APPX Software, Inc.

SPEED II to APPX 3.0.0 Conversion Notes

(using Customized Conversion Utility 2.00 Version 05 by APPX Software, Inc.)

October 21, 1996

SECTION 1: APPX Miscellaneous Bugs

1.1 Bug: Per APPX Technical Alert dated 08/04/95, the Toolbox utility *Renumber* is not working correctly in APPX 2.x. The problem occurs when renumbering any type of process and can lead to the corruption of ILF code in the process. (The renumbering of Files or Fields is not affected.)

1.2 Bug (**fixed** in APPX 3.0 Beta): In Application Parameters F/M in Application Design, APPX 2.3 gives false WARNING messages that menu names are not found.

1.3 Bug: In all releases of APPX, the *Cross Reference Utility* often does not find all references and is sometimes inconsistent in what it does find. This is particularly true of fields; however, the ILF X-Ref appears to work correctly. (See items 2.6 & 2.9.)

1.4 Bug: In APPX 3.0.0, there appear to be several bugs related to CANCEL OK with some dating back to at least APPX 1.8.4 (see items 7.11 & 7.17; see test function CPA SE Job "!!! TEST CANCEL"):

1) APPX does not enforce CANCEL OK ="N" (whether set explicitly by commands or implicitly via 'User Cancel OK?' flags. The operator can still cancel.
2) If an operator cancels an (update) process, the cancel does not take effect until End of Process but before the *End of Process* code is executed. This is true regardless of the value in CANCEL OK or 'User Cancel OK'.
3) If an output is printed to the screen, it can be cancelled at that point even though CANCEL OK and 'User Cancel OK' are both "N".
4) A Status process (even with 'User Cancel OK?'="N") sets CANCEL OK to "Y" after executing Status *Start of Process* but before executing *Post Invocation* for the Status job step. It remains at "Y" for *Pre-Invocation* of the next job step even though a previous job step explicitly set it to "N". This is true whether or not the Status presents a screen to the user.
5) Difference or Bug? If the manual is correct (p. 575), setting CANCEL OK ="N" in APPX is not equivalent to SP2 use of built-in function 'CANCEL NOT OK' on a job step. In SP2 this would disable PF32 CANCEL unless and until the built-in function 'CANCEL OK' reenabled it. In APPX, according to the manual, CANCEL OK is set to the 'User Cancel OK?' flag for each process.
6) 'Cancel Disposition' on automatic and optional children shows the value "DISABLED" on the Scan screen (although referred to as "DISABLE" on p. 360 of the manual and as "*prohibit*" on p. 575). However, if you enter "DISABLED" as the Cancel Disposition, it is accepted but if you check it again it has been changed back to "CANCEL PARENT". (Note: The HELP screen on 'Cancel Disposition' in the 0AD File 'CHILD' is not current but does show Prohibit as option 0. Is it legitimate to have value 0 for a token?)
7) Difference: For a SP2 Update, the default value for 'Is Function Cancelable?' is "N". For an APPX Update, the default value for 'User Cancel OK?' is "Y". (In APPX, there is a 'User Cancel OK?' flag for every type of process, not just updates.)

1.5 Bug: APPX 1.8.4 does not enforce opening files non-shared (either implicitly by flags or explicitly via OPEN).

1.6 Bug: APPX 3.0 Beta does not close all files in Design File Mgt following certain operations such as *Create Files* and *Initialize Files*. To avoid subsequent "File Not Closed" error messages, after each such operation you should exit to APPX menu and then reenter if needed.

1.7 Bug (**reported as fixed** in APPX 3.0.0): When creating design files in both APPX 2.3 and APPX 3.0 Beta, file PARAM may get the message "Requires Processing". Currently, the only work-around for this message is to reload application 0AD from the APPX Utility system tape.

1.8 Bug: In APPX 3.0.0 Data Dictionary the default value for a field may be improperly displayed. For example, the default value "??????" was displayed for a date field after previously stepping thru Additional Attributes for a numeric field which had default value 0. When Additional Attributes was reaccessed for the date field, the default value was properly displayed as "YYMMDD". (See item 2.12.)

1.9 Bug: In APPX 3.0 Beta, when using the *Change All References* utility, no input was permitted upon returning to the *Change All References (Processes)* screen after changing references for a job even though the fields appeared open for input. Entry was permitted after exiting and reentering.

1.10 Bug: In APPX 2.3 Application Design it is possible for a process to be missing after changing its name. When the process name is redefined at the process level, APPX can then find the remainder of the process. (See item 1.23 for orphan utility and reference to similar type bugs.)

1.11 Bug?/Note/Caution: In APPX 3.0.0, if you make a change to a domain in the data dictionary, processing the domains can take an inordinate amount of time (depending upon the number of applications with the same version). This is because APPX now addresses the domains in each application with that version in order to handle SAME AS domains. **CAUTION:** If you plan to change a domain, it is best to avoid having any other window or user in the Data Dictionary for any application with that version. For example, if one window is changing domains in one application and another window is looking at the domains in INQUIRE mode in another application with the same version and then exits, that window can trigger processing the domains changed in the first application. While this may not cause any harm, it probably is not desirable.

1.12 Bug: In APPX 3.0.0 it is possible to get an APPX system error MM.C.3873 ("mm free - attempting to free freed memory").

Pinellas got this in a query process which had an EXIT command (see item 8.26) while APPX Software, Inc.got this once under Beta just trying to return to the menu after printing something to the screen.

1.13 Bug: In APPX 3.0.0, it is possible to get an APPX system error DD1.C.308 ("Read bitmap fi - Error - Record already held by another user") if another user or another window is in Application Design on the same application.

APPX Software, Inc.got this while processing the data dictionary when another window was sitting on the Application Design menu. Pinellas got this while printing technical documentation.

1.14 Bug: APPX 1.8.4 produces an action log when a user attempts to run a program to which he does not have security rights. However, this action log can cause a printer error when printing the key value for certain field types. APPX is apparently not converting packed decimals properly for printing.

1.15 Note: The conversion utility (version 04) converts a SP2 RUN command running an external function as COPY in APPX. (Example: CPA SE Status Function 'COPY W2REPORT FILE TO W2S TAPE').

1.16 Bug (**fixed** in engines compiled later than 01/15/96): In APPX 3.0.0 an error message "Error - An internal error has been detected; proceed with caution" can appear on line 23 of the APPX screen.

Note: This was found to be caused by scanning in a multi-level function and has been fixed.

1.17 Bug: APPX 1.8.4 (thru 3.0.0) allows an application with data files but no structure files to run without any error or warning message. If a file structure is changed, there is no notification that a restructure is required.

Note that structure files are governed by the FMS group at the database level (in System Administration) so they may not be where the data is.

1.18 Conversion Bug: The conversion utility does not validate the application ID when the application to convert is specified on the VS. If the application doesn't exist, the conversion utility will still proceed without further ado. It will reprocess the last application converted and misidentify it as the specified application. The conversion messages will be different since it thinks it is processing the specified application.

1.19 Bug: APPX 3.0.0 Tech Doc (and probably X-Ref too) consume increasing amounts of memory, not freeing memory until APPX is ended. In fact, APPX can consume all available memory if a large number of items are selected. For example, running tech doc for all output processes resulted in an incomplete printout to the screen and caused other terminals to hang or get system errors for lack of memory. Problems can be avoided by limiting selections to a screen full at a time and exiting & reentering APPX between each run. Note that it can take a couple of minutes to return to the menu after viewing tech doc on the screen (the more pages, the longer it takes).

1.20 Bug: In APPX 3.0.0 when you pan on output printed to the screen, underline characters remain in place and do not pan with the text.

1.21 Bug: In APPX 3.0.0 a month-only date field can be set to "----00----------" but this should be invalid.

1.22 Difference: Whereas both SP2 and APPX use 'D" to signify a negative sign in a packed decimal field, SP2 uses 'F' to signify a positive sign and APPX uses 'C' (unless the special FD version of APPX is used). Also, in both SP2 and APPX, a date field is stored as packed decimal and a blank date is stored as -1. For instance, a blank month-only date field would be stored as HEX(001D) in both systems. However, month '01' is stored as HEX(001F) in SP2 and as HEX(001C) in APPX. This means if a blank month-only date field is used in a BEG AT command, month 01 will be skipped in APPX. Conversely, if month 01 is used, APPX will include blank months. (See item 8.30.)

1.23 Bug & Conversion Bug (**fixed in V05 after 08/27/96**): Orphan statements can exist in the APPX 0AD file 'STMT'. Some orphans may be code that is no longer needed (such as commands generated by the conversion utility for SP2 'CLEAR SORT' and 'RETURN MENU' and other job steps with no counterpart in APPX that had a function code). Other orphans may be code that was not properly deleted by APPX during Application Design (known to have occurred in APPX 1.8.4 thru at least 2.3.0). However, it is also possible that some code got orphaned in APPX by mistake. To check for STMT orphans, run the APPX Software, Inc.UTL CS utility "Check APPX for STMT Orphans" immediately after conversion and then again after addressing the conversion logs.

There are several items that pertain to something missing or lost in APPX (see items 1.10, 1.25, 1.26, & 7.2 for details).

1.24 Bug: When printing APPX 3.0.0 Tech Doc (to the screen and to hard copy), it is possible for background text not to print. For example, the descriptors for main/division/branch length did not print when printing PARAMETERS LIST in CGL SE. (Note: The missing descriptors are there when editing the output process and when printing the actual report.)

1.25 Bug: In APPX 3.0.0, User Documentation may get deleted in error.

Example: User documentation was missing from CPA SE Output 'PAYROLL CHECK FORMS' at APPX Software, Inc.even though it was present in the Portdata DOCUMENT file and it was not knowingly deleted. This is the only output process missing the 'T'ools designation (which may have happened by stepping thru User Doc), but it is not known at what point the documentation was lost. It is possible that APPX did not display the documentation (perhaps due to the sequence in which it was accessed), when it really was there and then stepping thru the blank screen caused it to be deleted (see item 2.12). It was actually missing from the DOCUMENT file accessed thru FILES EDIT. It is also possible that other processes with the 'T' are actually missing the documentation. (See item 1.23 for orphan utility and reference to similar type bugs.)

1.26 Bug: In APPX 3.0.0, when using the Toolbox utility *Check Images for Item Overlap*, it is possible to have the screen hang on a particular frame even though processing appears to continue to completion. This may be caused by an orphan frame or image in APPX. You can check for orphans by running the APPX Software, Inc.UTL CS utilities "APPX IMAGE File List" and "APPX ITEM File List" and entering "Y" for 'Check for Orphan?'. (See item 1.23 also.)

For example, this occurred at Pinellas on the same output frame for an obsolete process on two different systems. The process level for the frame was found to be missing on both systems. When they added the process name on the development system, everything subordinate to the process level came back. When they added the process on the live system, nothing came back and everything else related to the process was also gone (job, file, etc.). This suggests that the process had actually been deleted on both systems but execution of the deletion was not properly carried out. The orphan screen did not have any commands in SP2 so it did not appear on the STMT orphan utility.

1.27 Bug: In APPX 3.0.0 a domain with Type FORMAT prints as Type ALPHANUMERIC on the detail version of Domain Tech Doc. (It prints properly as Type FORMAT on the summary version.)

1.28 Bug (**fixed under APPX 3.0.0?**): APPX 2.2.2 (on an HP at Citrus World) can encounter problems while processing domains which cause the system to exit to the APPX menu. It appears to be related to improper APPX maintenance of information in the APPX 0AD file DOMAIN when fields are changed from domains to non-domains or vice versa. Also, when a file is deleted from the Data Dictionary, APPX 2.2.2 may not delete the hidden domain records for the non-domain fields in the file. In that case, one possible solution is to export the DOMAIN file, use *vi* to delete the records with ('Generated?'=)"Y" in the first byte from the DOMAIN file in Portdata, and reimport the DOMAIN file.

1.29 Conversion Bug (**fixed in V05**): The conversion utility sets cache priority to 1 for a process. It should be set to 50 which is the default (contrary to what the manual says).

1.30 Bug: In APPX 3.0.0, when you add a domain with type SAME AS, the pop-up box appears with the error message " /vv Not Found in System Administration" before you have a chance to enter the name of the SAME AS field.

1.31 Bug: In APPX 3.0.0, using an index variable as the appearance number on an AT FIELD command causes a **compile error**.

1.32 Bug (**job used as job step in other job flagged in V05**): In APPX 3.0.0, if a job calls another job and the 2nd job contains a QUERY and an OUTPUT using the query and the OUTPUT has 'Separate Task?'="Y", then operator specifications made during query setup (such as record selections) are lost. One fix is to have the 1st job call the 2nd job as SUBPROCESS (but be sure the query has Override Default Mode "ADD" to avoid another bug). Another fix is to set 'Separate Task?'="N". (Example: CPA SE job END OF YEAR REPORT)

1.33 Bug: In APPX 3.0.0, if query execution runs in background and reads a temporary job file in Start of Query Execution, the READ statement may improperly return a FALSE condition under certain circumstances. In one instance it can occur when using a split job. The 1st job contains an INPUT, QUERY, disposition JOB, and JOB calling the 2nd job. The 2nd job contains the OUTPUT using the query. Separate Task is set to "Y" only on the JOB step calling the 2nd job. There is no problem if the job is run in foreground.

1.34 **Action**/Bug: In APPX 3.0, the 'Save Em?' flag on a process is not visible or accessible but may be set to "N" if the SP2 flag 'Save Load Module?' was set to "N". The *Create Executable Modules* utility does not address a process with 'Save Em?'="N" (they are not presented for selection). To find these processes, run the APPX Software, Inc.UTL CS utility "APPX PROCESS File List" and enter "N" in the job specification for 'Save Em?' value. If any found, use Toolbox Option 97 to perform a syntax check for these processes (the Em created will not be saved).

1.35 Conversion Bug: Jobs created for SP2 Query functions should not rename the SORT and OUTPUT job steps (since these already exist under the SP2 Query name) even though the job itself must be renamed. Also, any SP2 Query job steps should be flagged.

1.36 Conversion EH (**implemented in V05 after 08/27/96**): The conversion utility should flag the use of "SPEED" and "EXPLAIN" in text (similar to "PF", etc.)

1.37 Conversion EH: The conversion utility should flag code present in non-executed SP2 command series and possibly delete it as well. It should also flag anytime SP2 command series are combined even when no code is present in one series.

1.38 Conversion EH: The conversion utility should flag the presence of footer windows in SP2 input functions.

1.39 Conversion EH: The conversion utility should provide the occurrence number when it names a field on the conversion log.

1.40 Conversion EH: The conversion utility should flag the presence of page footers in SP2 output functions.

1.41 Bug/Conversion EH/Difference: Programming bugs or questionable programming practices converted from SP2 can cause system errors or other misbehavior in APPX. If possible, the conversion utility should flag the presence of such conditions.

For example, the SP2 command "IF MODE EQ 12" converted the improper value '12' to some unknown value. Upon initial display of the command in APPX, it was presented as questions marks, then caused an APPX system error when editing was ended. Upon reentering the ILF editor, the code had been changed to a valid value (but not necessarily the value intended). (Note: At runtime, SP2 simply returned a False when executing this command; APPX would probably get a system error if this code was executed in its garbage state.)

1.42 Bug: APPX 3.0 skips the image of a NORMAL frame if ALTERNATE IMAGE NUMBER is explicitly set to 0 in *Select Image*. (See items 1.43 & 8.23.)

1.43 Conversion EH: In order to emulate SP2 behavior where the first standard window for a new level cannot be skipped, frame type should be set to NORMAL (not OPTIONAL) even if code exists in *Set Alternate Window*. (However, see items 1.42 & 8.23.)

1.44 Bug: In APPX 3.0, a user with a security group which was defined for one company only was not denied access under a second company. If the security group was defined for both companies, it denied access but did not say so; instead, the screen would show "in progress" and then just return to the menu.

1.45 Bug: Security group file maintenance does not work properly in APPX 3.0 System Administration. Unless you exit after editing each group, your changes may not be preserved.

1.46 Bug: In APPX 3.0 for Windows, using PAN PAN to pan to the right on an output image in Application Design can cause an "unhandled exception detected" error. Using PAN PAN PAN to the right works, however.

1.47 Bug: Scanning on an alternate key field which is a LOGIC field does not work properly in APPX 3.0. If you scan after entering "Y" or "N", you get a blank screen. If you enter "0" or "1", you get an "invalid value" error. If you scan with the logic field left blank, you get everything (starting with the "N"'s).

1.48 Bug: In APPX 3.0 for Windows, the cursor is not controlled properly when adding records using a scrolling lower level input process. If you enter data which fills the last field on the record, the cursor returns to the top of the screen, not to the first field on the record. If you happen to press ENTER to add the record when the cursor is not positioned on a field in the record, then the new record will appear in two places: at the top of the display and where it was actually added. This is a screen display problem; the data is actually correct.

1.49 Bug?/Difference/Conversion Bug?: In the APPX Data Dictionary, the flag 'Blank OK?' can be set for a logic field to indicate whether or not a user is permitted to leave the field blank (according to pages 121, 161, & 199 of the manual). The conversion utility currently sets this field to "N" (which is the default according to the manual) since there is no counterpart to this flag in the SP2 Data Dictionary. This might cause a problem if the manual was right.

In APPX 3.0, the default for this flag is actually "Y" and the flag itself seems to have no effect, not even to serve as a default for the flag 'Required?' on an image item. The flag 'Required?' is what really controls whether a user can leave the field blank in APPX and is set by the conversion utility from the corresponding SP2 flag 'Blank OK?' for the image item. As long as APPX continues to behave the way it does, there is no problem but if APPX is changed to resolve the contradiction with the manual behavior could be different from SP2. In the meantime, the conversion utility should probably set the APPX DD flag 'Blank OK' to "Y" in order to agree with the default.

1.50 Bug: The APPX Toolbox utility "Generate 'Standard' Output Process" has some problems. It creates a job with Override Default Mode set to CHANGE on the QUERY job step whereas it should be ADD. Also, it creates a QUERY that does not allow use of OPTION 2 (Value/Blank) at runtime on the record selections it generated. OPTION 2 will work on record selections added to the QUERY after generation.

1.51 Bug: In APPX 3.0 & 3.1, if the full default value for a work field is changed via OPTION 88, the new default value is not used at runtime even though it is displayed on the DD screen in Application Design. If the default is too long to edit without using OPTION 88, the work field must be deleted and added back in order to get the new default value to be recognized. This may also affect other fields, not just work fields.

1.52 Bug: In APPX 3.0 it is possible to get an APPX system error PCITM.C.359 ("bad tdlu read") when trying to create Em's. If you continue to press return, APPX may go on and provide a compile error listing but the error reported may not be correct. (For example, this system error was encountered on several Citrus World outputs with the compile error given as "Invalid Data Lookup specifications - Source Missing" and naming a field on a RANGE-START frame. What was found to be causing the error was a DLU on an unrelated field in the RECORD frame.) For an overview of DLU items, run the APPX Software, Inc.UTL CS utility "APPX ITEM File List" and enter 'Check for DLU?'="Y". To see lookup items too, also enter 'Lookup?'="Y" (note that the checks for DLU or Lookup are the only checks combined with an OR; the rest use AND). To list items for a single process, enter the process name. (See items 9.45 & 10.28.)

1.53 **Action**/Bug: In APPX 3.0, a job may not run properly in background if it uses a temporary file with only 1 byte (for example, it may not produce any output). One solution is to simply define another field in any 1-byte temporary files. The APPX Software, Inc.UTL CS utility "APPX FILE File List" will provide a warning for any temporary files with record size of 1. You can also list such files by entering 'Temporary File?' value "Y" as a job parameter and specifying Rec Size EQ 1 as a record selection.

1.54 **Action**/Bug: In APPX 3.0, for output/inquiry processes only, if a BLANK command is used to conditionally blank a field using an index variable as occurrence number or appearance number, the field is not displayed again until a DISPLAY command is executed. The APPX Software, Inc.UTL CS utility "APPX STMT 'Format 6' List" can be used to list BLANK commands. (Note: There is no problem on a universal BLANK command even though the screen contains items with occurrences.)

1.55 **Action**/Conversion Bug (**fixed 08/27/96 in V05**): The conversion utility may not properly convert SP2 Auto Access paths. For instance, it may convert Data Lookup Type to be DEFINE when it should be CONTINUE (this can happen when there is another field on the screen between components of a multi-part key). Also, it may convert Data Lookup Type to be CONTINUE when it should be DEFINE (this can happen when there are multiple occurrences of a field with Auto Access on the screen).

If the conversion was run before this problem was fixed, this problem requires action. It is not flagged on the conversion log and does not cause a compile error. The DLU will just get a NOF condition at runtime. To review DLU items, run the APPX Software, Inc.UTL CS utility "APPX ITEM File List" and enter 'Check for DLU?'="Y". To see lookup items too, also enter 'Lookup?'="Y" (note that the checks for DLU or Lookup are the only checks combined with an OR; the rest use AND). There are also options to check for repeated DLU and for continued DLU but it may still be worthwhile to print the complete DLU list (given the problems outlined in items 1.52, 9.45, & 10.28).

1.56 Conversion Bug (**flagged in V05**): If a SP2 field is designated as a Record Access Security field in the DD, the conversion utility converts occurrence and 'Lower Case OK?' fields to blanks. Occurrence should be 1 and 'Lower Case OK?' should be "N". (See item 4.8.)

1.57 Bug: In APPX 3.0 X-ref documentation, information that may only apply to the first item on the report is incorrectly printed in the page heading (for instance, process type). Also, the "* Referenced But Not Defined" message prints but applicable items are not actually flagged.

