DataProbeCLASSIC - A NEW VERSION OF THE CLASSIC DATA-ANALYSIS TOOL

John McCormick
Paul Ferrill
Avionics Test and Analysis Corporation
4540 East Highway 20, Suite 6
Niceville, Florida 32578

ABSTRACT

DataProbeCLASSIC is the new PC-based version of the classic tool for telemetry data analysis and visualization.

DataProbe was the brainchild of the Unites States Navy and its contractors. At a time when computer terminals were expensive and graphical visualization of data was cutting edge, this software product was specifically designed to process time-series data in an efficient manner.

The primary strength of DataProbe is the capability to read specific data items for specific time slices from very large data files rather than reading the entire data file into memory. The efficiency and versatility of the product was quickly noted, and it gained widespread use within the testing community.

This paper presents a brief history of the legacy product and discusses the features and strengths of new implementation.

KEYWORDS

Data Analysis, Visualization, Probe, Software, Time Series

INTRODUCTION

DataProbe was the preferred tool for data analysis and visualization for many Navy, Air Force, and Department of Defense (DoD) test programs. It was the brainchild of John Means of the Naval Underwater Warfare Center (NUWC) in Newport, Rhode Island. His vision for an efficient data-analysis tool was realized, and the documented savings by Navy programs amounted to tens of millions of dollars.

NUWC and the Avionics Test and Analysis Corporation (ATAC) worked together to resurrect the product and re-introduce it to the testing community. The new DataProbeCLASSIC implements the most useful features of the legacy product and allows users to leverage existing script files which represent a considerable expense and an irreplaceable cache of corporate knowledge.

DataProbeCLASSIC highlights:

- User-configurable data dictionary
- Read-access to arbitrary data sources
- Specifically designed to handle large sets of time-series data
- Command-line interface for ad-hoc analysis
- Extensive facilities for visualizing data with plots and tabulations
- Built-in high-level programming language for creating detailed analysis procedures
- Batch and interactive analysis programs can be created
- Standard mathematical and arithmetic operators
- Wide array of time-series operators and built-in functions
- Many built-in utilities for dealing with data sources and dictionaries.

This paper will discuss many of these features.

HISTORY

DataProbe was developed for NUWC by Bolt, Beranek, and Newman (BBN) Systems and Technologies. The product was widely used by the Navy in torpedo test programs. The Navy enjoyed considerable cost savings during numerous test programs due to the speed and efficient data-handling inherent in the design of DataProbe.

BBN maintained separate Navy and commercial versions of DataProbe for a time but eventually merged the products into a single version. Later still, the DataProbe operation was spun-off into a separate company. The product was ported to the PC, but this version never achieved the robustness and reliability of the previous Virtual Memory System (VMS) and Unix versions. Support for the PC version continued for several years, but the other environments were abandoned. For various reasons the product became too expensive to maintain, and support for DataProbe ceased entirely.

The Navy contracted with ATAC to resurrect the product and create a PC version based on the source code the Navy obtained via the escrow clause in the license agreement. ATAC created a project in Microsoft Visual Studio 6.0 that allowed the old code to be recompiled. This task was particularly challenging since the source code was written in C and Rational FORTRAN (RATFOR) and made use of a UNIX-to-NT translation program. ATAC produced several new releases of DataProbe, most recently version 3.5b released in 2002.

In 2003 the Navy decided to fund a project to rebuild DataProbe and create a modern, object-oriented software product that would be easier to maintain and could be distributed to other government and DoD organizations.

ATAC employed the latest Microsoft Visual Studio .NET environment and created the first alpha release in early 2004. After a series of beta releases in 2005, the first new production version of DataProbeCLASSIC was released in early 2006.

The current production version is intended to be a work-alike version with the same look-and-feel, command-line interface, scripting language, and menu systems employed by the original product. This allows users of legacy versions to leverage existing data dictionaries and script files created for earlier versions of DataProbe.

