Using Predict Documentation

for

Online Help

W. David Wimberly

Computing Services
University of Arkansas,
Fayetteville

March 6, 1989

Fifth Annual SAG University/College BIg Conference
OVERVIEW

This document describes standard field and screen help routines that have been developed by the University of Arkansas for use in Natural online applications. These routines provide sophisticated help facilities in a consistent and uniform manner, yet are extremely simple to implement. The topics covered include:

- why these routines were developed
- examples of the online help provided
- availability of hardcopy documentation
- considerations made during development
- what standards must be followed
- list of components used
- known limitations

These routines are not intended to meet all possible needs. There will be occasions for deviating from their use and writing unique and specific help routines. However they do offer powerful features in an easy to use fashion and should meet most needs.

OBJECTIVES

The objectives of providing standard help facilities were:

- Provide help facilities at the field and screen level but eliminate, or at least minimize, the need for writing custom code for that purpose.
- Where possible, provide a list of values for a field and allow selection of a value to be returned to the primary map.
- Maximize the use of the documentation that is prepared for an application.
- Eliminate the need for duplicate maintenance of documentation.
- Provide a means to generate in hard copy form, the same documentation that is available through online help.
- Minimize the amount of code required to provide these functions.

These objectives have been met by using Predict as the repository of information about fields and programs, Script as the documentation and formatting tool, and a set of centralized routines to support the process. Field documentation is maintained in Predict's COMMENT field, application documentation is prepared in GML (generalized markup language) using Waterloo Script under CMS, and the centralized routines reside on the Natural SYSTEM library and are used by including the appropriate "HE=" reference within a map. New code must be written only for subprograms needed to access unique files or tables in order to provide lists of possible field values.
EXAMPLES OF ONLINE HELP

Field help

Field help is available in two flavors. The differentiating factor is whether or not there is a set of discrete values which can be listed so that a value can be selected and returned to the map. This is the case with SEX where the codes used to represent male, female and perhaps unknown could be listed. This is not possible for a field like STREET-ADDRESS. When there is not a set of values to be listed, the help routine UAHTHER is specified as the field help routine. This routine will display, within a window, the element definition (Comments) from Predict. See Figure 1 for an example of a screen where UAHTHER has been invoked for the field NOTICE.

Make changes and hit ENTER to validate

NSMAN
Command: App
Action: U Cmd Se
Action: U Appl: NSM

Application Notice maintenance - AN

------------------------------------------

02/21/89 17:10
Notice Eff Date:

------------------------------------------

NOTICE
Notices which are available for broadcast to all application users, keyed by the date of the notice and the user's date of last logon. A total of 15 lines per notice is the maximum allowed. This field is maintained in mixed case.

Last Updated: 02
Enter-PF1---PF2---P Help Menu Q
PF1 PF3/ENTER Help Quit F9---PF10---PF11---PF12--- Save

Figure 1: Help for a field without a discrete set of values

Routines which provide a list of values and allow a value to be returned to the map, require a parameter that matches the type and size of the field. Therefore several field help routines are provided of this type, one for each possible size and type of field. These routines are named: UAHTFA1, UAHTFA2, UAHTFN1, UAHTFN2, etc. where the name is UAHT followed by the Natural type and size format. These routines first display the element definition from Predict. If the definition includes a list of values, a selection may be made directly from that screen by positioning the cursor
to the desired value and pressing ENTER. This Comment format is recommended for fields such as SEX which have a limited and fixed set of values. See Figure 2 for an example of a help screen invoked for the field APPLICATION-LOCK, which has a fixed set of values.

