplificado de várias formas, como foi visto na subseção 6.3.1. O presente trabalho e seus protótipos podem, ainda, ser estendidos em vários sentidos, além destas simplificações.

8.2.1. Extensões Orientadas à Semântica

Serão listadas, nesta subseção, extensões ao trabalho que possam melhorar a qualidade das inferências e a própria organização do conhecimento especificado.

Uma ontologia do meio científico ainda mais formalizada, em que as classes definidas fossem subclasses de metaclasses de ontologias "de topo" (vide seção 4.10.), como as metaclasses Objeto e Processo da ontologia de John Sowa, ou as metaclasses Vivo e Não-vivo do WordNet (vide figura 16), permitiriam deduções mais seguras por parte do sistema,

bem como a especificação de tipos de relacionamentos parte-todo e propriedades como transitividade e outras. A classe Tópicos de Pesquisa poderia, ela própria, se tornar uma meta-classe e incluir conceitos relacionados. Assim, áreas poderiam ser definidas em outras ontologias (como Ciências Exatas ou Computação) e aproveitadas na definição de reconhecedores e extratores para os agentes, ou seja, ao invés de conceitos baseados em palavras-chave, como fazem os agentes atualmente, os reconhecedores e extratores poderiam aceitar classes de ontologia, desde que fossem subclasses de Tópicos de Pesquisa.

Também a definição de axiomas através da linguagem axiomática do Protégé PAL (*Protégé Axiomatic Language*) enriqueceriam em complexidade as relações e restrições entre as diversas classes e seus atributos, habilitando o sistema a recusar dados inconsistentes, como, por exemplo, uma pessoa estar em duas conferências ao mesmo tempo.

Um passo importante constará da inclusão de ontologias lingüísticas como o WordNet e o GATE, integráveis ao editor de ontologias Protégé (vide subseção 4.7.2.). Lançando mão de uma delas, os agentes poderão refinar o processo de extração, colhendo informações a partir de padrões linguísticos de textos (vide subseção 7.3.1.2. para um exemplo destes padrões).

A extração refinada pode habilitar os agentes a um tipo de cooperação distinta das que eles podem executar hoje: a *cooperação de informação*. Por exemplo, o agente CFP, ao encontrar a âncora referente ao *chairman* da conferência, pode enviar não só a âncora para o agente dos pesquisadores, mas também o fato de que ele ter sido *chairman* deste evento, um atributo relevante que pode não constar na página do pesquisador, como na mensagem abaixo:

```
(tell :sender "cfp"
      :receiver "researchers"
      :language JessTab
      :content
  (researcher
     (First-name "John")
     (Last-Name "Smith")
     (chair
        (event "IJCAI")
```

```
(year 1999))))
```

Como este, outros atributos podem ser inferidos a partir de fatos recebidos declarativamente. Por exemplo, num domínio comercial, páginas distintas sobre uma empresa podem anunciar a ocupação de um mesmo posto, o que, com o auxílio de técnicas de extração, pode fazer com que agentes registrem que o antigo ocupante não é mais o titular daquele posto [Gatzauskas & Robertson 97]. Estas facilidades não são possíveis com códigos procedurais. Também não foram encontrados sistemas que realizem este tipo de cooperação para o problema de manipulação cooperativa de informação na Web.

Pretende-se, também, adicionar contextos à mensagens que forem enviadas. É interessante, por exemplo, ao adicionar uma sugestão de página de *workshop* que foi encontrada dentro de uma conferência o envio deste contexto, que, de imediato, identifica a cidade de realização, datas máxima e mínima para a consistência, entre outros. Semelhante ação pode ser tomada em relação à páginas de *frames* HTML. A grande maioria das páginas deste tipo, encontradas pelo agente CFP, são, de fato, chamadas de eventos científicos, portanto, um bom tratamento destas páginas exigiria um fato relacionando-as como se tratassem de uma só entidade, até mesmo tratando-as todas ao mesmo tempo. O agente CFP, hoje, apenas enfileira os links de páginas *frames* na fila de alta prioridade.

Ainda no tocante á representação de conhecimento, podem ser testados outros formalismos integráveis ao Protégé, como Prolog e F-Logic, capazes de representar *frames* nativamente, reduzindo sobremaneira o tamanho do conhecimento especificado. A desvantagem em empregar o motor de inferência Jess, é que o componente do Protégé JessTab converte cada atributo e faceta de cada classe em um novo fato na base de conhecimento, tornando a base volumosa. Por outro lado, as facilidades procedurais do Jess produzem ótimas funções de extração.

8.2.2. Extensões Operacionais

A extração, com todas as suas funções típicas, como crítica de cada dado, consistência dos dados em seu conjunto, mapeamento, formatação, deve substituir a simples identificação de atributos implementada nos protótipos. Especial atenção deve ser dispensada ao problema da co-referência de instâncias das entidades extraídas em páginas diferentes. O

sistema CiteSeer abordou esta problema, quando do cômputo do número de citações de cada artigo. Apesar da relativa estruturação das informações nas citações, os autores do CiteSeer enfatizam o grau de dificuldade do problema.

A aquisição automática de conhecimento relativos às tarefas de reconhecimento e extração não foi incluída entre os tópicos deste trabalho. A inclusão de aprendizado automático (*machine learning*) pode acelerar a aquisição, e dotar a arquitetura de adaptatividade, criando uma camada instintiva para os agentes, de forma a seguir a arquitetura de agente em três níveis [Bittencourt 97] (reativo, instintivo e cognitivo, vide seção 5.3. para observar como técnicas de aprendizado podem ser incluídas na arquitetura).

A adição destas duas tecnologias, processamento de linguagem natural e aprendizado, é de fundamental importância para o processamento de classes menos estruturadas, como páginas de pesquisadores, por exemplo.

Do ponto de vista da disponibilidade dos dados extraídos, pretende-se colocar um servidor para disponibilizar publicamente os dados extraídos; para isto, o mediador será melhorado, e torna-se necessário abrir espaço à colaboração dos usuários. Podem ser enviadas mensagens aos responsáveis pelas páginas extraídas, pedindo a confirmação dos dados contidos nela, oferecendo-lhes, em troca, um possível aumento de "*hits*" de acesso a elas.

A presença do mediador levanta um outro ponto sobre a integração entre ontologias e bancos de dados: qual a melhor forma de armazenamento em bancos de dados de um grande volume de dados representados em *frames* de ontologias? Como foi colocado na subseção 4.7.2., os *slots* dos *frames*, por poderem assumir múltiplos valores ao invés de um único como num campo de uma tabela relacional, não podem ser diretamente colocados em bancos de dados relacionais de forma normalizada. Diversos tipos de solução, acoplando as tecnologias de bancos de dados e dedução, cabem neste contexto: extensões da linguagem de manipulação de dados SQL, bancos de dados dedutivos, extensões de bancos de dados orientados a objetos e *data warehousing*.

Melhorias podem ser adicionadas, ainda, ao processo de recuperação. O uso de prefixos de URLs para a localização de novas entidades de uma mesma classe, num mesmo sítio (por exemplo, muitas páginas de pesquisadores localizam-se em diretórios com prefixo comum, como *www.cs.cmu.edu/staff/*), e a cooperação entre agentes. Há agentes em que, talvez,

seja mais compensatório alimentar-se apenas de recomendações advindas de outros agentes, do que filtrando mecanismos de busca, tanto pela falta de precisão destes, quanto pela inexistência de palavras-chave adequadas para a busca. Experimentos com cooperação entre agentes de classes distintas podem ajudar a mapear a forma como as classes de páginas se conectam no hiperespaço, delineando inclusive padrões de cardinalidade da relação (1:1, 1:N ou N:M) e mapas "topológicos" de grupos de classes na Web.

Finalmente, novos agentes e domínios devem ser confeccionados e testados. A partir do ambiente de desenvolvimento apresentado neste trabalho, acrescido de técnicas de aprendizado e linguagem natural, espera-se reduzir o tempo de desenvolvimento dos agentes pela indução das regras de classificação e extração junto com a engenharia de conhecimento, enquanto as técnicas de PLN prometem refinar ainda mais os resultados.

8.3. Palavras Finais

Os sistemas abertos, como a Internet, por sua riqueza, variedade e desorganização, requerem um tratamento não-convencional, que a programação imperativa tradicional não está capacitada a fornecer. Os agentes cognitivos, por basearem-se em modelos com conhecimento, têm condição de comunicar-se de forma a evitar redundância de tarefas dentro de um mesmo ambiente, favorecendo a cooperação, através de vocabulários comuns a eles, papel desempenhado pelas ontologias. Esses fatores podem proporcionar a inauguração de uma nova era na informática distribuída, a comunicação em nível de conhecimento dinamicamente, estabelecida durante a execução, e o processamento de nichos de informação consistentes [Oates et al 94], como o domínio científico, o domínio turístico, etc.

Por ora, foi projetada uma arquitetura de sistemas multiagentes para extrair, recuperar e classificar páginas, não apenas de páginas específicas, mas de regiões da Internet, definidas por grupos de classes de páginas. Outros domínios, além do domínio científico, podem beneficiar-se dela; aliás, qualquer grupo formado por classes de páginas inter-relacionadas aceitam sua aplicação. Alguns exemplos: no domínio comercial, alguns grupos podem ser identificadas, e.g., um grupo de classes de compras, incluindo shopping centers, lojas de departamento e fornecedores; um grupo de classes de turismo, envolvendo eventos, hotéis, e transporte, e muitos outros. De fato, ressaltamos que busca, recuperação, extração e

APÊNDICE A – EXEMPLOS DE PROCESSAMENTO

A.1. Conexão ao Roteador de Mensagens

A listagem seguinte mostra o agente CFP iniciando sua execução. Ele carrega as ontologias e regras no Jess, e se conecta ao roteador do JATLite, através do pedido de registro e conexão do agente. Após este processo, o meta-robô efetua a primeira consulta do agente CFP ao mecanismo de busca *Google*. Os *links* do resultado são extraídos e colocados na fila de URLs a serem processadas (fila de baixa prioridade).

```
Initialization success
Jess> TRUE
Jess> EOF
Jess> Server created
cfpServer Started
Start to register
Register accepted Before
Start to connect as cfp
(reconnect-agent :host garrincha :port 250 :sender cfp :receiver Router
:password jhc :email fred-pe@lcmi.ufsc.br)
Connection established
Router started after
Client Router running
Checking 'to Send' row: priority 9
1 query:
=========
http://google.yahoo.com/bin/query?p=call+for+papers&b=0
Links :
http://www.english.upenn.edu/CFP/
http://www.unt.edu/UNT_ISWorld/cfp.htm
http://www.linuxtag.org/cfp/cfp3-en.html
http://daily.daemonnews.org/view_story.php3?story_id=2668
http://www.computer.org/conferences/cfp-conf.htm
http://www.dice.ucl.ac.be/crypto/call_for_papers.html
http://www.php-homepage.de/?news=211
http://lgxserver.uniba.it/lei/cfp/cfp.htm
http://www.orcca.on.ca/~ilias/aca2002SpecialIssueJSC.html
Checking Row with Priority 2
hi Prio (8) Row's length:0
low Prio(5) Row's length:10
to Send Row's length:0
```

A.2. Exemplos e Contra-exemplos do Agente de Eventos Científicos

A página da figura 28 apresenta uma chamada de trabalhos de uma conferência.