TIME SERIES DATA

A time series is a sequence of data points measured at successive times. Telemetry data is typically expressed as a times series. DataProbeCLASSIC was designed specifically to provide the data analyst with an efficient means for processing time-series data. For example, Figure 1 shows a time series containing range data created with DataProbeCLASSIC. Note that each data value has an associated time.

```
+ show sequence range

range.value (1:700) [=(778.282, 631.770, 962.603, 1006.582,...)]
range.time  (1:700) [=(00:00:01, 00:00:02, 00:00:03, 00:00:04,...)]
```

Figure 1 Time Series Representation

The user is able to plot, tabulate and perform calculations on time-series data simply by using the name of the sequence or variable in a command. Figure 2 shows some example commands.

```
+ define sequence range_error = true_range - range_estimate

+ tabulate true_range, range_estimate, range_error
```

Figure 2 Commands with Time-Series Data

The fact that these commands use time-series data represented by two arrays of data is completely transparent to the user.

DATA DICTIONARY

The data dictionary contains the information about each data item (called a variable) that resides within a data file. The dictionary contains information such as the location of a variable within a frame of data, the data type of the variable, and the engineering units of the variable. There is no limit to the number of variables in a dictionary. The data dictionary consists of three files that are created and maintained with a set of menu-driven DataProbeCLASSIC commands.

The dictionary file (DIC) contains basic information about the variable, such as the name, engineering units, maximum/minimum values, and print format.

The format-descriptor table (FDT) describes the frames where data can be found, where the data are located within the frame, and how to unpack the data and convert it into engineering units.

The code file (COD) contains information for coded variables. A coded variable has integer values that can be translated into user defined text strings when displayed. Coded variables are commonly used to represent flags or status indicators.

An existing dictionary can be opened as either:

- System dictionary.
- User dictionary.

System dictionaries are always opened in read-only mode. The system dictionary is typically under the control of the data manager, contains all possible variables, and is uniform for all users. User dictionaries can be created by each user. They may contain a subset of the system dictionary, new variables, updated variables, or ad hoc variables for testing or other special cases. If the same variable name appears in both dictionaries, the definition contained in the user dictionary always takes precedence. The current implementation allows each user to have one system and one user dictionary open simultaneously.

COMMANDS

DataProbeCLASSIC uses a command-line interface along with a sophisticated command parser. The parser allows the user to enter command shortcuts, repeat commands, and quickly modify command qualifiers and parameters. The parser checks user input for validity and gives meaningful error messages. Context-sensitive and searchable help is available for all commands.

Commands can be entered from the command line or from script files. A script file is just a plain-text file that contains DataProbeCLASSIC commands. The set of commands, macros, and operators form a high-level programming language. It is sophisticated enough to allow users to create suites of complex data-analysis routines that can run interactively or in batch mode.

DataProbeCLASSIC commands fall into these categories:

- Create conditional logic blocks (if-then-else).
- Create/edit data dictionaries.
- Create/edit variables.
- Create subroutines.
- Define events/functions/sequences/symbols.
- Draw text/lines/arrows.
- Dump hexadecimal files.

- Create iterative and explicitly terminated blocks (repeat-until)
- Add labels to plots.
- Open/close data sources.
- Open/close output/log/journal files.
- Open/close windows.
- Plot data.
- Run/Step through script files.
- Save settings/events/functions/sequences.
- Set options.
- Search for events.
- Tabulate data.

For example, Figure 3 shows commands that open a data set and create a y-versus-x plot for a user-selected time slice.

```
; Text preceded by a semicolon is ignored.
;
+ @open_data.prb                          ; Run a script file to
                                          ; open the data.

+ set system = sysdict                    ; Set the dictionary.

+ plot ( vehy vs vehx, targy vs targx )   ; Set plot parameters.
: start = 20
: duration = 15
: sample_interval = 0
: color = ( blue, red )
: done                                    ; Create the plot.
```

Figure 3 Example Commands

TIME-SERIES OPERATORS

DataProbeCLASSIC has many built-in commands specifically designed to process time-series data. The time-series data may reside in a variable or in a sequence. Chains of expressions with calculations and logical operations allow the user to extract useful information from the time-series data quickly and efficiently.

