

Documentação técnica (2ª fase)

- secção destinada à implementação das variáveis -
- para ser considerada em conjunto com a documentação técnica desenvolvida na 1ª fase -

A implementação de variáveis (não confundir com nós variáveis) é conseguida pelo código reunido no ficheiro VAR.LSP. A necessidade de variáveis ocorre da necessidade de implementar funções de procura. Estas funções de procura estão devidamente referidas na documentação do utilizador e na respectiva documentação técnica.

Assim esta documentação, é exclusivamente dedicada ao ficheiro VAR.LSP.

De forma a suportar variáveis, criou-se uma nova estrutura, chamada var. Esta estrutura apresenta 2 campos: o identificador da variável (campo id) e o valor respectivo (campo valor).

Esquemáticamente, pode-se considerar que a estrutura var tem o seguinte aspecto:

Var:

ID:
Valor:

Surge ainda uma nova variável global, chamada *variáveis* e cujos elementos vão ser estruturas do tipo var, representando cada uma das variáveis solicitadas pelo utilizador da rede.

Juntamente com esta nova estrutura, vieram naturalmente, novas funções que a manipulam ou de alguma forma são necessárias ao suporte de variáveis na rede semântica. São essas funções que a seguir descrevem.

1.

var? (identificador)

Devolve t se o identificador recebido corresponder ao identificador duma variável; devolve nil caso contrário. Considera-se que um identificador corresponde a uma variável quando começa por ?.

O verdadeiro nome da variável acabará por ignorar o ?, o qual só serve na prática para assinalar à rede semântica que se requisita ou a criação, ou a actualização duma variável.

2.

cria-var (identificador valor)

Função de «baixo nível» que cria uma estrutura representativa duma variável, com o identificador "identificador" e com o valor "valor". Não se destina ao utilizador; até porque, por si só, não insere a variável na lista *variáveis*, o que na prática se traduz pelo não reconhecimento da variável criada, por parte da rede.

3.

devolve-var (identificador)

Função que recebe um identificador, devolvendo nil caso o identificador não corresponda ao identificador de nenhuma variável presente na rede. Caso exista uma variável na lista *variáveis* com este identificador, a estrutura que a representa é devolvida.

4.

valor? (identificador)

Eis a função que permite ao utilizador saber o valor da(s) variável(eis) por ele requisitada(s). Esta função espera o nome duma variável, devolvendo o respectivo valor - caso exista uma variável com aquele identificador - ou produzindo uma mensagem de erro adequada, se não existir uma variável em *variáveis* que lhe corresponda.

5.

existe-var (identificador)

Função que se socorre da função «devolve-var» para determinar a existência / inexistência duma variável com o identificador recebido como argumento na lista *variáveis*. Se tal variável existir, devolve t; caso contrário, devolve nil.

6.

tira-var (identificador)

Mais uma função destinada ao utilizador da rede, que lhe permite eliminar (caso exista) a variável cujo identificador é recebido como argumento. Se a variável não existir, é produzida uma mensagem de erro adequada.

A saber:

Todas as variáveis criadas durante uma sessão de interacção com a rede são perdidas a partir do momento em que se recupera uma rede qualquer dum ficheiro. A operação de gravar a rede em disco - «grava» - ignora as variáveis existentes e, embora não as apague, ao recuperar-se a rede pela operação «carrega», as variáveis anteriormente

existentes, desaparecem.

Esta documentação reporta-se exclusivamente ao ficheiro VAR.LSP, mas não deve ser considerada isoladamente de outras documentações; como as desenvolvidas propositadamente para o ficheiro PROCURA.LSP, para a 1ª fase de desenvolvimento da rede semântica e para o utilizador (fases 1 e 2).

Anexo:Pequeníssimas alterações ocorreram nos ficheiros NOS.LSP e CARREGA.LSP, motivadas pela existência de variáveis e pelas novas funções de procura. Essas alterações são absolutamente mínimas e estão comentadas nos locais em que ocorreram.

Em NOS.LSP, na função gera-nó, onde já antes era averiguada a validade do nome que se pretendia atribuir ao nó, está agora um novo teste cujo objectivo é não permitir a criação de nós cujo nome comece por ?.

De facto, assim evitam-se as ambiguidades que doutra forma surgiriam ao tentar fazer uma procura do género:

(procura 'membro '?quem) :- se existisse um nó chamado ?quem, era suposto a função procura devolver o(s) identificador(s) do(s) nó(s) que tivesse(m) a relação descendente membro para esse nó; no entanto isso não aconteceria, porque a função procura assume que o ? corresponde à requisição de criação ou actualização do valor duma variável (neste caso) de identificador "quem".

Esse problema não existe agora porque, ao ser verificada a existência de solicitação de criação dum nó com nome "proibido", a função gera-nó torna o nome "legal", por remoção do ?.

Como é óbvio, este facto é relato ao utilizador na documentação que lhe compete.

*A outra alteração (foram só duas alterações em ficheiros da 1ª fase) consiste no acréscimo de 1 linha à função CARREGA - essa linha encarrega-se de pôr a nil a variável global *variáveis*.*

Naturalmente que também a shell do utilizador teve de ser alterada de forma a reconhecer os novos comandos e a fornecer ajuda sobre os mesmos.

O pequeno número de alterações produzidas em código já existente, revela a facilidade de manutenção própria do código desenvolvido para a 1ª fase da rede semântica.