1.58 Difference: There is no way to verify references in APPX 3.0. There is no way to find references to undefined items unless you name the undefined items explicitly. There is also no easy way to find x-app references.

1.59 Bug?: In APPX 3.0, the Selection Expression (end user or designer) in a query is evaluated differently than one might expect. For example, suppose there are three record selections using the same field: 1) field EQ 20; 2) field GE 10; 3) field LE 15. The selection expression is "1 OR 2 AND 3". In this example, the record with field EQ 20 was selected contrary to expectations. Are the AND's being evaluated first? (If this were evaluated based on IF, OR, AND commands, the result would be False.)

1.60 Bug: In APPX 3.0 Application Design (Job Processes), if you step thru Additional Attributes for a job step which has an Override Default Mode specified, this value can be presented (in CHANGE mode) as the Override Default Mode the next time AA is accessed for another job step. If you then press RETURN, data would be changed accordingly.

1.61 Bug?/Warning: In APPX 3.0, if an item is edited on an image using an abbreviated version of the item, the contents of the item will be truncated to what is visible in the abbreviated field without any warning. (Option 88 is available to edit the full version.) In the case of "In Progress" messages, it is not always apparent that the abbreviated field does not contain the entire message.

1.62 **Action**/Conversion Bug: When the conversion utility increases 79-character rows to be 80-character rows (in order to avoid problems in APPX), it should also increase the report width.

1.63 **Action**/Conversion Bug (**alternates with different sizes flagged in V05**): If SP2 alternate windows have different sizes and if a window does not start in column 1, the conversion utility may specify the offset for the column in both the frame and the image which results in doubling the offset. This can cause an error in Application Design when attempting to access the image. It can also cause compile error "Size Invalid for Frame Size". However, since it could be wrong without causing an error condition, check all alternate images flagged on the conversion log for this condition. Note that column 1 can be specified on both the frame and the image without causing a problem. To list images with size specified, run APPX Software, Inc.UTL CS utility "APPX IMAGE File List" and enter "Y" to 'Check for Size on Image?'. (See items 9.23 & 10.26.)

1.64 Bug: When a record selection screen for a query is presented to a user at runtime, all positions open for input on the right side of a selection may not be honored when evaluating the selection. For example, employee name has 30 characters open for input but a match will not be found if the employee name exceeds 22 characters.

1.65 Conversion EH: When printing application ID in any message, application '0LA' should be converted to '---' to increase clarity.

1.66 Conversion Bug: Whenever the conversion utility encounters a TOTAL command in a subtotal (or any non-detail?) window, it generates a DISPLAY command following the TOTAL but it truncates the 22nd byte of the field name. Also, it should not generate a DISPLAY for an update function. (See item 8.13.)

1.67 Conversion EH: The message "Inspect *command* on Total-Type Field" should name the field.

1.68 Conversion EH: The message "Inspect Window with Min Extra Lines > 0" should print the number of extra lines.

1.69 Bug?/Conversion EH: The 'in progress message' field in job, output, update, inquiry, and status processes loses the last word if it ends in the last byte of the field (unlike the 'report title' field which retains all characters). The conversion utility should warn about truncation for messages over 59 bytes (not over 60 bytes as it does now).

1.70 Bug?/Difference: If a DLU has a multi-part key and the first component is blank, SP2 will perform the auto access whereas APPX will not.

1.71 Bug: When a job is sent to background, APPX may not use the proper security. For example, at Pinellas, a user ran a two-part job which had security on both parts and where the second part was sent to background. The user had security set up under both the blank company and company "PIN" but only the "PIN" security group had execute rights for the job. Under company "PIN" the user could run the job in foreground but not in background. The UniQue Job Queue showed return code 34 which according to APPX means not having execute rights to the job.

1.72 Bug: If a field is based on a SAME AS domain, field attributes that should be visible and/or modifiable are not (for example, field description and column headings). Also, APPX may not know to require restructure at runtime if the root of the SAME AS domain has been modified. However, APPX does know to restructure the file when restructuring thru Data File Management. There is also some question as to whether documentation is available thru the SAME AS path. If these issues cause a problem, one solution is to point the affected field directly to the real domain in the x-application, instead of to the SAME AS domain in the local application.

Another solution is to make the SAME AS domain a local domain. This approach is less desirable since field modifications must be made manually to keep in sync with the root in the x-application. However, it may be the best approach when the x-application is not being converted at the same time.

1.73 Not a Bug?: There may be a problem related to the conversion or use of 'Float' in an output process under some weird circumstances. Pinellas reported an instance where a SP2 float of '0' on a work field allegedly got converted to ' '. After specifying float '0' in APPX, it still didn't work (even with a DISPLAY command) until Auto-Display '0' was changed to '1'. Then it worked even after Auto-Display was changed back to '0'. However, this problem could not be reproduced at C-Side. The conversion of Float is very straightforward and appears to be OK. This should probably not be treated as a bug until another instance of the problem surfaces. To list items with float, run the APPX Software, Inc.UTL CS utility "APPX ITEM File List" and enter "Y" to 'Check for Float?'.

1.74 **Action**/Bug: An indexed pseudo-file in SP2 is converted to an indexed working-storage file in APPX. However, if the indexed working-storage file is used in an input process, APPX 3.0 blows up with a system error at runtime if the user does a SCAN. However, it works fine if the working-storage file is consecutive. Since there is no known difference in the way the two behave (remember, no I/O is performed for a working-storage file), for any indexed working-storage file, remove the key and change it to be consecutive. If the pseudo-file is used as a PCF file, remove the default key path if any and mark the process for testing. (See item 8.17.)

To list indexed working-storage files for a given application, run the APPX Software, Inc.UTL CS utility "APPX FILE File List" and enter file organization "3" and file type "0". To find any input processes using an indexed working-storage file as its PCF (including files defined in other applications), run the APPX Software, Inc.UTL CS utility "APPX PROCESS File List" and enter process type "INPUT", PCF file organization "3", and PCF file type "0".

1.75 Bug?/**Difference!** (**flagged in V05 on files log for local files**): In APPX 3.0, delete protection on a higher level file does not protect lower level records from being deleted; in SP2 it does. For instance, when a user tried to delete a higher level record that had delete protection (in a function with the higher level scrolling), the higher level record was not deleted but there was no error message and the lower level records **were** deleted. In a function where the higher level was not scrolling, the user got the error message "Record is Delete-Protected" (after pressing ACKNOWLEDGE DELETE) but the lower level records were still deleted. In fact, lower level records are deleted before the delete-protected message even appears.

One work-around is to add code to check for ACKNOWLEDGE DELETE in *Option Intercept* and give an error if the higher level record is protected. To give the user advance warning, check INTERACTIVE PHASE for DATA DELETION in *Pre-Display/ Verify* and give an error if the record is protected. (Do not put the ERROR in Post PCF Read since APPX behavior can be peculiar.) To list files with delete protection, run the APPX Software, Inc.UTL CS utility "APPX FILE File List and enter "Y" to 'Check for Delete Protect?'.

1.76 Bug (**compile error**): An "Invalid Init Value" compile error in APPX can occur if a work field which has both a default value and a validation table (either defined directly or via a domain) is referenced by commands in another application. There is no problem within the same application. One work-around is to remove the default value or the validation table. If the work field is based on a domain, the domain validation table may not be needed for the work field itself (assuming it is not used as a modifiable field in data entry). If this is the case, change the work field to be defined independently without the validation table.

SECTION 2: Conversion Hints & APPX Programming Suggestions

2.1 In APPX, avoid making changes in the DATA MODIFICATION phase of ADD mode. Call the target back up in CHANGE mode instead.

2.2 Consider exiting and reentering APPX to be sure that critical changes are effective or to flush memory.

2.3 In APPX, when critical changes are needed or as a last resort for problems, consider deleting and re-adding or copying and re-adding (using *Design Transfer* to copy back items as needed). APPX has many hidden fields that sometimes are not reset properly when certain critical data is changed.

Also, it may be wise not to rename processes. For example, this has been suspected to have caused some problems at Pinellas. They sometimes use *Design Transfer* to copy a process to another name to work on, leaving the live process under the original name. Then, when they want to activate the work process, rather than renaming, they do the following:

1) use *Design Transfer* to copy the live process to an interim name
2) delete the live process
3) use *Design Transfer* to copy the work process to the live name
4) when appropriate, delete the work process and the interim process

When using *Design Transfer* to copy multi-level processes or jobs that you intend to run under the new name, be careful to consider all process names imbedded within, such as on children or job steps or Use Query specifications.

2.4 In APPX, use Option 88 to access the full value of abbreviated fields such as default values and edit masks.

2.5 In APPX, process the data dictionary more than once following critical events such as conversion or sweeping changes. If cross-application references exist, process the data dictionary for each application in series more than once.

2.6 Do not rely too heavily on the APPX *Cross-Reference Utility* since it often does not find all references and is sometimes inconsistent in what it does find (see item 1.3). It is probably best to use both the APPX X-Ref and the SP2 X-Ref in combination. The SP2 X-Ref is reliable (with the exception that it mislabels built-in functions) but the function name may not always reflect the APPX process name and there is no visibility of references to or in APPX Inquiry processes. Remember a SP2 multi-level process is split into multiple processes in APPX and a SP2 Output can spawn both an Output & Inquiry in APPX if it is used on an INQUIRY job step in SP2.

2.7 When all else fails (provided you have the proper registration), consider accessing the APPX application 0AD (Application Design) to decipher or resolve problems. Enter Application Design for the desired application and then:

```
99) Invoke a Process
      App/Version    0AD 00     Application Design
      Database ID
      Process Type   MENU
      Process Name   FILES EDIT
```

<u>Warning</u>: This facility should be used with great caution since there is no editing to preserve the integrity of the design files.

2.8 In order to address conversion problems, it is very helpful to run and save SP2 Tech Doc, X-Ref, etc., on disk so they can be searched as needed using DL or DISPLAY. SP2 documentation is reliable except that the SP2 'Built In Function' X-Ref mislabels the name of the built-in function. Also, for field tech doc, specify 'Fields (By Name)' since 'Fields (By File)' can cause a disk full error or too many extents. To conserve disk space, use Space-Saver to release space from the libraries containing the saved documentation. (See 2.8 attachment for details.)

Selected SP2 documentation should be printed such as the SP2 Table of Contents found at the end of SP2 Tech Doc. It helps to use the SP2 Table of Contents as your bible to the application by marking pertinent facts or unusual conditions on it. This is also a good place to mark functions that need testing.

From the user installation, obtain printouts of all SP2 Installation Control lists and reports as well as SP2 'Edit File Specifications' and 'Display File Information' for all companies. Mark any files that are shared by any company on the SP2 Table of Contents.

By the way, when you want to search for something in SP2 Tech Doc from a specific point, say, starting at input functions, the easiest way is to see what page inputs begin on from the SP2 Table of Contents; then, search for "- nnn" where nnn is 1 less than the beginning page.

2.9 After importing applications into APPX, it is very helpful to run and print selected APPX Tech Doc & X-Ref so they are at hand when needed. Remember only the APPX ILF X-Ref may be totally reliable (see items 1.3 & 2.6). (See 2.9 attachment for details.)

Now that the application is in APPX you can mark other pertinent information on your SP2 Table of Contents. The V05 conversion log warns whenever lower level processes are created in the conversion so you could easily mark multi-level processes. The V05 conversion log for Inquiry Functions warns whenever SP2 outputs are converted to APPX inquiry processes or to both APPX inquiry/output processes. (See item 13.2.)

2.10 Before beginning conversion, review the conversion logs and the SP2 X-Ref for references to any applications not being converted (i.e., "unavailable"). The conversion logs do not report all such references so it is important to examine the SP2 X-Ref closely. Decide how to handle these references since they may cause compile errors in APPX. If the application will later be converted, change domain references to be defined locally and make note of functions referencing its files for testing later. There may even be references to applications that do not exist in SP2 or are not used anymore; if so, get approval to delete such references. (See item 2.15.)

2.11 There are several issues related to numeric fields and display masks in APPX 3.0.0 which are helpful to know (besides what is in the manual):

1) If a mask is not explicitly defined for an item, APPX seems to use anáinternal display mask of "9-" for a signed numeric field and "9" for an unsigned numeric field (note placeholders are not needed for decimal places).á The 'Blank if Zero?' flag determines whether or not a zero is displayed. (Note: It has been requested that APPX change the internal mask to "90-" & "90" respectively.)

2) Bug?/Difference: If a signed or unsigned numeric field is negative and there is no sign explicitly defined in the mask if present for the item or in the internal display mask if not, APPX displays question marks whereas SP2 only displays question marks if there is not room for the sign. (Both APPX & SP2 store a sign for an unsigned field. A negative value cannot be entered for anáunsigned field but it may be computed via statements.) (Note: It has been requested that APPX print question marks similar to SP2.)

3) Bug?/Difference: If a field with decimal places has no mask or if the mask does not have a zeroáin the units position, APPX does not display a leading zero like SP2 does (for example, APPX prints .00, not 0.00). (See item 2.11.1.)

4) Difference: If a leading "-" sign is specified in the mask for an item and the item isánot right-justified, APPX displays a leading space for a positive value whereas SP2 does not.

5) Bug: When a mask of "90-" is defined for a field with no decimal places and the item is right-justified, APPX does not properly reserve a space for the trailing sign. This causes positive and negative values to not be properly aligned.

Run the APPX Software, Inc.job '!TEST MASKS LIST' or input '!TEST MASKS' in CGL SE to see the behavior of a new APPX version.

The standard conversion utility only generates a numeric mask for exceptional SP2 conditions such as unsigned or zero-filled fields (from SP2 inputs & outputs) and for non-trailing sign, grouping, or currency characters (from SP2 outputs only). **The APPX Software, Inc.conversion utility (V05) always generates a mask for output/inquiry processes.** If generated, the mask normally contains a zero in the units position, thus avoiding the .00 problem, but there may be other problems. (See items 4.24, 5.12, 9.29, 9.30, 10.15, & 10.19 for more information.) (Also, see item 2.20 for APPX environmental variables affecting handling of mask.)

2.12 *Warning: To avoid possible repercussions (such as changing or deleting data accidentally), CANCEL out whenever you encounter something unexpected in APPX and then investigate by calling up the same screen in INQUIRE using the most direct path available or by using FILES EDIT. It is possible that the screen was just improperly displayed the first time (perhaps due to the sequence in which it was accessed) but if you step thru it data will be permanently changed or deleted.*

2.13 Note: If **strange APPX system errors** are suddenly encountered doing everyday tasks not recently changed, suspect corrupted Em's and/or corrupted system variables that can be reset by IPLing. The best course of action is for the system administrator to have everyone else log off the system. He should delete the Em's and Dbg's and then run APPX thru the system menu before allowing any users back on the system. If system menu Em's are being built while users log on, it can cause certain users to get **segmentation faults** when attempting to run APPX.

2.14 Difference: SP2 uses PF keys while APPX uses OPTION or other named keys (such as END, CANCEL, ADD, DELETE, INQUIRE, CHANGE, HELP, SCAN, SELECT, PREVIOUS ACTIVITY, NEXT RECORD, etc.). The conversion utility converts (PF KEY) references to OPTION and flags most situations that are in doubt. However, there may still be text references to "PF" which are no longer appropriate. The V05 conversion log also flags text references to "PF" (see items 6.4, 8.29, & 9.34; items 9.16, 9.32, & 9.35 will be addressed later). You will usually change PF1 or PF2 to SCAN, PF9 to ADD, PF10 to DELETE, PF11 to INQUIRE, PF12 to CHANGE, PF13 to END, PF16 to END (or possibly to CANCEL), PF17-24 to OPTION 17-24, and PF32 to CANCEL.

2.15 Bug/Note: When using the Toolbox utility *Create Executable Modules*, remember there can be legitimate reasons for APPX to get a compile error. For example, a subroutine process may have references to labels in other subroutines or in the process invoking the subroutine; such subroutines will not compile independently but will compile at runtime. Also, the function may not be executable in SP2 either; it may be obsolete and/or reference non-existent fields, files, or applications (see items 2.10 & 2.18). If the screen hangs while running this utility, there may be an orphan frame or image in APPX. (See items 1.26 & 1.34).

2.16 Difference (**flagged in V05**): SP2 allows 79 characters for Default Value and Edit Mask whereas APPX only allows 60 characters. ***In the case of Edit Mask, this is a critical difference since Edit Mask is used to size the field and must match SP2.*** If necessary, remove the APPX edit mask by defining the field as ALPHA, not FORMAT, and add commands to edit the field in input processes allowing field entry. For those fields with default values longer than 60 bytes, if it matters, add commands to set these values in any process referencing the field. Beware of commands to restore default values. ***Warning: The truncation of Edit Mask may not be flagged on conversion logs.*** (See items 3.2, 4.12, 5.4, & 9.36.)

2.17 Note: Underlining on reports can cause some printers to beat themselves to death (since APPX sends a character, backspace, underscore to do underlining). Laser printers do not have any problem.

2.18 Note: If it is not practical to address some conversion issues in a process that does not compile in SP2, add code to *Start of Process* in APPX to document this situation and to cancel the job. Use XCopy to copy this code from *Start of Process* in UTL CS Status function "*CANCEL* OBSOLETE SP2 PROCESS".

2.19 Difference: In SP2, when using a non-unique alternate key to access data, records with duplicate alternate keys are returned in primary key sequence. In APPX, the alternate key must be defined as 'forced' to ensure this behavior. In both SP2 and APPX, if a sort is performed which does not specify the sequence to a unique level, the sequence is unpredictable past the specified level.

Also, note that it is possible for various Wang utilities such as DISPLAY or DL to present data from indexed files in primary key order (due to the Wang DMS file system). On a UNIX machine there is no inherent file access method and programs such as *pg* present the data as it is stored in the file.

2.20 Note: There is a special FD version of APPX which uses HEX(F) to signify a positive sign (which is the same as SP2). The standard version uses HEX(C). There are several environmental variables that can be set on the command line running APPX 3.0.0 which will modify its behavior, such as the following:

```
APPX_PRT_SCRIPT=filename
APPX_BG_SCRIPT=filename
APPX_184_ALIGNMENT=x
APPX_200_ALIGNMENT=x
APPX_RUN_MEM=nnnnnn          (for example, use 256000 to increase memory on PC)
APPX_SCRIPT_IN=filename      (APPX will use file to generate keystrokes)
APPX_SCRIPT_OUT=filename     (APPX will record keystrokes in file)
APPX_TRAP_RESTRUCT=x         (inserts TRAP in action process)
APPX_DB_TRAPS=x              (enables TRAPs in production)
APPX_NET_HASP=1              (needed by APPX for WINDOWS for hasp registration)
APPX_184_MASKS=x
```

Run the APPX Software, Inc.job '!TEST MASKS LIST' or input '!TEST MASKS' in CGL SE to see the results of using a different mask. (See items 2.11 & 8.30.)

2.21 Difference: SP2 allows a GOTO with an undefined label to compile (and only gives an error at runtime if the GOTO is actually executed). APPX gets a **compile error** if the label does not exist in the same event point or in a called subroutine.

2.22 Note: In APPX 3.0.0, a record in the DOMAIN file in 0AD provides the definition of each field in the Data Dictionary. If the field is actually a domain, the 'Generated?' flag in the first byte is "N"; otherwise, it is "Y". This flag is part of the DOMAIN primary key. Records in the FIELD file in 0AD, on the other hand, provide instances of the definition. For a domain field, there may be several instances. Although the two files, DOMAIN and FIELD, have many fields in common, some of these fields (such as Storage Mask) are now obsolete in the FIELD file. APPX updates the BITMAP, ELEMENT, FILE, and INITIAL files when the Data Dictionary is processed. At runtime the ELEMENT file is used to provide the field definition.

2.23 **Note: Before running the conversion utility, it would be wise to process the SP2 Data Dictionary for all the conversion applications (as well as any reference applications) and correct any errors in SP2 first.** Otherwise, the conversion may create invalid data in APPX that is not readily apparent. For instance, the conversion utility does not properly handle a SP2 field which is 'same as' a field which is 'same as' another field. Although this condition causes a SP2 error when processing the Data Dictionary, it does not cause an error at runtime so it is easy to go unnoticed.

2.24 EH?/Difference/Conversion EH?: In the SP2 Data Dictionary, the flag 'Use AM/PM if 12-Hour Time?' can be set for a date/time field to control whether or not AM and PM should be displayed and printed for the field if 12-hour (rather than 24-hour) time is used. There is no such capability in the APPX Data Dictionary. However, the flag 'Discard AM/PM?' can be set for an image item in APPX. APPX should offer this flag at the Data Dictionary level in order to emulate SP2. If so, the conversion utility would need to be changed to properly set this field.

2.25 Note: The Validation Table in the APPX Data Dictionary corresponds to the Look Up Table for an alpha field in the SP2 Data Dictionary. The conversion utility will not create an entry in the APPX validation table for a blank value imbedded among other values in the SP2 look up table but this does not cause a problem. SP2 and APPX behave the same since a flag on the image item controls whether or not a blank value is permitted for the field (the flag 'Blank OK?' in SP2 and the flag 'Required?' in APPX).

2.26 Difference: In the SP2 Data Dictionary, Auto Access Path is specified separately for each field defined as 'Same As' another field but is not specified for the root field of a 'Same As' field. In the APPX Data Dictionary, a DLU is only specified for the domain, not for a field based on the domain.

2.27 Difference: Due to the nature of SP2 and APPX, there are some fundamental differences between the two products which can have repercussions.