The standard mathematical and arithmetic operators (+, -, *, /, trigonometric functions, logical operators, inequalities, exponentiation, max/min, absolute value, etc.) are included along with a large number of special-purpose operators that manipulate and filter time-series data.

Several unique operators are especially useful for finding events of interest that would be difficult to find otherwise. The $when, $while, $is_present, and $merge operators used with inequalities and logical expressions make it possible to locate multiple data points and single events of interest.

Figure 4 shows examples that use time-series operators to define sequences that contain useful information that would be difficult to obtain with other data-analysis tools.

```
+ define sequence bad_range = true $when($abs(range_error/true_range) > 0.10)

+ define sequence alt_at_bad_range = ac_alt $when($is_present(bad_range))
```

Figure 4 Time-Series Operators

MACROS AND BUILT-IN SYMBOLS

Macros and built-in symbols provide useful tools to extract information from time-series data, process text strings, process event information, and other macros that are useful for creating automated data-processing routines.

Figure 5 shows examples where the maximum value and the time of the maximum value are extracted from a sequence.

```
+ let max_alt = $extract_value( maximum, ac_alt )

+ let time_at_max_alt = $extract_value( t_at_max, ac_alt )
```

Figure 5 Extract Value Macros

SCRIPTING LANGUAGE

DataProbeCLASSIC has a built-in, full-featured, high-level programming language. The scripting language makes it easy to automate any manual task and is limited only by the imagination of the data analyst.

The data analyst can create small routines that perform mundane tasks (like opening data files) to increase efficiency. Entire suites of routines can be created to automate time-consuming data analysis tasks. These routines can be interactive or run in batch mode.

For example, a set of scripts could be designed to perform the following tasks:

1. Open the data files and perform validity checks.
2. Determine the start/stop time for each run in the mission.
3. Build sequences for variables of interest using the start/stop times for each run.
4. Save the sequences for later use in more detailed analysis.
5. Find times for events of interest and test point values that occurred during each run.
6. Create and save a set of plots to help the analyst visualize the results of each run.
7. Create and save a set of tabulations that allow the analyst to drilldown and research problems.
8. Compare measured data points to truth data to characterize specification compliance.
9. Build automated reports that summarize the results of each run.
10. Compile the saved plots and tabulation results into a final report.

DATA PLOTS

DataProbeCLASSIC provides a rich set of options for plotting data. The user can easily create line and scatter plots for specific start/stop times with custom labels on the axes, custom legends, automatic and interactive labels, and specific minimum/maximum axis limits. Custom plot layouts are available that allow multiple plots per page. The user can zoom and un-zoom individual plots or groups of plots. A special y-versus-x plot can also be created.

Plots are displayed in a separate plot window. The user can open several plot windows simultaneously. Each plot window has an interactive menu that allows the users to change the colors of the data plots, plot panel color, plot window background color, and add/move/delete labels.

Plots can be displayed in an animated mode with a user-selected animation rate. This feature is especially useful for visualizing moving vehicles and targets that are displayed in a y-versus-x plot. The contents of the plot window can be printed, copied to the paste buffer, or saved in one of four available file formats. Figure 6 shows an example with six plots displayed in a single plot window.