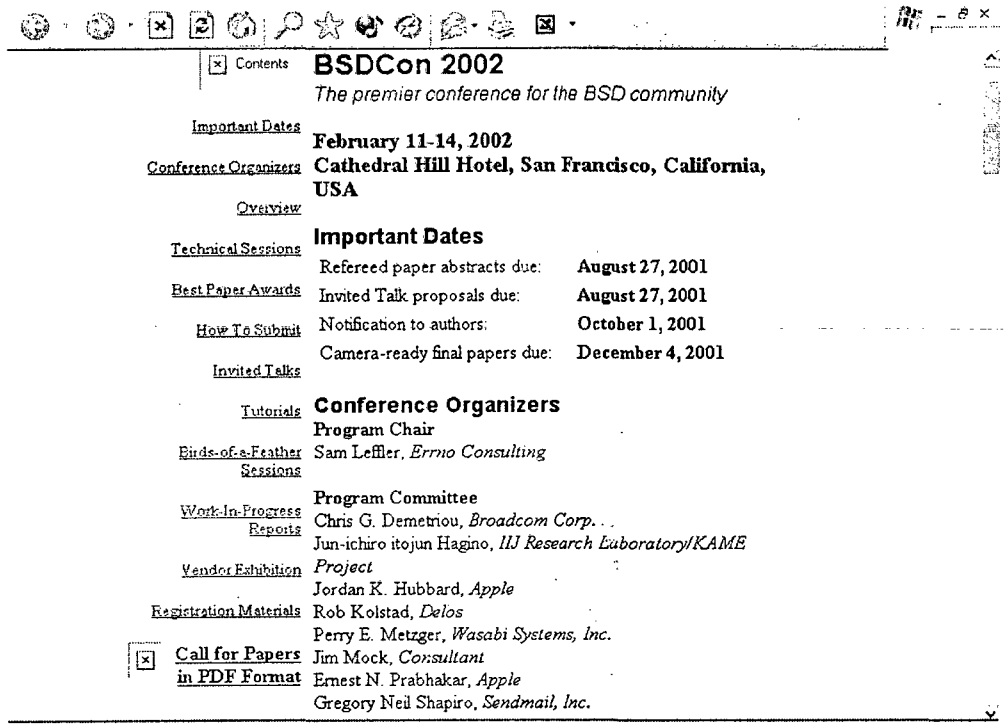

Figura 28. Página reconhecida como instância da classe Conferência pelo agente CFP.

Apesar de parecer obedecer ao padrão representado pelo caso da subseção 5.5.3. - em que uma página que possui, no seu início, os atributos data inicial do evento e localização (país ou estado americano em que irá ocorrer o evento) e algum termo relacionado a um evento ao vivo, como as expressões *"call for papers"* ou *"call for participation"* -, esta página foge ao caso apontado, pois os *links* laterais, como "*Important Dates*", "*Conference Organizers*" e outros foram codificados em HTML antes do local e período do evento, fazendo com que estes dados não sejam encontrados no início do código HTML da página.

Uma representação ou regra poderia ser confeccionada para resolver estes casos. No entanto, um caso de reconhecimento, apresentado a seguir, que sói ocorrer com freqüência entre chamadas de trabalho, indica que a existência de pelo menos quatro atributos, acompanhado da presença de um dentre os conceitos *cfp, call-for-participation, workshop, event* ou *conference* no início da página e da ausência do conceito *listing-of-call-for-papers* garante que a página deve ser uma chamada de trabalhos.

```
([cfp_00445] of Case
      (Concepts-in-the-Beginning
            [Call-for-participation]
            [Workshop]
            [cfp]
            [Conference]
            [event])
      (Description "sc-evnt:nr slots & concepts")
      (Nr-of-Slots 3)
      (Absent-Concepts [listing-of-call-for-papers])
      (Slots
            [Acceptance-Date]
            [Accomodation-Places]
            [Final-Date]
            [Final-Paper-Due]
            [Deadline]
            [Initial-Date]
            [Organizing-Chair]
            [Organizing-Committee]
            [Program-Committee]
            [Scientific-Chair]
            [takes-Place-at]
            [Tutorials]
            [Travel-Information]
            [Sponsors]
            [Social-Program]
            [Registration-Fees]
            [Event-Publication]
            [Program]
            [Subject-Areas])
      (Importance HIGH))
```

A página foi processada e a regra 910 (vide exemplo de regra na subseção 6.5.2.) que corresponde ao caso, foi atendida, reconhecendo a página como um evento científico, de acordo com a listagem abaixo:

```
Processing page from row with priority 5
Page : http://brown.lcmi.ufsc.br/cfp/Teste/b53.html
FIRE 1 MAIN::i_901_start f-189
FIRE 2 MAIN::v_314_valid f-828, f-826
FIRE 3 MAIN::i_905_filling f-829
duration 16.0s
FIRE 4 MAIN::i_907_fill-fields f-830
```

```
Found month ("February")
Found state ("California")
FIRE 5 MAIN::r_442_slots_me_ccpt f-852, f-712,,
SLOT FOUND : Scientific-Chair
FIRE 6 MAIN::r_442_slots_me_ccpt f-852, f-703,,
SLOT FOUND : Deadline
FIRE 7 MAIN::r_454_slots_me_ccpt_lnk_txt f-852, f-703,,
FIRE 8 MAIN::r_454_slots_me_ccpt_lnk_txt f-852, f-659,,
SLOT FOUND : Acceptance-Date
FIRE 9 MAIN::r_442_slots_me_ccpt f-852, f-659,,
FIRE 10 MAIN::r_442_slots_me_ccpt f-852, f-586,,
SLOT FOUND : Subject-Areas
FIRE 11 MAIN::r_910_slots_hi_funct f-852, f-1347, f-267, f-486,
FIRE 12 MAIN::c_606_recognized_concepts_title f-1350, f-240, f-603,
CLASS Conference
FIRE 13 MAIN::e_203_links f-1353, f-795, f-1354,, f-769, f-12,
FIRE 14 MAIN::e_203_links f-1353, f-139, f-1354,, f-769, f-12,
FIRE 15 MAIN::s_002_result f-1353, f-1356, f-1355, f-1354
fact : CLASSIFIED
Inserting as recognized...
CLASS Conference
```

O agente demonstra robustez mesmo ante chamadas de eventos científicos com poucos dados, como é o caso da página da figura 29, encontrado no teste com *corpus* desconhecido, considerada corretamente como *Evento Científico de Publicação Genérica*, pois a ontologia não abrange chamadas para capítulos de livros.

Contudo, o agente falha em portais sobre determinados assuntos ou organizações, como o da figura 30. Estas páginas costumam ser longas e possuir muitos dos atributos de um evento científico, como patrocinadores, comitês e outros.

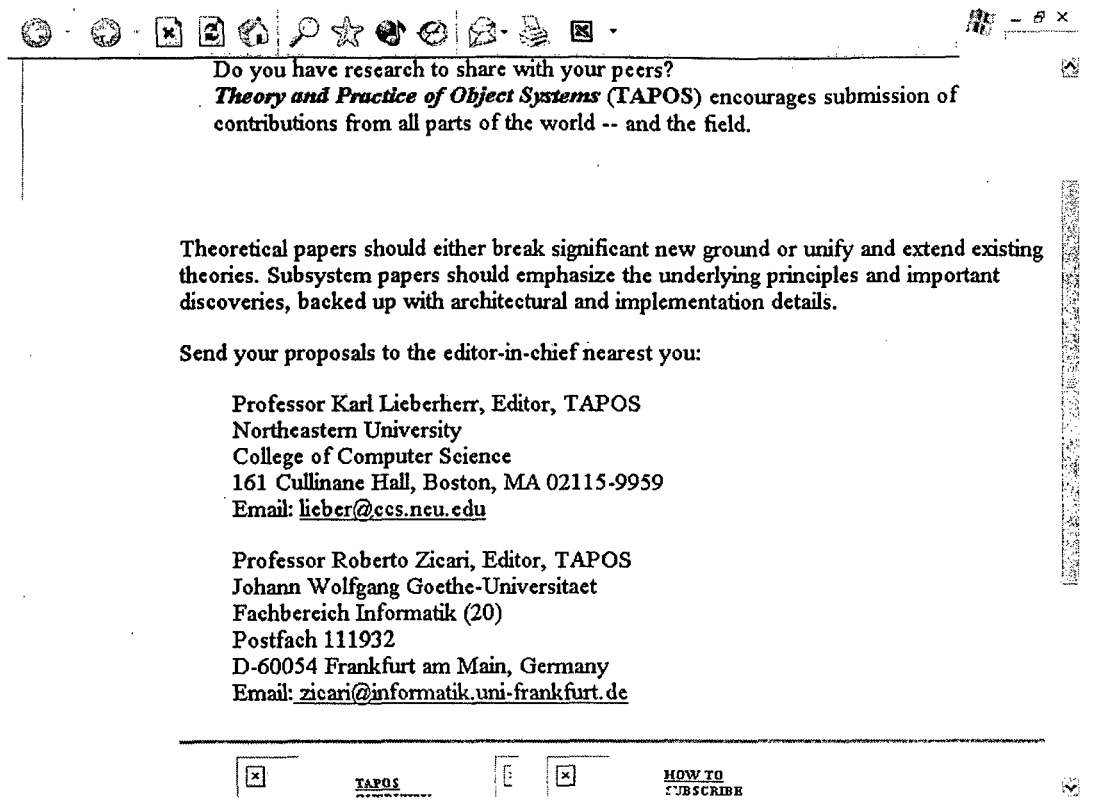

Figura 29. Página classificada como Evento Científico de Publicação Genérica.

