Identification

Command_arg
C. Marceau

Purpose

Because of restrictions in the EPL language a procedure written in EPL must have a fixed-length argument list. However, certain Multics commands are called with a variable number of parameters. The command_arg procedure allows a command to obtain all the arguments, regardless of the number of parameters in the compiled procedure.

Usage

To obtain m arguments beginning with the nth argument:

```plaintext
call command_arg (n, count, arg1, arg2, ..., argm);

dcl n fixed bin (17),
  arg1 char (*),
  count fixed bin (17); /*returned by command_arg,
  =total number of arguments
  passed to command_arg's
  caller*/
```

A declaration of char (*) for arg1 is necessary in the calling program, because command_arg supplies specifier and dope for arg1. If command_arg's caller was passed fewer than n+m-1 arguments, say k-1 arguments, then command_arg returns argk, ..., argm = "".

To pass a "return argument" to the calling procedure

```plaintext
call command_arg$return (n, count, arg);

dcl arg char (N); /*command_arg's caller must know
  the length of N*/
```

Implementation

Command_arg obtains the argument list of its caller, say, proc, and obtains from it the address of the nth argument to proc. Call this argument char_arg.
From its own argument list command_arg obtains a pointer to arg1. Because proc declared arg1 char (*), the pointer points to the following structure:

```
<table>
<thead>
<tr>
<th>garbage (pointer to data)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(pointer to dope)</td>
</tr>
<tr>
<td>(addressing offset) 0</td>
</tr>
<tr>
<td>(template) dope</td>
</tr>
</tbody>
</table>
```

Command_arg calls cv_string$cs to fill in the specifier and dope in this structure so that arg1 is equal to char_arg. Similarly, command_arg sets arg2 equal to the (n+1)st arg to its caller, and so on.

If n is not greater than count, command_arg$return (n, count, arg) calls stgop_g$cscs_ to set char_arg (n'th argument to command_arg's caller) equal to arg. If n exceeds count, command_arg simply returns.