Subject: MCR10105, Add Final Support to history_comment for .info Segments

Author: Gary Dixon

Date: November 27, 2021

Ticket: https://multics-trac.swenson.org/ticket/258
https://multics-trac.swenson.org/ticket/260

Info segment structure requirements divide an info segment into blocks, each containing a particular type of information. The designated block for the history comment begins with an :hcom: block divider; and that :hcom: block must be the final block in the info segment. Furthermore, if an info segment contains just one block, it may be missing elements of the multi-block structure required to add an :hcom: block to the end of the segment.

Issues

The MR12.7 version of the history_comment (hcom) cannot properly add the first history comment to an info segment. It always places the new comment at the very top of the info segment; and does not include an :hcom: block divider so the history comment is visible to the help_ subsystem.

This was a known flaw when hcom was changed to provide history comments in an .info segment. To overcome this flaw, special instructions were written telling the user how to manually edit the .info segment to add block structure, and to move the mis-positioned history comment to a new :hcom: block at the end of the info segment. However, such manual editing has proven cumbersome and error prone.

hcom already provides special treatment for the different segment types it now supports. A table maps segment name suffix onto handling protocols. Info segments are handled like PL/I source, so the first history comment is added at the beginning of the .info segment.

Multics Ticket #258 suggests further enhancing the hcom command by adding a new protocol especially for handling .info segments.

- hcom would call a new info_seg_ entry point to locate the :hcom: block in an existing info segment. This would help hcom locate a history comment at end of the segment without having to search the entire segment.
- hcom would call another new info_seg_ entry point if it needed to add an :hcom: block to an info segment not already containing history comments. This info_seg_ entry point would carry out the steps needed to add a new :hcom: block at the end of the segment, then return the location of that block. hcom would then add the new history comment at that point in the info segment.

This proposed enhancement fits the existing organization of the hcom code. It already provides a seg.type for handling the &version requirement in exec_coms. It can handle calls to new info_seg_ subroutines in a similar fashion.

This enhancement strategy would also keep knowledge of info segment structure within the info_seg_ subsystem, and would require only minimal adjustments to existing hcom code.

Multics Ticket #260 reports a different hcom issue: an unusual bug that occurs when hcom displays a comment, or adds a new comment to many kinds of segments, or updates contents of an existing history comment. The segment's page length (records used) can grow while its bit count remains constant, even for hcom operations that intend no changes to the segment. For example:

```
lf -lb o bound_full_cp_.bind
r 12:18 0.402 0

status bound_full_cp_.bind -length

bit count:          72324
records used:       2
max length:         261120

r 12:18 0.256 0

>tools>hcom display bound_full_cp_.bind

>user_dir_dir>Multics>GDixon>work>hcom>bound_full_cp_.bind:
  1) change(1986-01-30,KFleming),
  …
r 12:19 0.731 82

status bound_full_cp_.bind -length

bit count:          72324
records used:       3
max length:         261120

r 12:19 0.148 0
```

The extension of some segments (incrementing records used by 1 or more without changing that segment's bit count) occurs because an incorrect string length argument is passed to lex_string_$lex by the hcom_parse_.rd subroutine when parsing an existing history comment. The incorrect length tell lex_string_ to look beyond the last character within the segment's bit count range when parsing tokens in the history comment. When parsing touches any location in the physical pages beyond the page containing the last real bit of the segment, the segment's records used count grows as those extra pages are examined.

This problem is detected by programs like **archive** which looks for discrepancies between valid characters within the bit_count range of the segment versus that segment's records used. Such programs report any inconsistency between the segment's records used versus its bit count. That's how this issue was originally detected.

The incorrect length arguments passed by hcom_parse_.rd will be corrected by changes proposed in this MCR.

Proposed Changes

The following changes implement the suggestion to make hcom fully support adding history comments added to an info segment (Ticket #258). These changes include the following:

- Add two new entry points to the info_seg_ subsystem to support locating history comments in an info segment.
  - info_seg_$find_history_comment: locates the :hcom: or :Internal: info segment block containing a history comment and returns that location to the caller.
  - info_seg_$add_history_comment_block: checks for existence of a history comment block within an info segment. If found, returns its location to the caller.

    If not found, it does the following:

    1. Check whether the info segment begins with an :Info: block divider. If not, one is added specifying all names currently on the info segment (without their .info suffix).
    2. Add an :hcom: block divider at the end of the info segment, followed by an empty line.
    3. Return the index of that empty line to the caller (the hcom add operation).

