The standard for info segments documenting a subsystem request is to use a Syntax: section title to document usage of that request. This format distinguishes subsystem request info segs from command infos, which use a Syntax as a command: section title.

However, validate_info_seg has an old coding bug which identifies info segs using the Syntax: section title as UNKNOWN format info. Since it has decided their format is unknown, it makes no attempt to validate other section titles in such files.

I discovered this problem when validating the new mbuild.info segment, which documents the mbuild command and each of its subsystem requests.

This problem is described in Ticket #199: [http://multics-trac.swenson.org/ticket/199](http://multics-trac.swenson.org/ticket/199)

The following output fragment from validate_info_seg shows the problem:

```
> t> vis mb

>user_dir_dir>Multics>GDixon>work>mb>mb.info

-----------COMMAND info
LINE: 3   Syntax as a command
LINE: 6   Function
LINE: 22  Control arguments

-----------GENERAL info
LINE: 65  Summary of requests
Severity 2, line 65. Paragraph size (20 lines) exceeds limit of 15.
Severity 3. These lines of section exceed 71 chars: 67, 68, 70, 73, 74, 76, 78, 80, 82, 83, 84, 87, 91, 93, 95, 100

-----------UNKNOWN format info
LINE: 103 :Info: scan: sc: 2020-01-25 scan, sc
LINE: 105  Syntax
LINE: 109  Function
LINE: 113  Control arguments

-----------UNKNOWN format info
LINE: 125  Syntax
LINE: 129  Function
LINE: 135  Control arguments
```
Proposed Changes

The problem occurs because the Syntax section title appears twice in validate_info_seg's list of acceptable section titles.

/* In this list, all names which are the full 41 chars long must be an exact */
/* match to a section name, while the shorter ones only need to match their */
/* length's worth at the beginning. */
dcl std_section     (45) char (41) var int static options (constant) init (  
  " 1Access required                        
  2Arguments                              
  3Control arguments as a command         
  5Control arguments as a request         
  6Control arguments as an active function" 
  7Control arguments as an active request 
  8Control arguments for " ,
  9Entry points in " ,
  "10Examples                             
  11Function                              
  12List of " ,
  13Notes                                 
  14Notes on " ,
  15Syntax" ,
  16Syntax as a command                   
  17Syntax as an active function          
  18Syntax as an active request           
  19Syntax" ,
  "20<untitled>                           
  **** improper forms follow, they reference proper ones above. */
  "17Syntax as active function            
  16Syntax as command                     
  11Purpose ,
  " 2Argument", 
  2Where ,
  " 3Control argument", 
  10Example", 
  " 1Access requirement", 
  " 1Access requirements", 
  13Notes ,
  "17Active function syntax              
  17Active function usage                 
  2Active function arguments              
  2Active function argument               
  2Arguments as active function           
  6Active function control arguments      
  6Active function control argument       
  6Control arguments as active function   
  16Command syntax                       
  16Command usage                        
  " 2Command arguments", 
  2Command argument", 
  " 4Command control arguments", 
  " 4Command control argument", 
  " 4Control argument as command" );
I suspect this problem was introduced by a 1985 change to make a Usage: title no longer acceptable as a standard title in info segs. Instead of removing the “19Usage…” line (shown in RED in the earlier table), the earlier repair changed its title to “19Syntax…”. A constant called USAGE remains in the program, with a value of 19.

The correct repair is to:

- remove the “19Syntax…” line from the table above;
- eliminate all use of the USAGE constant and remove it;
- change the “20<untitled>…” entry in the table to “19<untitled>…”;
- change the value of the UNTITLED constant from 20 to 19.