If the possible field values are on a file or table, an option is provided to enter a starting value to be used in reading the file. A second screen then displays the list of possible values and allows either the selection of a value or the specification of another starting value. The starting value is prefilled to the next available value until the end of file is encountered. Note that the starting value does not have to be the same field for which help is being provided, e.g. help for SSN may list SSNs in name order and allow specification of name as the starting value. See Figure 3 on page 4 and Figure 4 on page 5 for examples of the two help screens invoked for the field USER-ID, where potential values are read from a file.
Screen help

Screen help, as used here, refers to the need to display one or more full screen "pages" of text. Typically this is to provide the operator with information regarding the use of a specific screen. The logical place to store this information is in Predict's extended description for the map being executed. Additionally, it is sometimes beneficial to provide other general use information that is common throughout an application. We chose to store this text in Predict's extended description for conceptual fields since it could be accessed and displayed in a similar manner as program descriptions. A help routine may then provide a menu of help topics and, based upon the user's selection, call the routine which will display the appropriate text based upon parameters. See Figure 5 on page 6 for a selection menu of help topics and Figure 6 on page 7 for a screen help example.

The screen help text is read from Predict's extended descriptions and displayed on screen pages of 23 lines each. Predict limits each line to 72 characters. The current, last and next page numbers are displayed to the right of the help text. The "Next page" value may be modified in order to back up or skip ahead in the text.

This screen help facility is available by specifying either NSHSHM or UAHSH as the default help routine for a map. NSHSHM is written specifically for function modules.
of applications which use the NSM Architecture and displays the menu of help topics as shown in Figure 5 on page 6. The map help routine is specified on the "Map Settings" screen of the map editor as either 'NSHSHM=', or 'UAHSH=', with the parameter "as field default" of N.
Make changes and hit ENTER to validate
NSMU User maintenance - U
Command: _____ Appl: NSM-MS User: Z38 Menu Cmd:
Action: U Cmd Sec Grp: Val Sec Grp:

Action: U User ID: Z38

SSN: 465-96-2408
Name: Wimberly, W. David
   Last Comma First and Midd
Short Name: DAVIDW

Department CD: No: 2067
Name:

Program Library: Dw85047 Proj
Password: 9999

Last Updated: 10/05/88 By: Z38

Select a topic and press ENTER
Help information is available for the following topics:

- Command U
- Commands in general
- Menus
- Key fields

Mark or position the cursor to the topic desired.
Enter PF1---PF2---PF3---PF4---PF5---PF6---PF for help info
Help Menu Quit DCode RStrt
Quit

Figure 5: Screen help selection menu
Enter the next page number to be displayed or PF3 to quit

Commands

Commands provide the transporation for traveling to various destinations
within an application while menus serve as roadmaps. More precise
definitions for these components are:

1. A menu is a list of functions or sub-menus available within an
   application
2. An application is a collection of functions designed to address a
   general need
3. A command is a short name used to identify an online function or menu

Control is passed from one function or menu to any other by entering the
desired command. A command will usually utilize only one screen,
however some complex functions may require several screens or windows.

When a value is entered in the 'Command' field, control is transferred
to the command dispatcher which validates and processes the request.
The dispatcher determines if the command is a menu or a function and
either displays the menu or transfers control to the function. If the
command is invalid, the previous menu is displayed along with an error
message.

Figure 6: Screen help

HARDCOPY DOCUMENTATION

A major goal of providing online help facilities was to eliminate any dual mainte-
nance of documentation. Since writing documentation is one of the most dreaded
tasks in applications development, it is natural to want to maximize the use of what
documentation is prepared. The idea is to prepare documentation that serves the
needs of an application from the point of systems analysis and design through appli-
cation use and maintenance. The areas that were felt to be most important were the
definitions of the elements used within an application and the discrete functional
processing which makes up a system, essentially the screens.