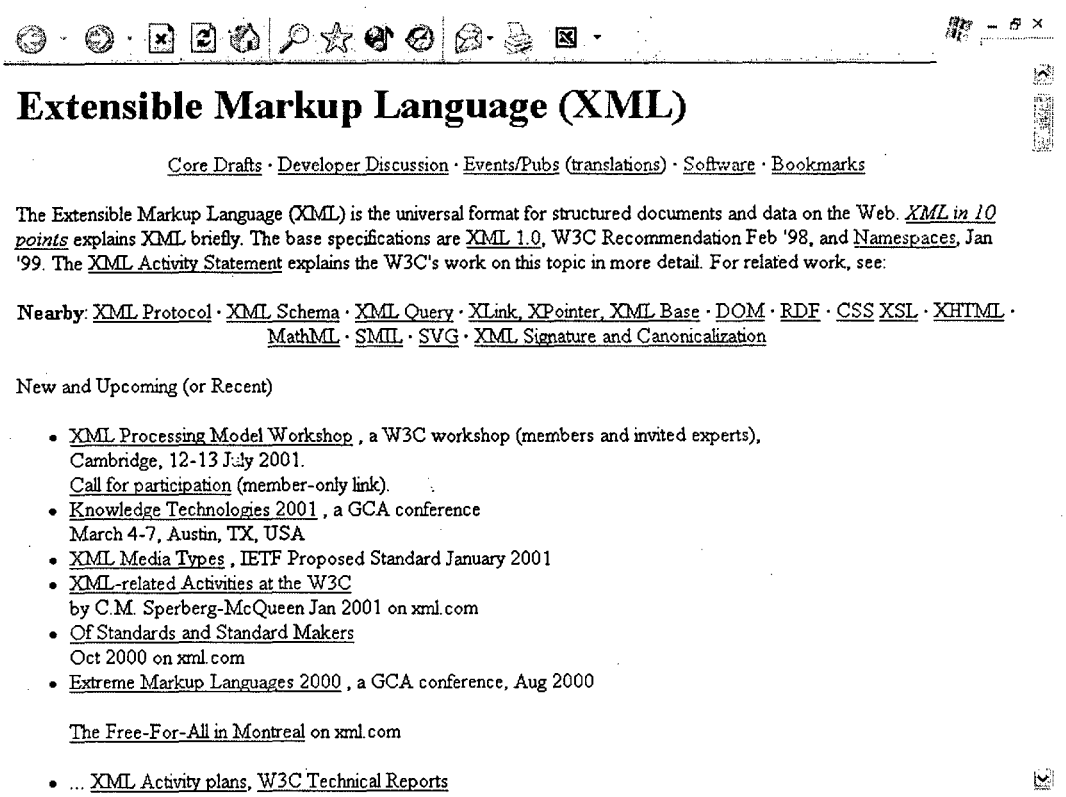

Figura 30. Falso positivo na classificação de Eventos Científicos.

A.2. Cooperação entre os Agentes

Embora a proposta de especificação da linguagem de comunicação KQML tente definir uma semântica formal para as performativas de comunicação que denotam as intenções dos agentes, de modo a provocar ações bem-definidas nos agentes emissores e receptores, tem se difundido a prática de atribuir às performativas comportamentos particulares a cada sistema multiagente. Procurou-se, neste trabalho, seguir as definições apregoadas pela especificação da linguagem, porém é possível que os comportamentos dependentes das mensagens difiram da semântica definida pelo KQML.

Apesar de ter funcionado adequadamente, como mostra este exemplo, a cooperação produziu poucas sugestões: apenas três recomendações foram enviadas do agente de artigos para o agente CFP, embora todas estivessem corretas e nenhuma outra recomendação pudesse ser encontrada nos artigos. Ao final desta seção, será visto que a cooperação entre o agente CFP e o futuro agente de pesquisadores pode produzir muitas páginas úteis a este último.

Antes, porém, a troca de mensagens entre os dois agentes será explanada. O agente CFP foi posto em execução e, após registrar-se e carregar ontologias e regras, envia várias mensagens ao agente de artigos.

```
Initialization success
Jess> TRUE
Jess> EOF
Jess> Server created
cfpServer Started
Start to register
Register accepted
Before
Start to connect as cfp
(reconnect-agent  :host garrincha
                  :port 250
                  :sender cfp
                  :receiver Router
                  :password jhc
                  :email fred-pe@lcmi.ufsc.br)
Connection established
Router started
after
```

A primeira das mensagens enviadas de todo agente é um anúncio de sua existência e de sua capacidade em tratar uma classe do domínio, no caso, a classe *Scientific-Event.* Esta mensagem é repassada a todos os agentes do sistema, devido ao uso da performativa intencional :*broadcast*, da linguagem de comunicação KQML.

```
Message to be sent:(broadcast:sender cfp
                              :receiver ppr
                              :language JessTab
                              :ontology Science
                              :content ([cfp] of Agent
                                    (treats-Class Scientific-Event)
                                    (name "cfp")))
Sending message
Sending success
```

Logo após, o agente CFP pedirá *links* que cumpram certas condições. Ele envia ao agente de artigos científicos instâncias de conceitos, casos e reconhecedores através da intenção :*inform*, que fará com que o agente receptor insira as instâncias em sua base de conhecimento.

```
Message to be sent:(inform    :sender cfp
                              :receiver ppr
                              :language JessTab
                              :ontology Science
                              :force tentative
                              :content ([conference] of Concept
                                    (Synonyms   "symposium"
                                                "congress"
                                                "meeting"
                                                "seminar")
                                    (name "conference")))
Sending message
Sending success

Message to be sent:(inform    :sender cfp
                              :receiver ppr
                              :language JessTab
                              :ontology Science
                              :force tentative
                              :content ([proceedings] of Concept
                                    (Keywords   "camera ready"
```

```
                                        "camera-ready"
                                        "published")
                                (name "proceedings")))
Sending message
Sending success

Message to be sent:(inform    :sender cfp
                              :receiver ppr
                              :language JessTab
                              :ontology Science
                              :force tentative
                              :content ([annual] of Concept
                              (Keywords   "th " "second" "first" "third"
                                          "1st"  "2nd"  "3rd"  "'9"  "'9"
                                          "'0" "'0" "200")
                              (name "annual")) )
Sending message
Sending success

Message to be sent:(inform    :sender cfp
                              :receiver ppr
                              :language JessTab
                              :ontology Science
                              :force tentative
                              :content ([ppr_00008] of Case
                                    (Description "cfp suggestions")
                                    (Importance MEDIUM)
                              (Concepts   [Call-for-participation]
                                          [annual] [conference]
                                          [proceedings] [Call-for-papers]
                                          [workshop])
                                    (All-of-Concepts-Keywords FALSE)))
Sending message
Sending success
```

Agora que o agente de artigos recebeu as instâncias às quais as condições vão referir-se, o agente pede (performativa intencional *:ask-all*) *links* com as palavras chaves das instâncias que estejam no início de páginas reconhecidas como de artigos de conferências ou *workshops*.

```
Message to be sent:(ask-all   :sender cfp
                              :receiver ppr
                              :language JessTab
```

```
                    :ontology Science
                    :content((object(is-a Anchor)(Link-Text?l))
                    (Result (Page-Status CLASSIFIED)
                    (Class "Conference-Paper"|"Workshop-Paper")
                          (Recommended-URLs $?u)
                          (Recommended-Agents $?ra)
                          (Recommendation-Rule $?rr))
                    (object (is-a Web-Page) (Contents ?co))
              (test (and (integerp
                          (str-index  ?l  (beginning-until
                          "abstract" ?co) ?co ))
                           (if-occur (words-of-concepts
                          (slot-get [ppr_00008]
                                Concepts)) ?t))))))
Sending message
Sending success
```

Após isso, o agente começa a executar normalmente, buscando páginas nos mecanismos de busca, no caso abaixo acessando o Google com a palavra-chave "*call for papers*":

```
Client Router running
1 query:
=========
http://google.yahoo.com/bin/query?p=call+for+papers&b=0
Links :
http://www.english.upenn.edu/CFP/
http://www.unt.edu/UNT_ISWorld/cfp.htm
http://www.dice.ucl.ac.be/crypto/call_for_papers.html
```

O agente de artigos também se inicializa e envia sua mensagem de anúncio, especificando que responsabiliza-se, dentro do sistema, por tratar da classe Documento-Científico, como pode ser verificado abaixo:

```
Initialization success
Jess> TRUE
Jess> EOF
Jess> Server created
pprServer Started
Start to register
Register accepted
Before
```

```
Start to connect as ppr
(reconnect-agent  :host maquiavel
                  :port 250
                  :sender ppr
                  :receiver Router
                  :password jhc
                  :email fred-pe@lcmi.ufsc.br)
Connection established
Router started
after
Client Router running
Message to be sent:(broadcast:sender ppr
                              :receiver cfp
                              :language JessTab
                              :ontology Science
                              :content ([ppr] of Agent
                                    (treats-Class Scientific-Document)
                                    (name "ppr")))
Sending message
Sending success
```

Recebe, então, as mensagens enviadas pelo agente CFP. O conteúdo da mensagem de anúncio ele inclui em sua base de conhecimento, i. e., o nome do agente e junto com a classe que é capaz de processar.

```
Received message:N 1026139434790 (broadcast    :sender cfp
                                               :content ([cfp] of Agent
                                 (treats-Class Scientific-Event )
                                 (name "cfp" ) )
                                               :ontology Science
                                               :receiver ppr
                                               :language JessTab)
Performative:broadcast
Contents:([cfp] of Agent (treats-Class Scientific-Event ) (name "cfp" ) )
Sender:cfp
Aware of the existence of agent and its capabilities : ([cfp] of Agent
(treats-Class Scientific-Event )
(name "cfp" ) )
```

Cada mensagem com a performativa :*inform* produz um fato na base de conhecimento do receptor, como abaixo. A palavra *asserted* indica o que ele inseriu em sua base de conhecimento.

```
Received message:N 1026139436220 (inform :sender cfp
                                         :content ([Call-for-papers] of
                                               Concept
                                         (Synonyms "call for papers" )
                                         (name "cfp" ) )
                                         :ontology Science
                                         :force tentative
                                         :receiver ppr :language
                                         JessTab)
Performative:inform
Contents:([Call-for-papers] of Concept   (Synonyms "call for papers" )
                                         (name "cfp" ) )
Sender:cfp
asserted : ([Call-for-papers] of Concept (Synonyms "call for papers" )
                                         (name "cfp" ) )

Received message:N 1026139445220 (inform :sender cfp
                                         :content ([ppr_00008] of Case
                                    (Description "cfp suggestions" )
                                    (Importance MEDIUM )
                                    (Concepts [Call-for-participation]
                                    [annual] [conference] [proceedings]
                                    [Call-for-papers] [workshop] )
                                    (All-of-Concepts-Keywords FALSE ) )
                                         :ontology Science
                                         :force tentative
                                         :receiver ppr
                                         :language JessTab)
Performative:inform
Contents:([ppr_00008] of Case(Description "cfp suggestions" )
                              (Importance MEDIUM )
                              (Concepts [Call-for-participation] [annual]
                                    [conference] [proceedings]
                                    [Call-for-papers] [workshop] )
                              (All-of-Concepts-Keywords FALSE ) )
Sender:cfp
asserted : ([ppr_00008] of Case (Description "cfp suggestions" )
                                    (Importance MEDIUM )
                                    (Concepts [Call-for-participation]
                                    [annual] [conference] [proceedings]
                                    [Call-for-papers] [workshop] )
                                    (All-of-Concepts-Keywords FALSE ) )
```