These changes are summarized in the following code comparison:

```plaintext
mbuild: cmp

---------- validate_info_seg.pl1

compare_ascii >ldd>tools>source>bound_lib_utility_.s.archive::validate_info_seg.pl1 ==

A141 need_usage init ("3This section must be ""Syntax"".")
Changed by B to:
B141 need_syntax init ("3This section must be ""Syntax"".")

A268 dcl std_section (45) char (41) var int static options (constant) init (268)
Changed by B to:
B268 dcl std_section (44) char (41) var int static options (constant) init (268)

A287 "19Syntax"
A288 "20<untitled>
Changed by B to:
B287 "19<untitled>

A337 USAGE init (19),
A338 UNTITLED init (20),
Changed by B to:
B336 UNTITLED init (19),

A347 need_iso_date, need_usage, no_entries, no_entry, no_hdr_name, no_sections,
Changed by B to:
B345 need_iso_date, need_syntax, no_entries, no_entry, no_hdr_name, no_sections,

A1388 L(2): goto L(SEC ( 2, 1, 1, 2, USAGE));
Changed by B to:
B1386 L(2): goto L(SEC ( 2, 1, 1, 2, SYNTAX));

A1410 L(2): goto L(SEC ( 2, 1, 1, 2, USAGE));
Changed by B to:
B1408 L(2): goto L(SEC ( 2, 1, 1, 2, SYNTAX));
```
A1865     if (SYNTAX <= section_id) & (section_id <= USAGE)
Changed by B to:
B1863     if (SYNTAX <= section_id) & (section_id <= SYNTAX_AS_AN_ACTIVE_REQUEST)

A1908     rtn (19) /* USAGE */ :
A1909     call CHECK_USAGE;
A1910     goto exit;
Deleted by B, preceding:
B1906
B1907     rtn (0) /* UNKNOWN_TITLE */ :

A1925     rtn (20) /* UNTITLED */ :
Changed by B to:
B1920     rtn (19) /* UNTITLED */ :

A1932     CHECK_CTL_ARGS_FORMAT: CHECK_LIST_OF: CHECK_USAGE:
Changed by B to:
B1927     CHECK_CTL_ARGS_FORMAT: CHECK_LIST_OF:

A2202     if (SYNTAX <= section_id) & (section_id <= SYNTAX_AS_AN_ACTIVE_REQUEST)
A2203     then do;
A2204     call ERR_MSG3 (local.need_usage, 0, need_usage);
A2205     section_id = USAGE;
Changed by B to:
B2197     if (SYNTAX_AS_A_COMMAND <= section_id) & (section_id <= SYNTAX_AS_AN_ACTIVE_REQUEST)
B2198     then do;
B2199     call ERR_MSG3 (local.need_syntax, 0, need_syntax);
B2200     section_id = SYNTAX;

This change is described by its mbuild build script:

Description:
1) Change validate_info_seg to correct ticket #199, so it properly identifies subsystem request info segments.

Installation_directory: >udd>m>gd>w>vis03;

Build_script: vis03.mb;
Bound_obj: bound_lib_utility_ IN: tools UPDATE;
source: validate_info_seg.pl1 REPLACE compiler: pl1 -ot;
Testing

The change was tested against several subsystem request info segments. All were now treated as REQUEST info, and section titles were properly validated. For example:

>user_dir_dir>Multics>G Dixon>work>mb>mb.info

----------COMMAND info
LINE: 3   Syntax as a command
LINE: 6   Function
LINE: 22  Control arguments

----------GENERAL info
LINE: 65   Summary of requests
Severity 2, line 65. Paragraph size (20 lines) exceeds limit of 15.
Severity 3. These lines of section exceed 71 chars: 67, 68, 70, 73, 74, 76, 78, 80, 82, 83, 84, 87, 91, 93, 95, 100

----------REQUEST info
LINE: 103  :Info: scan: sc: 2020-01-25  scan, sc
LINE: 105   Syntax
LINE: 109   Function
LINE: 113  Control arguments

----------REQUEST info
LINE: 125   Syntax
LINE: 129   Function
LINE: 135  Control arguments

Documentation

The validate_info_seg command interface is not changing, so no documentation updates are needed.

Version History

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision</th>
<th>Author</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>2020-02-04</td>
<td>1.0</td>
<td>Gary Dixon</td>
<td>Initial version of MCR.</td>
</tr>
</tbody>
</table>