Title:	Project and Employee Tracking System (PETS)
Course:	
Year:	1989
Author(s):	M. Chamberlain
Report No:	P89009

Abstract: This is a software package and a user manual for the development of the Responsibility Interface Matrix (RIM) in an organization. RIM is a tool for the study of responsibility patterns in the organization by identification of primary responsibilities, information flows and support roles in the completion of work packages in each project.

PROJECT AND EMPLOYEE TRACKING SYSTEM (PETS)

M. Chamberlain

EMP - P8909

26

PROJECT & EMPLOYEE TRACKING SYSTEM (PETS)

```
                        -----
                       ¦  0  ¦
                        -----
                          ¦
            -------------------------
            ¦                       ¦
          -----                   -----
         ¦  1  ¦                 ¦  1  ¦
          -----                   -----
            ¦
   -------------------
   ¦                 ¦
 -----             -----
¦  2  ¦           ¦  2  ¦
 -----             -----
```

AN EAS506 PROJECT SUBMITTED TO

DR. DUNDAR F. KOCAOGLU
ENGINEERING MANAGEMENT PROGRAM
PORTLAND STATE UNIVERSITY

BY
MARK T. CHAMBERLAIN

AUGUST 28, 1989

USER MANUAL
PROJECT & EMPLOYEE TRACKING SYSTEM (PETS)

TABLE OF CONTENT

USER MANUAL

PROJECT & EMPLOYEE TRACKING SYSTEM (PETS)

Use Of The Manual:

This manual was written with the intent of providing off-line help to the user who is unfamiliar with the PETS interface. It is not intended as a forum for educating the user on the virtues, drawbacks, advantages, etc. of linear responsibility charts. It is simply an aid to understanding the interface of the software program and is best used when sitting down in front of the computer and running PETS. You may wish to read the entire manual, skip the manual and take your chances with the program (I like to think it's intuitive if you have ever used a spreadsheet and know what an LRC is), or browse the table of contents and skim through the manual when necessary.

Where functions are described but not available in the actual executable program a '[NI]' legend is shown. This legend means NOT IMPLEMENTED, and is followed by a number which corresponds to a numbered paragraph in the Future Directions And Extensions section where it is explained in more detail.

Hardware Requirements:

To run PETS you will need at least 512K memory on your fully compatible IBM PC. A hard disk is recommended. DOS 3.0 or higher is also required. The software was written for monochrome monitors. The program was not tested on color monitors but should work if the color monitor is configured in "mono" mode.

Step-By-Step Loading Instruction:

To load PETS on your hard disk follow these simple instructions. I will enclose what you should type in 'single quotes'. Do NOT type in the single quotes. I abbreviate carriage return as <CR>. When you see <CR> press the return key.

Step 1) Boot up your PC using a version of DOS 3.0 or later. You can verify your DOS version by using the VER command.

```
>'VER     <CR>'
```

If you want your prompt to look like my prompts in the remainder of these steps enter:

```
>'PROMPT $p$g    <CR>'
```

Step 2) Create a new PETS subdirectory on your hard disk:

```
C:\>'MKDIR PETS <CR>'
```

Step 3) Move to the PETS directory:

```
C:\>'CD PETS    <CR>'
```

Step 4) Place the PETS distribution diskette in drive A.

Step 5) Copy the contents of the PETS diskette to the new C:\PETS directory:
```
C:\PETS>'COPY A:*.*     <CR>'
```

Step 6) Note that several files are on the distribution diskette in multiple directories. Those with the extension ".C" are the Borland Turbo-C V2.0 source files. The ".H" suffix means that they are header files that contain definitions included in the ".C" source files at compilation. The ".PRJ" file is a Turbo-C "project" list of the ".C" files to be compiled and linked into an executable file, the ".EXE" executable file is the only executable file on the disk and therefore the one of most importance. The ".TRE" files contain the users data saved by PETS, and are on the disk for example purpose only. Files other than the ".EXE" file and the ".TRE" files were provided for reference for those inquisitive minds that have a desire to look at or modify source code or documentation.

Step 7) After you have placed the original PETS diskette in a safe place you are now ready to run PETS software.

Step-by-Step Program Execution:

The remaining steps are easy to follow and will take about 20 minutes to complete. My wife is pro on our word processor but is otherwise computer illiterate and she completed the remaining steps and her own PETS tree in 30 minutes.