SP2 handles higher and lower level window files within a single function. APPX has a separate process for each PCF file and each process has its own store buffer for a file. For example, SP2 can do a STORE at a higher level and a RESTORE at a lower level whereas APPX cannot. (See item 8.35.)

2.28 Conversion EH: Each item in this document labeled as "**Action**" should be reviewed to see whether the conversion utility can or should be enhanced to either flag the item or to flag the item more precisely.

2.29 Note (FYI): Here are some notes in regard to record hold which apply to the PCF file only. Remember, only 1 record can be held at a time but APPX will restore the hold under certain circumstances. A read without hold on the same file does not lose the hold; a read with hold on the same file does. A read with hold on a different file does not lose the hold (because APPX restores it for the PCF file). A write on the same file loses the hold whereas a write to a different file does not. (See item 8.38.)

2.30 **Action** (**flagged in V05 after 08/27/96**): Some SP2 terminology was not flagged on earlier conversion logs (such as "PF" references in field default values or references to "SPEED" or "EXPLAIN" or their variations anywhere). Items that should be checked for such terminology include the following: field default values, menu/input images, */**/ERROR/WARNING/MESSAGE/CANCEL commands, and documentation.

2.31 Note (FYI): According to APPX, predefined field --- STATUS CODE (not in the manual) can be used to obtain text messages for failure conditions on commands. Here is one technique:

```
         OPEN     aaa filename   SHARE? Y  FAIL 2
   F     SET      --- MESSAGE ID                =  --- STATUS CODE
   F     READ     --- MESSAGE    HOLD 0    FAIL 0   KEY IS  MESSAGE ID
  (note: you can safely ignore the 'File Not Found' WARNING here;
   if in full screen edit, use OPTION 88 to accept the screen anyway.)
   FT    SET      --- TEMP 80                   =      OPEN aaa/filename
   FT    APPEND   --- TEMP 80                   1  --- MESSAGE TXT
   FF    SET      --- TEMP 80                   =      your error message
   F     CANCEL   --- TEMP 80
```

The --- MESSAGE TXT field is defined as X(500). However, in some cases, it may not contain much information beyond the obvious.

SECTION 3: Data Dictionary - Domains ***(from SP2 'Same As' Root Fields)***

3.1 Bug: The Format column may not properly show *Tbl* even though a Validation Table exists for domains with Type FORMAT (and possibly Type ALPHA). Note that a *pg* printout of file TOKEN in Portdata lists all fields that should have validation tables. Nothing works to force the display of 'Tbl' but it does not seem to cause a problem.

Conversion Bug (**fixed** in Version 04): Version 02 of the conversion utility can create TOKEN records with token type '1' in the first byte. Token type must be '0' for validation tables converted from SP2 to work properly.

3.2 Difference (**flagged in V05**): SP2 allows 79 characters for Default Value and Edit Mask whereas APPX only allows 60 characters. ***In the case of Edit Mask, this is a critical difference since Edit Mask is used to size the field and must match SP2.*** Address problem fields (see item 2.16).

3.3 Note (FYI): Be sure to exit and reenter APPX after making any changes that involve *Tbl* formats (such as validation tables and TOKEN fields); otherwise, the changes will not take effect.

3.4 Note (FYI): Domains with Type SYSTEM appear to convert correctly now. (Previously, System Info was blank for USER ADDED and USER CHANGED domains.)

3.5 **Action**: Check APPX for format missing in Format column and determine cause if not legitimate. The APPX Software, Inc.UTL utilities "APPX DOMAIN File List" and "APPX FIELD File List" provide a warning whenever format is missing in error. For example, this could be caused by an error in SP2 where values meant for Table Lookup were incorrectly entered in the SP2 Input Mask instead. If the field is never used in data entry, this situation has no effect in SP2 but it must be corrected in APPX. (Note: Format should be blank for field types GROUP HEADER, GROUP TRAILER, and SYNONYM although sometimes they are not blank due to what appear to be harmless APPX bugs.) (See items 3.8, 4.21, & 5.7.)

It may be possible to force the format to appear by making and unmaking a harmless change to the field (such as entering and removing a default value or temporarily changing the descriptive name.

3.6 Conversion Bug (**all cross-application domains flagged in V05**): The V05 **conversion log** gives the message ("Confirm Cross-Application Is Converted and Root Field Is Domain in Cross-Appl") whether or not the SP2 application 'aaa' actually exists. If application 'aaa' does not exist or is not converted, APPX gets an error ("SAME AS Application ID Not Found") when processing the data dictionary. If you change the field type in APPX, it does not preserve field values like SP2 does. You must change the field in SP2 from 'Same As' to the appropriate field type in order to see how to define the field in APPX. (See item 3.10.)

In addition, APPX 3.0 has several problems related to use of SAME AS domains which may justify using local domains anyway. For instance, certain field values are not accessible or visible for SAME AS fields. Also, APPX may not prevent use of a file where restructure is required due to a change in the root of a SAME AS field (although it does know to restructure if you actually run *Restructure* in Database Management).

3.7 Difference (**flagged in V05**): Since "PF" terminology is not applicable in APPX, modify "PF" text as appropriate. (See item 2.14.)

3.8 Difference/Conversion Bug (**all SP2 input masks flagged in V05**): SP2 allows invalid characters in the Input Mask and enforces entry of these characters as specified in the mask if the field is not left blank during input. APPX does not allow invalid characters in the Edit Mask. The conversion utility does not properly convert such a mask but it does flag the use of invalid characters on the conversion log. In addition, there is a conversion bug related to a SP2 input mask which contains an "*". The conversion utility will improperly delete the character in the "*" position from the default value and range values for the field in the Data Dictionary as well as from any constants used in sort record selections or SET/IF/OR/AND(/others?) commands with the field. This means if the SP2 mask has an "*", every use of the field must be examined.

Check out every SP2 input mask for agreement in APPX. If different or if it contains an "*", programming will probably be required to behave like SP2. The null value should also be considered. (See items 3.5, 4.26 & 5.14.)

For example, SP2 may have Input Mask '###-####' with Initial Value blank for a phone number field. The operator must enter the '-' when entering a phone number or the field may be left blank. In APPX, this converts to Edit Mask '###*####' with Default Value blank so the mask is effectively "### ####' which is not correct.

One work-around would be to change the Edit Mask to '###!####', define a substring field for the '-' position, and add code to any input program to edit a non-blank entry for the '-'.

Note: In our example (NAME PHONE, a non-domain field in application SUB PW), the conversion program improperly set the Default Display Mask in the FIELD file to "-" although it was properly set to blank in the DOMAIN file. However, the bogus "-" in the FIELD Default Display Mask was cleared when we changed the Edit Mask.

3.9 Bug (**all cross-application domains flagged in V05**): In APPX 3.0.0, a domain in one application may be defined as SAME AS a field which is not a domain in another application. This is a bug since APPX should give an error if the referenced field is not a true domain.

Note that the conversion utility can create this situation whenever a field in one application is same as a field in another application which does not have a 'Same As' use within its own application. Change the domain to be a local domain, not a SAME AS.

3.10 Conversion Bug (**duplicated names flagged in V05**): When the conversion utility encounters a field defined as 'Same As' a field in another application, it creates a domain in the conversion application defined as SAME AS the field in the other application. It names the domain by prefixing the root field name with the application ID (for example, "aaa ") and truncating if necessary. If this duplicates the name of a previously created domain, the new domain is not actually created. (See item 3.6.)

SECTION 4: Data Dictionary - Files

4.1 Conversion Bug (**flagged in V05**): A SP2 memory file created upon first use (i.e., location type 2 with creation type 1) is converted to an APPX disk/permanent file (although the APPX file does not show up in File Selection or File Specifications in Database Management). To correct, try changing the APPX file to memory, then back to disk, and finally back to memory (however, see item 4.2). It might be wise to check how the file is defined in APPX since there are hidden fields being maintained. To do this, run the APPX Software, Inc.UTL CS utility "APPX FILE File List".

4.2 Bug?: In certain cases the use of a memory file can cause problems in APPX. For example, sometimes a job using a memory file will not run to completion in background. Although many problems related to memory files have been fixed, there are still some unresolved issues. You might consider converting the file to disk if it will be used in any critical functions such as posting or updating. However, this may not be a total fix since the problem may be related to sharing issues with 'Separate Task OK?'="Y".

4.3 Bug (FYI): In APPX 1.8.4, if a file exists for a database currently sharing another database's file (via Use Database) and the shared file is restructured, "restructure required" error messages may be encountered unless any existing file is also restructured (by temporarily removing the Use Database value). Note that these unused files may be left-over from SP2 or may have been created to avoid the APPX bug on IF EXIST for a shared file (described in item 8.3). (See item 2.8 to identify SP2 shared files.)

4.4 Bug (Pete to fix in 3.1?)(**flagged in V05**): In APPX 2.x (thru APPX 3.0.0): Default Value is not editable or visible for TEXT fields/domains or for fields with Field Type DOMAIN referencing domains with Type FORMAT, raising questions as to whether the SP2 default value was preserved in the conversion or whether it will be honored. To view the default value (also called 'Init') which will be used at runtime, run the APPX Software, Inc.UTL CS utility "APPX ELEMENT File Detail List" and enter the field name as the element name. If necessary, add code to set the default value in *Start of Process* for any process referencing the field. (Default Value was editable in APPX 1.8.4.) (See item 5.11.)

4.5 Conversion Bug: Synonyms of concatenated fields in SP2 are converted to be synonyms of group headers in APPX. Such synonyms are not allowed in APPX and cause an **error when processing the DD**. Change these synonyms to be group headers instead.

4.6 Conversion Bug (**fixed in V05**): Root fields of domains within the same application lose their documentation in the conversion since the documentation is tied to the domain, not the root field. To allow access to this documentation, set the 'Use Domain Doc?' flag to "Y".

4.7 Bug (**flagged in V05**): In APPX 1.8.4 (thru APPX 3.0.0), if a field with Read Security is defined on an output or inquiry image and a DISPLAY command is given on that field, a user without the required security rights will get APPX system error RMUPD.C.156 ("upd_chr - characteristic doesn't fit on image"). In APPX 3.0.0, if the DISPLAY command is removed, the function works properly. This bug does not affect inputs nor does it affect users with the required security. (See items 5.1 & 10.1.)

4.8 Conversion Bug (**flagged in V05**): Record protection fields used for delete protection, key protection, and record access security are identified in the file header in APPX, not defined in the individual fields as in SP2. However, Record Protection Fields may not be set when appropriate in the conversion, particularly when the fields in question are based on domains, and should be checked. (See items 1.56, 4.9, & 4.10 too.)

4.9 Bug (FYI): In APPX 2.3, if Record Protection Fields are edited on the file header, APPX may improperly default these field names when the Record Protection pop-up box is accessed for another file with no record protection fields.

4.10 Bug (FYI): In APPX 2.3, if the Toolbox utility *Renumber Fields* is used after adding fields and the seq# of a delete/key/record protection field is changed, the Record Protection field names in the file header reflect the wrong fields.

4.11 Bug: The Format column may not properly show *Tbl* even though a Validation Table exists for fields with Field Type FORMAT (and possibly Type ALPHA). Note that a *pg* printout of file TOKEN in Portdata lists all fields that should have validation tables. Nothing works to force the display of 'Tbl' but it does not seem to cause a problem.

Conversion Bug (**fixed** in Version 04): Version 02 of the conversion utility can create TOKEN records with token type '1' in the first byte. Token type must be '0' for validation tables converted from SP2 to work properly.

4.12 Difference (**flagged in V05**): SP2 allows 79 characters for Default Value and Edit Mask whereas APPX only allows 60 characters. ***In the case of Edit Mask, this is a critical difference since Edit Mask is used to size the field and must match SP2.*** Address problem fields (see item 2.16).

4.13 Note (FYI): Be sure to exit and reenter APPX after making any changes that involve *Tbl* formats (such as validation tables and TOKEN fields); otherwise, the changes will not take effect.

4.14 Note (FYI): The AA option flags indicating the presence of non-default Additional Attributes may be improperly set after conversion. These are not necessarily corrected by *Synchronize Design Elements* which doesn't always follow the same rules as the Editor. (Most flags can be cleared by stepping thru the option or by making a temporary change.)

4.15 Note (FYI): Fields based on domains with a validation table do not indicate this since the validation table is only accessible (or visible) via the domain.

4.16 Note (Bug fixed): Root fields of domains with Data Lookup now appear to work OK if left as Field Type DOMAIN. (In earlier versions, it caused problems at runtime if the root field had a Data Lookup to itself.)

4.17 Note (Bug fixed): Fields with Field Type SYSTEM appear to convert correctly now. (In earlier versions, Format did not show properly for Date Add, User Add, Date Chg, and User Chg.)

4.18 Note (FYI): If nothing else works to overcome problems caused by Field Type DOMAIN, change the field type to not be a domain.

4.19 Bug (**fixed in V05**): If Conversion Utility 2.00 Version 04 dated 11/10/94 is used to convert to APPX 2.x and higher (at least thru APPX 3.0 Beta), field descriptions, column headings, and default values will be lost for non-domain fields. To avoid this problem, you must use *vi* (before running *convert.sh* and before importing) to delete records in file DOMAIN in Portdata that have 'Generated?' flag "Y" in the first byte. Since this flag is part of the key, all records for real domains (which have 'Generated?' flag "N") appear first. You can precede the *dd* command in *vi* with the number of lines (records) to delete.

4.20 **Action**/Difference (between the VS and UNIX): The V05 conversion utility provides a description of all files defined in the application on the File conversion log. Consecutive/permanent files should be checked for file transfer usage to see if a field for record delimiter is needed. Use the APPX Software, Inc.UTL CS utility "APPX FILE File List" to list them by entering File Organization "2" and 'Temporary File?' value "N" as job specs. Investigate any file which has all fields defined as ALPHA. If a UNIX file is to be transferred to a PC diskette or to tape, a HEX(0A) record delimiter may be needed to allow the file transfer program to recognize the record length (unless the file has fixed length records and the file transfer program can handle that). Most file transfer programs expect HEX(0A) as the standard UNIX record delimiter and HEX(0D0A) as the standard PC record delimiter and provide for this conversion. The VS does not need a record delimiter field.

To set the value of HEX(0A) in a one-byte alpha field, use the following code:
```
         CNV BIN  aaa alpha field                =      10
```
To set the value of HEX(0D) in a one-byte alpha field, use the following code:
```
         CNV BIN  aaa alpha field                =      13
```

4.21 **Action**: Check APPX for format missing in Format column and determine cause if not legitimate. The APPX Software, Inc.UTL utilities "APPX DOMAIN File List" and "APPX FIELD File List" provide a warning whenever format is missing in error. (See item 3.5.)

4.22 Conversion Bug?/Note: The conversion utility does not flag some conditions which cause **errors when processing** the data dictionary in APPX. For example, 'Groups Overlap' errors are not caught. These can be addressed by moving one group header to the end of the file and using synonyms for its components. Note that clearing up 'Groups Overlap' errors often eliminates other error messages as well (such as "Invalid 'Group Through' Field on Group Header").

4.23 Conversion Bug (**fixed in V05**): SP2 fields defined as 'Same As' with 'Copy Doc?'="Y" are converted with 'Use Domain Doc?'="N". It should be "Y". Note that SP2 Tech Doc does not show the 'Copy Doc?' flag. (See item 5.10.)

4.24 Difference: Since APPX may display question marks for unsigned numeric fields with negative values where SP2 would show the negative value, you may want to investigate any critical unsigned numeric field with the potential to go negative. The best way to find such fields is to look at the field format column in the APPX Data Dictionary for formats starting with "9(" or to search APPX File Tech Doc on the screen for "m 9(" (to allow for numeric fields based on domains). Mark these fields on the SP2 File Tech Doc and address as needed. (See items 2.11, 5.12, 9.30, & 10.19.)

However, it is probably **not practical** to address the question marks since this is supposed to be corrected in a future release of APPX.

4.25 Difference (**flagged in V05**): Since "PF" terminology is not applicable in APPX, modify "PF" text as appropriate. (See item 2.14.)

4.26 Difference/Conversion Bug (**flagged in V05**): SP2 allows invalid characters in the Input Mask and APPX does not. (See item 3.8.)

4.27 Note (FYI): According to Korry, in determining whether to preserve the contents of a field, APPX first looks for a matching field name. If that fails, APPX then looks for a matching date/time stamp for when the field was added. This technique allows the field name to be changed without losing data.

4.28 Note (FYI): If a file that needs restructuring is selected for *Rebuild Key Files*, the message "Key File Not Rebuilt, file/path not found" may appear.

4.29 Bug (FYI): APPX 1.8.4 may sometimes require that a data file be restructured or a structure file be rebuilt after seemingly innocent changes are made in the Data Dictionary.

4.30 **Action**/Difference: SP2 allows the use of some special characters in the file name whereas APPX does not. Also, SP2 allows a file to be defined in the Data Dictionary with a blank file description whereas APPX does not. The APPX Software, Inc.UTL CS utility "APPX FILE File List" will provide an error message if the file name is invalid and will provide a count of files with blank descriptions.

4.31 **Action**/Bug/Difference: SP2 allows a file to have an alternate key of up to 255 bytes. For a non-unique alternate key, APPX 3.0 requires that the alternate key PLUS the primary key be <= 255 bytes. Otherwise, this can cause an APPX system error at runtime (at Pinellas a runtime system error occurred after changing data on the screen in CHANGE mode of an input process where the PCF file had an alternate key over 255 bytes). The APPX Software, Inc.UTL CS utility "APPX FILE File List" will print a warning message if the primary key plus any alternate key exceeds 255 bytes. If the alternate key is non-unique, the key must be shortened to avoid errors.

4.32 **Action**/Bug?/Difference: The file access method used by APPX may impose stricter requirements for valid key structures than SP2 did. This can result in invalid key number errors during *Create Files* and *Restructure Files* even though the Data Dictionary processes with no errors. These requirements affect only disk files, not memory files.

If APPX is using its own proprietary file access method (called UNIX I/O), invalid key number errors can be caused by the following conditions:

1) There are more than 16 segments (fields) in the primary key.
2) There are more than 16 segments in any alternate key plus the primary key.
3) There are more than 15 alternate keys (this limit may be an APPX bug -- 16 alternate keys may be the actual limit).

The APPX Software, Inc.UTL CS utility "APPX FILE File List" will print error/warning messages as applicable for files found to have any of these conditions.

CAUTION: It may become necessary to redefine fields as substring fields in order to reduce the number of segments in a key. If so, use Design Transfer to make a copy of the file under another name before such modifications (perhaps renaming it with a "Z" in the first byte of the file name). Remove all keys from the "Z" file and change it to be a consecutive file. This file will be needed to import data for the file from the VS (be sure the name of the SP2 exported file in Portdata reflects the "Z" name before importing). Once imported, an update function can be used to transfer the data to the live file.

Also, **beware** of redefining FORMAT fields to be SUBSTRING fields since APPX will not enforce format requirements on fields that are substrings of FORMAT fields. Other editing that was inherent to the field definition may have to be programmed with commands instead.

SECTION 5: Data Dictionary - Work Fields

5.1 Bug (**flagged in V05**): In APPX 1.8.4 (thru APPX 3.0.0), if a field with Read Security is defined on an output or inquiry image and a DISPLAY command is given on that field, a user without the required security rights will get APPX system error RMUPD.C.156 ("upd_chr - characteristic doesn't fit on image"). In APPX 3.0.0, if the DISPLAY command is removed, the function works properly. This bug does not affect inputs nor does it affect users with the required security. (See items 4.7 & 10.1).

5.2 Bug (FYI): For work fields with Field Type DOMAIN referencing domains with Type ALPHA, Share Class is non-modifiable under APPX 2.x. Since share class is not accessible on the domain, this means Share Class cannot be changed for such fields. This is inconsistent since Share Class is modifiable if the domain referenced has Type NUMERIC. (Share Class is modifiable at the field level for all domain types in APPX 1.8.4.)

5.3 Bug: The Format column may not properly show *Tbl* even though a Validation Table exists for fields with Field Type FORMAT (and possibly Type ALPHA). Note that the *pg* printout of file TOKEN in Portdata lists all fields that should have validation tables. Nothing works to force the display of 'Tbl' but it does not seem to cause a problem.

Conversion Bug (**fixed** in Version 04): Version 02 of the conversion utility can create TOKEN records with token type '1' in the first byte. Token type must be '0' for validation tables converted from SP2 to work properly.

5.4 Difference (**flagged in V05**): SP2 allows 79 characters for Default Value and Edit Mask whereas APPX only allows 60 characters*. **In the case of Edit Mask, this is a critical difference since Edit Mask is used to size the field and must match SP2.*** Address problem fields (see item 2.16).

5.5 Note (FYI): Be sure to exit and reenter APPX after making any changes that involve *Tbl* formats (such as validation tables and TOKEN fields); otherwise, the changes will not take effect.

5.6 Bug (**fixed in V05**): If Conversion Utility 2.00 Version 04 dated 11/10/94 is used to convert to APPX 2.x and higher (at least thru APPX 3.0 Beta), field descriptions, column headings, and default values will be lost for non-domain fields. To avoid this problem, you must use *vi* (before running *convert.sh* and before importing) to delete records in file DOMAIN in Portdata that have Generated? flag "Y" in the first byte. Since the Generated flag is part of the key, all records for real domains (with Generated? "N") appear first. You can precede the *dd* command in *vi* with the number of lines (records) to delete.