Ao receber o pedido de *links* através da intenção *:ask-all*, o agente de artigos cria uma regra cujo lado esquerdo (o das premissas) são as condições enviadas e o direito (conseqüentes) a criação de uma recomendação para o agente que lhe enviou o pedido.

```
Received message:N 1026139446710 (ask-all:sender cfp
                                          :content
      ((object (is-a Anchor) (Link-Text ?l))
      (Result (Page-Status CLASSIFIED)
            (Class "Conference-Paper" |"Workshop-Paper" )
            (Recommended-URLs $?u ) (Recommended-Agents $?ra)
            (Recommendation-Rule $?rr))
      (object (is-a Web-Page ) (Contents ?co ) )
      (test (and (integerp (str-index ?l
                        (beginning-until "abstract" ?co ) ?co ) )
                  (if-occur (words-of-concepts (slot-get [ppr_00008]
                        Concepts ) ) (lowcase ?t ) ) ) ) )
                                          :ontology Science
                                          :receiver ppr
                                          :language JessTab)
Performative:ask-all
Contents:((object (is-a Anchor) (Link-text ?l))
          (Result (Page-Status CLASSIFIED )
                  (Class "Conference-Paper" | "Workshop-Paper" )
                  (Recommended-URLs $?u ) (Recommended-Agents $?ra )
                  (Recommendation-Rule $?rr ) )
          (object (is-a Web-Page ) (Contents ?co ) )
          (test (and
                        (integerp (str-index ?l
                              (beginning-until "abstract" ?co ) ?co ) )
                        (if-occur (words-of-concepts
                              (slot-get [ppr_00008] Concepts ) )
                              (lowcase ?t ) ) ) ) )
Sender:cfp
```

Aqui está a regra gerada e sua respectiva compilação pelo Jess:

```
      (defrule d_1_cfp
      (not (Processed ANCHOR ?l))
      (object (is-a Anchor) (Link-Text ?l))
      ?f <- (Result (Page-Status CLASSIFIED )
                  (Class "Conference-Paper" | "Workshop-Paper" )
                  (Recommended-URLs $?u ) (Recommended-Agents $?ra )
                  (Recommendation-Rule $?rr ) )
      (object (is-a Web-Page ) (Contents ?co ) )
```

```
      (test (and
            (integerp (str-index ?l (beginning-until "abstract" ?co)?co))
            (if-occur (words-of-concepts (slot-get [ppr_00008] Concepts))
                  (lowcase ?t )) ) )
      =>
      (assert (Processed ANCHOR ?l))
      (modify ?f (Recommended-URLs (create$ $?u ?l))
                  (Recommended-Agents (create$ $?ra "cfp"))
                  (Recommendation-Rule (create$ $?rr 1))))
MAIN::d_1_cfp: =1+1+1+1=1=1+1=1+1+1+1+1+1=1+1+1=1+1+2+2+2+t
```

O agente PPR continua a execução normalmente, buscando artigos no mecanismo de busca AltaVista, com as palavras-chave *"abstract"*, *"keywords"*, *"introduction"*, *"related work"*, *"overview"*, *"scenario"* e *"results"*.

```
1 query:
=========
http://www.altavista.com/sites/search/web?q=abstract+keywords+introductio
n+%22related+work%22+overview+scenario+results&stq=0
Links :
http://www.soft.com/QualWeek/QWE99/qwe99.abs.html
http://www.db.fmi.uni-passau.de/~kossmann/FKM00/paper.html
```

Enquanto isso, o agente CFP recebe a mensagem de anúncio do agente de artigos e procede de forma similar. Mais tarde a página da figura 31, processada pelo agente de artigos dispara a nova regra, criando uma recomendação ao agente CFP, que lhe é enviada, através da intenção *:tell*, como pode-se constatar.

Proceedings of the 2nd Workshop on Adaptive Systems and User Modeling on the WWW

Web Assistants: Towards an Intelligent and Personal Web Shop

Johan Åberg and Nahid Shahmehri

Department of Computer and Information Science
Linköping University, S-581 83 Linköping, Sweden
phone: +46-13-281465, fax: +46-13-282666
e-mail: johab@ida.liu.se, nahsh@ida.liu.se
url: Johan Åberg, Nahid Shahmehri

Abstract: Electronic commerce has recently shown enormous potential to take over the sales market. There is a need to provide services that can reach individual computer users with different information profiles and levels of expertise. In this paper we introduce the novel concept of *web assistants*, human assistants working in an electronic web shop. This human-computer cooperation provides intelligent and personal services via an integrated communication media. A prototype of a web assistant system has been implemented. While browsing through the system the user can call for human assistance should the need arise. We present the results of a usability study performed on our prototype system. The results are encouraging especially when it comes to the attitude aspects of usability. The subjects were extremely enthusiastic about the concept of web assistants and its implications.

Keywords: Adaptivity, electronic commerce, usability, www, data collection.

1 Introduction

Web-based electronic commerce is just in its youth. Still, the amount of shopping on the web in the USA has been estimated to

Figura 31. Esta página de artigo de Workshop, processada pelo agente de artigos científicos, possui um *link* a ser sugerido para o agente CFP.

```
Processing page from row with priority 2
Page:
http://www.contrib.andrew.cmu.edu/~plb/WWWUM99_workshop/aberg/aberg.html
FIRE 1 MAIN::i_901_start f-209
...
FIRE 10 MAIN::r_900_slots_hi_funct f-3779, f-3859, f-226, f-306,
FIRE 11 MAIN::c_607_recognized_concepts_begin_absent f-3862, f-684, f-
428,
CLASS Workshop-Paper
FIRE 12 MAIN::d_1_cfp f-905,, f-3866, f-3778,
FIRE 13 MAIN::e_205_non_links_text f-3865,
FIRE 14 MAIN::e_204_non_links f-3865,
FIRE 15 MAIN::s_002_result f-3865, f-3869, f-3868, f-3866
AGENT:cfp
Inserting "http://www.contrib.andrew.cmu.edu/~plb/WWWUM99_workshop/" into
'to Send' row.
Inserting as recognized...
Workshop-Paper
Checking 'to Send' row: priority 9
```

```
Message to be sent:(tell :sender ppr
                         :receiver cfp
                         :language JessTab
                         :ontology Science
                         :force tentative
                         :content
      (object     (is-a Recommendation)
                  (Anchor-to-Suggest
            http://www.contrib.andrew.cmu.edu/~plb/WWWUM99_workshop/)
                  (Recommendation-Rule 1)))
Saving Statistics...
Sending message
Sending success
```

O agente CFP recebe a sugestão e deposita a URL sugerida na fila de alta prioridade, de onde, posteriormente, ela foi retirada, processada e reconhecida como página-conteúdo.

```
Received message:N 1026099728110 (tell    :sender ppr
                                          :content
      (object(is-a Recommendation )
            (Anchor-to-Suggest http://www.acm.org/sigchi/cscw96/ )
            (Recommendation-Rule 1 ) )
                                          :ontology Science
                                          :force tentative
                                          :receiver cfp
                                          :language JessTab)
Performative:tell
Contents:(object  (is-a Recommendation )
                  (Anchor-to-Suggest http://www.acm.org/sigchi/cscw96/ )
                  (Recommendation-Rule 1 ) )
Sender:ppr
URL received and inserted in hi prio row:
      (Anchor-to-Suggest http://www.acm.org/sigchi/cscw96/)
```

APÊNDICE B – RESULTADOS DOS EXPERIMENTOS

B.1. Agente CFP

O agente encarregado de processar a classe de páginas representada por chamadas a eventos científicos acessou os mecanismos de busca *AltaVista* e *Yahoo-Google*, tanto para a geração dos *corpi* para os testes iniciais, como para o agente rodando diretamente na Web. Todos os testes seguiram os padrões de classificação descritos na subseção 6.7.1., exceto o último, em que as sugestões tiradas das listas apresentaram outros padrões e melhoraram substancialmente o desempenho do agente.

B.1.1. Teste do Agente CFP com Corpus *para Aquisição de Conhecimento*

Para este teste foram alocadas 250 das 400 páginas recuperadas, com o meta-robô capturando páginas da Web sem o agente. Deste total, seis delas não continham dados, e foram desprezadas. Os resultados da categorização funcional estão detalhados na tabela 7.

- Agente CFP – Categorização Funcional do *Corpus* de Aquisição	Reconhecidas	Listas	*Frames*	Recomendadas	Rejeitadas	Total	**Reconhecidas (%)**	Total (%)
Corretas	161	20	1	27	20	229	**95,8**	93,8
Falsas positivas	4	3	0	1	7	15	**2,4**	6,2
Falsas negativas	3	1	0	6	5	15	**1,8**	6,2
Ocorrência (%)	67,2	8,6	0,4	13,5	10,2	100	100	100

Tabela 7. Categorização funcional do agente CFP sobre o *corpus* de aquisição de conhecimento.

As recomendações de páginas referiam-se às classes *Organização*, inclusive comunidades virtuais da Internet, e *Publicação Divisível* (como livros, anais, etc), cujos agentes ainda não foram implementados. Em um terço destas recomendações, o agente conseguiu identificar a subclasse das recomendações, seis para a subclasse *Jornal* e três para a subclasse *Institutos de Pesquisa.* Páginas de lixo que continham termos relativos a listas foram tratados como tais, mas não geraram sugestões, e, portanto não comprometeriam a performance do agente.

Seguem, na tabela 8, os resultados detalhados do agente CFP classificando por conteúdo o *corpus* de aquisição.

- Agente CFP - Classificação por Conteúdo / ***Corpus* de Aquisição**	Conferência	*Workshop*	Jornal	Edição Especial de Jornal	Revista	Edição Especial de Revista	Evento Genérico ao Vivo	Evento Genérico de Publicação	Total	**Total (%)**
Corretas	119	15	5	2	2	0	3	7	161	**91,7**
Falsas Positivas	4	2	0	0	0	0	0	2	8	**8,3**
Falsas Negativas	2	1	2	0	0	0	2	1	8	**8,3**
Não-reconhecidas	1	0	1	0	0	0	0	1	3	**-**
Ocorrência (%)	74,4	9,8	4,9	1,2	1,2	0	3	5,5	100	**100**

Tabela 8. Resultados detalhados do agente CFP classificando por conteúdo o *corpus* usado para aquisição de conhecimento.

B.1.2. Teste do Agente CFP com Corpus *Desconhecido*

Aqui foram usadas as 150 páginas restantes coletadas pelo meta-robô, 3 das quais vieram sem conteúdo. Os resultados da categorização funcional encontram-se na tabela 9. Não surgiram novos padrões neste teste. Uma página gráfica, i. e., sem texto explicativo, de chamada de evento foi analisada pelo sistema e não pôde ser reconhecida, e outra que apenas tinha um *link* para outro foi contabilizada como um evento. As três recomendações foram para as classes *Organização* e *Instituto de Pesquisa*, uma delas erradamente.