Step 8) After completing steps one through eight you are ready to run PETS. Enter PETS and press return.
```
C:\PETS>'PETS   <CR>'
```

Step 9) The initial PETS tree logo should be displayed on the screen. Press any key to continue.

Step 10) The new screen is the level 0, root RIM (Responsibility Interface Matrix) screen. Notice the "TITLE:", "RIM PATH:", and "Responsible Authority:" information fields on the top few lines of the computer screen. For the level 0 RIM the designations in these fields are predetermined and cannot be changed. RIM screen information for these fields at all level other than zero is automatically updated [NI-2].

The initial screen contains the function key menu--the function performed by the function keys are listed on lines four and five of the screen.

Step 11) At this point in the program you may begin a brand new RIM screen. However, before you get too ambitious lets look at an example.

Try using the arrow keys to move the cursor around from cell to cell. Notice the cursor only goes in part of the screen. This part is reserved for cells and is the part you will be able to edit after a few more steps.

Take a quick look at the Help function by pressing function key F1. Read the first help page. When you are done browsing help toggle it off with the F1 key.

Step 12) Read the menu options on screen lines four and five. Select the '/' option by pressing the '/' key. This will take you from the function key menu to the main command menu.

Try your arrow keys again. Do they work? They shouldn't! The arrow keys are designed to work as part of the function key menu. If you wish to test your arrow keys again, go back to the function key menu by pressing the escape 'Esc' key, but come back to the main command menu by using the '/' to finish the step by step example.

Each of the main command menu options can be selected by pressing the first letter of the menu option. Try pressing the 'F' key to select the 'File' option.

Step 13) The File option submenu should be displayed. You can either Retrieve a file or Save a file. Since you have nothing to save try and retrieve a file. Press the 'R' key to Retrieve a file. At the flashing prompt enter the filename 'example' and press return. The file EXAMPLE-.TRE was read in and is displayed on the RIM screen. If you get an error message stating the file could not be opened then retry using correct spelling, or recheck to make sure you copied all of the files from the distribution diskette.

If you typed 'example' why was 'EXAMPLE-.TRE' retrieved? PETS allows you to name the first eight characters, if you use less than eight characters PETS pads it with as many dashes '-' as needed until there are eight. If you keyed in 'abc' PETS would try to retrieve a file from the current directory named 'ABC-----.TRE', then give you a diagnostic message that the file could not be opened because it does not exist. The unique '.TRE' name extension allows you to identify and organize your PETS TREes separately from your other files. These filename conventions hold for the Save option as well.

Step 14) Back out of the File option submenu by pressing the escape 'Esc' key (or the 'Q' for Quit-menu). Again back out of the main command menu using the 'Esc' key. Using arrow keys place the cell cursor over cell D6 which is in the intersection of the "Steve Montano" name column and the "New CAD/CAM Acquisitions" project row. Notice how the cell legend 'P' is in inverse video? What does 'P' mean? Press the F2 function key (F2's meaning is written on screen line four) to toggle on the display of the legend code window. You can see that 'P' means primary authority. Steve is the primary authority (primary responsibility) for the new CAD/CAM acquisitions project.

Hit the TAB key several times and watch the cursor go from legend field to legend field (SHIFT-TAB goes the other direction). Notice that the first four legends along the left side column are for informational purposes, the others represent possible employee tasks on a project. Informational legends are not modifiable, the others are[NI-3]. You can create your own legend codes and descriptions after you are through with the example [NI-3]. Press function key F2 again to toggle off the display and continue with the program.

Step 15) There is more to learn about this cell. With the cell cursor on D6 (same as in previous step) toggle on the comment mode by pressing function key F4. The comment window is both an information window and a data entry window. This is the only mode that allows you to enter names and projects. A begin and end date field is included for reference. The dates must be in the format 'ddMMMyy' (eg: 25Dec89). A 55 character comment field is also available. The comment should contain anything that you think is worth commenting. Toggle the comment mode off by pressing function key F4.

Step 16) Move the cursor to cell D1. Can you tell what the legend code symbols mean? (Use F2). Can you read the comment? (Use F4).

