# Common Airborne Processing System (CAPS) 2.0: Data Reduction Software on a Personal Computer (PC)

**Trent W. Hunt**
TYBRIN Corporation
Air Force Development Test Center
Eglin AFB, Florida
(850) 882-6308
huntt@eglin.af.mil

## ABSTRACT

CAPS 2.0 provides a flexible, PC-based tool for meeting evolving data reduction and analysis requirements while supporting standardization of instrumentation data processing. CAPS 2.0 will accept a variety of data types including raw instrumentation, binary, ASCII, and Internet protocol message data and will output Engineering Unit data to files, static or dynamic plots, and Internet protocol message exchange. Additionally, CAPS 2.0 will input and output data in accordance with the Digital Data Standard. CAPS 2.0 will accept multiple input sources of PCM, MIL-STD-1553, or DDS data to create an output for every Output Product Description and Dictionary grouping specified for a particular Session. All of this functionality is performed on a PC within the framework of the Microsoft Windows 95/NT graphical user interface.

## KEY WORDS

Data Reduction, Personal Computer (PC) Software, MIL−STD−1553, Common Airborne Processing System (CAPS), Engineering Unit Conversion (EUC)

## INTRODUCTION

Traditionally, instrumentation data systems for reduction and analysis have been specifically designed diagnostic tools built dependent on tightly integrated hardware and software components, causing a lack of adaptability, maintainability, and extensibility. CAPS 1.3 addressed instrumentation data reduction and analysis needs while providing a simple-to-use application using the open-architecture and low cost of the PC platform. CAPS was developed by the Air Force's 96$^{th}$ Communications Group at Eglin AFB, Florida for the Navy's Airborne Instrumentation System (AIS). The AIS Test Instrumentation Pod (TIP) is an AIM-9 type pod designed to support Operational Test and

Evaluation missions using tactical operational configured aircraft. The pod is designed for MIL−STD−1553, Global Positioning System (GPS), and inertial sensor measurement, acquisition, and recording. After satisfying the AIS requirements, CAPS was extended to accept additional data formats and added capabilities, making it a powerful generic data reduction tool.

CAPS 2.0 is an object-oriented re-engineering effort that incorporates data reduction in a 32-bit, multithreaded application with an extensible system architecture and a user-friendly interface. The goal of CAPS 2.0 is to perform the data reduction functionality of previous CAPS versions with the added flexibility of multiple input streams, Digital Data Standard (DDS) compatibility, and a modern graphical user interface (GUI). This paper shall describe the data reduction functionality of CAPS 2.0, its object-oriented design (OOD) and development, and future capabilities.

## DEVELOPMENT APPROACH

The design and development effort of the CAPS 2.0 software engineering project is based on two interdependent concepts. The first, to create an adaptive software infrastructure using object-oriented techniques. This infrastructure serves as the application framework upon which all data reduction functionality will be established. The second, to enable the user to manipulate the CAPS 2.0 functionality via a modern, user-friendly GUI.

The goal of creating CAPS 2.0 as extensible software was a proactive response to meet evolving requirements and lower future development costs. Object-oriented design enabled software component replacement or enhancements to be achieved with less system discord. CAPS 2.0 software design emphasized code reuse through abstraction and data encapsulation. Abstracting the application domain allowed a generic high-level program structure to be accomplished. This approach made it possible to maintain flexibility while supporting future capabilities. Functional modifications can be made within a component without affecting other areas of the software system. Computer Aided Software Engineering (CASE) tools were used to aid in the design process of CAPS 2.0. Visual C++ and the Microsoft Foundation Class (MFC) Libraries were used to implement the application framework, data reduction, and GUI functionality. The application was designed and developed utilizing threads to support multitasking capability and processor optimization.

## CAPS 2.0 SYSTEM DESCRIPTION

CAPS 2.0 performs the extraction and Engineering Unit (EU) conversion of instrumentation data parameters. Raw instrumentation data is input into CAPS 2.0, extracted via data description, EU converted, and output in a manner described by the

user. The primary functionality of CAPS 2.0 is its EU conversion capability. Secondary functionality is the low-level data manipulation necessary to enable conversion. This manipulation is made possible through to input and output data descriptions provided by the user. Required components for CAPS 2.0 operation are:

- Input — one or more streams of instrumentation data in raw, DDS, or real-time form. CAPS nomenclature refers to input sources as input stream(s) or physical stream(s).
- Input Data Description — a user-specified description of the physical location and data type of each instrumentation parameter the user wishes extracted from an input stream. CAPS nomenclature refers to this description as a Data Dictionary.
- Output Data Description — a user-specified description of the output format and order of EU converted instrumentation parameters. CAPS nomenclature refers to this description as an Output Product Description (OPD).