5.7 **Action**: Check APPX for format missing in Format column and determine cause if not legitimate. The APPX Software, Inc.UTL utilities "APPX DOMAIN File List" and "APPX FIELD File List" provide a warning whenever format is missing in error. (See item 3.5.)

5.8 Bug (FYI): APPX 3.0.0 appears to stop processing work fields after it finds the first error. It only reports the first error found.

5.9 Conversion Bug: The conversion utility does not flag a work field defined as 'Same As' a field in a non-existent file in the same application. APPX gets a "Domain Not Found" **error when processing** work fields in the data dictionary. The easiest solution is to delete the work field if it has no references on the SP2 X-Ref. (See item 5.8.)

5.10 Conversion Bug (**fixed in V05**): SP2 fields defined as 'Same As' with 'Copy Doc?'="Y" are converted with 'Use Domain Doc?'="N". It should be "Y". This condition is **not practical** to correct. (See item 4.23.)

5.11 Bug?/Difference (Pete to fix in 3.1?)(**flagged in V05**): In SP2 a text work field can have a default value. In APPX 3.0.0, default value cannot be accessed or viewed for a text work field. To view the default value (also called 'Init') which will be used at runtime, run the APPX Software, Inc.UTL CS utility "APPX ELEMENT File Detail List" and enter the field name as the element name. If needed, add code to set these values in *Start of Process* for any APPX processes referencing these fields. (See item 4.4.)

5.12 Difference: Since APPX may display question marks for unsigned numeric fields with negative values where SP2 would show the negative value, you may want to investigate any critical unsigned numeric field with the potential to go negative. The best way to find such fields is to look at the field format column in the APPX Data Dictionary for formats starting with "9(" or to search APPX File Tech Doc on the screen for "m 9(" (to allow for numeric fields based on domains). Mark these fields on the SP2 File Tech Doc and address as needed. (See items 2.11, 4.24, 9.30, & 10.19.)

However, it is probably **not practical** to address the question marks since this is supposed to be corrected in a future release of APPX.

5.13 Difference (**flagged in V05**): Since "PF" terminology is not applicable in APPX, modify "PF" text as appropriate. (See item 2.14.)

5.14 Difference/Conversion Bug (**flagged in V05**): SP2 allows invalid characters in the Input Mask and APPX does not. (See item 3.8.)

SECTION 6: Menu Processes

6.1 Conversion Bug (**flagged in V05**): The SP2 built-in function BEGIN LIVE OPNS is converted to use the predefined subroutine --- BEGIN LIVE OPERATIONS but needs attention in order to call the proper menu. (See item 7.15 for how to handle use on a job step.)

If used directly as a menu item (assuming the menu is only one level deep in the application), from the menu item optional process change the AA parent disposition from NORMAL to END AFTER (Invocation Type is SUBPROCESS) and modify code in *Post Invocation* to read:

```
         MENU     aaa                              DETACHED     END? N  FAIL 0
```

Delete the code to set --- NEXT APPLICATION and --- NEXT PROCESS to blanks.

6.2 Difference (**flagged in V05**): Since APPX navigation differs from SP2, text on APPX menu screens should be changed accordingly. Change "ENTER" or "RETURN" to "END". (See item 2.14.)

6.3 Note (**flagged in V05**): The conversion program has an option to delete menu item #n where #n is operator-entered to represent the number normally used to return to the previous menu (usually #0 for ENTER/RETURN or #16). If menu item #n does not call a menu, the item is not deleted. If menu item #0 was specified and text has been modified to change "ENTER" or "RETURN" to "END" (see item 6.2), an APPX menu may be left with a menu item #0 that has no corresponding text. If so, decide whether to add text for "0) ..." or to intercept END in *Option Intercept*.

6.4 Difference (**flagged in V05**): Since "PF" terminology is not applicable in APPX, modify "PF" text as appropriate. (Addressed by item 2.14.)

6.5 Conversion Bug/Difference (**flagged in V05**): If the company is specified, the SP2 built-in function SELECT COMPANY should be converted to the APPX predefined subroutine SELECT DATABASE, not the APPX predefined input process SELECT DATABASE. Beware other issues too (see items 7.3 and 8.14). (Should this only be used on the initial menu? Add MENU statement in Post Invocation?)

6.6 Conversion Bug/Difference (**flagged in V05**): The SP2 built-in function END APPLICATION is not needed in APPX. (See item 7.16.)

6.7 Note (**fixed and flagged in V05**): It is usually desirable to set Cancel Disposition to CANCEL PARENT if the menu item is calling a menu in the same application or the cancel menu.

SECTION 7: Job Processes ***(from SP2 Jobs, Query Functions, and Displayed Disposition Functions)***

7.1 Conversion Bug (**fixed in V05**): In V04 invocation Type is improperly set to DETACHED on SP2 job steps running other jobs. Change this to RELATED in order to share disposition values. (Note: SP2 job steps using SUB JOB are converted properly to RELATED.)

7.2 Bug (FYI): In APPX 2.3, changing the name of a job can sometimes cause the rules of the job to be lost; they come back if the job is changed back to the original name. (This happened once at APPX Software, Inc.when changing the names of several disposition jobs in succession but could not be reproduced at will.) (See item 1.23 for orphan utility and reference to similar type bugs.)

7.3 Bug/Difference (**flagged**): In APPX 1.8.4 and 2.0, using a job to change companies does not work right when the change is not just for the duration of the job (for instance, the SP2 job includes the predefined input/subroutine SELECT DATABASE followed by MENU job steps or the SP2 job does not change the company back too). APPX does not enforce security for the new company and then can give security violation errors when the operator returns later to the original company. What was found to work properly was to offer a menu pick to change to a specific company, define a MENU process as the menu pick, and set SELECTED DATABASE to the company ID in *Pre-Child Invocation*. The **conversion log** flags jobs which are used to change companies. (See items 6.5 and 8.14.)

If the SP2 job is changing the company simply for the duration of the job, change Invocation Type to DETACHED on the job step to be run under the new company. If more than one job step is under the new company (or if the job step has security), create another job containing those job steps and invoke that job as DETACHED. APPX will automatically restore the original company at the completion of the job. (Under APPX 3.0 at Pinellas, where a job step was used to change company and the next job step had security, APPX used the security of the old company whereas SP2 uses the security of the new company. This problem can be avoided by creating another job for the job step.)

7.4 Bug (**fixed** in APPX 3.0 Beta): In APPX 1.8.4, the predefined subroutine BEGIN LIVE OPERATIONS does not work. This means the jobs to end Recovery Processing or Initial Setup do not work. You must enter System Administration in order to change phase. (See item 7.15.)

7.5 Bug (FYI): In APPX 1.8.4 the *Cross Reference Utility* may not find jobs run by commands.

7.6 Difference (**flagged in V05**): The **conversion log** flags any job using function code for review since many do not convert properly. For instance, SP2 jobs may use function code to determine which of several sorts to use with a given output/update function or to determine which output/update to run following a given sort. In APPX a query does not get executed until a job step is invoked which specifies the query in its Use Query option. Also, SP2 external functions can be defined to set function code from the return code. (See items 7.7, 7.19, & 10.16.)

7.7 Difference (**flagged in V05**): APPX cannot handle a job with a single query serving the same process twice for different results as in the following SP2 example:

```
            CSA  INPUT         CUSTOMER ACTIVITY REPORT
            CSA  SORT SETUP    CUSTOMER ACTIVITY REPORT
            CSA  DISPOSITION   STANDARD REPORT
            CSA  SORT          CUSTOMER ACTIVITY REPORT
      D     CSA  OUTPUT        CUSTOMER ACTIVITY REPORT
      S     CSA  OUTPUT        CUSTOMER ACTIVITY REPORT
```

This SP2 job uses the same output function to print detail and/or summary versions of the report by keying on the function code. If both versions are requested, the function code is initially set to "D" in the input function and then changed to "S" by commands in the Output End of Function.

This approach will not work in APPX 1.8.4. When a query is reused in a job (i.e., not run again), the shared resources of the original query replace the current values. This means that although the 2nd output job step starts out with a PROCESS CODE of "S", during execution PROCESS CODE reverts back to "D", the value at query runtime. Update Type in the Use Query cannot be set to 2 (Clear) since this will discard any operator changes to record selections or sort order. One work-around is to create a summary version of the output (for example, CUSTOMER ACTIVITY SUMMARY) which does not rely on testing PROCESS CODE within the output and change the job to use the summary version. (See item 10.16 too.)

7.8 Difference (FYI): Certain job steps that are self-explanatory in SP2 have no direct counterpart in APPX and job clarity is lost. For instance, DISPOSITION functions convert to JOB or SUBROUTINE processes in APPX, and certain built-in functions convert to rules on a blank job step. (See items 7.9 & 7.11.)

7.9 Suggestion: To distinguish disposition jobs or subroutines from ordinary ones, use *Change All References* to rename them to begin with the prefix *DISP* (for example, *DISP* STANDARD LIST). Beware of any cross-application use. ***Warning:*** ***If application JJJ installed, use the procedure in item 7.10 instead.***

7.10 Suggestion: Consider installing the APPX Software, Inc.Job/Report Disposition Control System for APPX (application JJJ).

If JJJ installed, perform the following steps for jobs converted from SP2 displayed disposition functions:

1) Print APPX Tech Doc for 'old' disposition jobs (converted status).
2) Create new disposition jobs by copying JJJ Job "*DISP*".
3) Define new disposition jobs in JJJ Disposition Definitions F/M.
   If background not implemented (on PC?), set background options to "N".
4) Delete old disposition jobs.
5) Run *Change All References* from old disp jobs to new disp jobs.

Also, perform the following steps for subroutines converted from SP2 non-displayed disposition functions:

1) Print APPX Tech Doc for disposition subroutines (converted status).
2) Print APPX X-Ref for disposition subroutines (& compare to SP2 X-Ref).
3) Create new disposition jobs by copying JJJ Job "*DISP*".
4) Define new disposition jobs in JJJ Disposition Definitions F/M
   specifying "N" for display options.
   If background not implemented (on PC?), set background options to "N".
5) Delete disposition subroutines.
6) Modify all references to disposition subroutines to call new disposition
   jobs.

Finally, print APPX Process X-Ref on --- INPUT DISPOSITION to confirm there are no references. There may be references in jobs that used the SP2 built-in disposition function as well as references in inquiry processes on key entry frames (see optional process for the image). These should all be changed to reference JJJ INPUT DISPOSITION instead. Remember any changes made on key entry frames must be reimplemented if key entry frames are rebuilt.

7.11 Suggestion: To provide greater visibility into what a job is doing, consider replacing blank job steps with SUBROUTINE job steps that call self-explanatory or do-nothing subroutines appropriately labelled, such as: (see items 1.4, 7.17, 7.18, & 7.19)

```
*SET* CANCEL OK
*SET* CANCEL NOT OK
*SET* PROCESS CODE    (do-nothing subroutine)
*GOTO*                (do-nothing subroutine)
```

For do-nothing subroutines, the rules should still be attached to the job step. Create the self-explanatory subroutine with the appropriate code and create the do-nothing subroutine with a single comment line.

7.12 Suggestion: Consider printing a completion report as the last step in jobs which perform critical updates. Otherwise, if the job is run in background, there will be no visibility that it did not run to completion. For example, an error message does not print if the job is cancelled.

7.13 Note (FYI): APPX handles queries differently than SP2 handles sorts and these differences can cause problems in the way jobs are converted. Background processing is also handled differently (see item 7.21).

7.14 Difference: In SP2 the title of the job appears at the top of screens presented by STATUS and DISPOSITION job steps. In APPX the title of the job does not appear. If desired, review STATUS processes to determine if a warning message naming the job is needed. Check SP2 Status X-Ref for use. (See item 14.1.)

7.15 Conversion Bug (**flagged in V05**): The SP2 built-in function BEGIN LIVE OPNS is converted to use the predefined subroutine --- BEGIN LIVE OPERATIONS but needs attention in order to call the proper menu. (See item 6.1 for how to handle use directly from menu.)

If used as a step in a job, from the job step change the AA Invocation Type from RELATED to SUBPROCESS (Parent Disposition is NORMAL) and add the following code to *Post Invocation*:

```
         MENU     aaa                               DETACHED     END? N  FAIL 0
```

Then, from the menu item optional process for the job, change the AA parent disposition from NORMAL to END AFTER (Invocation Type is DETACHED). This assumes that the menu is only one level deep in the application.

7.16 Note (**flagged**): The conversion log may generate the message "Revise Use of END APPLICATION on Job Step". Since there is no equivalent of this SP2 built-in function in APPX, delete any remaining references to the job and then delete the job. If the job is associated with menu item 0 in SP2 (as is normally the case) and if the application was converted with the conversion option to delete menu item 0, there may not be any references to the job in APPX. However, change menu text to read "Press END to End Application".

7.17 Note (**flagged in V05**): In light of the APPX bugs related to CANCEL OK (documented in item 1.4), extra work is needed to protect jobs from being cancelled that used the SP2 built-in function 'CANCEL NOT OK'. Check each process following this job step and set 'User Cancel OK?'="N". Also, insert another job step to set CANCEL OK="N" following any STATUS job step. The **conversion log** flags any job using CANCEL NOT OK. (See items 1.4 & 7.11).

7.18 Note (**flagged in V05**): The **conversion log** flags any job using GOTO since these jobs may not convert properly. If a job loops via GOTO to reuse a query, Update Type in Use Query should be changed to 2 (Clear) so that APPX will not reuse the same query file and so that different user criteria can be applied. Also, once background processing is enabled, it is illegal in APPX to branch to a job step before the job step with 'Separate Task?'="Y". Remember that GOTO is implemented in APPX by setting the NEXT CHILD ID predefined field. (See item 7.11.)

7.19 Difference (**flagged in V05**): It may be necessary to add an UPDATE or SUBROUTINE job step in APPX to replace any updating performed in the query (including setting PROCESS CODE). (See item 12.11 for details.)

7.20 Difference (**flagged in V05**): If a job has 2 or more OUTPUT's but only one SORT, confirm that both OUTPUT's are using the sort by checking the APPX job step UQ. If so, there is the potential for bogus blank lines in one of the outputs. (See items 10.16 & 12.16.)

7.21 Bug/Difference (**flagged in V05**): APPX handles background processing differently than SP2 and provides a new flag 'Separate Task?' defined at the job step level. The **conversion log** flags any suspect jobs for review (see items 1.32 and 1.33 too):

1) Any single-step job should have 'Separate Task?'="N".
2) A multi-step job should have 'Separate Task?'="Y" only on its final job step and then only if the job is not run by another job with 'Separate Task?'="Y".
3) If a job has more than one job step following its disposition job step, these job steps should be moved to a new job (or jobs if necessary).

For example, to address a job which has both an update and an output following its disposition job step, use *Design Transfer* to create a new job named the same but ending in "2". In the original job, delete the update and output job steps and add a JOB job step calling the new job as the final job step. Set 'Separate Task?'="Y" only on the new JOB job step. In the new job, delete all job steps except the update and output and set 'Separate Task?'="N" on all job steps.

Note that the disposition job will control whether or not the operator can actually select background processing from the disposition screen.

<u>SECTION 8: Commands</u>: Use APPX ILF X-Ref (reliable) and APPX PDF X-Ref (not so reliable) from item 2.9. Check for APPX Inquiry Process too!

8.1 Bug (**fixed** in APPX 3.0): APPX is only checking the first 22 characters of name on the COPY command. This can cause a duplicate label error at compile time. This can be avoided by using GOSUB instead (now that you can GOSUB to another application).

8.2 Bug (FYI): When editing rules in Application Design, APPX may not properly handle the options to Move or Copy commands! The screen may not redisplay properly and it is wise to exit and reenter the event point before doing any further editing. Upon reentering you should confirm that the commands were moved or copied properly since some commands can be deposited in the wrong place.

8.3 Bug (**flagged in V05**): APPX 1.8.4 can erroneously return a FALSE condition following an IF EXIST on a shared file if the current database does not own the shared file. One work-around is to create the file within the current database before specifying the Use Database. However, this work-around has a drawback since a related bug (described in item 4.3) means that any restructures must be performed on both files. Check files used in IF EXIST commands to see if they are shared and then decide how to address. (See items 2.8 & 4.3.)

8.4 Bug (**fixed** in APPX 3.0.0): APPX does not always process a CALC command correctly. The problem seems to occur when there are too many significant digits. Consider the following example: CALC X = 47*(21+(7/12)) where the correct answer is 1014.41666666667. APPX 2.3 gets a numeric overflow error (whereas APPX 1.8.4 and APPX 2.1 both get the wrong answer of 14.41666666667). To avoid problems, rewrite code to use COMPUTE commands, not CALC. Run APPX Software, Inc.UTL CS utility "Check APPX CALC Statement" to check a new version of APPX. (See item 8.32.)

8.5 Conversion Bug (**fixed in V05**): The V04 conversion can generate references to SUBMIT CLASS which is not a valid PDF in APPX. Run a X-Ref in APPX and delete all such references. The V05 conversion changed SUBMIT CLASS to SUBMIT PRIORITY. (Note: These may been deleted already by item 7.10.)

8.6 Conversion Bug (**fixed in V05**): The V04 conversion can generate references to DISPOSITION OPTIONS. Although this is a valid PDF, it is not documented and it behaves peculiarly. It does not appear to be needed so the best approach is to delete all such references. The V05 conversion does not generate such references. (Note: These may been deleted already by item 7.10.)

8.7 Difference (**flagged in V05**): APPX differs from SP2 in the behavior of CNV BIN; SP2 truncates a decimal number to an integer whereas APPX rounds. This means that the SP2 technique to use CNV BIN to truncate will not work in APPX. The converted code will typically look like this:

```
         CNV BIN  aaa WORK alpha field          =  aaa number field
         CNV BIN  aaa number field              =  aaa WORK alpha field
```

Rewrite code to use CALC commands as follows (provided X is not being used):

```
         SET      --- X                         =  aaa number field
         IF       --- X                         GE
   T     CALC     X  = @FLOOR(X)
   F     CALC     X  = @CEIL(X)
         SET      aaa number field              =  --- X
```

8.8 Difference (**flagged in V05**): APPX differs from SP2 in maintaining a hold on a record when changing its primary key. In SP2, you can read a record with hold, (optionally store it), change the primary key, write it, (optionally restore it), and delete it in that order. In APPX 1.8.4, you must DELETE the record before the WRITE with the new key to avoid getting a "record not held" error at runtime. The **conversion log** flags all uses of DELETE for inspection. In addition, the APPX Software, Inc.UTL CS utility "APPX STMT I/O List" can be used to review these commands.

8.9 Difference (**flagged in V05**): APPX differs from SP2 in that APPX considers each BEG READ or READNEXT command to be a separate thread with its own pointer. This means that if a BEG AT or END AT command is applicable, each BEG READ or READNEXT command must have its own.

For example, the conversion message "Inspect READNEXT within BEG READ loop" should be researched to determine if both read's affect the same file. If so, a BEG AT should be inserted before the READNEXT to position the pointer. In another example, there may be a single BEG AT serving two alternating READNEXT commands in SP2. This would not work in APPX (one READNEXT would start at the beginning of the file). A solution would be to put the READNEXT in a subroutine at the end of the event point and replace the READNEXT's with GOSUB's.

Note that SP2 and APPX behave the same where there is an initial BEG AT/END AT for a READNEXT and then a single BEG AT or END AT affecting the same READNEXT statement. The BEG AT with no END AT will establish a new beginning pointer and effectively set the END AT pointer to end of file. The END AT with no BEG AT does not change the current pointer (it continues reading from wherever it was) but it does establish a new ending pointer.

Likewise, SP2 and APPX behave the same where there is a READ followed by a READNEXT with no BEG AT. The READ will establish the pointer for the READNEXT (assuming the same key path is used).

Also, note that SP2 and APPX behave the same in regard to appending HEX(00)'s or HEX(FF)'s where the field specified in the BEG AT or END AT is shorter than the key used to read the file.

The APPX Software, Inc.UTL CS utility "APPX STMT I/O List" can be used to review all BEG AT/END AT/BEG READ/END READ/READNEXT commands in the application.

8.10 Difference (**fixed in 3.0**): In earlier releases of APPX, the BEG AT and END AT commands must be in the same event point as the associated BEG READ or READNEXT in order to be effective. In SP2 they could be in different command series. The APPX Software, Inc.UTL CS utility "APPX STMT I/O List" can be used to review these commands. *Warning: This does not cause a compile error in unfixed releases of APPX; the program just does not run properly!*

8.11 **Action**/Difference (**compile error**): The END LOOP and END READ commands must have matching BEG LOOP and BEG READ commands or it causes a **compile error** in APPX whereas SP2 allowed this. Both SP2 and APPX allow BEG LOOP and BEG READ commands without a matching END LOOP and END READ. To find BEG LOOP/END LOOP commands, run the APPX Software, Inc.UTL CS utility "APPX STMT 'BEG LOOP/END LOOP' List". To find BEG READ/END READ commands, run the APPX Software, Inc.UTL CS utility "APPX STMT I/O List". If there are any mismatches, you must rewrite the code in APPX to eliminate these.

8.12 Difference (**flagged**): The **conversion log** flags the use of MODE since many (if not most) commands that reference MODE should be modified to reference INTERACTIVE PHASE instead. Remember that it is possible to be in ADD mode at the same time interactive phase is DATA MODIFICATION (for instance, when returning to a higher level via PF4 PREVIOUS ACTIVITY or when editing a scrolling record.) However, beware of complicating factors. For instance, do not change the use of MODE in a footer window (since INTERACTIVE PHASE will never be DATA ADDITION). Also, reconsider if testing NE condition or if testing to switch modes. Note that a scrolling frame executes *After PCF-Read* twice in the DATA SCROLL INTERACTIVE PHASE; this may have repercussions if mode was being used to set a flag which is then tested independently. (See item 9.12.)