Step 17) Move the cursor to cell D2. Toggle the comment mode on (F4). Notice that the comment mode window is already filled out in the name and project input fields. Lets enter the date so TAB once to put the cursor in the date field. Enter today's date in ddMMMyy format (eg:25Dec89). TAB twice to get to the end date field, enter another date in the same format. Pressing either the return key or the F4 key should save the new information and toggle off comment mode. if you make a mistake you can backspace. The delete key is not functional. Pressing the 'Esc' key will not save the new information, but will toggle off the comment mode.

Try step 17 again but this time edit in the comment field, and use SHIFT-TAB to move the cursor around.

Step 18) Suppose column B should be headed by David Jensen, and not Doug Jensen who left the company last month. See if you can modify the name. Start by moving the cursor to any cell in the B column and then toggle on comment mode (F4). Enter the correct name and press return. Does the name look a little off center? PETS allows for 18 character names, but only the first 16 of those will fit on the display, and only characters one through eight fit on the top line, the next eight fit on the bottom line. To help you determine centering position after the first eight characters are typed the shape of the cursor changes. If you like your names left justified start typing the next name (Jensen) when the cursor changes shape. Names like mine (Mark Chamberlain) must be contracted, truncated, or spilled onto the next line. When you are done making the modification exit comment mode (F4, <CR>, or Esc).

Step 19) The CAD company Computer Vision was purchased by PRIME Computer. Update project row 2 with the new information. Place the cursor on any cell in row 2 and enter comment mode (F4), TAB to the appropriate field. The projects can have up to 15 characters per line. As in step 18 watch for the cursor to change shape. Toggle off comment mode when you are done.

Step 20) Oops! The content of cell C5 should have been placed in B5. To fix this place the cursor in cell C5. Enter edit cell mode by pressing F3. The cursor cell is made up of eight 2 character fields. The TAB and SHIFT-TAB keys will allow you to move from field to field in edit cell mode. To delete an unwanted legend code simply place the flashing cursor on top of it and hit the space bar, TAB to the next unwanted legend and do the same. Pressing F3 or <CR> will save the new changes and exit edit cell mode, Esc will exit without saving changes (this is the same convention exists in edit mode, comment mode, and legend window). The C5 cursor cell should now be blank.

Move the cursor to cell B5. Enter edit cell mode by pressing F3. Enter the same legend codes which were deleted (if you can remember them) from C5. Remember you must TAB to get to the second legend field. When done, exit edit cell mode.

Step 21) Now that you have made changes, you may wish to save them for later retrieval. From the function key menu press '/' to enter the main command menu. From the main command menu press 'F' for File. From the File submenu press 'S' for Save. Enter a PETS tree name and press return. If you enter 'example' you will have completely overwritten the old example file. PETS does not implement any file backup processes. See step 13 for file naming conventions. The file can only be saved in the current directory. Only characters that comprise legal DOS file names are allowed.

Try pressing the '*' as part of the file name and listen for your rejection notice. Press Esc and go to the main command menu.

Step 22) The main command menu presents several options: Traverse [NI-4 also F5 and F6], Reports [NI-5], and List [NI-6]. Try pressing 'T' or 'R' or 'L' and explore all of these menu structures. Screen line four will always indicate which menu you are in. Remember to back out of a menu press 'Esc'.

Step 23) If you want a print out of the current RIM use the SHIFT-PRINTSCREEN key (assuming you are configured to a printer that is on and on-line).

By now you are probably ready to take a break. To exit go to the main command menu and press 'Q' to quit. Since you have already saved press 'N' to not save before quitting (unless you made other changes then press 'Y' to save). This will take you back to the DOS level. Verify your newly created PETS tree by entering 'DIR *.TRE' at the DOS prompt.

Step 24) After completing these steps you may wish to immediately solidify what you have learned by creating your own PETS tree.

You now have enough information to run PETS. To start your own PETS tree make sure you are in the same directory that PETS is in (or have PETS in the DOS path) and enter 'PETS   <CR>' at the DOS prompt.

I have included my wife's WORKPETS.TRE for your perusal (it represents our efforts to clean the apartment in preparation for moving out of state after completing the Engineering Management Program. Included in the RIM tree are Nathan and Chet, our preschool sons, and Benjamin their imaginary dog).

PETS Overview:

The PETS software will allow you to model a hierarchical tree (PETS tree) [NI-1] to represent and track the breakdown of projects, and responsibilities of personnel within an organization. This tree, with its labeled levels, is depicted by the initial PETS screen called a RIM screen. RIM is an acronym for Responsibility Interface Matrix.