A tabela 10 apresenta os resultados da classificação por conteúdo do *corpus* de teste, que se comportou de forma semelhante aos outros testes.

- Agente CFP – Categorização Funcional do *Corpus* de Teste	Reconhecidas	Listas	*Frames*	Recomendadas	Rejeitadas	Total	**Reconhecidas (%)**	Total (%)
Corretas	89	7	0	2	40	138	**93,9**	93,9
Falsas positivas	4	0	0	1	4	9	**2,7**	6,1
Falsas negativas	5	0	0	2	2	9	**3,4**	6,1
Ocorrência (%)	63,9	4,8	0	2,7	28,6	100	100	100

Tabela 9. Categorização funcional do *corpus* de teste do agente CFP.

- Agente CFP - Classificação por Conteúdo *Corpus* de Teste	Conferência	*Workshop*	Jornal	Edição Especial de Jornal	Revista	Edição Especial de Revista	Evento Genérico ao Vivo	Evento Genérico de Publicação	Total	**Total (%)**
Corretas	62	8	0	1	3	1	3	5	83	**93,3**
Falsas Positivas	2	3	0	0	0	0	0	1	6	**6,7**
Falsas Negativas	3	0	0	0	0	0	2	1	6	**6,7**
Não-reconhecidas	3	0	0	0	0	0	1	0	4	-
Ocorrência (%)	72,3	8,5	0	1	3,2	1	7,4	6,4	100	**100**

Tabela 10. Resultados do agente CFP classificando por conteúdo o *corpus* de teste.

B.1.3. Teste do Agente CFP na Web

Este teste recuperou e classificou páginas diretamente da Web, com uma diferença de meses em relação à recuperação de páginas para os testes anteriores, o que fez com que o conjunto de páginas processados neste teste possuísse mais de 60% de páginas desconhecidas. Houve mais erros do que acertos nas páginas recomendadas, conforme

exposto na tabela 11 logo abaixo. Duas conferências foram confundidas com organizações. A classificação por conteúdo também não trouxe novidades e pode ser observada na tabela 12.

- Agente CFP – Categorização Funcional do Teste na Web, sem usar Listas	Reconhecidas	Listas	*Frames*	Recomendadas	Rejeitadas	Total	**Reconhecidas (%)**	Total (%)
Corretas	89	15	0	1	16	121	**94,7**	93,9
Falsas positivas	1	2	0	3	2	8	**1,1**	6,1
Falsas negativas	4	0	0	1	3	8	**4,3**	6,1
Ocorrência (%)	72,1	11,6	0	1,6	14,7	100	100	100

Tabela 11. Categorização funcional do teste na Web do agente CFP, sem usar *links* vindos da categoria funcional Lista.

- Agente CFP - Classificação por Conteúdo Teste na Web, sem uso de Listas	Conferência	*Workshop*	Jornal	Edição Especial de Jornal	Revista	Edição Especial de Revista	Evento Genérico ao Vivo	Evento Genérico de Publicação	Total	**Total (%)**
Corretas	66	9	3	1	1	0	1	2	83	**93,3**
Falsas Positivas	1	3	1	0	0	0	0	1	6	**6,7**
Falsas Negativas	3	0	0	1	0	0	1	1	6	**6,7**
Não-reconhecidas	1	0	0	0	0	0	0	0	1	-
Ocorrência (%)	78,5	9,7	3,2	2,1	1	0	2,1	3,2	100	**100**

Tabela 12. Resultados do agente CFP classificando por conteúdo páginas da Web, sem usar *links* vindos de listas.

B.1.4. Teste do Agente CFP na Web Usando Listas

Aqui, padrões novos e completamente diferentes dos testes anteriores, face ao uso de listas, foram encontrados e podem ser observados na tabela 13. Estes padrões estão descritos na subseção 6.7.1., junto com comentários sobre ganhos a partir do uso de listas. Ressalte-se a evidente redução numérica em nas categorias funcionais inúteis para o agente, como recomendações e rejeitadas, e de listas, que não são diretamente úteis. Em contra-partida, a categoria de *frames,* recheada de páginas-conteúdo, passou a ser muito freqüente. Dos 48 frames achados, apenas 8 não constituem chamadas de eventos.

- Agente CFP – Categorização Funcional do Teste na Web, usando Listas	Reconhecidas	Listas	*Frames*	Recomendadas	Rejeitadas	Total	**Reconhecidas (%)**	Total (%)
Corretas	109	8	48	0	16	181	**94,8**	96,3
Falsas positivas	2	3	0	0	2	7	**1,7**	3,7
Falsas negativas	4	0	0	0	3	7	**3,5**	3,7
Ocorrência (%)	60,1	4,3	25,5	0	10,1	100	100	100

Tabela 13. Categorização funcional do teste na Web do agente CFP, usando *links* de listas.

A classificação por conteúdo é mostrada na tabela abaixo.

- Agente CFP - Classificação por Conteúdo Teste na Web, usando Listas	Conferência	*Workshop*	Jornal	Edição Especial de Jornal	Revista	Edição Especial de Revista	Evento Gen. ao Vivo	Evento Gen. de Publicação	Total	**Total (%)**
Corretas	84	9	2	1	0	0	0	4	100	**91,7**
Falsas Positivas	2	2	2	0	0	0	1	2	9	**8,3**
Falsas Negativas	2	1	1	2	1	0	2	0	9	**8,3**
Não-reconhecidas	2	0	0	0	0	0	0	0	2	-
Ocorrência (%)	78,8	8,8	2,7	2,7	0,9	0	1,8	4,4	100	**100**

Tabela 14. Resultados do agente CFP classificando por conteúdo o *corpus* de teste.

B.2. Agente de Artigos Científicos (PPR)

B.2.1. Teste do Agente PPR com Corpus *para Aquisição de Conhecimento*

Além dos números apresentados na tabela 15, três artigos com poucos dados, i. e., impossíveis de ser reconhecidos, foram categorizados como rejeitados.

- Agente PPR - Categorização Funcional do *Corpus* de Aquisição	Reconhecidas	Recomendadas	Rejeitadas	Total	**Reconhecidas (%)**	Total (%)
Corretas	68	5	111	184	**93,1**	96,8
Falsas positivas	0	0	6	6	**0**	3,2
Falsas negativas	5	1	0	6	**6,9**	3,2
Ocorrência (%)	38,4	3,2	58,4	100	100	100

Tabela 15. Categorização funcional do agente de artigos científicos sobre o corpus de aquisição de conhecimento.

- Agente PPR – Classificação por Conteúdo do *Corpus* de Aquisição	Artigo de Conferência	Artigo de *Workshop*	Artigo de Jornal	Artigo de Revista	Tese	Dissertação	Relatório Técnico	Relatório de Projeto	Capítulo de Livro	Publicação Genérica	Total	**Total (%)**
Corretas	21	2	1	3	1	1	1	2	1	33	66	**97**
Falsas Positivas	1	0	0	0	0	0	0	0	0	1	2	**3**
Falsas Negativas	0	0	0	0	0	0	1	0	0	1	2	**3**
Não-reconhecidas	0	1	0	0	0	2	0	0	0	5	8	**-**
Ocorrência (%)	**27,6**	**3,9**	**1,3**	**3,9**	**1,3**	**3,9**	**2,6**	**2,6**	**1,3**	**51,3**	**100**	**100**

Tabela 16. Resultados do agente PPR classificando por conteúdo o *corpus* usado para aquisição de conhecimento.

B.1.2. Teste do Agente de Artigos com Corpus *Desconhecido*

Aqui, os artigos "impossíveis" eram em número de seis.

- Agente PPR - Categorização Funcional do *Corpus* de Teste	Reconhecidas	Recomendadas	Rejeitadas	Total	**Reconhecidas (%)**	Total (%)
Corretas	43	11	87	141	**82,7**	94
Falsas positivas	0	0	9	9	**0**	6
Falsas negativas	9	0	0	9	**17,3**	6
Ocorrência (%)	34,7	7,3	58	100	100	100

Tabela 17. Categorização funcional do *corpus* de teste do agente PPR.

As recomendações referem-se a páginas com "instruções para autores" com dados de um evento a serem sugeridas ao agente CFP. Este tipo de página só aconteceu nos testes com *corpus*, não possibilitando cooperação nos testes da Web.

- Agente PPR – Classificação por Conteúdo *Corpus* de Teste	Artigo de Conferência	Artigo de *Workshop*	Artigo de Jornal	Artigo de Revista	Tese	Dissertação	Relatório Técnico	Relatório de Projeto	Capítulo de Livro	Publicação Genérica	Total	Total (%)
Corretas	6	3	0	0	0	3	0	0	0	28	40	**93**
Falsas Positivas	0	0	0	0	0	0	0	1	0	2	3	7
Falsas Negativas	1	0	0	0	0	0	1	0	0	1	3	7
Não-reconhecidas	1	2	3	0	0	0	0	0	0	9	15	-
Ocorrência (%)	**13,8**	**8,6**	**5,2**	**0**	**0**	**5,2**	**1,7**	**0**	**0**	**65,5**	**100**	**100**

Tabela 18. Resultados do agente PPR classificando por conteúdo o *corpus* de teste.

B.1.3. Teste do Agente de Artigos na Web

Este teste apresentou mais páginas rejeitadas que os outros, provavelmente devido ao processamento de páginas muito grandes, que nos testes fora da Web, eram eliminados pelo próprio meta-robô. Foram encontrados cinco artigos impossíveis neste teste.

- Agente PPR - Categorização Funcional do Teste na Web, sem Listas	Reconhecidas	Recomendadas	Rejeitadas	Total	**Reconhecidas (%)**	Total (%)
Corretas	43	13	119	175	**87,8**	95,1
Falsas positivas	0	1	8	9	**0**	4,9
Falsas negativas	6	3	0	9	**22,2**	4,9
Ocorrência (%)	26,6	8,7	64,7	100	100	100

Tabela 19. Categorização funcional do teste na Web do agente de artigos científicos, sem usar *links* vindos da categoria funcional Lista.

- Agente PPR – Classificação por Conteúdo Teste na Web, sem uso de Listas	Artigo de Conferência	Artigo de *Workshop*	Artigo de Jornal	Artigo de Revista	Tese	Dissertação	Relatório Técnico	Relatório de Projeto	Capítulo de Livro	Publicação Genérica	Total	**Total (%)**
Corretas	9	0	1	0	0	0	1	1	0	23	35	**81,4**
Falsas Positivas	0	0	0	0	0	2	1	0	0	5	8	**18,6**
Falsas Negativas	3	0	1	0	2	0	0	0	1	2	8	**18,6**
Não-reconhecidas	1	0	2	0	0	1	0	0	0	7	11	-
Ocorrência (%)	**24,1**	**0**	**5,6**	**0**	**3,7**	**1,8**	**1,8**	**1,8**	**3,7**	**57,4**	**100**	**100**

Tabela 20. Resultados do agente PPR classificando por conteúdo páginas colhidas na Web, sem usar os *links* contidos nas listas.