The user defines these Dictionaries and OPDs via the GUI prior to processing. At runtime the user specifies the input streams and edits the Dictionary and/or OPD configuration files as necessary. These components are used by CAPS 2.0 in the production of flexible output results. Output can be obtained in several commercial off-the-shelf (COTS) formats as well as virtually any user-defined ASCII or binary format. CAPS 2.0 supports file and local and distributed inter-process communications output.

## APPLICATION DESIGN

The CAPS 2.0 software is designed based on four main components referred to as engines. Figure 1 illustrates the interdependent relationship shared by these engines. The Executive engine controls the GUI as well as the creation and constant monitoring of the Input, EUC, and Output Engines. Input streams are ingested via the Input Engine, which imports individual time and data blocks (based on their specific input source formats) to the EUC engine for conversion. The EUC engine performs required conversions and transfers the raw data, time, and EUC data to the Output Engine. The Output Engine produces output results in user-defined order and format when the conditional output criteria is met. The Output Engine time-merges the multiple stream input using current value tables and extracts EU converted parameters from the range of inputs for user desired output results.

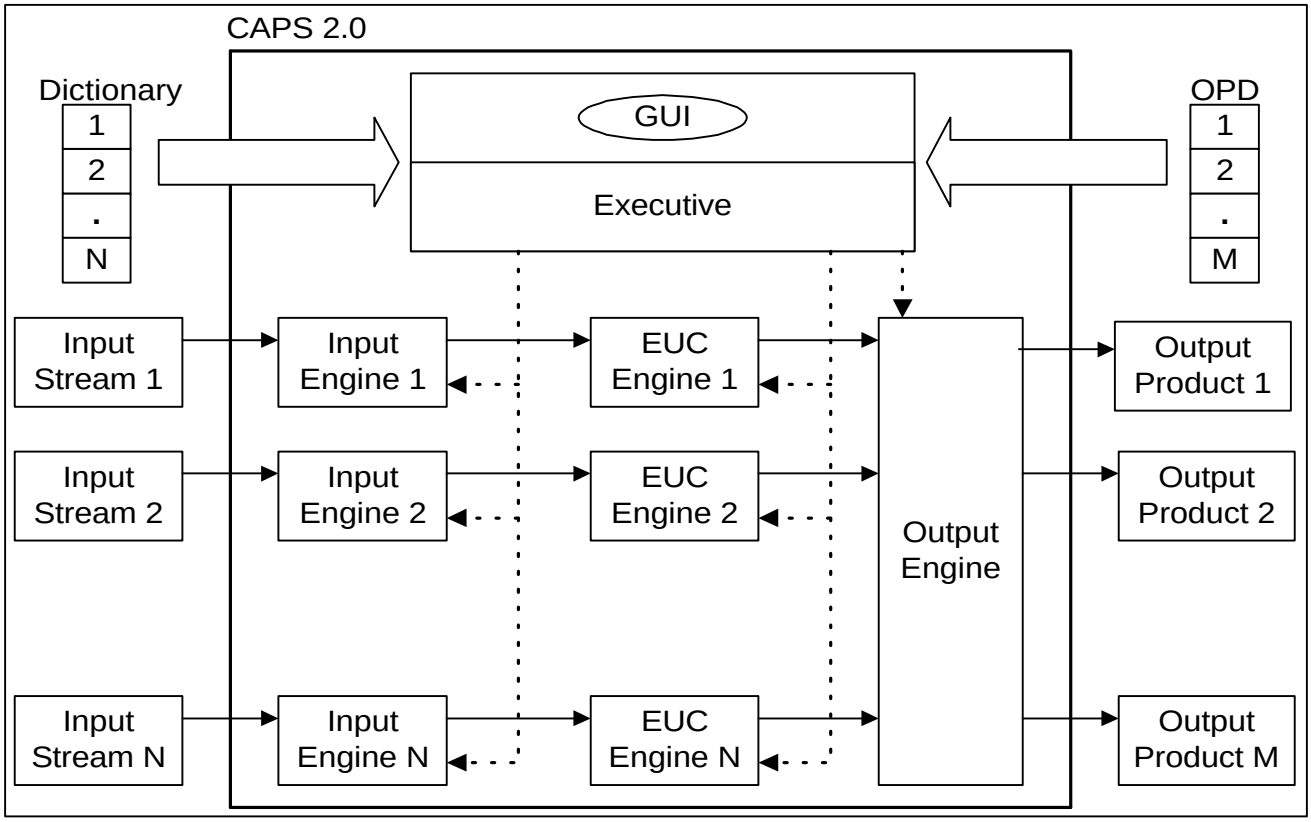


**Figure 1 - CAPS 2.0 Application Design**

## APPLICATION FEATURES

At any point in CAPS 2.0, the user has the ability to access the Help functionality and create or edit the Session, OPD, or Dictionary files, as well as execute the data processing functionality. All user interface controls perform validation on user inputs. If conflicts exist, the user is prompted for correction. All user inputs entered via the GUI controls can be saved to file for subsequent reuse.

A dictionary is a low-level description of how a single input data stream can be extracted and converted to engineering unit results. Data dictionaries are used specifically to extract data bits from virtually any binary data source. The basic unit of dictionary information is the parameter, which at a minimum includes the start word location(s), field length(s), scale factor, and the parameter name. This allows the user to be completely unconcerned about how EU results are developed. The user simply references parameters by name (i.e., Heading, Pitch, Roll).

A dictionary parameter can be created by concatenating multiple bit level data fragments together. The Location page of the Dictionary Editor allows the user to define a parameter by bit-level fragment description(s). A fragment can have individual bit, byte, word, and

double word swapping attributes specific to its binary storage orientation. This information is used in conjunction with the binary storage type of the parameter in the physical stream to correctly import the data into CAPS 2.0. A dictionary parameter will be extracted from the raw data type in the physical stream and converted to a user-specified EUC type representing the data type for CAPS internal storage. During data extraction, the gain and offset bias constants will be used as conversion factors, if applicable. This conversion is necessary to enable proper conversion of the parameter during the EUC engine processing. Other enhancements to dictionary parameters include the more visible manner of editing the dependent variable test list and extending test capability. Dependent variables are accessed using the respective editor page. A dictionary can be saved for reuse and has a default extension of ".dic". Figure 2 provides an example of the Dictionary Editor.