MODE:	ADD	INTERACTIVE PHASE:	DATA ADDITION
	DELETE		DATA DELETION
	INQUIRE		DATA INQUIRE
	CHANGE		DATA MODIFICATION

8.13 **Action**/Difference (**partially flagged in V05**): Fields can be defined as TOTAL fields in SP2 whereas there is no such field type in APPX. Instead, APPX associates two areas with a NUMERIC field, the current area and the total area.

The TOTAL command (as well as automatic totaling) affect the total area while other commands such as SET, COMPUTE, and RESTORE affect the current area. The SET TOT command is used to move the total area into the current area.

In order to emulate SP2, the conversion utility inserts a SET TOT command in front of any command using a SP2 total-type field in non-detail windows. Although the first SET TOT is necessary for a given field, needless repetitions only serve to obscure the code and can actually cause a bug at runtime if the SP2 total-type field has been modified by SET/COMPUTE/RESTORE commands.

The best approach to emulate SP2 is to have a single SET TOT (to itself) for each SP2 total-type field (watch out for use in loops!) at the beginning of the event point and delete all the other SET TOT'S. In addition, if the SP2 total-type field is modified by SET/COMPUTE/RESTORE commands and if it is defined on the image, add a DISPLAY command following the SET/COMPUTE/RESTORE commands and change the Auto-Display type on the image from '4'=Auto-Total Value to '0'=None. Also, if there is a TOTAL command on a SP2 total-type field in a non-detail window, add a SET TOT following the TOTAL command and if the field is defined on the image, change the Auto-Display type on the image from '4' to '0' and check for a DISPLAY command. (The conversion utility generates a DISPLAY command following a TOTAL command in a subtotal window but it may drop the 22nd character of the field name.) Note that if a SET TOT is generated for a SP2 command library which is copied only into detail windows, the SET TOT should be deleted.

The conversion log currently warns to inspect SET TOT commands and to inspect SET/COMPUTE/RESTORE commands on total-type fields. **However, it does not yet warn if a TOTAL command is used in a non-detail window. Use the APPX Software, Inc.UTL CS utility "APPX STMT 'TOTAL/SET TOT' List" to find these by blanking the opcode for SET TOT and selecting Frame Class NE RECORD in the sort.** (See items 1.66, 8.27, 10.23, & 11.4.)

8.14 Difference (**flagged in V05**): NEXT DATABASE is cleared in APPX by OPEN, CREATE, and SCRATCH commands. Any use of NEXT DATABASE should be carefully reviewed to see if this causes a problem. (See item 10.25.)

8.15 **Action**/Bug?/Difference (**DATE ADD flagged in V05 + compile errors**): SP2 can handle some uses of DATE ADD which cause a 'date invalid' **error at runtime** in APPX 3.0.0. For example, when subtracting one month from a 'month only' date field with month 01, SP2 got month 12 whereas APPX got an error (APPX handled adding months OK - it got month 02 when adding 13 months). Also, when subtracting one year from a 'year only' date field with year 00, SP2 got year 99 whereas APPX got an error. Errors appear at risk when subtracting from a date field without a higher component than the unit. One work-around is to perform the DATE ADD using a work field with the next higher component defined. The **conversion log** flags all DATE ADD commands. Unit values are defined as: 8=CC, 7=YY, 6=MM, 5=DD, 4=hh, 3=mm, 2=ss, 1=th.

SP2 also allows some uses of DATE BTW commands that will cause a **compile error** in APPX. For instance, SP2 allows days (unit 5) on a DATE BTW command for a month/day date field whereas APPX does not (although DATE ADD compiles OK).

Use the APPX Software, Inc.UTL CS utility "APPX STMT 'DATE ADD/DATE BTW' List" to review all DATE ADD/DATE BTW commands in the application.

8.16 Difference (**flagged in V05**): In SP2 the short version of SET MNTH (type S) returns the three-character month in upper case; APPX returns it in upper/lower case (just the first letter is capitalized). Check usage to see if this matters.

8.17 Difference (**compile error**): SP2 and APPX differ in regard to commands allowed on WORKING-STORAGE files (pseudo-files). In SP2, IF EXIST/SCRATCH/CREATE commands can be executed and READ/WRITE/REWRITE/BEG READ/READNEXT commands can be compiled but cause a fatal error if executed at runtime. All such commands cause a **compile error** in APPX and should be deleted. Use the APPX Software, Inc.UTL CS utility "APPX STMT I/O List" for the WORKING-STORAGE file to find these commands. (See item 1.74.)

8.18 Difference (**flagged in V05**): APPX handles the cursor option of the POSITION command differently than SP2. For example, the placement of the cursor will be one position to the left on the screen in SP2 than it will be in APPX for the same CURSOR COLUMN. Check usage to see if this difference matters. (See item 9.15.)

8.19 Difference (**flagged in V05**): SP2 uses the RUN command to invoke execution of a menu, job, or function. The conversion utility converts this to be INPUT, INQUIRY, JOB, MENU, OUTPUT, QUERY, STATUS, SUBR, UPDATE, etc. (SP2 RUN commands invoking external functions are converted to COPY) and Version 05 sets invocation type to RELATED. If memory files or such do not need to be shared, invocation type can be set to DETACHED. The **conversion log** flags all uses of the SP2 RUN command.

8.20 Difference (**flagged in V05**): ENTRY LEVEL in APPX pertains only to DATA ADDITION and is undefined and hence unreliable in DATA MODIFICATION, unlike SP2 where (ENTRY) was always 2 in CHANGE mode. To check for the equivalent of SP2 secondary entry, code as follows in APPX: (see item 9.10)

```
         IF       --- ENTRY LEVEL                GE     2
         AND      --- INTERACTIVE PHASE          EQ     DATA ADDITION
         OR       --- INTERACTIVE PHASE          EQ     DATA MODIFICATION
         OR       --- INTERACTIVE PHASE          EQ     DATA INQUIRE
```

8.21 Note (FYI): The AT FIELD command only works in the *Option Intercept* event point (it always returns a False condition in other event points). This is not a difference since the AT FIELD command is only valid in *PF Intercept* in SP2.

8.22 Note (FYI): You can use Option 88 to bypass a bogus ERROR message when editing commands. ERROR messages can be caused by the following conditions:
   1) APPX 2.x bug is only checking the first 22 characters of name
      on COPY command (**fixed** in 3.0.0).
   2) A field defined as >= 0 is being set to or multiplied by negative value
      (it is OK to multiply by -1 in order to force a positive value
      but any other use should probably be investigated).
   3) A validation table exists for a field and the field is being set to a
      value not in the table (this condition should be corrected if possible
      - you can use the APPX Software, Inc.UTL CS utility "APPX STMT 'Format 5'
List"
      to list commands where that field is being set or tested.)

8.23 Bug/Difference (**flagged in V05**): APPX 2.3 (thru 3.0.0) differs from SP2 in its handling of ALTERNATE IMAGE NUMBER as discussed below (see items 1.42, 1.43, 9.26 & 10.11). If (ALTERNATE NUMBER) is not being explicitly set in SP2 wherever it is used, add code in APPX to do so. The Input/Output conversion logs flag all *Set Alternate No.* command series which have commands. In addition, the Commands conversion log flags all commands setting (ALTERNATE NUMBER) which are not in a *Set Alternate No.* command series. (The APPX Software, Inc.UTL CS utility "APPX STMT 'Format 5' List" can be used to review code by entering --- ALTERNATE IMAGE NUMBER as app/field 1 in job specs.)

SP2 sets (ALTERNATE NUMBER) to 1 before the first execution of a standard window each time thru and sets (ALTERNATE NUMBER) to 0 before the first execution of a repeating window each time thru.

APPX initializes ALTERNATE IMAGE NUMBER at 0 at start of process and never changes it (except in INQUIRY processes by selection of a key-entry frame); otherwise, it remains as last set by commands.

Also, note the following HELP text for Frame Type in APPX 3.0.0:

"Frame Type governs the execution of an image (or images) for the frame. The Frame Type options are:

> NORMAL executes the first (lowest numbered) alternate image unless another image is explicitly selected with statements and the predefined field ALTERNATE

IMAGE NUMBER. (See bug in item 1.42.)

> OPTIONAL executes the image that is identified by the Normal Frame Type above, with the following exception: If the value in the predefined field ALTERNATE IMAGE NUMBER is 0, do not execute an image.

> REPEATING executes the image repeatedly until the value in the ALTERNATE IMAGE NUMBER is set to 0 by statements."

Note that OPTIONAL frames are converted from SP2 windows with code in *Set Alternate No.* or with 'Alternate Window No.'<>"1". REPEATING frames are converted from SP2 windows with 'Repeat?'="Y". If there are no commands to set ALTERNATE IMAGE NUMBER, REPEATING frames execute only if the current value is non-zero.

8.24 Difference (**flagged in V05**): (RECORD NUMBER) can be used in SP2 to provide a record count whereas RECORD NUMBER has not yet been implemented in APPX and always returns zero. This is flagged on the **conversion log**.

8.25 Bug (**fixed** in APPX 3.0 Beta): In APPX 1.8.4, commands referencing screen items with appearance numbers do not always work properly. For instance, OK INPUT and NO INPUT do not work right. For some operations the problem is caused by APPX always using appearance #1. Sometimes the use of a constant other than 1 as appearance number causes a **compile error** (for example, DISPLAY on appearance #2). (See items 9.3 and 10.3.)

8.26 Difference (**flagged in V05**): SP2 could successfully use END FN in a sort (for example, to quit sorting after processing a certain number of records). This is converted to EXIT in APPX which causes a system error at runtime when used in a query (see item 1.12). Modify code to set INCLUDE RECORD to "0" or "N" instead.

8.27 **Action**/Difference (**partially flagged in V05**): When SP2 executes a TOTAL command, it rounds the source value before accumulating; APPX does not. In certain instances, this can result in printing reports where detail values do not sum to total values. One way to overcome this is to set the TOTAL field to the source field and TOTAL the TOTAL field as follows (see item 8.13 for how TOTAL works in APPX):

```
         SET      CIC WORK TOTAL COST 9 2        =  CIC WORK DETAIL COST 6 3
         TOTAL    CIC WORK TOTAL COST 9 2        +  CIC WORK TOTAL COST 9 2
```

The conversion log flags any TOTAL commands where rounding is a problem with a message to revise rounding. However, prior to 07/12/96, it was not flagging TOTAL commands adding the variable (D). These can be found by using the APPX Software, Inc.UTL CS utility "APPX STMT 'Format 5' List" to search TOTAL commands (opcode 78) for "--- D" in the 2nd application ID/field name.

8.28 Difference (**flagged in V05**): APPX differs from SP2 in some respects in its treatment of the SET command when used to set alpha field = number field. The problem occurs if SP2 is testing for a specific result since APPX may format the result differently. The only known difference is when the number is an indexvariable with value zero (SP2 gets 0 whereas APPX gets 0.000000000000000 unless setting a format field with right alignment and 0-pad in which case there is no difference). This can be overcome by using a numeric work field, rather than the indexed variable, to set the alpha field. Non-zero values appear to convert the same. If other situations causing differences are discovered, it may be possible to use the CNV TEXT command instead. The **conversion log** flags this condition but it may not be worthwhile to investigate since differences are rare. The APPX Software, Inc.UTL CS utility "APPX STMT 'Format 5' List" can be used to list the actual commands by entering "ALPHA" as basic field type 1 and "NUMERIC" as basic field type 2. To see only commands using indexed variables, also enter "---" as app ID 2.

8.29 Difference (**flagged in V05**): Since "PF" terminology is not applicable in APPX, modify "PF" text in ERROR, WARNING, and MESSAGE commands as appropriate. (See item 2.14.)

8.30 Difference (**all BEG AT's flagged in V05**): Whereas both SP2 and APPX use 'D" to signify a negative sign in a packed decimal field, SP2 uses 'F' to signify a positive sign and APPX uses 'C' (unless the special FD version of APPX is used). Also, in both SP2 and APPX, a date field is stored as packed decimal and a blank date is stored as -1. For instance, a blank month-only date field would be stored as HEX(001D) in both systems. However, month '01' is stored as HEX(001F) in SP2 and as HEX(001C) in APPX. This means if a blank month-only date field is used in a BEG AT command, month 01 will be skipped in APPX. Conversely, if month 01 is used, APPX will include blank months.

Although a single component date field is the likeliest scenario to cause a problem, using other date fields with BEG AT may also be at risk. Fortunately, the most common types of date fields such as ymd, cymd, or md are OK but beware ym. The APPX Software, Inc.UTL CS utility "APPX STMT I/O List" can be used to review BEG AT commands. Also, the "APPX FILE File List" will list files with dates in key fields by entering "Y" to 'Only Files w/Date in Key?'. (See item 1.22.)

8.31 Difference/Conversion EH? (FYI): SP2 and APPX can behave differently when setting partial components of a date field. For example, when setting a ym date field from a non-blank year-only date field, SP2 sets the month component to the current month whereas APPX sets it to month '01'. (See item 8.36.)

8.32 Bug (**fixed in later engines**): APPX 3.0.0 Optimized may not divide properly. For example, in the divide test 14782712.00 / 197064.93 = 75.01 (rounded to 2 decimal places), the CALC command (as well as COMPUTE commands using work fields, not variables) may get the wrong answer of 29.34. Run APPX Software, Inc.UTL CS "Check APPX CALC Statement" to check a new version of APPX. (See item 8.4.)

8.33 Difference (**all WRITE commands flagged in V05**): Due to the increased speed in performance of computers running APPX, WRITE commands for files with a date/time stamp in the key may fail since the key may no longer be unique. If the date/time stamp is being used to make the key unique, code must be present to loop back to reset the date/time stamp and retry the WRITE on a FALSE condition. The **conversion log** flags all WRITE commands for inspection but it only matters on those files with a date/time stamp in the primary key or in any alternate/unique keys.

The APPX Software, Inc.UTL CS utility "APPX FILE File List" will list files with dates in key fields by entering "Y" to 'Only Files w/Date in Key?'. The APPX Software, Inc.UTL CS utility "APPX FIELD File List" will list all date fields by entering "DATE" as 'Basic Field Type'. The date field list is more useful for this purpose if you change the sort order to be file name 1st and field name 2nd (you might want to specify subheading to be NO on file name to conserve paper since the file name is given on each detail line). If desired, the APPX Software, Inc.UTL CS utility "APPX STMT I/O List" can be used to review all WRITE commands. (See item 8.36 too.)

8.34 Difference (**fixed/flagged**): SP2 allows BL, NB, and BW as the relation on IF/AND/OR commands whereas APPX does not. The conversion utility converts BL to EQ and NB to NE. It flags the use of BW with the message "Revise Relation".

8.35 **Action**/Difference/Conversion EH: SP2 handles higher and lower level window files within a single function. APPX has a separate process for each PCF file and each process has its own store buffer for a file. As a result, SP2 can do a STORE at a higher level and a RESTORE at a lower level whereas APPX cannot. The conversion utility does not flag this condition. Run the APPX Software, Inc.UTL CS utility "APPX STMT 'STORE/RESTORE' List" to review these commands.

8.36 **Action**/Difference/Conversion EH (**all WRITE commands flagged in V05 but more action needed**): When setting or comparing a date field to a non-date field or a non-date field to a date field, SP2 and APPX may get different results. This is because APPX forces date components to have valid values. For instance, APPX forces month value 00 to 01 and month values 13-99 to 12; day value 00 to 01 and day values 29-99 to 28-31 (depending upon the month); hour values 24-99 to 23;

minute values 60-99 to 59; and second values 60-99 to 59. If the date/time field is being used to make a key to a file unique, this can result in records not being written in APPX.

For example, Citrus World sets an hh-ss date key field from a 6 digit number field being incremented by 1. This works in SP2 since SP2 allows the time to be set from 00:00:00 thru 99:99:99 but in APPX this produces duplicate keys due to capping hours, minutes, and seconds. In another example, if an alpha field is set from a hh-mm date field, SP2 gets "00000000hhmm0000" whereas APPX gets "00000101hhmm0000".

Since the conversion program only flags one potential trouble spot and does not flag this problem directly, further investigation is required. To list all date to non-date commands, run the APPX Software, Inc.UTL CS utility "APPX STMT 'Format 5' List" and enter DATE for basic field type 1 and NONDATE for basic field type 2. To list all non-date to date commands, enter NONDATE for basic field type 1 and DATE for basic field type 2. The APPX Software, Inc.UTL CS utility "APPX FIELD File List" will list all date fields by entering "DATE" as 'Basic Field Type'. To see selected fields, just enter the field names as record selections in the sort. (See item 8.33 too.)

8.37 Difference (**flagged**): The conversion utility flags use of the CNV ZONE command in SP2 since there is no counterpart in APPX. However, the equivalent functionality can be programmed. See the subroutine processes "*EX* CNV ZONE ALPHA TO NUMERIC" and "*EX* CNV ZONE NUMERIC TO ALPHA" in application UTL CS for examples of the necessary code.

8.38 **Action**/Difference: When a record has been read with hold, SP2 preserves the hold after a WRITE to the same file has returned a False condition whereas APPX loses the hold. This means the following code works in SP2 but does not in APPX (regardless of whether the file is the PCF file or not):

```
         READ     aaa file1     HOLD 1    FAIL 0    KEY IS  file1 key
     ... various commands without T/F flags ...
         WRITE    aaa file1               FAIL 0
   F     REWRITE  aaa file1               FAIL 0
```

One solution would be to modify the code following the WRITE as follows:

```
   F     STORE    aaa file1                         RECORD
   F     READ     aaa file1     HOLD 1    FAIL 0    KEY IS  file1 key
   FT    RESTORE  aaa file1                         RECORD
   FT    REWRITE  aaa file1               FAIL 0
```

Since the conversion log does not flag this condition, run the APPX Software, Inc.UTL CS utility "APPX STMT I/O List" and investigate any code where the same file has both WRITE and REWRITE commands in the same event point. (See item 2.29.)

SECTION 9: Input Processes

9.1 Bug (**fixed** in APPX 3.0 Beta): In APPX 1.8.4, fields with Modifiable flag blank are never modifiable. They are supposed to be modifiable in Initial Setup or Recovery Processing but not in Live Operations. Fields with Type 4 (Setup) in SP2 are converted with the Modifiable flag set to blank and thus are not modifiable. If this bug has not been fixed, add commands to control input and set 'Modifiable?' to "Y".

9.2 Bug (**flagged in V05**): In APPX 1.8.4 (thru APPX 3.0.0), if the Modifiable flag for an alternate key field is blank, the field is not open for input during key entry. Key entry can be allowed by setting this flag to "N". ***(Warning: In APPX 1.8.4, primary keys with Modifiable? "N" are not open for input in key entry but they are in 3.0.)*** The V05 conversion log will flag SP2 screen items with type="4" (Setup) which may be converted with Modifiable flag blank. If the field is an alternate key, set Modifiable flag to "N" in APPX.

9.3 Bug (**fixed** in APPX 3.0 Beta): In APPX 1.8.4, commands referencing screen items with appearance numbers do not always work properly. For instance, OK INPUT and NO INPUT do not work right. For some operations the problem is caused by APPX always using appearance #1. Sometimes the use of a constant other than 1 as appearance number causes a **compile error** (for example, DISPLAY on appearance #2). (See item 8.25 & 10.3.)

9.4 Bug (FYI): In APPX 1.8.4, *Pre-Display* is not executed upon return from invoking an optional or automatic child process on the same file as the PCF. (*Verify*, *Pre-PCF Update*, and *Post PCF Update* event points are executed before invoking the optional child and *Post PCF Read** and *Post Child Completion* are executed upon return from the child.) This means that commands must be used in *Post Child Completion* to force redisplay of any common fields. **This bug does not affect conversions.**

*In INQUIRE mode in APPX 1.8.4, *Post PCF Read* is not executed upon return from the child and the record is not reread (whereas they are in CHANGE mode). **This bug also does not affect conversions.**

9.5 Bug (**fixed** in APPX 3.0 Beta): In APPX 1.8.4 in a two-level input, *Pre-Display* (i.e., *Start of Image*) is executed once for each entry level in DATA ADDITION at the higher level and ENTRY LEVEL is incremented properly. However, Pre-Display is executed only once in DATA ADDITION at the lower level ***and ENTRY LEVEL is always 1. Warning: Since ENTRY LEVEL does not work at the lower level, it may be necessary to define all fields at entry level 1 and use code to control processing (see items 8.20, 9.9, 9.10, and 9.11 too).***

9.6 Bug (**all lower level processes flagged in V05**): In APPX 1.8.4 (thru APPX 3.0.0) in a two-level input, the PDF fields ERRORS and WARNINGS are not reset to zero when dropping to the lower level. This can cause strange behavior on the first line item when an error/warning was encountered on the header record. To avoid this, add the following code to *Start of Process* for the automatic child process:

```
SET      --- ERRORS                   =
SET      --- WARNINGS                 =
```

9.7 Bug (FYI): In APPX 1.8.4, if a child input process is constrained on primary key to a single record and it uses a scrolling frame, APPX returns the next record (not the record requested). **This bug does not affect conversions.**

9.8 Difference (**secondary entry 'type 2' fields flagged in V05**): In the **lower level process** of a multi-level function, APPX 3.0 does not honor image commands for a field (such as BLANK, BLINK, BRIGHT, DIM, DISPLAY, NO INPUT, OK INPUT, POSITION) until ENTRY LEVEL reaches the entry level of the field. For instance, contrary to SP2, APPX will not honor a DISPLAY on an entry level 2 field when executing *Pre-Display/Verify* after initial entry. ENTRY LEVEL does not become 2 until just prior to executing *Pre-Display* for secondary entry. However, if the entry level 2 field has Auto-Display '1' and a value is set during *Pre-Display/Verify*, it is displayed when entry level reaches 2 unless BLANK'ed in *Pre-Display* for that level. **Caution: If entry level 2 fields exist, APPX treats non-PCF fields as entry level 2 even though they are defined as entry level 1.** Since PCF fields are converted to have Auto-Display '1' but non-PCF fields are converted to have Auto-Display '0', it takes an extra RETURN for values set into non-PCF fields to be displayed in a lower level process. **If the non-PCF field is non-modifiable**, add code to Pre-Display to display the field IF --- INTERACTIVE PHASE EQ DATA ADDITION AND --- ENTRY LEVEL GE 2. Note that Auto-Display is normally changed to '1' for modifiable non-PCF fields per item 9.48 so they are OK. However, do not change Auto-Display to '1' for non-modifiable non-PCF fields since this causes other differences in behavior. (See items 9.14, 9.22, 9.27, 9.46, & 9.48 too.)

