ACKNOWLEDGMENT

I express my deepest gratitude to Dr. Herbert Haynes for his time and patience in advising me on this project. Your preliminary work, insight, and wisdom proved to be very valuable to me during this semester.
ABSTRACT

A panel is defined as a tailored, pre-defined screen image. Panels are defined by the user in accordance with his needs, and may contain fixed text or variable references. Each panel may have any number of fields, each of which may have uniquely defined visual and logical attributes.

This project involves the use of a panel description language and its translated form. The panel description language is modeled after the one used with the IBM program product Interactive System Productivity Facility, as modified by Dr. Herbert Haynes. The project includes various programs to display, manipulate, and process the panels that are written in the panel description language.


**TABLE OF CONTENTS**

CHAPTER I.  INTRODUCTION ........................................... 1

CHAPTER II. THE PANEL LANGUAGE ................................. 6

CHAPTER III. THE PANEL TRANSlator AND THE
INTERMEDIATE LANGUAGE ...................................... 12

CHAPTER IV. PANEL DISPLAY ROUTINES ....................... 18

CHAPTER V. PANEL PROCESSING ................................. 37

CHAPTER VI. CONCLUSION ........................................ 41

APPENDIX A PROGRAM LISTINGS ............................... 45
CHAPTER 1

INTRODUCTION

A panel is defined as a tailored, pre-defined screen image. The panel display routines written for this project may be used to merely display text, but they are more powerful and versatile than a mere text display routine. The user is able to design and customize his own panels through use of visual attributes available in the panel language. Variables may be defined in the panel, which the user may access to initialize, change, or retrieve values. Panels may be linked with one another in association with one window (screen display area), and shown in some predetermined order as documentation or as an expository presentation. The order of panel display need not be determined beforehand, however. Panel processing is available so the user may enter variable values to dynamically direct the order of panel presentation.

This project is hardware dependent and is designed to be used specifically with the VT 220 terminal and the Ultrix-32 operating system in support of the Digital Equipment Corporation’s Vax 8200 computer. All display
routines are coded in the C programming language, and actual display is done with system write calls. Cursor positioning and realization of particular visual attributes for the panel text are dependent on the terminal being used. For instance, to position the cursor at row \( j \), column \( k \) on a VT 220 terminal, the following string must be written to the terminal: "ESC[j;kH" [Digital Equipment Corporation, 1984]. Naturally, each terminal model may require a different string to achieve the same effect. Similarly, particular visual attributes are created by writing strings peculiar to the VT 220 hardware.

The panel description language used in this project is based on the IBM Interactive System Productivity Facility and has many of the same options and features. In the version used here, there are three sections to the panel language - an attribute section, a body section, and a processing section.

The attribute section provides three default attribute characters for predefined visual and logical attributes for the display of the panel text and variables. In addition, if the user prefers, he may redefine the default characters, or he may add characters and define his own attributes.

The body section is that part of the panel where the developer specifies the text to be shown in the panel display. Along with the text, display details are
specified via the attribute characters, one of which precedes each area or field of text. The body section may be as long as desired, since a panel may be scrolled within a window. In addition to ordinary text, the developer may choose to utilize fields containing variables in the panel. There are two types of variables, input and output, and these are often referred to as window variables.

An input field has the feature that it may be modified by the user. The input value is stored in a window variable. Modification of window variables may be achieved dynamically during the running of the display routines, or it may be realized through the user's application program which may make use of several window variable modification routines. Input variables also may be used to specify any one of a number of options available in the processing section of the panel. A field containing an input variable has visual and logical attributes, just as text is specified.

The output fields are not available for direct modification, but they may refer to the value of a window variable which may be modified indirectly. For instance, there may be an input field referencing a variable named value1 in a panel, and somewhere else in that panel, value1 may be used in an output field. Actually, these are referring to the same variable. Any change to the input field also changes the value of the output field. The
display routine will dynamically display the new values for both the input and output variables as it redisplay the panel. An output variable is specified by the single character '&amp;' followed by the variable name, and inherits the attributes of the field containing it.

The processing section is optional and follows the body section in the panel. In the processing section the developer typically assigns to a selection variable the value of a particular input variable in the panel. Based on the value of that variable, the panel display routine will display another panel as dictated by the user. In a sample scenario the user may have six panels related to one another and associated with one window. The first panel may be used as a master menu and be arranged so it is displayed first. In that first window there may be one input variable which is used as a selection variable for the processing of panel number one. Based on the value entered for the selection variable, the panel display routine may be terminated or any one of the other five panels may be displayed. As another of the panels is displayed, its processing section is executed (if present) and any of the other panels, including the master panel, may be displayed. Eventually control returns to the calling panel and the stack of calling panels is unwrapped one by one.

A typical program application would utilize the panel
package as follows. The user/programmer would design his panels using the guidelines set forth by the panel description language. He would then call PDT, the panel language translator developed by Dr. Haynes, and each panel would be translated into a form we shall call the intermediate language. This intermediate language is the form that the panel display and processing routines use as input for their processing. The user then incorporates into his application program any of the panel functions he wishes to use. Naturally he would want to first define an area or areas on the screen (a window) that he wishes to use as the display area. Next he may choose one of his previously designed panels to be displayed initially in that window. In addition, he may initialize some or all of the window variables defined in the panels linked with the window. Normally the actual display and processing routines would be called at this time. At any time during the application he may also access the values of the window variables to use them for any processing his application may require.