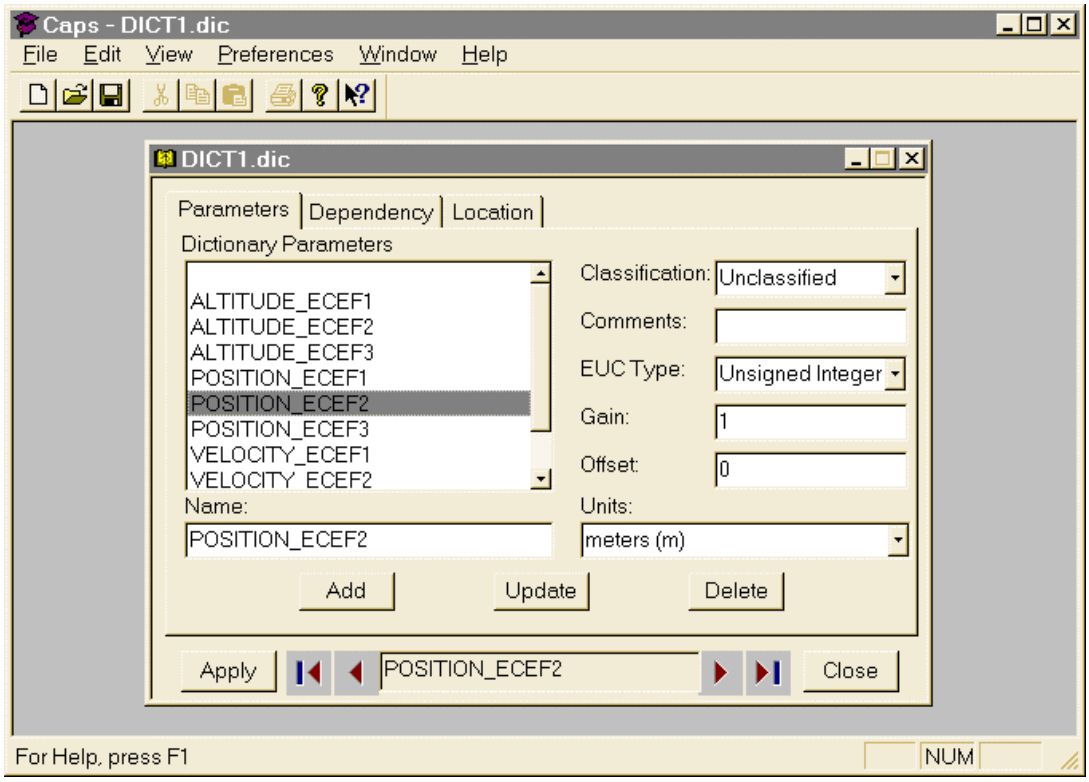


**Figure 2 - Dictionary Editor, Parameter Page**

An OPD allows the user to create and save an ordered, formatted output list independent of specific input details. This list describes one output result of the desired EU converted data and is used in the Session to control processing. The generated output can be saved to a file for follow-on applications or processed to a local or distributed inter-process communication application for reduction, analysis, or presentation. CAPS 2.0 can produce virtually any output file format with user-defined ASCII and/or binary output descriptions. OPDs contain a trigger variable that represents the criteria for output of the OPD elements. The description of an output product can be saved for reuse and has a default extension of ".opd".

The key component of the OPD file is the list of parameters the user specifies for a particular ordered output. The user selects parameters from various dictionaries and input sources as needed for extraction by switching between logical streams. In order to effectively use multiple input sources and dictionaries concurrently, we must make each variable unique via logical stream and dictionary association. Each dictionary incorporated into an OPD for data extraction is assigned a logical stream value representing