  The new entry points will be undocumented and used only by hcom_process_seg_. A description of their use is provided as code comments in info_seg_.pl1. Since info_seg_ and hcom_process_seg_ reside in different bound segments, the two new entry points must be retained in info_seg_'s bound segment; bound_help_.bind will therefore be modified.
- Change pnotice_language_info_.incl.pl1 and pnotice_language_info_.cds to add a new Type=6 segment designator for .info segments. Both the add_pnotice and history_comment commands share this pnotice_language_info_ data array.
- Change the add_pnotice.pl1 segment to recognize Type=6 segments, and report an error when attempting to add a protection notice (copyright or trade secret notice) to an .info segment. Neither adding a protection notice to, nor displaying a protection notice in an info segment is supported.
- Change hcom_process_seg_.pl1 to recognize Type=6 segments as follows:
  - The get_language_info internal procedure of hcom_process_seg_ will call info_seg_$find_history_comment and: record the location of any existing comment; or remember that no comment currently exists. The existing seg.text_pos variable (used for locating history comments within exec_com files) will record this location for an info segment; a negative value indicates that no history comment currently exists.
  - The OPER(1): /* ADD operation */ label of hcom_process_seg_ will call info_seg_$add_history_comment_block for an info segment which does not contain any history comments and remember the location of the empty :hcom: history comment block returned by that subroutine in the seg.text_pos variable.
  - Existing code at various points in hcom_process_seg_ knows how to use the seg.text_pos value to operate on an existing history comment, or add a new comment

box. This code will be extended to handle Type=6 info segments which will share code used for Type=4 (PL/I source comment begin/end delimiters, etc.).

- The info_seg_$add_history_comment_block entry will use the existing pnotice_mlr_.alm subroutine to shift contents of the info segment when inserting a new :Info: block divider at the beginning of segment; and to append an :hcom: history comment block to the end of the segment. This tricky ALM routine uses a single MLR or MRL instruction to move text. An elegant opening comment in this code diagrams which entry point to use when the source and target string locations of the move overlap. But the diagram layout is clouded by use of overstrike character sequences. These will be removed to make the diagram understandable on today's video terminals.
- Special instructions that describe manual insertion of the first history comment into an info segment are stored in info_seg.hcom.info. These instructions are no longer needed and will be removed. References to info_seg.hcom.info that appear in hcom.add.info (and elsewhere in hcom.info) will be removed as well.

Code comparisons describing the above changes are not included in this MCR. The new entry points involve too many lines of code to show here. The changes made within hcom_process_seg_.pl1 are too intricate to be presented properly in compare_ascii output.

The following change corrects the bug reported in Ticket #260.

- Change hcom_parse_.rd to pass correct segment boundary length information to lex_string_$lex. The proposed code change is:

```
cpa [lpn hcom_parse_.rd] ==

Inserted in B:
B32         7) change(2021-11-11,GDixon):
B33            Fix call to lex_string_$lex to pass correct length of comment to be
parsed.
B34            This prevents lex_string_ from referencing beyond end of input segment,
B35            thereby extending records used in that segment without changing its bit
count.
Preceding:
A32                                                        END HISTORY COMMENTS */


A318      call lex_string_$lex (addr(seg), length(cmt)+Lignore, Lignore, proc_ptr,
"0000"b,"","","","","",BREAKS,
Changed by B to:
B322      call lex_string_$lex (addr(seg), length(cmt), Lignore, proc_ptr,
"0000"b,"","","","","",BREAKS,
```