Predict is the obvious choice for storing and maintaining field definitions. The
University of Arkansas chose to write our own dictionary report to generate the
needed hardcopy documentation of these field definitions. An example of the output
produced can be found in the appendix of the paper "Natural Secured Menus Architec-
ture", which is included in the proceedings of this conference. Some of the reasons
for developing our own report are identified in the section "Development Considera-
tions" on page 8.
The use of GML and Script for the preparation and formatting of text was also a natural choice for the University of Arkansas. These tools had been used for years to produce both formal and informal documents. We were comfortable with their capabilities and knew that through the use of parameters, a document could be formatted with different physical dimensions while still incorporating the same formatting features (heading levels, indentation, lists, page references, etc.). Given this approach, each function of an application is documented in an individual file. These are processed and loaded into Predict's extended description for the associated map. To produce a reference manual, these individual files are imbedded by a driver file to produce the needed hardcopy documentation. Screen images from the applications are also saved and imbedded to provide a complete reference. Further details on how this is accomplished are included in the section "Development Considerations". The "Natural Secured Menus Architecture" paper, included in the conference proceedings, is an example of this approach since some of the sections have been loaded to Predict to serve as Help information (note that Figure 6 on page 7 is contained in that document).

DEVELOPMENT CONSIDERATIONS

There were many considerations made during the development of these routines. Understanding what some of the issues were and how they were addressed may be helpful in the design and development of similar facilities.

Field Help

To assist in understanding how field help processing is performed, an overview is provided in Figure 7 on page 9. This diagram depicts the flow of processing that occurs for any field where a value is being returned. Reference to this diagram may be beneficial when reading the discussion below.

Multiple help routines are required in order to match the size and format of the field value, which is passed as a parameter. Although the code required in the field help routines is not extensive, there was a great reluctance to maintain a copy in each help routine since the processing is essentially identical for all field types. Copy code could have been used, but instead we chose to develop one subprogram to serve all needs, UANFMCT. This is accomplished by having each help routine call the subprogram with a generic field of format A20. Upon return from the subprogram, the help routine transfers the A20 value returned to the original parameter field. This is made to work for numeric fields by first checking their validity with the "IS" logical condition criteria and then using the "VAL" function to make the conversion.

There is definite need to have one master location for the definition of a field. Predict standard files had been used but were accompanied by restrictions the University of Arkansas could not live with. Additionally, standard files do not tightly couple the field Comment, which was our primary need. For these reasons, we created our own pseudo standard files. These were merely conceptual files where, by definition, we would store a master list of fields with their official definition in the Comment. We chose to have a universal file, UA-FIELDS, where elements common to
multiple applications are maintained. Additionally, each major application area has its own standard file, aa-FIELDS. The aa represents an acronym and prefix that is assigned for all application areas. Field definitions may then be obtained by first looking to the application's standard file and, if not located there, on to the universal file. The main advantage of this approach is to avoid having to update a fields definition in multiple files.

With field definitions centrally located, a report was needed that would use this definition regardless of the file being reported. This was accomplished by developing a custom dictionary report program which would read the description from aa-FIELDS or UA-FIELDS. The value "aa" is obtained from the Predict KEYS for the file being reported. The online help routines obtain this same prefix from the APPLICATION file for the application currently being executed.

Another problem associated with locating a needed field definition is the field name itself. Within a program and the associated map, the same logical field may have various and multiple names. Through the field help specification, you may specify a literal name (must be very short) or a variable that contains the name of the field for which help is requested. However the easiest and most convenient specification
is "=", which will pass the name as it is defined on the map. To facilitate this usage, the subprogram UANFHTC strips away any characters up to a period (the view name), any leading "#" (typically used for local fields), and any trailing "-D" (denotes a date data type field which was a redefinition of a P6 data base field, required since Adabas V4 does not support date data types). With these provisions and some forethought, the help routine for most fields can then be specified as: 'UAHFxx=',.

When possible, it seemed logical to include the values for a field within the fields definition. This significantly contributes to the meaning provided by the field definition and can be used by the help routine to provide a value selection. The need was for a uniform way to distinguish the field values from the other text included in the definition. Since a description for each value is usually appropriate, the format of one value per line with the syntax "value = description" was adopted. If the cursor is on a line with an "=" then the value to the left is returned as the value selected.