9.9 Bug?/Feature (**all 'repeating' windows flagged in V05**): APPX 2.3 does not reset ENTRY LEVEL to 1 for a **repeating frame** when the frame repeats in DATA ADDITION. (See items 8.20, 9.10, and 9.11 too.)

9.10 Difference (**flagged in V05**): ENTRY LEVEL in APPX pertains only to DATA ADDITION and is undefined and hence unreliable in DATA MODIFICATION, unlike SP2 where (ENTRY) was always 2 in CHANGE mode. To check for the equivalent of SP2 secondary entry, code as follows in APPX: (Addressed by item 8.20; see items 9.9 and 9.11 too).

```
IF       --- ENTRY LEVEL                GE     2
AND      --- INTERACTIVE PHASE          EQ     DATA ADDITION
OR       --- INTERACTIVE PHASE          EQ     DATA MODIFICATION
OR       --- INTERACTIVE PHASE          EQ     DATA INQUIRE
```

9.11 Difference (**all 'repeating' windows flagged in V05**): In ADD mode APPX executes *Select Image* and *Pre-Display* once for each ENTRY LEVEL whereas SP2 just executes them once. In an attempt to provide compatibility, the conversion program inserts the following code at the beginning of each routine: (See items 8.20, 9.9, and 9.10.)

```
      IF       --- INTERACTIVE PHASE       EQ       DATA ADDITION
      AND      --- ENTRY LEVEL             GT       1
   T  END
```

For a repeating frame, however, this code can cause problems given the (item 9.9) behavior of APPX 2.3 which does not reset ENTRY LEVEL to 1 when the frame repeats. This means you would exit before reaching any code to change or reset ALTERNATE IMAGE NUMBER, possibly causing an endless loop. It also means only the first frame execution would have initial entry. (APPX 2.3 does retain the ALTERNATE IMAGE NUMBER so there is no harm done by exiting for a non-repeating frame.)

A more remote problem relates to the unreliability of ENTRY LEVEL in DATA MODIFICATION. Since APPX executes once per ENTRY LEVEL, these routines could be re-executed in DATA MODIFICATION if ENTRY LEVEL happened to get changed. Code like 8.20 if needed to protect against that.

9.12 **Action**/Difference (**all lower level processes flagged in V05; combined event points partially flagged in V05**): In APPX there is only one set of process level event points (*Set Run-Time Defaults*, *Post PCF Read*, *Pre-PCF Update,* and *Post PCF Update*) whereas in SP2 there may be multiple event points for each depending upon the number and nature of windows defined for the current level file. Remember, in SP2, *Post Load* executes on the first standard window of a file level whereas *Pre-Save* and *Post Save* execute on the last standard window of a file level. (See item 9.42 for separate problem regarding *Set Default Values*.)

***Warning**: The conversion program combines this code together into the corresponding event point in APPX; such code should be carefully reviewed in multi-frame processes. The conversion program may not flag code that was combined when one event point had no code even though the code may behave differently.*

For example, in SP2, if a multi-level function has header window(s), line item window(s), and trailer window(s), the equivalent set of process level event points exists separately for header, line item, and footer windows. In APPX, there is only one set for the higher level file which combines code from the SP2 header and footer windows and a line item set in a different process for the lower level file (the automatic child).

To adjust for this situation, determine if there is code that needs to be conditionally executed in any of these event points. If so, create another process for the higher level file and call it as an automatic child (preferred method). Note that MODE will never be ADD and INTERACTIVE PHASE will never be DATA ADDITION for the child so beware of SP2 code testing (MODE)! Also, beware of any code setting PCF values in *Post PCF Read* at the higher level since this event point will be executed 3 times, once at the higher level and once when returning from each child. If PCF values are changed, the record will be rewritten. (See item 9.44 too.)

Another less preferred method (assuming there is just one automatic child and no other complicating factors) is do the following:

```
      Create a one-digit numeric work field named WORK PROC LEVEL.
      In Start of Process for parent,
         SET      aaa WORK PROC LEVEL            =      0
      In Start of Frame for first header frame for parent,
         SET      aaa WORK PROC LEVEL            =      0
      In Start of Frame for first footer frame for parent,
         SET      aaa WORK PROC LEVEL            =      2
      In Set Run-Time Defaults for parent, test for footer frame by
         IF       aaa WORK PROC LEVEL            EQ     1
      In Post PCF Read for parent, test for footer frame by
         IF       aaa WORK PROC LEVEL            EQ     1
      In Pre-PCF Update for parent, test for footer frame by
         IF       aaa WORK PROC LEVEL            EQ     2
      In Post PCF Update for parent, test for footer frame by
         IF       aaa WORK PROC LEVEL            EQ     2
      In End of Process for automatic child
         SET      aaa WORK PROC LEVEL            =      1
         IF       --- OPTION                     EQ     PREVIOUS ACTIVITY
   T     SET      aaa WORK PROC LEVEL            =      0
```

9.13 Difference (**fixed in APPX 3.0 Beta**): SP2 allows input during key entry on display type fields (this technique is often used in SP2 for change-only inputs to protect the primary key). For APPX 1.8.4 to allow input, a primary key must have Modifiable? "Y" although an alternate key can have Modifiable? "Y" or "N". Add NO INPUT commands to protect the primary key and change the flag to "Y".

9.14 Difference (**flagged in V05**): APPX differs from SP2 in regard to image commands in non-image event points. In APPX, these commands (BLANK, BLINK, BRIGHT, DIM, DISPLAY, NO INPUT, OK INPUT, POSITION) have no effect on the image and must be moved to image event points to behave like SP2. (APPX image event points are *Pre-Display, Pre-Display/Verify, Option Intercept,* and *Verify*). Such **commands** are flagged on the **Commands conversion log**. **Warning: When moving commands from *Post PCF Read* to *Pre-Display*, commands should be conditional on INTERACTIVE PHASE NE DATA ADDITION.** Normally, DISPLAY commands in *Set Run-Time Defaults* and *Post PCF Read* need only be deleted **as long as the item has Auto-Display '1'** (Current Value) on the image. Non-PCF fields are converted to have Auto-Display '0'=None. See comments in item 9.48 before changing any non-PCF fields to Auto-Display '1'. (See items 9.8, 9.25, 9.46, & 9.48.)

9.15 Difference (**flagged in V05**): APPX handles the cursor option of the POSITION command differently than SP2. For example, the placement of the cursor will be one position to the left on the screen in SP2 than it will be in APPX for the same CURSOR COLUMN. (See item 8.18.)

9.16 Difference (**flagged in V05**): APPX differs from SP2 in regard to exiting from a lower level. SP2 will exit the input function upon use of PF16 from a lower level whereas APPX will return to the higher level upon use of END from a lower level. Both PF13 *Next Section* and PF16 *End* in SP2 were converted to END in APPX. The conversion program flags the use of PF13 and PF16 for inspection. Check usage to see if this difference matters. (See item 2.14.)

For standard TOM transaction entry programs, modify code in *Option Intercept* of the detail image for the lower level process to read as follows:

```
        IF       --- OPTION                     EQ     PREVIOUS ACTIVITY
   T    ERROR    Program Function Key Not Allowed
        IF       --- OPTION                     EQ     END
   T    SET      --- OPTION                     =      CANCEL
```

9.17 Difference (**all one-record input processes flagged in V05**): APPX differs from SP2 in regard to exiting from input on a one-record **job file**. SP2 would not allow PF16; the operator had to press RETURN or use PF32 to cancel the job. APPX allows the use of END and will continue the job even though the record was not saved. For one-record **job files (**not one-record permanent files), insert the following code in *Option Intercept* for the image**)**:

```
        IF       --- OPTION                     EQ     END
   T    SET      --- OPTION                     =      CANCEL
```

9.18 Difference (**all sequence numbers and/or auto-incrementing flagged in V05**): APPX 1.8.4 differs from SP2 in regard to the handling of auto-sequence numbers for a key field. In SP2, the new sequence number is available in *Set Default Values*. In APPX, it has not yet been assigned and thus is not available in *Set Run-Time Defaults*. This can cause problems if code exists which tests for specific values or which modifies the sequence number. Move the appropriate code to *Pre-Display* but remember to test for INTERACTIVE PHASE and ENTRY LEVEL. To list items with automatic sequencing, run the APPX Software, Inc.UTL CS utility "APPX ITEM File List" and enter "Y" to both 'Print DV?' and 'Check for Auto-Sequencing?'.

In the case of the standard TOM applications, this situation also causes a problem. Most transaction entry programs use auto-sequencing on trx number but also default the trx number from a higher level file in *Set Default Values*. In APPX the auto-incrementing takes place after *Set Run-Time Defaults* is executed and ignores the programmatic defaulting. To correct the problem:

In the Data Dictionary:
Set the field Default Value to 0001 for the higher level file
(e.g., CPA TIMETRX1 NEXT TRX NO)

In the input process for the lower level file:
Remove the field Auto-Increment factor, set Save As Default? "N", and set Override DD? "N" for the trx number field on the input screen
(e.g., CPA TIMETRX2 TRANS NO)
Modify the code in *Set Run-Time Defaults* similar to the following:

```
        IF       --- INTERACTIVE PHASE          EQ     DATA ADDITION
  [T    SET      CPA TIMETRX2 TRANS NO          =  CPA TIMETRX1 NEXT TRX NO]
```

9.19 Difference (FYI): In SP2, WARNING/ERROR messages are not issued in CHANGE mode until the operator presses RETURN on a screen. In APPX 1.8.4, they are presented with the initial display of the screen. This can sometimes result in bogus messages. To avoid this, move the code to the *Verify* event point.

9.20 Difference (FYI): APPX may not always position the cursor in the same manner as SP2 at runtime. For example, SP2 positions the cursor on the field for the first ERROR condition encountered; APPX 1.8.4 positions it for the last ERROR.

Also, if a key field is key protected, SP2 automatically positions the cursor on the next field; APPX 3.0.0 positions the cursor on the key field which is bright even though it is not modifiable (any characters entered overwrite the next field).

9.21 Difference (**all display-only windows flagged in V05**): APPX may differ from SP2 in the behavior of a display-only window. APPX 2.3 does not honor setting OPTION in *Pre-Display* of a display-only window. Move this code to *Pre-Display* of the detail frame.

Also, SP2áexecutes *Access/Verify* and *Pre-Display* in all modes; APPX 2.3 executes *Pre-Display/Verify* and *Pre-Display* in all modes except it does not execute *Pre-Display/Verify* in ADD mode (this may or may not be **fixed** in APPX 3.0). APPX 3.0 does not execute Pre-Display/Verify (or Verify) in interactive phase DATA ADDITION.

9.22 Difference (**all lower level processes flagged in V05**): APPX 1.8.4 does not always display higher level fields that were automatically displayed in SP2. For example, it may be necessary to change entry level to 0 in order to automatically display higher level fields in a lower level display-only or detail frame. Remember entry level 0 denotes that an item is always displayable; if it is modifiable, it is always editable too. (See items 9.8, 9.14, 9.21, & 9.27 too.)

9.23 Bug (somewhat **fixed** in APPX 3.0.0) (**flagged in V05**): SP2 allows repeating alternate windows to have different sizes. In APPX 1.8.4 alternate images must have the same size for repeating frames. In any case, scrolling alternates behave differently in APPX when they are different sizes. (See items 1.63 & 10.26.)

9.24 Difference: APPX presents zeros in a numeric field at initial entry (unless 'Blank if Zero?'=Y") whereas SP2 always presents blanks. This is **not practical** to address. However, in the event it is necessary to emulate SP2, set 'Blank if Zero?'="Y" if possible; otherwise, add code in *Pre-Display* such as:

```
         IF       --- INTERACTIVE PHASE       EQ       DATA ADDITION
         AND      --- ENTRY LEVEL             EQ       1
         AND      aaa numeric field           EQ
   T     BLANK    aaa numeric field
```

9.25 Difference (**flagged in V05**): In SP2, OK INPUT can be used to make a work field defined as Type 3 (Display) modifiable. In APPX, OK INPUT only works on a work field which is modifiable and a display-only field is converted to Modifiable? "N". (Both SP2 and APPX will not honor OK INPUT on a PCF field which is not modifiable.) (See items 9.8, 9.14, & 9.46 too.)

9.26 Bug/Difference (**flagged in V05**): APPX 2.3 (thru 3.0.0) differs from SP2 in its handling of ALTERNATE IMAGE NUMBER. (See item 8.23.)

9.27 Difference (**flagged in V05**): When there are entry level 2 fields defined on the screen, a user may have to press RETURN an additional time when adding a record in APPX compared to SP2. This should only be a problem in lower level processes. (See items 9.8, 9.14, & 9.22 too.)

Example: In SP2, on the CPA Time Cards trx entry input screen, the account number field is defined as type '2'=Secondary with Auto Access and the account description field is type '3'=Display with Auto-Display '1'=Yes. Then in SP2 Access/Verify there is code to default the account number and display the description, both of which appear on the screen at secondary entry.

In APPX the account description field on the screen is converted to Entry Level '1' with Auto-Display '2'=Lookup. However, the DISPLAY command in *Pre-Display/ Verify* is not honored and only the default account number appears on the screen at secondary entry. Account description only appears after the operator presses RETURN again. To eliminate the extra RETURN, the field must be defined as Auto-Display '0'=None or '1'=Current Value.

Comment: This situation appears to be an example, where entry level 2 fields are defined, of APPX treating a non-PCF field as secondary entry (see item 9.8). Note that no extra RETURN is involved in another situation where all fields are entry level 1 and both account and description are defaulted and displayed in *Pre-Display*; this works fine.

9.28 Conversion Bug: The conversion utility may convert Data Lookup Type wrong (for example, as CONTINUE when it should be DEFINE). This will cause a **compile error**.

9.29 Conversion Bug?/Difference (FYI) (**signed** numeric fields): In SP2 inputs, zero **signed** numeric fields with decimals display with a leading zero (e.g., 0.00) and negative **signed** numeric fields display with a leading sign (e.g., -5). In APPX inputs, unless a mask is defined, such fields display without the leading zero (e.g., .00) and with a trailing sign (e.g., 5-). The conversion utility only generates a mask for a signed field in an input if it is zero-filled. This is normally not a problem. (See items 2.11 & 9.30.)

9.30 Conversion Bug?/Difference (**unsigned** numeric fields): In SP2 inputs, zero **unsigned** numeric fields with decimals display with a leading zero (e.g., 0.00). Negative **unsigned** numeric fields display with a leading sign if there is room (e.g., -5); question marks are displayed only if there is not room for the sign in SP2. However, in APPX inputs, since the conversion utility generates a mask with no sign for an unsigned numeric field (e.g., 90), zero fields will display properly but negative values may cause questions marks in APPX where they would not in SP2. (See items 4.24 & 5.12; see items 2.11 & 9.29 too.)

9.31 Difference (**flagged in V05**): In a SP2 multi-level input, the Initial Access Key for the lower level could be different from the Binding Key without affecting response time. When converted, the SP2 Binding Key becomes the Child PCF Field on the Child Constraint for the Automatic Child and the SP2 Initial Access Key, if specified, becomes either the Default PCF Key in AA for the Automatic Child or the Default PCF Key in the input process created for the automatic child. (Note that the one specified in AA overrides; if none are specified, the primary key is used.) However, if left as converted in APPX, response time may be measured in minutes because APPX must read the entire child PCF file in order to find the records constrained. Since such response will not be acceptable to most users, one alternative is to add an alternate key to the child PCF file which can then be used both in the constraint and as the Default PCF Key.

For example, suppose LEVEL1 file has SSN# as primary key while LEVEL2 file has SSN#/Seq# as primary key and Date/Time Added as an alternate key. If the SP2 input has SSN#/Seq# as the Binding Key with Date/Time Added as the Initial Access Key, add a new alternate key for SSN#/Date/Time Added to LEVEL2 file in APPX. Specify the new key in the constraint and as the Default PCF Key.

9.32 Bug?/Difference (**flagged in V05**): In SP2, setting (PF KEY) is only valid in *Pre-Display* and *PF Key Intercept* but if (PF KEY) is set by commands in *Access/Verify*, it is initially ignored and then cleared so there is no consequence. In APPX, however, if OPTION is set by commands in *Pre-Display/Verify* (which is how it converts), it is initially ignored but not ever cleared so it can ultimately change behavior. If found in *Access/Verify*, delete corresponding code in APPX. (See item 2.14.)

9.33 Bug/Difference: APPX 3.0.0 has some difficulty handling ERROR commands in *Post PCF Read* compared to SP2 with ERROR commands in *Post Load*. In SP2 the normal screen is displayed but with no data other than the key entered and the ERROR message. In APPX the key entry screen is displayed with the ERROR message but without the key value. If it happens on a scrolling record, the APPX screen does not display properly. This condition is **not practical** to address.

Note that WARNING commands are handled properly. If a WARNING is encountered, the record is returned with the WARNING message displayed; the operator may then press RETURN to go on to the next screen, if any.

9.34 Difference (**flagged in V05**): Since "PF" terminology is not applicable in APPX, modify "PF" text as appropriate. Other SP2 terminology (SPEED and EXPLAIN) is also flagged in V05 after 08/27/96.

9.35 Difference (**flagged in V05**): SP2 uses PF keys while APPX uses OPTION or other named keys. The conversion utility converts (PF KEY) references to OPTION and flags most situations that are in doubt. However, it is wise to check (PF KEY) conversion in *Option Intercept* since the log may not catch everything. The V05 conversion log warns to inspect whenever code exists in 'PF Key Intercept' command series. (See item 2.14.)

For example, if the SP2 code was checking for (PF KEY) GT 24, modify code in *Option Intercept* to read as follows:

```
         IF       --- OPTION                    EQ     DIRECT PROCESS 1
         OR       --- OPTION                    EQ     DIRECT PROCESS 2
         OR       --- OPTION                    EQ     HELP ITEM
         OR       --- OPTION                    EQ     PRINT SCREEN
         OR       --- OPTION                    EQ     CANCEL
```

9.36 Difference (**flagged in V05**): SP2 allows 79 characters for Default Value and Edit Mask whereas APPX only allows 60 characters. If necessary, add commands to edit the field in input processes allowing field entry. Also, if needed, add commands to set default values longer than 60 bytes. (See item 2.16.)

9.37 Difference (**all 'Set Default Value' command series flagged in V05**): SP2 and APPX handle default values differently in some instances. In SP2, a SET command in *Set Default Values* will modify the default record buffer whereas a SET command in *Set Run-Time Defaults* in APPX will not. This may cause different behavior if the SET is conditional and there is no 'Save as Default'.

Also, SP2 will preserve default values modified at a lower level via 'Save as Default' across higher levels whereas APPX will not. For example, in APPX, defaults established via 'Save as Default' for order line items are honored for the current order but are reset from the Data Dictionary when the lower level line item process is called for the next order. (This is due to the nature of SP2 and APPX; SP2 handles higher/lower level file processing within a single function while APPX has a separate process for each file.)

9.38 Difference (**all 'Set Default Value' command series flagged in V05**): In SP2 you may RESTORE the entire record in *Set Default Values* but only DISPLAY selected fields for default purposes. In APPX, you should only RESTORE individual fields as needed.

9.39 Bug (**all scrolling windows flagged in V05**): In APPX 3.0.0, at runtime in a scrolling first-level file maintenance function with 'Read-First?'="N", if you switch from INQUIRE mode to DELETE, you will get "Error - No Record Currently Held" after pressing ACKNOWLEDGE DELETE. You can avoid this error by setting 'Read-First?'="Y".

To emulate SP2 behavior, the conversion utility sets 'Read-First?' to "N" on first-level functions and to "Y" on lower-level functions.

9.40 Bug (**flagged in V05**): Mod Log does not work in APPX if the input process is invoked as RELATED. Inputs are normally invoked as RELATED from a job step but invoked as DETACHED from a menu item. A good work-around is to run the input from a menu as a DETACHED process or from the job as SUBPROCESS or DETACHED. By the way, Mod Log does work when a default input process is invoked by a user during SCAN. The **conversion log** flags the use of Mod Log.

9.41 Difference: SP2 allows higher level components of the key to be modifiable at a lower level (although it is not wise to do so). APPX does not since constraints by definition cannot be modified.