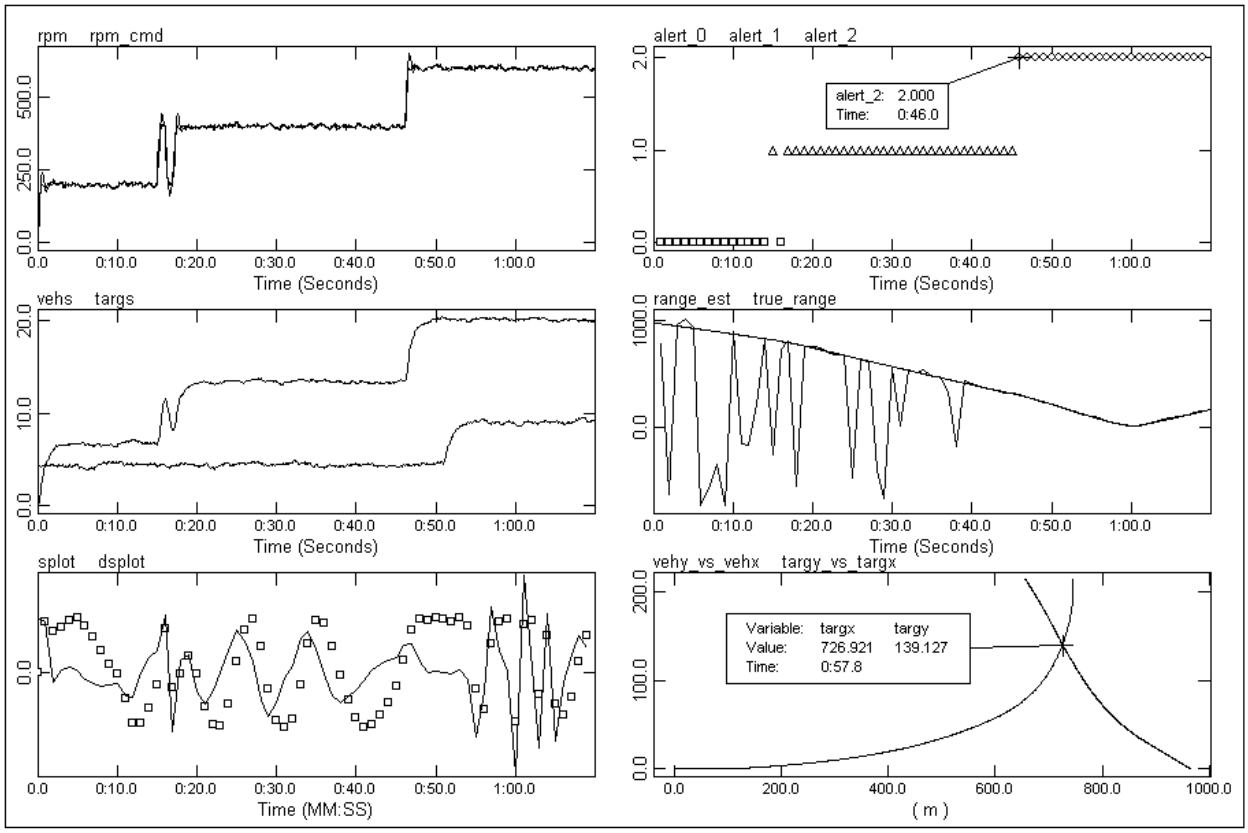

Figure 6 Example Plot

TABULATIONS

Tabulations are columnar tables of data values along with the associated time values. The output can be written to the screen or to an output file. The user has many options for formatting time and data, selecting change-only options, inserting underscores, and setting the general apperance of the tabulation. A column delimiter can also be added to the tabulation. For example, if a comma is chosen for the delimiter, then a comma-separated value (CSV) formatted output will be created. These files can be imported directly into Microsoft ® Excel.

Unlike the legacy versions of DataProbe, the data columns in tabulations are always perfectly aligned. Values that are too large or too small to fit in the selected print format are converted to scientific notation. Invalid results from floating-point calculations are shown as "NaN" in tabulations. Floating point operations that result in infinite values are shown as "Infinity" with the appropriate sign. Figure 7 shows a tabulation example.