When values are read from a table and listed for selection, the same format of "value = description" is used for consistency. This is accomplished by having the subprogram which reads the table also format this data and place it into an array which is displayed by UANFHTC. The more significant issues were: what subprogram should be called and, since we wanted to allow the specification of a starting value, how do we prompt for a starting value? This information is needed by the UANFHTC subprogram. Since it varies for each field and the Predict element record is already being accessed, it was logical to place these two pieces of information within the element definition on Predict. We chose the seventh synonym, currently unlabeled but maintainable from within Predict, to store both the name of the subprogram to be called and the prompt to be used for a starting value. This information may have to be relocated if a future version of Predict uses this occurrence. Figure 4 on page 5 is an example of how the end result appears to the user.

Screen Help

An overview of the processing that occurs for screen help is shown in Figure 8 on page 11. A discussion of some of the problems encountered in providing this generalized facility follows.

First, the GML layout used for preparing manuals was not appropriate for small discrete pieces of documentation that were to be prepared and presented individually. The layout GMLfoils, which is designed for the preparation of overheads (foils) for presentations, was found to work very nicely. The layout itself required some slight modifications and the heading level for the main topic needed to be zero. Generally, a topic's heading level within the context of a document was always some greater value. This was addressed with the use of a utility to pre-process the Script input and change all heading levels so that an H2 became an H0, H3 an H1, etc. A Script driver was then developed to establish the necessary parameters for the online environment: page size of 72 characters by 24 lines, horizontal and vertical spacing for different tags, and the definition of a bottom title to include the last page number generated (more on this later). This driver then includes an imbed for the preprocessed Script text, which is provided as a parameter. The driver then serves as the primary input for Script.
The output produced by Script also required processing before it could be stored in Predict. Since the destination of Script output is normally a printer, the file that is produced contains ASA carriage control. The overstrike, double space, triple space, and top of page characters had to be accommodated by the program written to load the text to Predict. Development of the program to load the text was only made difficult due to Predict's file structure and standards. Predict's extended descriptions are maintained within an MU field 27 occurrences (lines) per record within records keyed with a binary record number where the last record number value is H'9999'. This format is expected by Predict's extended description editor. By using this format the text can be viewed or updated from Predict, although the intent is that the documentation be maintained under CMS and the Predict copy be rebuilt following any update.

Displaying the help text in a screen format also required compliance with the format of the text under Predict's extended description. Given that the technique for loading the text had already been worked out, retrieving it was not a problem. This fixed structure allows the location of any page of text desired to be computed and accessed by reading no more than two records. This made it possible to provide as an input option, the next page number to be displayed. The only remaining problem was preventing the specification of a page number beyond the end of the text. It was also desired to let the user know, from the beginning, how many total pages of text were available for viewing. To provide this "last page" information to the subprogram UANSHSS, it was decided to place it within the text. This is a common requirement in the preparation of hard copy documents and easily accommodated by Script. The procedure is to process the document in two passes and include a top or
bottom title that references a variable whose value is assigned at the end of the
document. The result we choose was to include a right justified bottom title of
"Screen x of y". This will then occur on every 24th line of the text. The online
program does not display this line but reads and uses the last page value that is
stored there.

Help text was needed for some general topics in addition to specific maps or func-
tions. The obvious place to store this text was under some other Predict entity's
extended description. Since there was no entity that was best suited for this mis-
cellaneous type information, we chose to associate it with conceptual fields. These
are not true fields, but are defined on the conceptual files UA-FIELDS or aa-FIELDS
solely for the purpose of storing this help text. The index used for accessing an
element's extended description is slightly different from other entity's extended
descriptions, but the text is stored in exactly the same elements and the same for-
mat. It was then relatively easy to accommodate which type of entity you wish to
process as a parameter to the programs that access this data. This allows a program
to display a menu of help topics, establish the appropriate parameters based upon
user input, and then call UANSHSS to display the text.

