ASPECTOS RELEVANTES NO PROJETO DE LINGUAGENS PARA ACESSO A BASES DE DADOS BIBLIOGRÁFICAS

Carlos Alberto Mamede Hernandes
Sérgio Dagnino Falcão
Departamento dê Informática
Instituto Brasileiro de Informação em Ciência e Tecnologia
70710 Brasília, DF

RESUMO

Descrevem-se as funções que uma linguagem de recuperação de informações bibliográficas deve abranger e discutem-se as modalidades básicas existentes: linguagem natural por menu e e *linguagem orientada a comandos (CCLj, analisando-se as vantagens e desvantagens de cada uma. São sugeridas características que tornam uma linguagem orientada a comandos amigável e própria para o ambiente de buscas bibliográficas.*

1 - INTRODUÇÃO

Em nossos dias o volume de informações disponíveis vem aumentando a taxas cada vez mais elevadas. Tornaram-se necessárias ferramentas, como as linguagens de recuperação, que permitem aos usuários o acesso às fontes de informação de seu interesse profissional, acadêmico e pessoal. Neste texto são abordadas funções e características que devem ser consideradas no projeto de linguagens de recuperação.

Este texto é o resultado da experiência dos autores como funcionários do Departamento de Informática do Instituto Brasileiro de Informação em Ciência e Tecnologia (IBICT), onde participaram do projeto e dá implementação da LINCE (Linguagem Comum de Recuperação de Informações em Linha). Esta linguagem foi desenvolvida em conjunto por este Instituto e pelo Centro de Informações Nucleares da Comissão Nacional de Energia Nuclear (CIN/CNEN).

2- FUNÇÕES DE LINGUAGENS DE RECUPERAÇÃO

Uma linguagem de recuperação é um instrumento que possibilita a seleção e a recuperação de informações em bases de dados. De acordo com Cochrane, deve incluir as seguintes funções:
- controle operacional da sessão de trabalho;
- formulação e controle de buscas;
- gerenciamento de saídas;
- ajuda ao usuário.

O controle operacional da sessão engloba as tarefas de iniciar, definir seus atributos e finalizá-la; gravar, recuperar e apagar arquivos; e gerenciar atividades postergadas (executadas em *batch),* se estiverem disponíveis na linguagem.

A formulação e o controle de buscas se referem às facilidades que permitem a especificação do perfil das informações de interesse do usuário, incluem operadores (lógicos, relacionais, de proximidade, mascaramento e truncamento) e funções que possibilitam buscas auxiliadas por estruturas de vocabulários controlados (e. g. tesauros estruturados).

O gerenciamento de saídas abrange a exibição em tela e a impressão de informações recuperadas. Permite o controle da exibição e do formato de apresentação da informação.

A ajuda ao usuário consiste em fornecer informações interativamente sobre o sistema, a linguagem e o ambiente da sessão. Por exemplo, sintaxe dos comandos, histórico de consultas, bases disponíveis, parâmetros básicos *(default),* atributos da sessão etc.

3 - MODALIDADES

Existem basicamente três modalidades diferentes para se elaborar uma linguagem de busca de fácil utilização: linguagem orientada a *menus,* linguagem natural de formato livre e

linguagem orientada *a* comandos e argumentos com sintaxe preestabelecida.

As linguagens orientadas a *menus* permitem que a -etapa de treinamento do usuário seja rápida e simples. Além disso, têm a vantagem de reduzir o esforço na análise das ordens do usuário, tornando mais simples sua implementação. Por outro lado, urna tela de *menu* pode apresentar uma série de informações que terão pouca utilidade para o usuário, se este já tiver em mente o que vai fazer. Essas linguagens podem então gerar fluxos desnecessários de informação entre sistema e usuário, que as tornam menos indicadas para ambientes de rede. Apesar de técnicas modernas de navegação minimizarem este problema, permitindo o estabelecimento de seqüências completas de *menus* em um único passo, elas apresentam o inconveniente de obrigar o usuário a memorizar as seqüências de que mais necessita.

