

Published: 01/24/68

## Identification

Procedure for Monitoring Operator Functions

op\_checker

K. J. Martin

## Purpose

The procedure op\_checker helps implement a nearly continuous watch on operator functions which involve unsolicited requests. The operator's login responder, op\_listener (described in BX.15.01), adds op\_checker to the end of every command sequence typed in by every operator. Op\_checker goes into an unending wait for all signals associated with unsolicited-request functions. The operator can get out of op\_checker only by quitting. Thus, an explicit action is required not to monitor unsolicited-request functions rather than to monitor them.

Op\_checker also waits for a signal from System Control indicating that a new function has been added to this operator's duties. (See op\_report described in BX.15.02). The op\_here procedure (also described in BX.15.03) is called as a result of a signal from System Control over the appropriate event-call channel.

## Usage

The procedure op\_checker may be explicitly invoked by an operator at command level by typing:

```
op_checker
```

The operator's login responder causes op\_checker to be invoked by appending

```
; op_checker
```

to each command sequence.

Any procedure may call it normally:

```
call op_checker;
```

## Implementation

The procedure op\_checker uses the op\_function segment in the operators process-group directory to determine for what unsolicited-request functions (if any) the operator is responsible. The following structure in op\_function contains that information:

```

dc1 1 op_fcn based (p),
    2 op_pid bit (36), /* operator's process id */
    2 new_tabl_chn bit (70), /* channel for System Control
                             to signal a change in
                             this table */
    2 n_fcns fixed bin (17), /* number of functions for
                             which this operation has
                             responsibility */
    2 fcn (p→op_fcn.n_fcns),
    3 unsol_req bit (1), /* = "1"b if this is an
                          unsolicited-request
                          function */
    3 just_in bit (1), /* = "1"b if this function
                       was added on the most
                       recent addition to the
                       table */
    3 wait_chn bit (70), /* event channel to receive
                          unsolicited-request
                          function on */
    3 name_lgth fixed bin (17), /* no. of significant
                                chars in name of this
                                function */
    3 function_name char (32), /* name of function */
    3 n_hpc fixed bin (17), /* number of significant
                              chars in here_procedure */
    3 here_procedure char (32), /* name of procedure
                                function-here called by
                                op_here */
    3 n_apc fixed bin (17), /* number of significant
                              chars in attach_procedure*/
    3 attach_procedure char (32); /* name of procedure to
                                attach special devices */

```

The steps in op\_checker are:

- 1) If no unsolicited-request functions are assigned to this operator, return.
- 2) For each unsolicited-request function, in case an event signal for this function may have been lost because of quits or changes of operators, call that function's service procedure directly (using fake~~entry~~<sup>call</sup>, BY.10.01). The service procedure returns after servicing all requests which are currently outstanding. The media command (BX.15.09) is one such service procedure.

- 3) When all unsolicited-request functions have been processed, call wait (BQ.6.06) to wait on all the event channels specified in the op\_function segment. When an unsolicited request arrives, op\_checker finds out what function it is associated with and calls the appropriate service procedure. Op\_checker also waits on the event channel associated with the assignment of new functions to the operator. That channel is specified in p→op\_fcn.new\_tbl\_chn. The op\_here procedure (BX.15.02) is called when that event is signalled.

*check entry*