generic configuration information with respect to input source. Logical stream identifiers provide a unique method enabling users to represent input source information while remaining independent of input source specification. In this manner, users can create reusable OPD files that postpone specific input configuration information until runtime. This one-to-one logical stream-dictionary relationship eliminates errors when isolating a parameter within a particular input source while simultaneously processing data from multiple input sources. The Variable List Page, as shown in Figure 3, illustrates the ordered list of dictionary parameters (associated with various logical streams) that will be output. The Variables Page describes how an individual variable will be formatted for output.

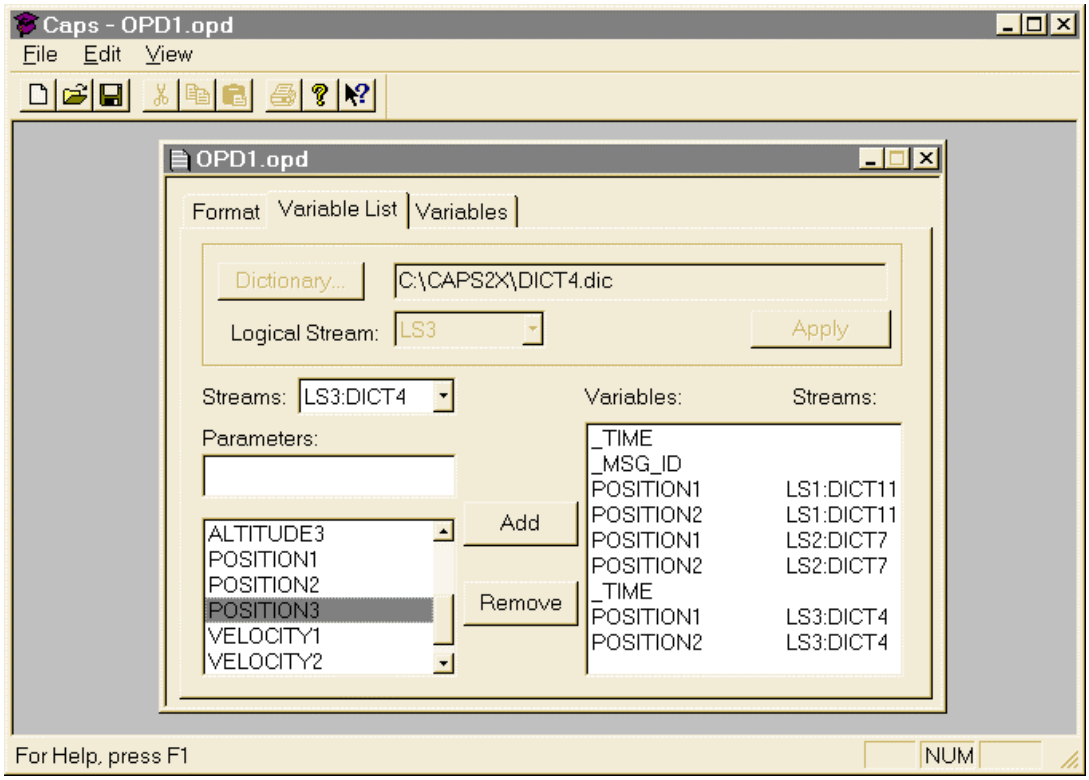


**Figure 3 - OPD Editor, Variable List Page**

A session corresponds to a specific data reduction effort performed on a specific date. The primary user interface of CAPS 2.0 is the Session Editor. This editor represents all information related to inputs, outputs, dictionaries, and OPDs. Each session contains a list of inputs to be processed and a list of outputs to be generated. Each input is defined by physical stream identifier, location, and data type. Another attribute closely related to input