The screen help facility is invoked only if the cursor is not on a field that has
its own field help routine defined. The idea of having a user position the cursor
off of any input field and then press the PF key for screen help is not desirable.
Therefore it is necessary that an input field be defined without field help in order
to allow access to the screen help. As an example, NSM applications have no field
help defined on the 'Command' field. The routine NSHSHM was developed with this
knowledge and provides an option for help on the field Command. Similar routines
can be developed for other application architectures and can invoke the subroutine
UANSHSS which actually does the screen displays.

Regardless of all other considerations, the primary adjustment has, and continues to
be, writing documentation for an application that not only serves as online help
but, when combined with other parts in a manual format, serves as a definitive refer-
ence source for an application. The goal being to maintain only one type of docu-
mentation throughout the life of an application. Planning to use documentation for
online help significantly influences how and what is written. This goal may yet
prove to be too difficult, the jury is still out since applications using this
approach at the University of Arkansas have not been tested by fire, the production
users.

STANDARDS

Adherence to certain standards is required for the effective use of these help facil-
ities. Many of these are the responsibility of the dictionary administrator
and will require coordination with the project leader to see that they are properly
implemented.

1. Each application must have an application area assigned and available to an
online program. This is a two character prefix at the University of Arkansas
and is stored on the APPLICATION file. This application area is used to build
the conceptual file name where the routines will look for an element's defini-
tion. See item 2.
2. The definitions for all elements must be stored within Predict's element comments on either the UA-FIELDS or aa-FIELDS files. The aa refers to the application prefix previously mentioned. These conceptual files serve as a centralized source for the element definitions. This is similar to Predict's standard files but without the restrictions that accompany standard files. Note that a custom batch dictionary report is required to report the element definitions from this same centralized source.

3. Fields which include their possible values within the element description must have these listed one per line in the format: value = description (e.g. M = Male). No other use of the equal sign should be used within the description.

4. It is most convenient if the field names on the map correspond to the dictionary names, since that will allow the ",=" syntax to be used when specifying the help routine. (This designates that the field name as defined in the map will be passed to the help routine as a parameter.) The help routine will strip off any characters preceding a ",." (view names), any leading ",#" (equivalent locally defined field), and any trailing ",D" (indicates this is a Natural date field that had to be redefined since Adabas V4 does not support D data types). The routine then searches for this field definition on the aa-FIELDS file and then the UA-FIELDS file. If the field name on the map does not correspond to the dictionary name, the name will have to specified as a literal or assigned to a variable that is specified with the help routine.

5. Fields which require table access subprograms, require those subprograms to be written and available in either the application library or in the system library.

6. Fields which require table access subprograms must have the starting value prompt and the table access subprogram name specified in Predict as the seventh synonym. The syntax is ssssssssssppppppp where the s's represent the subprogram name and the p's represent the description to be used in the starting value prompt. The colon is required in position nine.

7. The documentation for screen help must be prepared in GML or formatted suitably by some other means (word processor). If GML is used, the text must be isolated in a file of its own (may be included or imbedded in a larger document for generation of manuals). Additionally, the highest GML heading level in the file must be two (:h2). If the text is prepared by some other means, the first column must be standard ASA carriage control (blank is acceptable) and every 24th logical line must be a page footing with the last page (screen) number in columns 71 and 72.
COMPONENTS

A complete list of all components used by the UAF help facilities is provided below. A programmer using these facilities need only be aware of the help routines and the model subprogram UANHxxxx. Other components are included to provide complete documentation regarding these facilities.

Help routines

UAHOTHER The help routine which displays an element definition but does not allow a value selection.

UAHFxxxx The help routines which display an element definition and also allow selection of a possible value, either from the definition or from a table. The xxxx represents the field's format and size, e.g. an A8 field would use UAHFA8.

UAHSH The help routine which displays Predict's extended program description, for the map being executed, in online paged format.

NSHSHM The help routine which displays a menu of help options specifically for function modules of NSM architecture applications. The options include help for the map being executed and for definitions of commands, menus and key fields.