The RIM Screen:

The second line of the RIM screen depicts the level of the RIM being displayed. The RIM at the top of the hierarchy is always Level 0 and each subsequent level down is incremented by one. There can only be one Level 0 per PETS tree, the maximum number of Level 1 RIM screens is equal to the number of projects shown on the Level 0 screen. Each PETS tree is comprised of one or more RIM screens. Each tree is contained in exactly one DOS file.

The top row of text on the RIM screen lists the title of the project. For the Level 0 rim it is always "ROOT RIM", and Responsible Authority is always "LEVEL 0 AUTHORITY". For subsequent RIMs at levels greater than zero the title is the same as the project name in the level above, and the responsible authority is always the name of the person in the level above who has been 'P' designated.

The current date appears on the screen in the upper right hand corner for reference.

The fourth and fifth lines of the RIM screen contain a dynamic menu. The dynamic menu will either be the function key menu or the main command menu or one of the main commend menu submenus. The menus can be thought of as a tree -- a menu tree. Climbing down the tree is done by selecting the menu item desired, climbing up the tree (backing out of a menu) is done by hitting the ESC escape key. Try the Step-By-Step example to get a feel for the menus.

PETS provides modeling and tracking capability through the use of a RIM screen. The RIM screen is a 8x6 (projects x employees) matrix of Legend Cells. A Legend Cell is defined as the intersection of a Project Row and an Employee Column and is outlined by a highlighted cursor cell in the software implementation. The Legend Cells may be edited to contain up to eight, double character Legend Symbols which represent either the given individuals responsibilities (or workpackage) on the given project, or information Legend Codes (see the Step-By-Step example to distinguish the different types).

The names of the employees are entered on the RIM screen row just below the letters A B C D E F. The employee names serve as a header for the column in which they are located. All information in the column pertains to the employee listed at the top. The project names are listed down the row in the left most column and are numbered from 1 to 8. The project names also serve as header for their row. All information on a given row pertains to the project listed on that row on the left most column. Cells may be referenced by their alpha-numeric intersection, for example A1 or D5.

Function Keys (HotKeys):

Function keys F1-F6 are used in PETS to allow the user to execute some of the most commonly used functions. You may access the Function Key Menu from the Main Command Menu by pressing the 'ESC' key. The Function Key Menu lists the function keys and a one or two word description of what that function key does. The function key functions are not always accessible (for example, while you're in HELP only F1 has meaning, the others do nothing, F1 turns HELP off). You need not worry about executing a function key at an improper time. The PETS program controls which functions are accessible at all times during execution. Function keys F1, F2, F3, and F4 are toggles and therefore not only turn the function on but turn it off as well.

Main Command Menu:

To access the Main Command Menu from the Function Key Menu press the '/' key. Menu selections are performed by pressing the keyboard key that matches the first character of the function in the menu prompt. For example, if I wanted to Traverse the PETS tree I would press 'T'[NI-4], if I wanted to Quit I would press 'Q'. The Main Command Menu is comprised of several submenu multi-depth command structures. To Quit a command menu press 'Q' or to back up to the next higher command structure press 'ESC'.

Entering Data Into The RIM:

Names and Projects are entered in comment mode. Comment mode is entered by pressing F4 when the function key menu is displayed. After pressing F4 a comment mode window should be displayed that will allow you to enter data in the selected fields (name, begin date, project, end date, and comment). Pressing F3 allows editing of the legend cells. F2 allows editing of the legend symbol codes.

Data Entry Conventions:

All of the data entry conventions follows the same scheme:

- To go to the next field use the TAB key.
- To got to the previous field use the SHIFT-TAB key.
- To enter data simply type in the name of the data.
- To delete data in a field overwrite it with new data, eg: a single space will do.
- To quit editing and not save the data hit the escape 'Esc' key.
- To quit editing and save the data hit either <CR> or the function key that was used to enter the mode. For example to save the data in the comment mode window I could either <CR> or F4.
- Use of the backspace key is permissible, use of the delete key is undefined.

Note: In the comment window 8 screen spaces are allowed for the name on the top and on the bottom row, and 15 screen spaces are allowed for the top and bottom project names. To enhance readability, the user may wish to use contracted names. When the user has keyed in information sufficient to fill the top row the input cursor will change from a tall cursor to a short cursor. Information keyed in over the tall cursor goes on the top, and subsequent information will be displayed on the bottom row of that name or project field.