A segunda modalidade praticamente elimina a fase de treinamento e oferece uma grande flexibilidade de uso. Entretanto, apesar dos avanços nas áreas de lingüística computacional e inteligência artificial, apresenta uma implementação cara, difícil e com ambigüidades durante a interpretação dos comandos. Estudos realizados por Marcus([6]) indicaram duas razões adicionais para se questionar o valor do uso de linguagens naturais para buscas. Em primeiro lugar, o sucesso obtido com estas linguagens é normalmente alcançado em situações em que o significado das palavras é-restrito a um contexto bem definido, A segunda razão deriva do fato de que as implementações atuais apresentam poucas funções relativamente ao número de opções disponíveis em uma linguagem realmente natural (caso, por exemplo, da linguagem falada ou escrita). Estas duas razões implicam restringir a tal ponto o vocabulário e a liberdade do uso da linguagem que ela perde a sua principal característica, a de ter um formato livre. Setzert([7]) vai mais além, afirmando que as implementações hoje existentes não passam de caricaturas de linguagem natural. Esses sistemas simulam um comportamento inteligente com muitas restrições, obrigando o usuário a conhecer exatamente os itens de dado (nomes das bases, campos etc.) e a usar uma sintaxe restrita.

Existindo estas restrições, é mais adequado treinar o usuário em uma linguagem orientada a comandos com sintaxe preestabelecida. A prática indica ser esta a solução mais viável, pois está mais próxima de um formalismo lógico, possibilitando uma formulação de consulta mais precisa e permitindo implementações mais simples, que consumam menos recursos computacionais.

4 - CARACTERÍSTICAS DESEJÁVEIS

Em uma linguagem comum de comandos (CCL), o usuário expressa cada ação a ser executada pelo sistema através de uma estrutura formada por um comando seguido ou não por argumentos que o qualificam. Para qua a linguagem seja eficiente, atenda às necessidades dos usuários e seja de fácil utilização *(user-friendly),* deve levar em conta os seguintes aspectos:

- ser independente do formato interno das bases, podendo acessar bases com diferentes estruturas e diminuir ou eliminar os efeitos de possíveis alterações nas estruturas das bases pesquisadas;
- possibilitar a mudança de bases de dados dinamicamente, isto é, permitir consultas a mais de uma base durante uma mesma sessão;
- usar espaço como delimitador e minimizar pontuações especiais;
- ser indiferente quanto ao uso de múltiplos espaços em branco e de letras maiúsculas e minúsculas;
- aceitar qualquer comando em qualquer instante, independentemente de seqüências previamente definidas;
- os nomes dos comandos e os símbolos da linguagem devem lembrar claramente as funções que desempenham;
- os comandos deverão permitir formas simplificadas ou completas para atender tanto a usuários iniciantes quanto a experientes;
- permitir aos usuários conhecer o universo de termos (palavras-chave) que formam os índices da base de dados;
- empregar a forma consagrada de expressões booleanas para especificar expressões de busca, utilizando operadores lógicos (união, interseção e subtração) e parênteses para alterar a ordem de precedência dos operadores;
- possuir características para pesquisar em campos numéricos, tais como: aceitar buscas por intervalos de valores e por extremos (maior e menor valor de um campo), bem como o uso de operadores relacionais ($=$, $<$, $>$ etc.);
- suportar operadores de truncamento e mascaramento para executar buscas por radicais de palavras;
- permitir o proveitamento de resultados de consultas anteriores, possibilitando refinamentos sucessivos;
- possuir operadores de proximidade para permitir buscas em campos de texto livre (resumos, textos completos etc.);
- permitir buscas apoiadas por vocabulários controlados, tornando possível associações