9.42 Difference/Conversion EH (**all 'Set Default Value' command series flagged in V05**): In SP2 there can be a different *Set Default Values* command series executed for each alternate window whereas in APPX there cannot. The *Set Run-Time Defaults* event point is defined at the frame level in APPX. The conversion log attempts to flag combined event points but it does not catch them all. Currently, it does not directly flag *Set Default Value* commands found on alternate windows > 1. The conversion utility will combine the code but the code must be modified to make it conditional based on the conditions used in the SP2 *Set Alternate No.* command series.

9.43 Difference (FYI): In CHANGE mode, screen edits do not get executed in SP2 unless something is changed. They get executed in APPX even if nothing is changed.

9.44 Bug: APPX 3.0 does not always initialize INCLUDE CHILDREN at 1. If any automatic child of an input is optional, add code to set INCLUDE CHILDREN to 1 at the beginning of Pre-Invocation for each child. Consider this when converting a SP2 footer window to be another automatic child. (See item 9.12.)

9.45 Difference (**compile error**): A valid SP2 auto-access can cause a compile error ("Invalid Data Lookup specifications - Source Missing" in APPX 3.0 and 3.1. It seems to occur when the number of key segments in the DLU key exceeds the number of key segments in the file key. This can affect input, output, and inquiry processes. The only work-around is to remove the DLU from the image and program the equivalent functionality as detailed in the following paragraphs.

For an overview of DLU items, run the APPX Software, Inc.UTL CS utility "APPX ITEM File List" and enter 'Check for DLU?'="Y". To see lookup items too, also enter 'Lookup?'="Y" (note that the checks for DLU or Lookup are the only checks combined with an OR; the rest use AND). To list items for a single process, enter the process name. (See items 1.52 & 10.28.)

For input processes, do the following (see item 10.28 for output/inquiry processes):

1) For all components of the key on the image, change the Data Lookup Type to NONE. Be sure to make note of DLU specifications (whether the file must exist, action if NOF, etc.) and the entry level for the key.
2) For all associated lookup fields on the image, change the Auto-Display to 0 (None) and make note of lookup value (null or ?'s) if NOF.
3) Add code in *Pre-Display/Verify* to do the DLU read (consider entry level), issue error/warning and set lookup values as appropriate on F condition, and display lookup fields.
4) Add code in *Option Intercept* as follows:
```
         IF       --- OPTION                     EQ     SCAN
   T     AT FIELD <each component of key>
   TT    SCAN     <each component of key or group header>
```
   Repeat the AT FIELD/SCAN code for each component of the key on the image.

Note that in SP2, if a user uses PF2 Find Key to retrieve a multi-part key defined with Auto Access, all components of the key are returned. Also, both Auto Access and READ appear to leave the record buffer in the same status on an unsuccessful read (the key fields retain their value from the read while non-key fields have the same values they had prior to the read).

9.46 **Action**?/Difference/Conversion EH?: In SP2, executing OK INPUT or NO INPUT (and possibly other image commands as well) would force the screen to be re-displayed even though nothing else had changed. In APPX, it doesn't. The conversion log does not flag image commands found in valid image event points. To list such commands, run the APPX Software, Inc.UTL CS utility "APPX STMT 'Format 6' List" and enter the desired op codes. Additionally, record selections can be entered to select certain event points. Image event points are 41-44 (see Section 16 for complete list of event points). (See items 9.8, 9.14, & 9.25 too.)

It may not be practical to address this problem until it is reported by users. However, one solution is to put the OK INPUT/NO INPUT in *Option Intercept* and set --- OPTION to blanks. This will force the image to be redisplayed immediately without executing other event points (similar to the SP2 technique of setting PF KEY to some unused value such as 17, etc.).

9.47 Bug?/Difference (**all 'Set Default Values' command series flagged in V05**): APPX may not behave like SP2 if a value is set into a key field in *Set Run-Time Defaults*. For example, if SP2 *Set Default Values* executes a SET DATE on a date/time stamp key field with no DISPLAY command and with 'Save as Default?'="N", the default is displayed in ADD mode but not in key entry. In APPX the default is always displayed. To emulate SP2, add code to only set the default if INTERACTIVE PHASE EQ DATA ADDITION. Note that this problem may affect any type of key field.

9.48 **Action**/Bug/Difference/Conversion EH? (**modifiable non-PCF fields flagged in V05 after 08/27/96**): The conversion utility converts non-PCF fields as Auto-Display '0' but this may cause APPX to behave differently than SP2 when the non-PCF field is modifiable. For modifiable non-PCF fields, change Auto-Display to '1' and add code in *Set Run-Time Defaults* to clear any of these fields that have no default value and 'Save as Default?'="N". Do not clear modifiable non-PCF fields that have a default value or that have 'Save as Default?'="Y". **CAUTION: Make note to test the function if the modifiable non-PCF field is used as the default key field for a consecutive file (same as initial access key in SP2); this behavior is unknown.** (See item 9.8.)

This fix has one known difference for modifiable non-PCF fields not being specifically set in *Set Run-Time Defaults*. If the modifiable non-PCF field is being changed in *Option Intercept* and there is no DISPLAY command, neither SP2 nor APPX show the changed value when the screen is redisplayed but APPX does use the changed value as the default value for subsequent records whereas SP2 does not. If there is a DISPLAY command, both SP2 and APPX redisplay the screen with the changed value. The conversion log flags *PF Key Intercept* command series.

Changing Auto-Display to '1' will also overcome an APPX bug for modifiable non-PCF date fields. If a modifiable non-PCF date field has Auto-Display '0' and there is no DISPLAY command, APPX presents blanks in the display rather than the normal date mask and will not accept input for the date (it gives an invalid date error even though the date entered is valid).

To list modifiable non-PCF fields, run the APPX Software, Inc.UTL CS utility "APPX ITEM File List" and enter 'Check for Modifiable?'="Y", 'Check for Non-PCF Item?'="Y", and 'Print DV?'="Y". If non-standard field names are used and you want to identify the files they belong to, run the APPX Software, Inc.UTL CS utility "APPX FIELD File Col Hdg List".

9.49 Difference (**all sequence numbers and/or auto-incrementing flagged in V05**) (**compile error**): SP2 allows a non-key field to be defined as a sequence number with an increment value (contrary to what the manual says) although it probably ignores it. APPX gets a compile error if a non-key field has an auto-increment value. Remove the auto-increment value and flag the process.

SECTION 10: Output Processes ***(Check for APPX Inquiry Process!*** **See item 13.2.)**

10.1 Bug (**flagged in V05**): In APPX 1.8.4 (thru APPX 3.0.0), if a field with Read Security is defined on an output or inquiry image and a DISPLAY command is given on that field, a user without the required security rights will get APPX system error RMUPD.C.156 ("upd_chr - characteristic doesn't fit on image"). (See items 4.7 & 5.1.)

10.2 Bug (**all windows with non-zero Minimum Extra Lines flagged in V05**): APPX 2.x has a known bug in the use of Minimum Extra Lines which can result in an output running forever if the value of this field is too large. When SP2 encounters a Minimum Extra Line condition, it does a page eject and then prints the window. APPX does a page eject and then rechecks the Minimum Extra Line condition. If large enough, APPX will continue to page eject and print page headings. To list images with extra lines, run APPX Software, Inc.UTL CS utility "APPX IMAGE File List" and enter "Y" to check for extra lines.

Also, there may be other problems associated with the use of Minimum Extra Lines such as causing blank lines to print. In APPX 1.8.4, extra lines are incorrectly output if an alternate image with extra lines has 'CR/LF?'="N" at the frame level (same as 'Start Print Beside?'="Y" in frame AA in APPX 3.0.0). (See item 10.20.)

10.3 Bug (**fixed** in APPX 3.0 Beta): In APPX 1.8.4, screen items with appearance numbers do not always work properly. Sometimes the problem is caused by APPX always using appearance #1 for some operations. Sometimes the use of a constant other than 1 as appearance number causes a **compile error** (for example, DISPLAY on appearance #2). (See items 8.25 & 9.3.)

10.4 Bug (**flagged in V05**): In APPX 3.0, if printing standard column headings and the first field is a right-justified numeric field or a format type field with right alignment, the column headings will not print properly if the width of the column heading exceeds the width of the field and there is not sufficient space to the left on the line. The **conversion log** flags to inspect column headings if the first field on any row is RJ or RA and the report uses standard column headings.

To easily inspect column headings for all the fields identified in such messages on the log, run the APPX Software, Inc.UTL CS utility "APPX FIELD File Col Hdg List" and enter the field names as record selections. The column heading if any will print in the format column below the format (and below the descriptive if 'Print Descriptive?' set to "Y"). Remember, nothing prints in the standard column heading for a field if it has no column headings; the description is not used. If the column headings are wider than the field format, check the output to see if there is sufficient space to the left or if the field is on a row which will not be used to generate the heading. If feasible, move fields to make room for the heading (but consider alignment with other images) or shorten the column heading if appropriate. If all else fails, run the "APPX FIELD File Col Hdg List" naming the fields that make up the standard column heading and use it to add a corresponding PAGE-START frame to the output (be sure to change 'Standard Column Heading?' to "N").

10.5 **Action**/Difference: APPX requires a higher level PCF record be read with hold before being rewritten or deleted at a lower level whereas SP2 does not. This will cause an error at runtime. Check APPX ILF X-Ref for use of REWRITE or DELETE in lower level output and investigate. You can also use the APPX Software, Inc.UTL CS utility "APPX STMT I/O List" to review REWRITE/DELETE commands. (See item 11.2.)

10.6 Bug/Difference/Conversion EH (**flagged in V05**): SP2 supports the use of Auto-Total on lower level files whereas APPX does not. In APPX you must use a work field to TOTAL into at the lower level and then add RANGE-END frames using Auto-Display '4' values at the Sort File level. Also, be sure to remove auto-totaling from the image on the lower level process; otherwise, the function can misbehave in APPX. The **conversion log** flags the use of Auto Total. To list items with auto-totaling, run the APPX Software, Inc.UTL CS utility "APPX ITEM File List" and enter 'Check for Auto-Total?'="Y".

Note that it can be a little tricky to emulate the SP2 use of Auto-Total since auto-totals only appear on generated subtotal or grand total windows. You have to look at the sort function in combination with the output and even then you may not know what will happen at runtime. If subtotal and grand total windows are defined in the output for the default sort specifications, it may be that the SP2 auto-totaling was defined in error and is not even being used.

Both SP2 and APPX ignore any auto-totaling defined on subtotal windows. However, since this flag is only accessible or visible on APPX 3.0 RANGE-END images by using INQUIRE to list all image items, it would probably be a good idea if the conversion utility set it to "N".

10.7 Difference (**fixed in V05**): Outputs designed for the screen were limited to 79 columns in SP2. Change these to 80 columns in APPX to avoid possible editing errors.

10.8 Difference: APPX 1.8.4 inserts a blank line after a standard page heading whereas SP2 does not.

10.9 Difference (FYI): APPX differs from SP2 in handling an output which has no detail. APPX displays a "No Lines Were Output" message if printing to the screen; otherwise, nothing happens. SP2 prints a report with zero grand totals if applicable.

10.10 **Action**: Any outputs using special forms should be thoroughly reviewed and tested. They will usually require work to behave the same as SP2. To find outputs using preprinted forms, check SP2 Table of Contents for special dispositions and then refer to SP2 Disposition X-Ref to see where these are used. Also refer to the installation's SP2 Forms List from Installation Control (see item 2.8). Mark special form outputs on SP2 Table of Contents. (See item 10.21 too.)

For example, in APPX 3.0.0, the following changes were required to print checks in MAS SE:

```
      Change Footer Height in process AA from 2 to 3
         (to compensate for APPX blank line).
      Change Skip Lines Before from 4 to 3 on PAGE-START image
         (to allow for same alignment).
          Note: If the APPX form accommodated more than one program form,
             you would need to move the blank line to the bottom of the
             program form, not just remove it from the top (for example,
             where 2 33-line W-2 program forms print on a 66-line APPX form).
```

In APPX 1.8.4, the following changes were required to print CAP A/P checks:

```
      Set Footer Height to 25 (3 more than APPX footer size, allowing for line
         at beginning of form, line before footer, and dummy? line suppressed).
      Change Skip Lines Before from 1 to 0 on 1st record frame.
      Set 'Start New Page After?'="Y" on last record frame.
      Change Skip Lines Before from 1 to 0 on End of Page frame.
```

10.11 Bug/Difference: APPX 2.3 (thru 3.0.0) differs from SP2 in its handling of ALTERNATE IMAGE NUMBER. (Addressed by item 8.23.)

10.12 Difference: APPX differs from SP2 in regard to image commands in non-image event points. In APPX, these commands (BLANK, DISPLAY, etc.) have no effect on the image and must be moved to image event points to behave like SP2 or, if feasible, Auto Display type should be set to '1'. Note that non-PCF fields are converted to Auto Display '0' (None). (The only output/inquiry image event point is *Pre-Display*.) Such **commands** are flagged on the **Commands conversion log**.

10.13 Difference (FYI): The option to provide for "Continued" subheading messages has not been implemented in APPX.

10.14 Difference: If the binding key is not a valid key, SP2 ignores it (and uses the primary key if there is no sort) whereas APPX gets a **compile error** for invalid PCF key. Change it to be the primary key.

10.15 Conversion Bug?/Difference (FYI) (**signed** numeric fields): In SP2 outputs, zero **signed** numeric fields with decimals display with a leading zero (e.g., 0.00) and negative **signed** numeric fields display with a leading or trailing sign or special sign depending upon Sign Type. In APPX outputs, unless a mask is defined, zero fields display without the leading zero (e.g., .00) and negative fields display according to the internal display mask. The V05 conversion utility always generates a mask for output/inquiry processes so this is normally not a problem. (See items 2.11, 4.24, 5.12, & 10.19 too.)

10.16 **Action**/Bug?/Difference: In APPX, bogus blank lines may appear in the output if inappropriate subtotals (or subheadings?) are specified in the associated query but not defined in the output. In SP2, these are ignored. Delete such subtotals (or subheadings?) if not needed; otherwise, define do-nothing range frames in the output (or clone the query without them and change the job to use the clone). (See item 12.16.)

This problem is difficult to find so you may want to limit your review. Outputs/inquiries that are the most vulnerable are those with special forms (per item 10.10) or those that share queries. Remember a query may be used by multiple processes within a job (found by items 7.6, 7.7, 7.18, & 7.20). To find those used by multiple jobs, check the SP2 Sort X-Ref.

10.17 Bug?/Difference: APPX 1.8.4 (thru APPX 3.0.0) prints a blank line as the first line of a 'form', referring to the form as defined in APPX System Administration. This blank line is not visible if output is printed to the screen. The APPX blank line does not normally cause a problem except in the following situations: (see item 10.21 for more details)

1) You need to print on every line of the form. You're in big trouble because it can't be done. The APPX blank line will cause a blank page between forms.
2) You need to align forms on the printer the same way you did in SP2. If there is a blank line at the top of the form, it could be moved to the bottom of the form in APPX and forms would then align the same; otherwise, you're out of luck.

10.18 Note: If you have optional frames and you need to specify 'Skip Lines Before' or 'Skip Lines After', do so on the image level (not the frame level); otherwise, the lines will be skipped causing blank lines even though the image itself is suppressed. (However, see item 10.20 first & also see item 10.21.)

10.19 Conversion Bug (**fixed in V05**)/Difference (**unsigned** numeric fields): In SP2 outputs, zero **unsigned** numeric fields with decimals display with a leading zero (e.g., 0.00). Negative **unsigned** numeric fields display with a sign if there is room; question marks are displayed only if there is not room for the sign in SP2. However, in APPX outputs/inquiries, the standard conversion utility generates a mask with no sign for an unsigned field with SP2 trailing sign but generates a mask with a leading sign (which consumes an extra space) for an unsigned field with SP2 leading sign. This means the leading zero displays properly but a negative value for a SP2 trailing sign may produce question marks while a SP2 leading sign can cause a **compile error** (if items no longer fit on the image) **or** can cause data to be **misaligned**. (See items 2.11, 4.24, 5.12, & 10.15 too.)

The V05 conversion utility always generates a mask with no sign for an unsigned field, thus fixing the conversion bug. To print display masks, run the APPX Software, Inc.UTL CS utility "APPX ITEM File List" and enter "Y" to both 'Check for Display Mask?' and 'Print DM?'.

10.20 Difference (**flagged in V05**): APPX differs from SP2 in its approach to Print Beside which may cause conversion problems or which may prevent the output from looking like what you get in SP2. In SP2, Print Beside is specified by 'Location'="1" (Beside Prev Wndw) and controls where the window will print in relation to the previous window printed. In APPX, Print Beside is actually initiated in the frame before the SP2 Print Beside window by setting 'Start Print Beside?'="Y" in frame AA. The SP2 default is 'Location'="2" (Below Prev Wndw) and the APPX default is 'Start Print Beside?'="N".

One problem in the APPX approach is that it may not be known what frame will actually print at runtime immediately prior to the SP2 Print Beside window. For example, if it is a subheading which may be changed at runtime, it may be desirable to start Print Beside on every subheading if possible (for example, if they start in column 1). Other problems can occur if there are optional or repeating windows preceding the SP2 Print Beside window. Also, in APPX, if the Start Print Beside frame has Skip Before lines, these lines should be specified on the frame, not the image; otherwise, the SP2 Print Beside window will start printing beside the blank lines. (However, see item 10.18 and also see item 10.21.)

The upshot of this is that any output using Print Beside should be thoroughly reviewed and tested. The **conversion log** flags any SP2 window with location 'Beside'. To list processes using Print Beside, run the APPX Software, Inc.UTL CS utility "APPX IMAGE File List" and enter "Y" to 'Check for Print Beside?'.

10.21 Note (**1.8.4 Upgrade Alert**): To understand what causes some differences in output between SP2 and APPX, it might be helpful to review the findings below which are applicable to APPX 3.0.0 (see items 10.2, 10.8, 10.10, 10.16, 10.17, 10.18, 10.20, & 10.31 too):

1) Every APPX form begins with a HEX(0A), or line feed, caused by APPX itself, where 'form' refers to each page of the form as defined in APPX Sys Admin. The APPX blank line is not counted in the APPX line count for the form.
2) The last line of the form ends with a HEX(0C), or form feed, not a HEX(0A), regardless of whether there is text on the line. Where the last line is a blank line, there is only the HEX(0C), not a HEX(0A0C).
3) A frame with 'New Page After?'="Y" and lines specified in Skip Lines After produces a single HEX(0C)for the lines to skip.
4) If the application program accounts for every line of the form and does not end in a blank line, the APPX blank line will cause a blank page to print between forms.
5) If the application program uses a PAGE-END frame (same as SP2 Page Footing), Footer Height in the process AA must be increased by 1 to account for the APPX blank line.
6) APPX 1.8.4 inserts a HEX(0A) upon return from a lower level. **If there is a PAGE-END frame**, Footer Height must be increased by 1 to compensate or a leading blank line in the PAGE-END frame must be dropped. (Note: This is not a problem in APPX 3.0.0 but it is in APPX 1.8.4. **Programs must be changed when upgrading from 1.8.4.**)
7) APPX does not maintain Footer Height automatically if changes are made to PAGE-END frames.
8) Normally, printers do not do another form feed if already at top of form during printing when a HEX(0C) is received.
9) APPX handles its print buffering differently than SP2. In SP2 the print buffer was large enough for SP2 to assemble 2-3 pages if necessary before committing the first page to print; this allowed after-the-fact changes to be made to the first page as warranted. APPX commits to print much earlier in comparison. This may be the reason APPX 3.0 does not properly handle scroll down. (See items 10.20 & 10.31.)

The V05 conversion utility flags the use of SP2 scrolling or repeating windows or any with location 'Beside'. It also flags any alternate windows with different attributes (position, size, & scrolling). The APPX Software, Inc.UTL CS utility "APPX IMAGE File List" can be used to provide an overview of pertinent features for selected processes or for all processes with specific features.

10.22 Difference (FYI): If a subtotal is generated, SP2 prints zeros (for example, 0.00) for a zero total field auto-totaled from a detail field defined as 'Blank if Zero' whereas APPX prints blanks. Both SP2 and APPX print blanks for the detail field.

10.23 Difference (**flagged in V05**): SP2 allows a total-type work field to be used as an ordinary work field in the detail window of an output/update function and sets the value to zero before entering the detail window. There is no such thing as a total-type work field in APPX (see explanation in item 8.13) and APPX does not change the current area of a numeric work field without explicit commands to do so. If code exists in a detail window which relies upon the zero value, the SP2 total-type work field must be SET to zero at the beginning of the event point in APPX. In addition, there are other complications if commands are used on a SP2 total-type work field in a subtotal window (see item 8.13).

The **conversion log** flags the use of commands such as SET, COMPUTE, and RESTORE on SP2 total-type fields. (See items 8.13 & 11.4.)

10.24 Note: SP2 can have a DISPLAY command on a field and then modify it and not display the modified value. APPX 3.0 behaves the same as SP2 (at least in the record frame) even when Auto Display type is '1' (Current Value).

10.25 Difference: APPX differs from SP2 in its opening of files for an output process. APPX 3.0 opens all files referenced by domains or DLU's in the PCF file for the current company. This may cause unexpected problems.

For example, an "**element not found**" error may occur which is not appropriately labeled. Such an error may actually be caused by bad data in a domain DLU in a referenced application but the bad data may not be visible in the Data Dictionary and may not cause a DD processing error. To find bad data, run the APPX Software, Inc.UTL CS utility "APPX DOMAIN File List" to verify domains in all applications. It will print an error message if the element record is not on file for a DLU.