F1: HELP

The HELP function contains an overview of the basic concepts underlying the PETS software tool. Information on the meaning and use of the commands and menus is also contained in HELP. You may wish to read HELP now to familiarize yourself with the PETS tool, or you can read it almost anytime by pressing the F1 function key.

F2: LEGEND WINDOW

Legend symbols are defined within the legend window. The legend window will be displayed and editable after pressing function key F2. Only a few of the legend symbols are not modifiable [NI-3]. They are 'P' for Primary Authority, 'D' to signal project is Done , 'C' to flag user defined Critical data, and '*' to indicate a 'P' or 'D' designated cell has been previously exploded and contains sub RIMs. These four legend codes are informational in nature and except for 'P' do not represent a task assignment. The others are given as defaults to aid you in getting started. You may change the defaults either now or later [NI-3]. You may add new Legend Symbols in the Legend Window as well. The legend symbols are global in nature. In other words, if you modify one, the new modification will replace the old legend symbol on every RIM in the current PETS tree [NI-3]. Legend symbols should be chosen and updated with care.

F3: EDIT CELL

To edit the contents of the cursor cell press function key F3. The cursor cell may contain up to eight double character legend symbols (however, we use single character codes to improve readability). The legend symbols added to the cursor cell must first exist in the legend window [NI-3]. Putting a legend symbol in the legend window validates that legend symbol for use within cursor cells.

One and only one individual may be designated 'P' Primary Authority per project. This means that only one cursor cell on any given row of the RIM screen may contain a 'P' legend code [NI-6]. The legend code 'D' for Done is reserved to replace the 'P' code when the project is complete. The 'D' code should be identical to the 'P' code with the exception that it tells the viewer that the project is no longer current. Only cells containing a 'P' or 'D' may contain sublevel RIM screens (may be exploded). Upon explosion the program automatically changes the 'P' to 'P*' and the 'D' to 'D*' to indicate that the cell has pointers to lower level RIM screens [NI-6].

F4: COMMENT MODE WINDOW

The comment mode window is used to edit projects and names in the current RIM which correspond to the current cursor cell. The function key F4 will toggle the comment mode window on and it will be editable.

F5 and F6: EXPLODE AND IMPLODE:

Only cursor cells containing the 'P' or 'D' legend code may be 'exploded'. A 'P' legend cell may be exploded whenever a user wishes to breakdown the given project into subprojects. The explosion creates a whole new RIM at the next level down the PETS tree. Each 'P' cell in the new RIM (or at any level RIM) can then be exploded to create yet another RIM, and so on. Since the PETS tree is hierarchical in nature, only the highest level projects should be listed in the root RIM (the root RIM is the top RIM on the tree, the RIM is built from the top downward). An explosion moves down one level, an implosion moves up one level. Implosion at level 0 is meaningless [NI-1].

Description Of Software Implementation:

The source code was written using Borland's Turbo-C version 2.0. No attempt was made to write portable C code. The least portable C code in the software is that portion which manages the windows and the display. Although this portion is non-portable the C code on the platform to which it is ported will most probably have equivalent functions with slightly different names.

I used the Integrated Development Environment through all phases of coding. The debugger was an essential tool, and will be for anyone that wants to modify the code. I suggest "viewing" all of the global variable values through the debugger during program execution before making a single change to the code. This will give you a feel for the functions and the order in which they are called. No function calling map is provided because the debugger provides an error-free map called the call-stack.

When the executable code got up to about 70K I could no longer use the debugger on my 512K machine. To bypass this memory shortage I temporarily decreased the size of COMMLENGTH in sizes.h to 10 and then correspondingly changed the padding in PadCurrentRim() so the display would continue to work. Doing this will allow you to use the debugger if you are limited to a 512K machine. However, there is one side effect and that is that the ".TRE" files created when the COMMLENGTH is reduced are not compatible with the ".TRE" files when it is at full value since the models have a different yet constant byte length.

One or more functions are include in each ".C" file. The main() function is in the MENU.C file. Each function is documented with a header (except for a few that are obvious stubs to not implemented functions) and should be sprinkled with appropriate documentation when necessary.