REFERÊNCIAS

[Ait-Kaci & Podelski 93] Aït-Kaci,H.,Podelski, A.; 1993 Towards a Meaning of LIFE. *Journal of Logic Programming,* 16(3&4).

[Akman & Surav 96] Akman,V., Surav,M.;1996.*Context.*AI Magazine. Fall 96, pp 54-72.

[AgentBuilder 98] AgentBuilder. 1998. www.agentbuilder.com/

[Alvares & Sichman 97] Alvares, L. O., Sichman, J. 1997. *Introdução aos Sistemas Multiagentes*, Anais do EINE – Escola de Informática do Nordeste, Sociedade Brasileira de Computação – SBC, Brasil.

[Ambite & Knoblock 97) Ambite, J.; Knoblock, C.: 1997..*Agents for Information Gathering.* In Software Agents. Bradshaw, J. Ed., MIT Press, Pittsburgh, PA, USA

[Appelt & Israel 99] Appelt, D. E.; Israel, D. J.: 1999. *Introduction to Information Extraction Technology.* International Joint Conference of Artificial Intelligence.

[Arens et al 96] Arens Yigal, Hsu, Chun-Nan, Knoblock, Craig, 1996. *Query Processing in the SIMS Information Mediator,* in Huhns, Michael, Singh, Munindar, Eds. 1997. *'Readings in Agents'*, Morgan Kaufman Publishers, USA.

[Artale et al 96] Artale, A., Franconi, E., Guarino, N., Pazzi, L. 1996. *Part-Whole Relations in Object-Centered Systems: An Overview.*

http:// www.ladseb.pd.cnr.it/infor/Ontology/Papers/Parts.ps

[Ashish & Knoblock 97] Ashish, N.; Knoblock, C.: 1997.*Wrapper Generation for Semi-structured Internet Sources.* SIGMOD Record, 26(4):8-15. USA.

[Austin 62] Austin, J. L.; 1962.*How to Do Things with Words.*Clarendon Press,Oxford, UK.

[Baalen & Fikes 93] Van Baalen, J., Fikes, R..; 1993. *The Role of Reversible Grammars in Translating Between Representation Languages.* Disponível na Internet em: http://ksl-web.stanford.edu/knowledge-sharing/papers/index.html#reversible-grammars

[Barros & Robin 96] Barros, F., Robin, J.: 1996. *Processamento de Linguagem Natural.* JAI/SBC - Jornada de Atualização em Informática, Sociedade Brasileira de Computação – SBC, Brasil.

[Bayardo *et al* 96] Bayardo Jr., R. J., Bohrer, W., Brice, R., Cichoki, A., Fowler, J., Helal, A., Kashyap, V., Ksiezyk, T., Martin, G., Nodine, M., Rashid, M., Rusiunkiewicz, M., Shea, R., Unnikrishnan, C., Unruh, A,Woelk, D. 1996. 'InfoSleuth: Agent-Based Semantic Integration of Information in Open and Dynamic Environments', in Huhns, Michael, Singh, Munindar, Eds. 1998 .*Readings in Agents.* Morgan Kaufman Publishers, USA.

[Bear et al 98] Bear, John; Israel, David; Petit, Jeff; Martin, David. 1998. *Using Information Extraction to Improve Document Retrieval.* http:// www.cpe.ku.ac.th/~arnon/trec6_papers/trec6_27.ps.gz

[Benjamins et al 98] Benjamins, R.; Fensel,D.; Pérez, A. 1998. *Knowledge Management through Ontologies.* Proceedings of the 2nd International Conference on Practical Aspects of Knowledge Management, Basel, Suiça.

[Bertino et al 2001] Bertino, E., Catania, B., Zarri, G. 2001. *Intelligent Database Systems.* Addison Wesley, Inglaterra.

[Bikel et al 98] Bikel, D., Miller, S., Schwarz, R., Weischedel, R. 1998. *Nymble: a High-Performance Learning Name-finder.* http://www.csi.uottawa.ca/tanka/ArtDB/anlp.corrected.ps

[Bittencourt 98] Bittencourt, Guilherme; 1998. *Inteligência Artificial : Ferramentas e Teorias.* Editora da UFSC. Florianópolis, SC.

[Bittencourt 97] Bittencourt, G. 1997. *In the Quest of the Missing Link.* International Joint Conference of Artificial Intelligence. Nagoya, Japan.

[Bollacker et al 98] Bollacker, Kurt; Lawrence, Steve; Giles, C. Lee. 1998. *CiteSeer: An Autonomous Web Agent for Automatic Retrieval and Identification of Interesting Publications.* Proceedings of the 2nd International ACM Conference on Autonomous Agents, USA.

[Botelho & Ramos 2000] Botelho, L., Ramos, P. 2000. *Extending the FIPA ACL Language. From Object Based Descriptions to Relational Representations.* Em

Alvares, L. 2000. Proceedings of the Workshop on Distributed Artificial Intelligence and Multi-Agent Systems (DAIMAS2000), International Joint Conference SBIA'2000 (the Brazilian Artificial Intelligence Symposium) and IBERAMIA'2000 (the Ibero-American Articial Intelligence Conference), Edusp, Sâo Paulo.

[Bouquet 97] Bouquet, P. 1997. *The Problem of Context from the Stand-point of Artificial Intelligence.* Technical Report # 9707-01. Istituto Trentino di Cultura, Trento, Itália.

[Brachman & Schmolze 85] Brachman,R., Schmolze, J. 1985. *An Overview of the KL-ONE Knowledge Representation System.* Cognitive Science, 9(2):171-216, USA.

[Burke et al 97] Burke, R.; Hammond, K.; Kulyukin, V.; Lytinen, S.; Tomuro, N.; Schoenberg, S. 1997. *Question Answering from Frequently-Asked Question Files: Experiences with the FAQ Finder System.* Technical Report TR-97-05. University of Chicago, Department of Computer Science, EUA.

[Chaib-draa 94] Chaib-draa,B. 1994. Industrial Applications of Distributed AI. In Huhns, Michael, Singh, Munindar, Eds. 1997. *Readings in Agents.* Morgan Kaufman Publishers, USA.

[Chandrasekaram 86] Chandrasekaran, B. 1986. *Generic Tasks in Knowledge-based Reasoning: High-Level Building Blocks for Expert System Design.* IEEE Expert, 1986 (Fall), p23-30, USA.

[Chandrasekaram 99] Chandrasekaran, B. 1999. *What Are Ontologies, and Why Do We Need Them?.* IEEE Inteligent Systems, Janeiro 7 Fevereiro 99, EUA.

[Chandrasekaram & Josephson 97] Chandrasekaram, B. and Josephson, J. R.; 1997. *The Ontology of Tasks and Methods.* In Proceedings of th Symposium on Ontological Engineering, 1997, Stanford, CA, USA.

[Chaudri et al 98] Chaudri, V., Farquhar, A., Fikes, R., Karp, P., Rice, J. 1998. *OKBC: A Programmatic Foundation for Knowledge Base Interoperability.* Proceedings of AAAI-98, Madison, WI.

[Chekuri et al 95] Chekuri,C.;Goldwasser,M.;Raghavan,P.;Upfal,E. 1995. *Web Search Using Automatic Classification.* http://theory.Stanford.edu/people/wass/publications/Web_ Search.

[Chen et al 96] Chen, H.; Schuffels,C.; Orwig,R. 1996. *Internet Categorization and Search: A Self-Organizing Approach.* Journal of Visual Communications and Image Representation, 7(1):88--102, March 1996, USA.

[Chen et al 96a] Chen, H.; Schatz, B.; Ng, T.; Martinez, J.; Kirchhoff, A.; Lin, C. 1996. *A Parallel Computing Approach to Creating Engineering Concept Spaces for Semantic Retrieval: The Illinois Digital Library Initiative Project.* IEEE Transactions on Pattern Analysis and Machine Intelligence, USA.

[Clark 99] Clark,D. 1999. *Mad cows, metathesauri, and meaning.* IEEE Intelligent Systems. Jan/Fev 99, USA.

[Classic 2002] http://www.research.att.com/sw/toolsclassic/imp-systems.html.

[Cohen 96] Cohen, W. 1996. *Learning Rules that Classify E-mail.*

http://www.parerox.com/istl/ projects /mlia/papers/cohen.ps

[Cohen & Singer 96]. Cohen, W. and Singer, Y. 1996. *Learning to Query the Web,* AAAI Workshop on Internet-Based Information Systems.

[CORBA 98] 1998. '*Object Menagement Group Home Page*'. http://www.corba.org,.

[Corcho & Gómez-Pérez 96] Corcho, O., Gómez-Pérez, A. 1996. *Guidelines to Study Differences in Expressiveness between Ontology Specification Languages: A Case of Study.* http://

[Coulouris et al 94] Coulouris,G.; Dollimore,J.;Kindberg,T. 1994. *Distributed Systems: Concepts and Design*, Second Edition. Addison-Wesley. USA.

[Cowie & Wilks 96] Cowie, J., Wilks, Y. 1996. *Information Extraction.* ACM, 39(1):80--101, 1996. USA.

[Craven et al 99] Craven,M., McCallum,A.M., DiPasquo,D., Mitchell,T., Freitag,D., Nigam,K., Slattery,S. 1999. *Learning to Extract Symbolic Knowledge from the World Wide Web.* Technical Report CMU-CS-98-122. School of Computer Science. Carnegie Mellon University,USA.

[Croft 93] Croft, W. B. 1993. *Knowledge-Based and Statistical Approaches to Text Retrieval.* IEEE Expert. Abril 1993. USA.

[Crow and Shadbolt 99] Crow, L.; Shadbolt, N. 1999. *IMPS – Internet Agents for Knowledge Engineering*. http://ksi.ucalgary.cal /KAW98S/Shadbolt.

[Cruz et al 98] Cruz, I.; Borisov, S.; Marks, M. A.; and Webb, T.R.; 1998. *Measuring Structural Similarity Among Web Documents: Preliminary Results*. Proceedings of the Electronic Publishing, Artistic Imaging, and Digital Typography 7th International Conference on Electronic Publishing, EP'98. St. Malo, France.