The following build script file summarizes these changes and the source programs affected. Note that this script assumes info-seg-related source programs have been moved into bound_help_ (per MCR10103).

```
Description:
 1) Change hcom to call info_seg_ subsystem for assistance in: adding the first
    history comment to an info segment; and locating an existing pnotice within
    an info segment.  See: Ticket https://multics-trac.swenson.org/ticket/258
```

```
 2) Change pnotice_language_info_.cds to support a new Type=6 for info segments.
    This supports code in hcom_process_seg_ to call info_seg_ for assistance in
    locating/positioning a history comment within an info segment (change 1 above).
 3) Change pnotice_language_info_.incl.pl1 to document new Type 6 in comments
    describing this include segment.
 4) Change hcom to parse existing history comments in a segment without extending
    records used (number of pages in) that segment.
    See Ticket https://multics-trac.swenson.org/ticket/260
 5) Change pnotice_mlr_.alm to remove overstrike sequences making diagram comment
    at start of file unreadable.
 6) Change add_pnotice to disallow protection notices in .info segments.
 7) Remove the info_seg.hcom block from info_seg.gi.info.  This block described
    a process for manually adding first history comment to an info segment.  The
    changes above implement this process automatically when using: hcom add SEG.info
 8) Remove references to info_seg.hcom.info from hcom.add.info block (and elsewhere
    in hcom.info).

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

 Build_script:             hcom.mb;

 Bound_obj:                bound_help_                IN: sss   UPDATE;
     bindfile:             bound_help_.bind            REPLACE;
     source:               info_seg_.pl1               REPLACE  compiler: pl1 -ot;

 Bound_obj:                bound_pnotice_             IN: tools UPDATE;
     source:               add_pnotice.pl1             REPLACE  compiler: pl1 -ot;
     source:               hcom_parse_.rd              REPLACE  compiler:
                                                                    rdc -ot -trace off;
     source:               hcom_process_seg_.pl1       REPLACE  compiler: pl1 -ot;
     source:               pnotice_language_info_.cds  REPLACE  compiler: cds;
     source:               pnotice_mlr_.alm            REPLACE  compiler: alm;

 Include:                  pnotice_language_info_.incl.pl1
                                                      IN: sss.include  REPLACE;

 Info:                     history_comment.info       IN: priv.info    REPLACE;
                             add_name:
                               hcom.info
                               hcom.add.info
                               hcom.add_field.info
                               hcom.af.info
                               hcom.check.info
                               hcom.ck.info
                               hcom.compare.info
                               hcom.cmp.info
                               hcom.display.info
                               hcom.ds.info
                               hcom.exists.info
                               hcom.format.info
                               hcom.fmt.info
                               hcom.get.info
                               hcom.install.info
                               hcom.replace_field.info
                               hcom.rpf.info
                               hcom_validation_rtn_.info;
 Info:                     info_seg.gi.info           IN: sss.info    REPLACE;
                             add_name:  info_seg.info;
```

Testing

This proposal includes two separate changes. Test for each change was done separately.

- Ticket #258 allows hcom to add the first history comment to an .info segment; and provides a more efficient way for hcom to locate existing history comments in an .info segment (all of which must appear at the end of the info segment).
- Ticket #260 corrects an hcom bug affecting display of history comments found in any segment type.

Testing for Ticket #258

The proposed change introduces a new pnotice_language_info_ seg.type designator for .info segments. This designator is used by hcom_process_seg_.pl1 to separate handling of info segments from that performed for other segment types.

Prior to this change, .info segments were treated as seg.type = 4 (PL/I-style comment delimiters with history comment near top of segment). After the change, new hcom_process_seg_ code handles .info segments as seg.type = 6 in which history comments appear in a dedicated info block at the end of the .info segment. But most handling snippets do not require involvement of the new info_seg_ subroutines, and therefore use the existing seg.type = 4 code snippet as in the past.

Therefore, testing was focused on three areas:

A. Thorough testing of the new code in info_seg_$find_history_comment.

B. Thorough testing of the new code in info_seg_$add_history_comment_block.

C. Focused testing of code in hcom_process_seg_ which: locates the comment box containing the history comment in an info segment; or adds the first history comment in a new comment box at the end of the info segment.

AREA A: Testing of info_seg_$find_history_comment focused on the following characteristics:

- Did the subroutine find the correct location of existing history comments in .info segments already having a history comment which: appeared in an :hcom: block; or in an obsolete :Internal: info block?
- Did it correctly report no history comment present in .info segments not having one of these two block types?

RESULTS:

Most errors detectable by info_seg_$find_history_comment have already been detected before hcom calls info_seg_. This was code tested to ensure that it properly finds any existing :hcom: or :Internal: block in the input info segment, and properly reports its location back to hcom_process_seg_.

AREA B: Testing of info_seg_$add_history_comment_block focused on the following characteristics.

- Did the subroutine correctly add an :hcom: block to the end of any valid info segment block structure? Info structures tested include: segments having no :Info: block dividers; segments with :Entry: subroutine block dividers; segments with :Info: block dividers.
- Did it properly handle info segments having unnecessary whitespace characters at the beginning or end of the info segment? Such whitespace is often present and is ignored by info_seg_parse_; but must be removed when adding an :Info: block to the beginning of an info segment; and/or an :hcom: block to the end of the segment.

RESULTS:

This code was validated using a variety of existing or fabricated info segments which start out with no history comments. Testing included command/AF info segs, subroutine description info segs, general information info segs, etc. Variants included single block info segments (both with and without a leading :Info: block divider); multi-block info segments (both :Info: with :[Info]: dividers, and :Info: with :Entry: dividers). Test files all started out as accepted by verify_info guidelines. Some candidates included leading and trailing whitespace at start and end of the info segment. Extreme test cases were then tested, such as (almost) empty files that would not be accepted by verify_info.