The ".H" headers are in separate files. They are essentially grouped by what they contain. What do you think would be in the header file sizes.h? Right, the length (sizes) of the variables and arrays. Anytime you make a change to a header file you must make sure that all of the modules object files are deleted so that Turbo-C will recompile (and preprocess all the headers) each of the functions before linking all of the new object files into an executable image. If you forget to recompile every function after making any significant change in a header file the consequences are bound to be quite confusing.

Last of all good luck! I learned C mostly on my own from library books and other manuals that I purchased. PETS is my first and only C program. You may want to know a little more C than I did before I started (or you'll have as much fun as I did learning it), and you should have a pretty solid programming background. I am open for phone calls if you decide to make this your PETS project too. Again, good luck. I think that a multi-dimensional LRC is a neat concept and over time could be developed into a useful product.

Not Implemented Functions:

Some functions were discussed which were not implemented in the model. These were designated by the "[NI]" code when they were previously described. A more complete explanation follows:

1.0 The multidimensional concept was not implemented due to the time requirements of the already large project scope. Every effort was made to complete this part of the implementation. In fact, most (90%) of the code for this feature is done but was not compiled because it was useless without the other 10%. The files menu.cc, kaboom.cc, and files.cc in the "3d-Code" directory contain not only the code but the pseudo-code for what has been written thus far. A sincere perusal of these files (especially kaboom.cc and files.cc) to an experienced C programmer will be of much benefit.

Future work on this feature would require a direct access pointer in functions WriteRim and ReadRim (this is the 10% not begun) so that each time they are called they access (write to or read from) a specific byte starting point in the file. The file length is constant so it shouldn't be much problem determining the file position. Each RIM contains ByteCnt number of bytes, so the file pointer must be incremented or decremented by ByteCnt for each RIM it passes over to find the target RIM. The global variable AlphaLevel should start at 'A' for the root rim, when the second RIM is created AlphaLevel should be incremented to 'B', and so on up to 'Z'. Yes, this means an artificial 26 RIM tree limit. Before quitting the very first byte in the ".TRE" should be set equal to AlphaLevel. Upon retrieving a tree AlphaLevel should be set equal to the very first byte in the ".TRE".

2.0 Since the multidimensional concept was not fully implemented the automatic updating of Title, RIM PATH, and Responsible Authority fields in the first three lines of the screen could not take place. Essentially this requires probably less than 30 lines of code in display.c. If the current level is level zero leave it as it now is, if it is greater than zero display the appropriate names. The Responsible Authority can be found in currentRIM.name[currentRow] at the time of explosion, the Rim Path is the RimLevel global variable, and the Title is the currentRIM.projects[currentRow] at the time of explosion.

3.0 Editing legend codes was not implemented due to time constraints and project scope. Currently the user is on Scouts Honor to only use the legend codes contained in the legend window.

No checking is performed whatsoever. See the editwin.c functions to learn the methods used to obtain data for the other data entry modes. This should readily adapt to this feature as well. A table of permissible codes could be made and checked each time a user enters a code. When an unrecognized code is entered the user should have the option of defining it and adding it to the list of permissible codes. The permissible codes should also be saved in the ".TRE" file--perhaps in the first reserved 500 bytes or so, meaning that the other things in the file start at byte 501.

4.0 The Traverse option and suboptions are not yet implemented. They are functionally equivalent to function key options F5 and F6 which explode and implode a cell. See the explanation 1.0 above. When 1.0 is implemented 4.0 will be almost done. I recommend using the convention that the only cell that can be exploded be the current cursor cell and only if the top left corner of the cell contains a 'P' or a 'D' legend code. All cursor cells on a given RIM (excluding the ROOT RIM) will implode to the same RIM. The next higher RIM structure is always in memory and is called upOneRIM.

5.0 The Report and List features implementation is presently limited to getting a hot print by using the SHIFT PRINT-SCREEN keys. The suggested options for this are described in the Future Directions and Extensions section.

6.0 Whenever a cursor cell is exploded the 'P' or 'D' designation should be automatically changed to 'P*' or 'D*'. Where '*' represents the condition of having lower level RIMS. When implementing the legend window the '*' code must be made non-modifiable. Since one of the basic ideas of the project was to ensure a hierarchical task reporting structure it would not do to have two 'P' primary authorities on a given task. Therefore, whenever a 'P' is entered in a cursor cell it should only be entered after a check has been made that no other 'P' codes preexist for the same project.