[Decker et al 96] Decker, K., Sycara, K., Williamson, M. 1996. *Middle-Agents for the Internet*. http://www.cs.cmu.edu

[Dunkel et al 96] Dunkel, Brian; Soparkar, Nandit; Weinstein, Peter. 1996. *Customized Metadata for Internet Information* .http://www-personal.engin.umich.edu/~bedunkel/ papers/kes.ps

[Elliott 1994] Elliott, C. 1994. *Research problems in the use of a shallow Artificial Intelligence model of personality and emotion*. 12th National Conference on Artificial Intelligence (AAAI'94). AAAI Press. Seatle, USA

[Embley et al 98] Embley, D., Campbell, D., Liddle, S., Smith, R.: 1998. *Ontology-Based Extraction of Information from Data-Rich Unstructured Documents*. http://www.deg.byu.edu/papers/cikm98.ps

[Erdmann et al 2000] Erdmann, M., Maedche, A., Schnurr, H.-P., Staab, S. 2000. *From Manual to Semi-automatic Semantic Annotation: About Ontology-based Text Annotation*. http://www.aifb.uni-karlsruhe.de

[Eriksson 2000] JessTab plugin for Protégé. http://www.ida.liu.se/~her/JessTab/

[Falcão et al 97] Falcão, P. F., Salgado, A. C., Meira, S., 1997. *'BRight: A Distributed System for Web Information Indexing and Searching'*. Anais do Simpósio Brasileiro de Engenharia de Software(SBES'97), Fortaleza - CE, Brasil.

[Farquhar 96] Farquhar, A., Fikes, R., Rice, J.; 1996. *The Ontolingua Server: a Tool for Collaborative Construction*, Computer Science Department, Stanford University.

[Farquhar 97] Farquhar, A. 1997. *Ontolingua tutorial*. http://www.ksl.stanford.edu/software/ontolingua/tutorial.pdf

[Fensel 2001] Fensel, D. 2001. *Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce.* Springer-Verlag. Berlim.

[Figueira Filho & Ramalho 2000] Figueira Filho, C., Ramalho, G. 2000. *JEOPS – The Java Embedded Production System.* Proceedings of the International Joint Conference SBIA'2000 (the Brazilian Artificial Intelligence Symposium) and IBERAMIA'2000 (the Ibero-American Articial Intelligence Conference). Springer-Verlag. Berlim.

[Fikes 98] Fikes, Richard; 1998. *Reusable Ontologies: A Key Enabler for Eletronic Commerce.* Disponível pela Internet em: http://www-ksl.stanford.edu.

[Finin et al 94] Finin, T.; Fritzson, R.; McKay, D.; and McEntire, Robin; 1994. *KQML as an agent communication language.* Proceedings of the International Conference on Information and Knowledge Management. ACM Press, NY.

[Finin & Labrou 99] Finin,T., Labrou,Y. 1999. *Agent Communication Languages.* Slides do Tutorial no ASA/MA. www.umbc.edu

[Florescu et al 98] Florescu,D., Levy,A., Mendelzon,A. 1998. *Database Techniques for the Worlrd Wide Web.* SIGMOD record, 27(3):59-74.USA.

[Forgy 82] Forgy, C. L.; 1982. Rete: A Fast Algorithm for the Many Pattern/ Many Object Pattern Match Problem, *Artificial Intelligence* 19(1982), 17-37. USA.

[Franklin & Graesser 1996] Franklin & S., Graesser, A. 1996. *Is it an Agent, or just a program?: A Taxonomy for Autonomous Agents.* Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages. Springer-Verlag, Berlin.

[Freitas et al 99] Freitas, F., Siebra, C., Ferraz, C., Ramalho, G.: 1999. *Mediation services for agents integration.* Proceedings of SEMISH'99. Soc. Brasileira de Computação (SBC). Rio. Brazil.

[Freitas & Bittencourt 2000] Freitas, F.; Bittencourt, G. 2000. *Cognitive Multi-Agent Systems for Integrated Information Retrieval and Extraction over the Internet.* Proceedings of the International Joint Conference SBIA'2000 (the Brazilian Artificial Intelligence Symposium) and IBERAMIA'2000 (the Ibero-American Articial Intelligence Conference). Springer-Verlag. Berlim.

[Freitas 2001] Freitas, F. 2001. Ontology of Science. http://protege.stanford.edu/plugins/ontologyOfScience/ontology_of_science.htm)

[Freitas et al 2001] Freitas, F.; Bittencourt, G., Calmet, J. 2001. *MASTER-Web: An Ontology-based Internet Data Mining Multi-Agent System.* Proceedings of the Second International Conference on Advances in Infrastructure for E-Business, E-Science and E-Education on the Internet SSGRR-2001. L'Aquila, Itália.

[Freitas & Bittencourt 2002] Freitas, F.; Bittencourt, G. 2002. *Comunicação entre Agentes em Ambientes Distribuídos Abertos: o Modelo "peer-to-peer".* Revista Eletrônica de Iniciação Científica (REIC). Ano II No. II Vol. II, Edição de Junho de 2002. Sociedade Brasileira de Computação (SBC). Brasil. http://www.sbc.org.br/reic/edicoes/2202e2/informativos/ComunicacaoEntreAgentesEmAmbientesDistribuidosAbertos-OModeloPeerToPeer.pdf

[Friedmann-Hill 2000] Friedmann-Hill, E. 2000. *Jess, The Java Expert System Shell.* http://herzberg.ca.sandia.gov/Jess

[Gatzauskas & Robertson 97] Gatzauskas, R., Robertson, A. 1997. *Coupling Information Retrieval and Information Extraction: A New Text Technology for Gathering Information from the Web.* Proceedings of RIAO, Computer-Assisted Information Search on the Internet. Montreal, Canadá.

[Genesereth & Fikes 92] Genesereth, M. R., Fikes, R. E.; 1992. *Knowledge Interchange Format, Version 3.0 Reference Manual*, Logic-92-1, Computer Science Department, Stanford University, EUA.

[Giarratano & Riley 98] Giarratano, J., Riley, G. 1998. *Expert Systems: Principles and Programming.* Brooks/Cole Pub Co. USA.

[Gibson et al 98] Gibson,D., Kleinberg,J., Raghavan, P. 1998. *Inferring Web Communities from Link Topology.* Proc. 9th ACM Conference on Hypertext and Hypermedia, USA.

[Gómez-Pérez 94] Gómez-Pérez, A. 1994. *From Knowledge Based Systems to Knowledge Sharing Technology: Evaluation and Assessment.* http://

[Grishman 97] Grishman, R. 1997. *Information Extraction: Techniques and Challenges.* In Maria Teresa Pazienza, editor, Information Extraction. Springer-Verlag, Lecture Notes in Artificial Intelligence, Rome, Itália.

[Gruber 92] Gruber, Thomas .R.; 1992. *Ontolingua: A Mechanism to Support Portable Ontologies*. Technical Report KSL-91-66. Stanford University, Knowledge Systems Laboratory, EUA.

[Gruber 95] Gruber, Thomas R.; 1995. Towards Principles for the Design of Ontologies Used for Knowledge Sharing. *International Journal of Human and Computer Studies*, 43(5/6): 907-928.

[Guarino 97] Guarino, N. 1997. *Some Organizing Principles for a Unified Top-Level Ontology.* Proceedings of the American Association of Artificial Intelligence Conference (AAAI-97) in the Spring Symposium on Ontological Engineering, EUA.

[Hayes-Roth et al 78] Hayes-Roth, F., Waterman, D. A. and Lenat, D. B.; 1978. Principles of Pattern-Directed Inference Systems. In *Pattern-Directed Inference Systems,* Academic Press, New York, NY, USA.

[HPSearch 2002] HPSearch. 2002. http://hpsearch.uni-trier.de/

[Huffman 95] Huffman, S. 1995. *Learning information Extraction patterns from examples.* Proceedings of the International Joint Conference of Artificial Intelligence. Workshop on new approaches to learning for natural language processing.

[Huhns & Singh 97] Huhn, Michael N. and Singh, Munindar P.; 1997. *Agents and Multiagent Systems: Themes, Aproaches, and Challenges.* In Huhns, Michael, Singh, Munindar, Eds. 1997. Readings in Agents. Morgan Kaufman Publishers, USA.

[Huhns & Singh 97a] Huhn, Michael N. and Singh, Munindar P.; 1997. *Ontologies for Agents*. IEEE Internet Computing 1(6):81-83. Na coluna "Agents on the Web". EUA.

[Huhns & Singh 97b] Huhn, Michael N. and Singh, Munindar P.; 1997. *The agent test.* IEEE Internet Computing. Setembro / Outubro de 1997. Na coluna "Agents on the Web". EUA.

[JATLite 98] *'JATLite'*, http://java.stanford.edu.

[JDBC 99] Sun Microsystems. 1999. *'JDBC – Connecting Java and Databases'*, http://www.javasoft.com/products/jdk/1.1/docs/guide/jdbc/index.html.

[Jini 2000] The Jini Community. 2000. *'What is Jini Connection Technology?'* http://www.jini.org/whatisjini.html

[Karp et al 95] Karp, P. D., Myers, K., Gruber, T.; 1995. *The Generic Frame Protocol.* 14th International Joint Conference on Artificial Intelligence, Montreal, Canada.

[Kifer et al 95] Kifer, M., Lausen, G., Wu, J. 1995. *Logical Foundations of Object-Oriented and Frame-Bsed Languages.* Journal of the ACM. 42:741-843. EUA.

[Knapik & Johnson 98] Knapik, M., Johnson, J. 1998. *Developing Intelligent Agents for Distributed Systems*, Mc-Graw-Hill. EUA.

[Kushmerick 99] Kushmerick, N: 1999. *Wrapper Induction.* http://www.compapp.dcu.ie/ ~nick/ research/wrappers

[Kushmerick 99a] Kushmerick, N: 1999. *Gleaning the Web.* IEEE Intelligent Systems. Mar/Abr 99. EUA.

[Lamounier 95] Lamounier, I. 1995. *A Framework for Comparing Text Categorization Approaches.* http:// www.parc.xerox.com/istl/projects/mlia/papers/moulinier.ps

[Lazarinis 97] Lazarinis, F. 1997. *Combining Information Retrieval with Information Extraction for Efficient Retrieval of Calls for Papers.* ICSG.Glasgow, Escócia.

[Lenat 90] Lenat, D. 1990. *Towards Programs with Common Sense.* Communications of ACM. 33(8):30-49. USA.

[Leong et al 96] Leong,H.; Kapur, S.; de Vel,O. 1996. *Text Summarization for Knowledge Filtering Agents in Distributed Heterogeneous Environments.* In Working Notes of the AAAI--97 Spring Symposium on Natural Language Processing Tools for the World-WideWeb, pages 87--94, Stanford, CA, EUA.

[Lesser et al 97] Lesser, Victor; Horling, Bryan; Klassner, Frank; Raja, Anita; Wagner, Thomas; Zhang, Shelley. 1997. *Information Gathering as a Resource Bounded Interpretation Task.* Computer Science Technical Report 97-34. Computer Science Department. University of Massachusetts, Amherst, EUA.

[Lopez 98] Lopez, Mauricio 1998. *Access and Integration of Distributed, Heterogeneous Information.* http://www.dyade.inrialpes.fr/mediation/pub/ white_paper. html.

[Luke et al 96] Luke, Sean; Spector, Lee; Rager, David. 1996. *Ontology-Based Knowledge Discovery on the World-Wide Web.* Proceedings of the Workshop on Internet-based Information Systems, AAAI-96 (Portland, Oregon). USA.