Subprograms

UANHxxxx The model table access subprogram which should be used as the starting point for any field for which table values are to be listed.

UANFHCT The subprogram which performs the field help processing for any field where a value is to be selected and returned. All UAHFxxxx routines call this subprogram.

UANHSSS The subprogram which reads pages of text from Predict's extended descriptions and displays them on the screen. This routine will work for any Predict entity type that has an extended description that has been properly formatted. It is called by NSHSHM and UAHSH to do the real work.

Copy Code

UACFHFND The code that determines the field name to be used when searching Predict for the field definition. It strips any leading view name or '#' and trailing 'D'. It is used by UAHOTHER and UANFHCT.

UACFHRFC The code that reads the field comments from Predict. It first searches the application specific file aa-FIELDS and the universal UA-FIELDS and includes error handling when nothing is found. It is used by UAHOTHER and UANFHCT.

M10-16
Local Data Areas

UALFH  The local data area used by routines that provide field level help (UAOTHER and UANFHCT).
UALSH  The local data area used by the routines that load or provide screen level help text, UANSHSS and PDBUPDD.

Maps

NSMSHM  The map used by NSM SHM to present the available help topics from within an NSM Architecture function module.
UAMCOMM  The map used by UANFHCT when table access is not involved.
UAMHELPC  The help text provided when UAMCOMM is used.
UAMHELPO  The help text provided when UAOTHER is used.
UAMHELPN  The help text provided when UAMTABLE or UAMTABLE2 is used.
UAOTHER  The map used by UAOTHER to display an element definition.
UANSHSS  The map used by UANSHSS to display extended descriptions as screen level help.
UAMTABLE  The first map used by UANFHCT when there is a table access routine for an element. It includes the element definition and allows entry of a starting value.
UAMTABLE2  The second map used by UANFHCT when there is a table access routine for an element. It displays a list of values and allows value selection or entry of another starting value.

Batch programs

PDBUPDD  The batch program which reads formatted text from a work file and loads it to Predict's extended description area. The text may be associated with a program or an elementary field on Predict.
PDXUPDD  A CMS EXEC that spools a batch job to alter the heading levels within a script file, script the text using GMLoth (specialized version of GMLfoils), and load the results into Predict with PDBUPDD.

LIMITATIONS

There are several known limitations of these help facilities. However, the intent is that these facilities meet most requirements for online help in a productive and efficient manner. Situations where these facilities are not appropriate can still be addressed by developing custom routines. Where patterns of special needs develop, it will always be possible to expand the core help facilities to address additional and more specialized requirements. Current limitations of the facilities include:

1. Strict adherence to standards is required and includes the coordination of numerous components (element names on maps that correspond to Predict fields, table access subprograms that are identified within Predict field definitions, special formats for program documentation so that it can be loaded to Predict, etc.).
2. The number of components that must be maintained in-sync with each other is significant and complicates the management of test and production environments.

3. Field definitions, at least for these help facilities, are limited to the size of Predict's Comment field, 30 characters by 16 lines.

4. Table or file access for fields which require multiple keys is not accommodated since there is only one starting value field. An example is the need to look up institutions on a table by name within state. Starting values for both state and institution name are needed.

5. There can be only one table access routine defined for a field. This prevents one program from providing table access for institutions via state code if another program is already providing table access for institutions via institution name.

6. A map may include the same field several times and therefore must have unique names for the field, however the help facilities only accommodate a limited number of variations for naming the same field given the ",=" notation is used for the help routine.

7. Local program fields that do not exist on a file can not use these facilities unless the element is added to Predict's aa-FIELDS file. Similarly, fields which are broken into multiple fields for display and entry on a map can only be accommodated if the sub-fields are defined on the aa-FIELDS file. There may be some reluctance to define such fields within Predict.

8. There is a heavy reliance and responsibility on the dictionary administrator to maintain all definitions correctly.

9. There is a danger that the use of the seventh synonym on Predict may be designated for some purpose in the future.