no nível semântico, isto é, buscas segundo conceitos e não apenas por palavras-chave;
— rapidez na resposta às solicitações do usuário e exibição de resultados parciais sempre que o resultado final exigir maior tempo de processamento;
— retorno ao usuário *(feed-back)* de informações a respeito da avaliação de sua solicitação. Por exemplo: exibir o número de referências recuperadas após uma busca, desmembramentos dos termos truncados, o número de ocorrências na base de cada termo de uma expressão de busca etc;
— ajuda interativa ao usuário *(help)* ampla e de fácil acesso;
— as mensagens de erro devem ser claras, indicar o ponto do comando em que foi detectado o erro e ter um tom amigável. Em nenhuma hipótese deverão desestimular o usuário;
— possibilitar a configuração de parâmetros da sessão, tornando o ambiente adaptável às necessidades e hábitos de cada usuário;
— permitir que comandos que exijam grande tempo de processamento sejam executados de modo postergado *(batch).*

Uma CCL que incorpore estas características oferece um formato mais próximo ao da linguagem natural. O inconveniente da CCL consiste no fato de exigir por parte do sistema uma validação dos comandos e argumentos digitados (análise sintática) e, por parte do usuário, uma memorização das regras da linguagem.

5 - CONCLUSÃO

Existem várias modalidades de interfaces sistema-usuário para recuperação de informações. No contexto de informações bibliográficas, a que tem se mostrado mais adequada é a de linguagens orientadas a comandos (CCL), pois representa o ponto de equilíbrio entre simplicidade de uso e eficácia. Uma CCL que leve em consideração as características apresentadas neste artigo oferece ao usuário um ambiente ao mesmo tempo ergonômico e poderoso. Além disso, é fundamental contar com a participação dos usuários durante a etapa de projeto da linguagem, pois são eles que conhecem a aplicação e todas as suas particularidades.

É importante ter em mente que uma linguagem de recuperação, por mais sofisticada que seja, não passa de uma ferramenta dentro de um sistema de recuperação de informações. Para que um sistema deste tipo realmente cumpra seus objetivos, é necessário que seus usuários recebam treinamento adequado para conhecerem a linguagem de recuperação, as características das bases (objetivos, área de conhecimento etc.) e as técnicas de formulação de estratégias de busca. Buscas em bases bibliográficas requerem não só recursos computacionais, onde a linguagem de busca se insere, mas também recursos humanos qualificados.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] BUXTON, Andrew & TRENNER, Lesley. An experiment to acess the friendliness of error mensages from interactive information retrieval systems. *Journal of Information Science, 13:* 197-209, 1987.

[2] COCHRANE, Pauline A. Can a standard for a online common language be developed?. *Online,* 7(1): 36-37, Jan. 1983.

[3] FALCÃO, Sérgio D. & HERNANDES, Carlos A. M. *Linguagem de recuperação de informações.* Brasília, UnB, 1988. 2v.

[4] ISO/TC46. *Working draft for search and operational support protocols (command language for interrogation of information retrieval systems).* Paris, 1980. 15p.

[5] NATIONAL INFORMATION STANDARDS ORGANIZATION. *Common command language for online interactive information retrieval,* s. 1., 1 986, 18 p.

[6] MARCUS, S. R. & RElNTJES, J. F. A translating computer interface for end-user operating of heterogeneous retrieval systems. *Journal of the American Society of Information Science,* 32(4): 287-303, jul. 1981.

[7] SETZER, Valdemar W. *Bancos de dados; conceitos, modelos, gerenciadores, projeto lógico e projeto físico. 2. ed.* rev. São Paulo, Editora Edgard Blücher, 1987. 289 p.

RELEVANT ASPECTS IN PROJECTING BIBLIOGRAPHIC DATABASE RETRIEVAL LANGUAGES

ABSTRACT

Describes the functions a bibliographic retrieval language must have and discusses the basic existing types: menu driven, natural language and common command language, analysing the advantages and disadvantages of each one. Sugests features to make it user-friedly and adequated for the bibliographic retrieval environment.