[Maes 86] Maes, P. 1986. *Introspection in Knowledge Representation.* Proceedings do ECAI 7, Brighton, Inglaterra.

[Maes 94] Maes, P. 1994. *Modeling Adaptative Autonomous Agents.* Artificial Life Journal. 1(1,2). MIT Press. EUA.

[Maes 97] Maes, P.1997. *Pattie Maes on Software Agents: Humaninzing the Global Computer.* Entrevista na IEEE Internet Computing, julho-agosto de 1997.

[Magnanelli et al 97] Magnanelli, Mario; Erni, Antonia; Norrie, Moira. 1997. *ACADEMIA: An Agent-Maintained Database based on Information Extraction from Web Documents.* http://www.globis.ethz.ch/publications/docs/1998a-men-emcsr.ps.gz

[McCallum et al 1999] McCallum, A.; Nigam, K.; Rennie, J., Seymore, K. 1999. *Building domain-specific search engines with machine learning techniques.* In AAAI Spring Symposium on Intelligent Agents in Cyberspace.

[McCarthy 58] McCarthy, John; 1958. *Programs with Common Sense.* In Proceedings of the Symposium on Mechanisation of Thought Processes , 1:77-84, London. Her Majesty's Stationery Office, UK.

[McGuinness et al 2000] McGuinness, D., Fikes, R., Rice, J., Wilder, J.. *The Chimaera Ontology Environment. Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI 2000).* Austin, Texas. July 30 - August 3, 2000.

[Mediation 98] 1998. '*Mediation Components*', http://www.dyade.inrialpes.fr/mediation/pub/white_paper.html.

[Miller 95] Miller, G. 1995. *WordNet: A Lexical Database for English.* Communications of the ACM. 38(11):39-41. EUA.

[Miner 2002] Miner. 2002. http://miner.uol.com.br

[Minsky 75] Minsky, M.; 1975. A Framework for Representing Knowledge. In *The Psychology of Computer Vision*, p.211-281,McGraw-Hill, New York. USA.

[Minsky 86] Minsky, M.; 1986. *Society of Minds.* Simon & Schuster, Inc. EUA.

[Mooney et al 98] Mooney, Raymond J.; Bennett, Paul N.; Roy, Loriene. 1998. *Book Recommending Using Text Categorization with Extracted Information.* Proceedings of

the AAAI-98/ICML-98 Workshop on Learning for Text Categorization and the AAAI-98 Workshop on Recommender Systems, EUA.

[Moreover 2002] Moreover. 2002. http://www.moreover.com

[Müller 99] Müller, M. 1999. *An Intelligent Multi-Agent Architecture for Information Retrieval from the Internet.* http://mir.cl-ki.uni-osnabrueck.de/~martin/onlinepubs/paam-99/ oyster-paam.html

[Muslea 99] Muslea, I. 1999. *Extraction Patterns for Information Extraction Tasks: A Survey.* http://www.isi.edu/~muslea/RISE/ML4IE/ml4ie.muslea.ps

[Neches 91] Neches, R., Fikes, R., Finin, T., Gruber, T.R.,Patil, R., Senator, T., Swartout, W.; 1991. *Enabling Technology for Knowledge Sharing.* In AI Magazine, 12(3): 36-56.

[Newell 82] Newell, A.; 1982. The Knowledge Level. *Artificial Intelligence* 18(1):87-127.

[Newstracker 2002] Newstracker. 2002. www.newstracker.com

[Noy 97] Noy, N. 1997. *Knowledge Representation for Intelligent Information Retrieval in Experimental Sciences.* Tese de Doutorado, Faculdade de Ciência da Computação, Universidade Nordestina de Boston, Massachusetts, EUA.

[Noy et al 2000] Noy, N., Fergerson, R., Musen., M. 2000. *The knowledge model of Protege-2000: Combining interoperability and flexibility.* 2th International Conference on Knowledge Engineering and Knowledge Management (EKAW'2000), Juan-les-Pins, France .

[Oaks & Wong 99] Oaks, S., Wong, H. 1999. *Java Threads.* O'Reilly & Associates, Inc. Sebastopol, CA, EUA.

[Oates et al 94] Oates, T.; Prasad, M.; Lesser, V. 1994. *Cooperative Information Gathering: A Distributed Problem Solving Approach.* Computer Science Technical Report 94-66- version 2. University of Massachusetts, Amherst, USA.

[O'Brien & Nicol 98] O'Brien, P.; Nicol, R. 1998. *FIPA – towards a standard for software agents.* http://www.

[ODBC 98] 1998. Open DataBase Connectivity., http://www.obdc.com/index.html

[Pachet 1994] Pachet, F. (ed.). 1994. *Workshop on Embedded Object-Oriented Production Systems (EOOPS). Proceedings of the OOPSLA'94* Portland, Oregon, USA.

[Pachet 1995] Pachet, F. 1995. *On the embeddability of Production Rules in Object-oriented Languages*. Journal of Object-Oriented Programming (Joop).Jul-ago 1995.

[PC Week 2000] PC Week. 2000. *DAML could take search to a new level.* February 7,2000. USA.

[Petrie 96] Petrie, Charles J., 'Agent-Based Engineering, the Web, and Intelligence', *IEEE Expert*, Dezembro de 1996. USA.

[Pirolli et al 95] Pirolli,P.,Pitkow,J.,Rao,R.:. 1995. *Silk from a Sow's Ear: Extracting Usable Structures from Web.* http://www.acm.org/sigchi/chi96/proceedings/papers/Pirolli_2/pp2.html

[Price 95] Price Waterhouse. 1995. *Welcome to Knowledge View.* Dallas. TX. USA.

[Rich 89] Rich, E. 1989. *Stereotypes and User Modelling.* In: User Models in Dialog Systems, A. Kobsa e W.Wahlster (eds).

[Riley 99] Riley, G. 1999. *CLIPS: A Tool for Building Expert Systems.* http://www.ghg.net/ clips /CLIPS.html

[Riloff 94] Riloff, E.. 1994.. *Information Extraction as a Basis for Portable Text Classification Sytems.* PhD.thesis.Dept. of Computer Science.Univ. of Mass. Amherst. USA.

[Russel & Norvig 95] Russel, Stuart J. and Norvig, Peter; 1995. *Artificial Intelligence: A Modern Approach.* Prentice Hall, Upper Saddle River, NJ. USA.

[Sahami et al 97] Sahami, M., Yusufali,S., Baldonado,M. 1997. *Real-time Full-text Clustering of Networked Documents.* Proceedings da Conferência do AAAI-97. EUA.

[Sahuguet & Azavant 99] Sahuguet, A. & Azavant, F. 1999. *Building light-weight wrappers for legacy Web data-sources using W4F.* International Conference on Very LargeDatabases(VLDB).

[Schank 91] Schank, Roger C.; 1991. *Where's the AI?.* In AI Magazine.12:38-49. USA.

[Schreiber et al 94] Schreiber, A., Wielinga, B., de Hoog, R., Akkermans, H., van de Velde, W. 1994. *CommonKADS: A Comprehensive Methodology for KBS Development.* IEEE Expert, Dezembro,28-37, EUA.

[Smith & Lopez 97] Smith,D., Lopez, M., 1998. *Information extraction for semi-structured documents,* http://www.research.att.com/~suciu/WORKSHOP-PAPERS/ paper09. ps.

[Staab & Maedche 2000] Staab,S., Maedche, A. 2000. *Axioms are Objects, too – Ontology Engineering beyond the Modeling of Concepts and Relations.* Proceedings of ECAI 2000, 14th European Conference on Artificial Intelligence , Workshop on Aplications of Ontologies and Problem Solving Methods.

[Steier & Belew1993] Steier, A., Belew, R. 1993. *Exporting phrases: A statistical analysis of topical language.* In Second Annual Symposium on Document Analysis and Information Retrieval.

[Steele 2001] Steele, R.; 2001. *Techniques for Specialized Search Engines.* http://www-staff.it.uts.edu.au/~rsteele/SpecSearch3.pdf

[Sun 98a] Sun Microsystems. 1998. *'The source for Java Technology',* http://www.javasoft. com/

[Uschold & Jasper 99] Uschold, M., Gruninger, M. 1999. *A Framework for Understanding and Classifyig Ontology Applications.* http:// sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-18/11-uschold.pdf

[Valente & Breuker 96] Valente, A., Breuker, J. 1996. *Towards Principled Core Ontologies.* In B.R. Gaines and M. Mussen, editors, Proceedings of the KAW-96, Banff, Canada.

[Valente et al 99] Valente, A.,Russ, T., MacGregor, R., Swartout, W. 1999. *Building and (Re)Using an Ontology of Air Campaign Planning*, IEEE Expert, January/February. EUA.

[van de Velde 95] van de Velde, W.; 1995. *Reuse in Cyberspace.* Abstract at the Dagstuhl seminar 'Reusable Problem-Solving Methods'. Musen, M.; Studer, R. Orgs. Dagstuhl, Germany.

[Villain 99] Villain, M. 1999. *Inferential Information Extraction.* Em Pazienza, M. (Ed.) 1999. Information Extraction – Towards Scalable, Adaptable Systems. LNAI 1714, Springer-Verlag, Berlin. Germany.

[Visser & Cui 98] Visser, P.; Cui, Z. 199. *On Accepting Heterogeneous Ontologies in Distributed Architectures.* Proceedings of the ECAI'98 workshop on Applications of Ontologies and Problem-solving methods, Brighton, UK.

[Warren 83] Warren, D. H. D.; 1983. *The runtime environment for a prolog compiler using a copy algorithm.* Technical Report 83/052, Suny and Stone Brook, New York. USA.

[Watt 89] Watt, David A .;1989. *Programming Language, Concepts and Paradigms.* Prentice-Hall International, UK.

[Wiederhold & Genesereth 97] Wiederhold, Gio, Genesereth, Michael, 1997. *'The Conceptual Basis for Mediation Services'*, IEEE Expert, pp. 38-47, Setembro-Outubro 1997. USA.

[Wielinga *et al* 92] Wielinga, B. J.; Schreiber, A T. and Breuker, J. A.; 1992. KADS A modelling approach to knowledge engineering. *Knowledge Acquisition.* 4(1):5-53.

[Winograd 75] Winograd, T.; 1975. Frame Representations and the declarative/ procedural controversy. In *Representation and Understanding Studies in Cognitive Science*, 185-210, Academic Press, New York, NY. USA.

[Woods 75] Woods, W. A.; 1975. What's in a link: Foundations for semantic networks. In *Representation and Understanding Studies in Cognitive Science*, Academic Press, New York, NY. USA.

[Wooldridge & Jennings 95] Wooldridge, Michael, Jennings, Nick. 1995. *Intelligent Agents: theory and practice. The Knowledge Engineering Review*, 10 (2), 115-152.. 751-763, Dec. 1996.