Time (secs)	test_data	test_exp	test_exp_10	test_log
0.000000	0.000000	1.000000	1.000000	-Infinity
1.000000	-1.000000	0.367879	0.100000	NaN
2.000000	-4.000000	0.018316	0.000100	NaN
3.000000	-8.000000	0.000335	1.00000E-008	NaN
4.000000	-10.000000	0.000045	1.00000E-010	NaN
5.000000	-14.000000	8.31529E-007	1.00000E-014	NaN
6.000000	-18.000000	1.52300E-008	1.00000E-018	NaN
7.000000	-22.000000	2.78947E-010	1.00000E-022	NaN
8.000000	-26.000000	5.10909E-012	1.00000E-026	NaN
9.000000	-100.000000	3.72008E-044	1.00000E-100	NaN
10.000000	-1000.000000	0.000000	0.000000	NaN
11.000000	10000.000000	+Infinity	+Infinity	9.210340
12.000000	100000.000000	+Infinity	+Infinity	11.512925
13.000000	1.00000E+009	+Infinity	+Infinity	20.723266
14.000000	1.00000E+012	+Infinity	+Infinity	27.631021
15.000000	1.00000E+015	+Infinity	+Infinity	34.538776

Figure 7 Tabulation Example

INCREASED EFFICIENCY

The main goals for this project were to create a backward-compatible product that is easier to maintain and upgrade. These goals were achieved. In the past, bugs took days or, in some cases, weeks to fix. The new development environment coupled with the object-oriented design allows bugs to be fixed in a matter of hours.

Execution speed was not a primary concern for the initial production release, and additional code optimizations are planned for later releases. Nevertheless, DataProbeCLASSIC does provide some performance advantages over the legacy versions.

Legacy versions running on PCs and VMS machines have a data-file size limitation. For example, the PC version is unable to process a 500MB file. No such limit applies to

DataProbeCLASSIC, which has been successful tested against a 2.5GB file. The 2.5GB file contained nearly three days of data collected at 20 samples/second. In this test, two data points were collected from the entire time range without sampling. The total time required to read the raw data, convert the raw data to engineering units, and write the data to a formatted output file (423,280 pages) took less than 6 hours on a PC with an Intel Pentium 4 and 512MB RAM. This process took roughly 0.05 seconds per page of data.

FUTURE PLANS

The capability to read data files recorded under the Inter-Range Instrumentation Group (IRIG) 106 Chapter 10 solid-state recorder standard will be added in the near future. This feature will be quite valuable as solid-state recorders come into wider use.

A number of enhancements are also planned, including the addition of a separate window for tabulations, allowing more than two data dictionaries to be used simultaneously, and adding more options to the plot window. Short-term plans also call for adding some lower priority and lesser-used features.

Long-term plans call for creating user-interface using standard Windows controls for a more modern look and feel. The scripting language is one the major strengths of the product and will continue to be compatible with scripts written for legacy versions.

CONCLUSION

DataProbe provided the Navy, Air Force and others in the testing community with a purpose-built tool that provided years of service and millions of dollars in cost savings. This new version will provide similar benefits and will allow users to reuse legacy script files and dictionary files. The cost involved in recreating that capability and recovering lost corporate knowledge would be considerable.

DataProbeCLASSIC has been in production only a few months but has already stirred considerable interest among the testing community. This new version will be a welcome replacement for the poorly-performing legacy PC version and the unsupported legacy versions running on VMS and UNIX environments. The availability of a new PC version will offer new and veteran users a more efficient option for analyzing time-series data.

ACKNOWLEDGMENTS

ATAC would like acknowledge the continued support of Eric Spigel, Teresa Cunha, and Matthew Lavoie of the Naval Underwater Warfare Center in Newport, Rhode Island. Their support of this project allowed the vision of John Means to be realized, thus providing weapon testers with the tools they need to accomplish their mission.

REFERENCES

Domain Manufacturing Corporation, "Getting Started," Release 3.2, 1999.

Domain Manufacturing Corporation, "Command Reference Manual and User's Supplement," Release 3.2, 1999.

Domain Manufacturing Corporation, "System and Data Management Guide," Release 3.2, 1999.