Future **Directions and Extensions:**

A. Using the date, project, tasks, and name information captured by PETS output a Gantt chart for a given project or set of projects on multiple levels.

B. Using the date, project, task, and name information captured by the PETS interface with a CPM or PERT program for CPM or PERT output.

C. Create historical records of all completed and current projects so that end of year summaries for a given employee can be processed that would list tasks and duties on all projects and provide some type of scoring mechanism for comparison purposes against predefined standards or other employees.

D.	Rate task assignments on a value to the company basis, and determine the employees having valuable tasks, and those not having valuable task assignments.

E.	Provide the necessary function to start a new RIM without having to return to DOS to execute the PETS program. Allow the user to delete every entry on the current RIM. This could easily be done merely by making a call to the PadCurrentRim() routine in menu.c.

F.	Provide a delete feature that would selectively allow the user to delete any particular RIM.

G.	Provide a file compression/decompression scheme to save space since the ".TRE" files are constant length they may get large. Also, allow for entry by password only. If the other things developed fully this could be modified into a system that could be accessed on a network by simultaneous users (dreaming).

H.	The HELP function can be context sensitive by making the helpCounter variable global and setting it equal to the page number of the help screen that describes it at the beginning of each function that is described in help. This means that the user would obtain help for the function they are currently in instead of always starting at page 1 and searching for the needed help page. This is a relatively easy task and only involves adding one line of code in in each of the functions described in help.

I.	To be truly functional in an industrial environment the PETS program may need to be extended to include more than 8 project rows and 6 employee names on a given screen. This is probably not much of a coding task but would require much thought and planning in algorithmic development. It would be an addition to the present program rather than a modification of the existing code.

J.	Complete the reporting functions described below. Examples of the reports are contained in a Turbo-C source file called reports.cc found in the same directory as the source code.
The Reports function enables you to generate reports. This is a time consuming but worthwhile venture. Reports could be tailored using the following suggestions. Note that the present menu structure implements calls to stubs in this structure. Reports contains the following options:

```
Print       ==> Send output from suboptions to the default
                lineprinter.
Screen      ==> Send output from suboptions to the screen.
Record      ==> Write the current RIM out to an ASCII file.  Provide
                option for removing 'recorded' end node RIMS from the
                PETS tree.
File-report==> Send the next report generated to an ASCII file.
Quit-menu   ==> Return to the Main Command Menu.
```

Report-Print:
The Reports-Print submenu enables you to generate report to the line printer. It contains the following options:

```
RIM         ==> Print the current RIM.
Workpackage==> Print all of the responsibilities in the PETS tree
               for a given individual.
History     ==> Print the 'filename'.HST file.  This file was created
               using the  Record option.
Critical    ==> Menu of options which enables printing of critical
               reports.
Quit-menu   ==> Return to the Report menu.
```

Reports-Print-Critical:
The Reports-Print-Critical option enables you to output reports to the line printer regarding critically designated items. It has the following options:

```
Projects    ==> Print a list of projects designated as critical.
Incomplete ==> Print projects not designated as 'Done' that have an
               end date    prior to the current system date.
Chain       ==> Print the chain of command from the current cell to
               the root RIM.
Employees   ==> Print a list of employees designated as critical.
Quit-menu   ==> Return to the Reports-Print menu.
```

Reports-Screen:
The Reports-Screen submenu enables you to list report to the CRT screen. It contains the following options:

```
RIM         ==> List the current RIM.  (It should already be listed)
Workpackage==> List all of the responsibilities in the PETS tree for
               a given individual.
History     ==> List the 'filename'.HST file.  This file was created
               using the Record option.
Critical    ==> Menu of options which enables printing of critical
               reports.
Quit-menu   ==> Return to the Report menu.
```

Reports--Screen--Critical:
The Reports--Screen--Critical option enables you to list output reports to the CRT screen regarding critically designated items. It has the following options:

```
Projects    ==> Echo a list of projects designated as critical.
Incomplete ==> Echo projects not designated as 'Done' that have an
               end date1 prior to the current system date.
Chain       ==> Echo the chain of command from the current cell to
               the root RIM.
Employees   ==> Echo a list of employees designated as critical.
Quit-menu   ==> Return to the Reports--Screen menu.
```