In another example, a file may not exist in the current company (in SP2 it may actually be opened under a different company). In this case, a possible work-around could be to set 'Use Database' to that company in Database Management. (See item 8.14.)

10.26 Difference (**flagged in V05**): Scrolling alternates behave differently in APPX than in SP2 if they have different sizes. To list the frame/image characteristics for alternates, run the APPX Software, Inc.UTL CS utility "APPX IMAGE File List" and enter "Y" to 'Check for Opt/Repeat Frame?'. If desired, enter "OUTPUT" or "INQUIRY" for process type or enter the process name. (See items 1.63 & 9.23.)

10.27 Difference (**flagged in V05**): Expanded print does not work by setting the flag in APPX (this flag can only be accessed by pressing INQUIRE while on the image). However, you can provide for expanded print by imbedding a HEX(0E) in your print line. For example, in the Data Dictionary, define a new work field WORK HEX(0E) as a one-byte alpha field; then, in your output function, add this field with Auto-Display '1' somewhere on the print line containing the field flagged for expanded print and add the following code to *Start of Process*:

```
SET      --- X                      =     14
CNV BIN  aaa WORK HEX(0E)           =  --- X
```

10.28 Difference (compile error): A valid SP2 auto-access can cause compile error "Invalid Data Lookup specifications - Source Missing" in APPX 3.0 and 3.1. It seems to occur when the number of key segments in the DLU key exceeds the number of key segments in the file key. This can affect input, output, and inquiry processes. The only work-around is to remove the DLU from the image and program the equivalent functionality as detailed in the following paragraphs.

For an overview of DLU items, run the APPX Software, Inc.UTL CS utility "APPX ITEM File List" and enter 'Check for DLU?'="Y". To see lookup items too, also enter 'Lookup?'="Y" (note that the checks for DLU or Lookup are the only checks combined with an OR; the rest use AND). To list items for a single process, enter the process name. (See items 1.52 & 9.45.)

For output/inquiry processes, do the following (see item 9.45 for input processes):

1) For all components of the key on the image, change the Data Lookup Type to NONE. Be sure to make note of DLU specifications (whether the file must exist, action if NOF, etc.).
2) For all associated lookup fields on the image, change the Auto-Display to 1 (Current Value) and make note of lookup value (null or ?'s) if NOF.
3) Add code in *Pre-Display* to do the DLU read and to issue error/warning and set lookup values as appropriate on F condition (DISPLAY commands are not necessary).

10.29 **Action**/Bug?/Difference (**flagged in V05 after 08/27/96**): If a multi-level output function has subheadings and/or subtotals defined at the lower level, SP2 would not print them if there were no records at the lower level. APPX will print RANGE-START and RANGE-END images even though there is no detail. To emulate SP2, in *Pre-Invocation* of the automatic process calling the lower level, add code to do a BEG AT/END AT/READNEXT using the constraint and set --- INCLUDE CHILDREN to 0 on a False condition. (You do not need to set INCLUDE CHILDREN to 1 on a True or in other children. Outputs do not suffer from the input process bug described in item 9.44, which can improperly skip children if any children are optional.)

10.30 Bug?/Difference: If a query has a temporary file as its PCF and if the temporary file is populated in the job after the QUERY job step, an output using the query may not find the populated file when the job is run in background. (See item 12.11.)

10.31 Difference/Bug?/Conversion EH (**all scrolling windows flagged in V05**): APPX cannot handle certain scrolling features in outputs which worked in SP2. For example, a SP2 subtotal window could specify Scroll Across whereas APPX only allows access to scroll type on a RECORD frame. The conversion utility will convert it as Scroll Across but it does not work right. At Pinellas, the last control break did not scroll properly; it was in the right column but down a line (the detail window was not being printed).

Also, a SP2 output window could be programmed to scroll down (although one might question why) and it worked. In APPX 3.0 this can cause an APPX system error RMIMG.C.2472 ("write_img() - Error - Already on File"). Alternatively, it can cause an output to end prematurely as though it were finished, printing an incomplete report.

To list images in scrolling frames, run APPX Software, Inc.UTL CS utility "APPX IMAGE File List" and enter "Y" to check for scrolling frames. If desired, enter OUTPUT or INQUIRY as process type.

10.32 **Action**/Bug (**flagged in V05 after 08/27/96**): APPX does not handle a display mask with CR or DR properly. For example, the mask '9,999CR' produced "-5,678.11R" whereas the mask '$9,999CR' was OK. Also, it appears to not be clearing CR from following fields and it prints DR for positive whereas SP2 didn't. Change the mask to use "-" or "+" instead. To list such items, run the APPX Software, Inc.UTL CS utility "APPX ITEM File List" and enter "Y" to 'Check for CR/DR in Mask?'.

SECTION 11: Update Processes

11.1 Bug (**supposed to be fixed** in APPX 3.0.0): APPX 1.8.4 can sometimes skip the PCF record if it is being held and there is no explicit code to read the record with hold.

11.2 **Action**/Difference: APPX requires a higher level PCF record be read with hold before being rewritten or deleted at a lower level whereas SP2 does not. This will cause an error at runtime. Check APPX ILF X-Ref for use of REWRITE or DELETE in lower level output and investigate. You can also use the APPX Software, Inc.UTL CS utility "APPX STMT I/O List" to review REWRITE/DELETE commands. (See item 10.5.)

11.3 Bug (**flagged in V05**): In APPX 3.0.0, the action log for an Update process does not print any ERROR or WARNING messages issued in *End of Process*. However, it does print those issued in *Start of Process*. Unexpected type ERROR's (like those following READ's or WRITE's) can probably be ignored but any messages that provide needed information of some kind will require alternative programming. If the update process uses a query, one solution would be to modify the query as needed to have STANDARD grand totals with 'Editable?'="N" (be sure to check the X-ref for other use of the query), add an unbounded RANGE-END frame as needed to the update, and move the necessary code to this frame from *End of Process*.

11.4 Difference (**flagged in V05**): SP2 allows a total-type work field to be used as an ordinary work field in the detail window of an output/update function and sets the value to zero before entering the detail window. There is no such thing as a total-type work field in APPX (see explanation in item 8.13) and APPX does not change the current area of a numeric work field without explicit commands to do so. If code exists in a detail window which relies upon the zero value, the SP2 total-type work field must be SET to zero at the beginning of the event point in APPX. In addition, there are other complications if commands are used on a SP2 total-type work field in a subtotal window (see item 8.13).

The **conversion log** flags the use of commands such as SET, COMPUTE, and RESTORE on SP2 total-type fields. (See items 8.13 & 10.23.)

11.5 Bug?/Difference: If a query has a temporary file as its PCF and if the temporary file is populated in the job after the QUERY job step, an update using the query may not find the populated file when the job is run in background. (See item 12.11.)

SECTION 12: Query Processes ***(from SP2 Sort Functions)***

12.1 Bug (FYI): In APPX 1.8.4 Application Design when changing Runtime Editability on the Sort Order screen, incorrect values for Subheading or Subtotal could appear and inadvertently cause the design to be changed.

12.2 Bug (**flagged in V05**): In APPX 1.8.4, if a job file is created by the CREATE statement (as opposed to created implicitly) within an APPX query process, it will not be used by a later job step. Also, in APPX 3.0, a permanent file created via CREATE in Query Execution could not be found until the CREATE was moved to Query Setup. (See items 7.19 & 12.11.)

12.3 Bug (**fixed** in APPX 3.0.0): In APPX 1.8.4, record selections that compare a field in the PCF file to a corresponding field in another file are not handled properly, such as:

```
                  CIC PRODWARE ACTIVE              CIC PRODUCT ACTIVE
```

This construct allows the operand to be filled in by the operator at runtime, for instance, "NE" to select records where these flags do not agree. One bug is that APPX either selects no records or an incomplete set of records when this selection is activated. Another bug is that just the presence of this selection can cause the query to misbehave, even when it is not activated. If the selection is deleted from the query, the problems go away.

12.4 Bug (**flagged in V05**): An APPX 2.3 bug will not allow a query to run in background if there are no enduser selections and no designer selections. The easiest solution is to add a designer selection testing a field EQ itself. It may be necessary to have no enduser selections in order to suppress the display (since the Display Sort flag only affects the sort order display). However, it is also permissible to not have a QUERY job step and still use the query; nothing is displayed and it behaves as though the operator just pressed return thru the screens with the only exception that End of Process for Query Setup is not executed. (Note: A QUERY job step with enduser selections can follow the disposition and run OK in background without displaying record selection screens, but if run in foreground it will still display screens.)

12.5 Difference (FYI): APPX differs from SP2 in its definition of INCLUDE RECORD. APPX defines INCLUDE RECORD as follows:

```
"0" - Do not include this record (used in both Pre-User & Post User Selections)
"1" - Include this record (used in Pre-User Selection to force record inclusion
          EVEN IF IT FAILS Enduser or Designer selections; N/A in Post User)
" " - Let inclusion depend upon meeting Enduser and Designer selections
          (used in Pre-User Selection but N/A in Post User)
```

Note that APPX resets INCLUDE RECORD to " " before each record is presented to query selection.

Since SP2 code to set (INCLUDE) to 1 is converted to set INCLUDE RECORD to " " in APPX, this difference does not normally cause a problem. The conversion log will flag (INCLUDE) for inspection if it is not used on the left side of a SET command with a constant of 0 or 1 on the right side.

12.6 Difference (**compile error**): SP2 ignores invalid date constants in record selections but this causes a **compile error** in APPX. Remove the constant.

12.7 Note (**compile error**): If you encounter an invalid frame type **compile error** creating the EM for a query with designer selections, try using *Invoke A Process* to run Update Process "FRAME (UPD OPTS)" in Application 0AD 00.

12.8 Note (**flagged in V05**): If AND's and OR's are used in SP2 record selections, the *Enduser Selection Expression* or *Designer Selection Expression* generated may be incomprehensible or wrong. Any AND's for fields with no OR's can be ignored (unless parentheses were used in SP2). For fields with both AND's and OR's, consider the following example:

```
         100     CIC PRODUCT NO                   GE
         200     CIC PRODUCT NO                   LE
         300     CIC PRODUCT NO                   EQ
         400     CIC PRODUCT NO                   EQ
         500     CIC PRODUCT DESCRIPTION          GE
         600     CIC PRODUCT DESCRIPTION          LE
```

where *Enduser Selection Expression* = ((100 AND 200) OR 300 OR 400)

12.9 Bug (**fixed** in APPX 3.0.0): In APPX 1.8.4, the only way to tell if an *Enduser Selection Expression* exists is to select the option for the pop-up screen (i.e., the option is not highlighted).

12.10 Note (**flagged in V05**): The conversion utility splits displayed & non-displayed record selection criteria into enduser & designer selections in APPX. This may cause problems for any SP2 sort which OR's displayed with non-displayed selections. The conversion log will flag any OR's on non-displayed selections.

12.11 **Action**/Note (**updating in query flagged in V05**): APPX handles queries differently than SP2 handles sorts and such differences can cause problems at runtime. In APPX a query does not get executed until an output/update process is invoked which "uses" that query. If the SP2 sort does any updating of any kind, including setting (FUNCTION CODE), an update or subroutine should be added to the APPX job to perform this code and it should be removed from the query. Remember to check the SP2 Sort X-Ref for all jobs using the SP2 sort. (See item 7.19.)

If such updating is not removed from the query, it may not be completed before the process using the query starts. Also, if a temporary job file is being updated in the query, another copy of this file will be created by the process using the query which will not have the updated values. While a dummy update could be inserted to run the query and solve these problems now, this approach runs the risk of not working later if APPX ever changes query functioning to agree with its documentation.

If the APPX query is setting PROCESS CODE in *End of Process*, you could add a SUBROUTINE job step prior to the process using the query. The new subroutine could be a do-nothing subroutine named "*SET* PROCESS CODE" with a single comment line (see item 7.11). *Post Invocation* of the new SUBROUTINE job step should contain the appropriate code from the query *End of Process* (use Xcopy to copy code). Then this code should be deleted from the query *End of Process.*

If you decide to add an update, avoid the use of CREATE on temporary job files in the new update (see item 12.2). Also, both the new update and the process already using the query should have query update type 1 (normal). In certain cases you might find that the query is no longer needed and the new update can simply replace it (for example, where the query is sorting on primary key and the operator cannot change the sort or impose record selections). If so, remember to remove the query from UQ besides deleting the query job step.

Beware a query which has a temporary file as its PCF **(this condition is not currently flagged)**. If the temporary file is populated in the job after the QUERY job step, the output/update using the query may not find the populated file whether or not the job is run in background. This may be a valid difference in behavior since the temporary file will be created during query setup by virtue of being the PCF file. Then, when the query is used, the status at query setup is restored. One solution is to invoke the output/update as SUBPROCESS but this may not work if there are other job steps following the output/update since work fields, etc., will be shared; also, this solution may not work in a split job. Another solution is to set 'Separate Task OK?' to "N" in the job. To find such queries plus the jobs that use them, run the APPX Software, Inc.UTL CS utility "QUERY'S with Temporary PCF" and enter "Y" to 'Complete Job?'. Also, check the

SP2 X-Ref for the use of queries from cross-applications. (See items 10.30 & 11.5.)

If desired, the APPX Software, Inc.UTL CS utilities can also be used to list updating in QUERY processes (be sure to enter "QUERY" as process type). Use the "APPX STMT I/O List" to list desired I/O commands and use the "APPX STMT 'Format 5' List" to list SET commands on PROCESS CODE.

12.12 Note (FYI): A query will not behave as expected if it comes up in mode other than ADD. The mode is specified in the Override Default Mode field in the job step.

12.13 Difference (FYI): The option to provide for "Continued" subheading messages has not been implemented in APPX.

12.14 Conversion Bug (**fixed and flagged in V05**): If any SP2 record selection has a non-editable component, the conversion utility may improperly set the 'Editable at Runtime?' flag to "N" for the *Enduser Selection Expression*. (See item 12.15.)

12.15 Bug (FYI): APPX 3.0.0 does not enforce the 'Editable at Runtime?' flag for the *Enduser Selection Expression*. If "N", the operator may still edit the expression at runtime.

12.16 Bug?/Difference: In APPX, bogus blank lines may appear in the output if inappropriate subtotals (or subheadings?) are specified in the associated query but not defined in the output. (See item 10.16.)

12.17 Difference (**compile error**): At runtime SP2 ignores a record selection which references an invalid field (for instance, field does not exist) whereas APPX gets a **compile error**. Delete the record selection and remove it from the expression if any.

12.18 Bug?/Difference (**flagged in V05 after 08/27/96**): At runtime SP2 offers the flag 'Print Totals Only (No Detail)?' on the sort selection screen which, if selected, suppresses detail lines on the associated report. APPX provides the same capability but in a different place; at runtime it offers the flag 'Print Summary Only' on the disposition screen. If a SP2 sort is programmed with this flag set to "Y", the conversion utility generates code in *Start of Process (Query Setup)* to set --- PRINT SUMMARY ONLY = 1 (Yes). This allows the disposition screen to default the flag at "Y". Unless the disposition overrides this value, it should work properly as converted **provided there is only one output in the job and provided the job is not used as a job step on another job**. However, since PRINT SUMMARY ONLY is only changed by commands or disposition entry, it will remain set throughout the job which can cause subsequent outputs to behave differently. The safest thing to do is to add a job step following the output using the query and have the job step set --- PRINT SUMMARY ONLY = 0 (No). Consider using a subroutine named "*TURN OFF* PRINT SUMMARY ONLY". To find queries setting PRINT SUMMARY ONLY plus the jobs using those queries, run the APPX Software, Inc.UTL CS utility "QUERY's with PRINT SUMMARY ONLY" and enter "Y" to 'Complete Job?'. Also, check the SP2 X-ref for the use of queries from cross-applications.

12.19 Difference (**flagged in V05**): At runtime SP2 offers the option to specify 'Maximum No. of Records to Print' on the sort selection screen which, if entered, determines the maximum number of records from the sorted file to be included in the report. All records must still meet all sort selection criteria and are sorted in the correct order. Subtotals and grand totals include only those records that actually print. When implemented, APPX will provide the same capability but in a different place; at runtime it offers the option to specify 'Record Limit' on the disposition screen but this currently has no effect (since PRINT RECORD LIMIT has not been implemented).

If a SP2 sort is programmed with 'Maximum No. of Records to Print' set, the conversion utility generates code in *Start of Process (Query Setup)* to set PRINT RECORD LIMIT accordingly. This allows the disposition screen to default the proper value (unless overridden by the disposition job definition). However, the job will still not work properly in APPX unless and until PRINT RECORD LIMIT is implemented. The conversion log flags this condition.

Until PRINT RECORD LIMIT is implemented, there does not appear to be an easy way to address this problem short of adding an update following the query which will build a job file of pointers with the proper number of records. The new update could then be followed by a new query which used the pointer job file to determine which records to include. The new query would have no enduser selections, a designer selection to avoid the BG bug (see item 12.4), the same sort order as the original query, and would not display to the user. The output would also have to be changed to use the new query.

12.20 Difference (**flagged in V05**): SP2 allows record selections to have relations for which there is no counterpart in APPX. For example, the SP2 relation 'BW' (Beginning With) on a user-specified selection can be modified in APPX to use 'GE/LE' but the user specification must be entered differently to behave the same way. The conversion log will flag these conditions with the message "Revise Relation".

12.21 Conversion Bug (**fixed in V05**): Default mode for query processes should be set to ADD (1).

12.22 **Action**/Bug (**all designer selection expressions flagged in V05**): In APPX 3.0, if a designer record selection has nothing on the right side of the selection and if it is included in a designer selection expression, APPX will treat the selection as though comparing to blanks (rather than ignoring it). Delete the selection and remove it from the expression.

Although the problem only exists if there is a designer selection expression, all 'do nothing' designer record selections should probably be deleted. To find such selections, run the APPX Software, Inc.UTL CS utility "APPX SELECT File List" and enter process type "QUERY", frame class "QUERY-SLCT-HIDE", and right basic field type "BLANK".

12.23 **Action**/Bug: In APPX 3.0, if a record selection has a field with right-alignment and/or padding on the left and a constant hard-coded on the right without the corresponding alignment or padding, the record will not be selected. In the case of an end user selection, the proper alignment and padding will be displayed at runtime but it will not be honored unless the user changes something on the record selection screen. In application design, hard code the constant with the proper alignment or padding.

To find such selections, run the APPX Software, Inc.UTL CS utility "APPX SELECT File List" and enter process type "QUERY", 'Check Left for Pad Char?'="Y", 'or Alignment?' ="Y", and right basic field type "CONSTANT". To see the field formats for any selections found, run the "APPX FIELD File Col Hdg List" and name the desired fields as record selections. (Alignment code values are: 0=None, 1=Left, 2=Justified, 3=Right, 4=Centered.)

SECTION 13: Inquiry Processes ***(from SP2 Output Functions invoked as INQUIRY)***

13.1 Conversion Bug (**fixed in V05**): In V04 the first two lines of code in the SP2 Output *Start of Function*, if any, are lost (overwritten) when an output is converted to an inquiry process in APPX. The conversion writes two sets of identical code in *Start of Process* with the exception that the first set uses 1/0 for flags while the second set uses Y/N.

13.2 Note: In general, APPX inquiry processes suffer from the same problems that APPX output processes do except that those converted from SP2 are not associated with a query (since SP2 cannot link a sort to an output invoked as an INQUIRY). These problems should be addressed for the inquiry when addressing the output. **The V05 conversion log for Inquiry Functions warns whenever SP2 outputs are converted to APPX inquiry processes or to both APPX inquiry/output processes.**

13.3 Bug (FYI): APPX technical documentation for inquiries is incomplete.

13.4 Difference (**flagged in V05**): APPX presents a key entry screen upon entry and exit from an inquiry on a one-record file whereas SP2 does not. If desired, this screen can be suppressed in APPX by defining a logic work field 'WORK DISPLAY' with default "N" and adding the following to *Select Image* of the Key Entry frame:

```
         IF       aaa WORK DISPLAY               EQ     0
   T     SET      aaa WORK DISPLAY               =      1
   T     SET      --- OPTION                     =      RETURN
   F     SET      --- OPTION                     =      END
```

This approach will remove access to print disposition but SP2 does not offer this either.

13.5 Difference (**fixed in V05**): Outputs designed for the screen were limited to 79 columns in SP2. Change these to 80 columns in APPX to avoid possible editing errors.

13.6 Bug (**fixed** in APPX 2.3): APPX 1.8.4 holds the record returned as the next key in an inquiry. (Note: In APPX 2.3 the screen may show CHG Mode but the MODE value is actually INQUIRE so the record is not held.)

SECTION 14: Status Processes

14.1 Difference: In SP2 the title of the job appears at the top of screens presented by STATUS job steps. In APPX the title of the job does not appear. (See item 7.14.)

14.2 Difference (**flagged in V05**): SP2 provides for a Screen Line 23 Warning Message (which is displayed if any WARNING and no ERROR commands are executed) and for a Screen Line 23 Error Message (which is displayed if any ERROR commands are executed). APPX does not provide this capability.

14.3 Difference: SP2 presents ERROR/WARNING/MESSAGE's in the order in which they are programmed. APPX presents them in the order of severity (from highest to lowest) with ERROR's first, then WARNING's.

SECTION 15: Subroutine Processes ***(from SP2 Command Libraries, External Fns, and Non-Displayed Disposition Functions)***

15.1 Note (**flagged**): Subroutines converted from SP2 non-displayed disposition functions require work. (See items 7.9 & 7.10.)