In all valid test cases in which the leading :Info: block was missing, any leading whitespace was removed, and a properly-named :Info: divider was added. Any trailing whitespace was removed, and a proper :hcom: history block was appended to the info segment. And the extended info seg length and correct location of the new :hcom: block was reported back to the hcom command. The extreme test cases were all properly diagnosed by error codes returned by the new code.

AREA C: Testing of hcom_process_seg_ changes focused on the following characteristics:

- Are all errors returned by info_seg_$add_history_comment properly reported by hcom?
- Does the subroutine react as expected if info_seg_$find_history_comment reports no history comments found?
- If the info segment is changed by hcom, does it still pass verify_info guidelines?

RESULTS:

The changed hcom code was tested using the Area B test info segments. hcom add, add_field, display, and check operations were tested to ensure they worked properly against info segments. [Other hcom operations share those same code paths.]

All changed info segments were tested with verify_info to ensure that any changes were implemented according to design guidelines.

The add_pnotice and display_pnotice commands were tested against a few info segments to ensure they accepted info segments as a known segment type, but refused to add or display a protection notice in any info segment.

Testing for Ticket #260

The proposed change to hcom_parse_.rd involves only correct specification of the boundaries within the segment containing a history comment: number of characters preceding the history comment entry to be parsed; and number of characters in that history comment entry. hcom_parse_.rd correctly calculates these lengths, but mistakenly combines them when passing in the argument specifying number of characters in the history comment entry.

Testing focused on examining the actual tokens created by lex_string_$lex to ensure only the expected comment string was parsed. Debug code added to display these tokens will remain (as a comment in the program) for possible re-use to explore any future problems in this code.

RESULTS:

History comments were displayed in several different types of files (PL/I source, info segments, bind files, etc) to ensure the fix is working as designed. Segment length were checked before and after displaying history comments to ensure segment's record used did not grow.

I also reviewed code in all other callers of lex_string_$lex to ensure they were passing correct length values.

Finally, I checked records used against bit count value for all info segments in >doc>info and >doc>priv to see if any files experience this unintentional lengthening had been installed. Only one was found:

```
ec needs_abc ([segs -absp >doc>(sss priv)>**.info])
--- >doc>priv>mbuild.info ---------------
records_needed:  13
records_used:    25
```

Non-info files that include a history comment are usually archived. The archive command diagnoses any discrepancy between bit_count and records_used for segments being archived.

The needs_abc.ec test exec_com was run against non-archived files that might experience this problem: in >ldd>include>** and >ldd>(h sss tools unb)>s>**. No problems were found in any of these directories.

Documentation

This MCR changes the hcom.add.info block to no longer refer to the info_seg.hcom.info block, which describes the manual procedure for adding the first history comment to an info segment. That manual procedure is not longer needed.

```
cpa >doc>info>hcom.info ==

A230      :Info: hcom.add: 1988-03-10  hcom add operation
Changed by B to:
B230      :Info: hcom.add: 2021-11-13  hcom add operation


A240      To add the first history comment to an info seg,
A241      type:  help info_seg.hcom
A242
Deleted by B, preceding:
B240
B241      Arguments:


A1197                                                     END HISTORY COMMENTS */
Changed by B to:
B1194       5) change(2021-11-13,GDixon):
B1195          Remove references to info_seg.history_comment.info.  That info segment
B1196          is no longer needed when adding first history comment to an info seg.
B1197                                                     END HISTORY COMMENTS */

Comparison finished: 3 differences, 10 lines.
```

The info_seg.gi.info segment is changed to remove references to the info_seg.hcom.info block, as well as removing the block itself.

```
cpa >doc>info>info_seg.gi.info ==

A328      Adding a history comment to an info seg: The "Multiple-entry info segs"
A329      format described above provides a method for adding a history comment
A330      to the end of an info seg.  Type: "help info_seg.hcom" for instructions
A331      on this feature.
Changed by B to:
B328      Adding a history comment to an info seg: A history comment may be
B329      added to most info segments using the command:
B330         hcom add INFO_SEG_NAME.info


A357      :Info: info_seg.hcom.gi: info_seg.history_comment: info_seg.hcom:
A358      2021-06-09  Adding a History Comment to an Info Seg
…
A545
A546      :hcom:
A547
Changed by B to:
B356      :hcom:


A558                                                      END HISTORY COMMENTS */
Changed by B to:
B367       4) change(2021-11-13,GDixon):
B368          A) Remove references to info_seg.hcom.info now that the history_comment
B369             command can add the first history comment to an :hcom: block in the
```

```
B370             info segment.
B371          B) Delete the block info_seg.hcom from this info segment.
B372                                                      END HISTORY COMMENTS */
```

Version History

Date	Revision	Author	Comment
2021-11-28	1.0	Gary Dixon	Initial version of this MCR.