is what dictionary will be used to extract and convert the input data. The user will need to carefully select a dictionary file for every input. The dictionary source selection should reflect the dictionary used with the OPD to enable the system to automatically match physical and logical streams based on dictionary use. When the user specifies the OPDs to be used, CAPS 2.0 will resolve the Logical Streams from the OPDs versus the Physical Streams previously selected during this Session. The input data is extracted from raw bit form via its physical stream and data type. This data is then retrieved via dictionary attributes and converted to satisfy the OPD requirements. This EU converted data is then output in the order specified by the OPD into the proper output form. The Start/Stop times page will display time constraints to be applied across all data streams. The Execute Page of this editor contains the Run button that activates the CAPS data processing functionality. A session file can be saved for reuse and editing using a ".ses" extension. A sample Session Editor is shown in Figure 4.

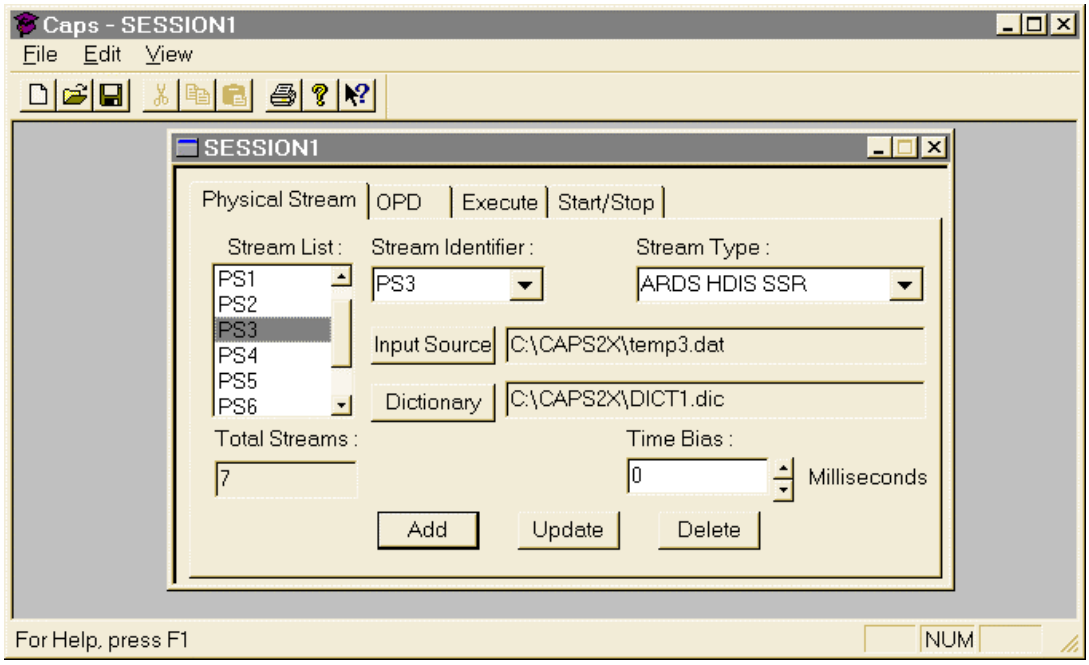


**Figure 4 - Session Editor, Physical Stream Page**

## FUTURE CAPABILITIES

Future development will continue to add new input source formats to extend the range of instrumentation and telemetry data reduction capabilities. Planned development for CAPS 2.0 includes input and output of real-time data and inter-process communication applications, both locally and distributed. Currently a real-time data standard is being developed. Needs also require CAPS to act as a real-time EU converted data server to provide data to any device across the network. Current plans call for ActiveX development to meet the CAPS data server need.

Additional CAPS 2.0 enhancements will include the ability to provide data to local and distributed applications. These applications include graphical presentation, analysis, and post-EU conversion of CAPS file and real-time data. Plans also exist to test the ability of CAPS to process in a distributed environment. Future requirements include developing CAPS functionality to better meet PCM data reduction needs. This development will support PCM reduction by allowing multiple occurrence definitions for parameters within a DDS structure.

Needs also dictate that an OPD will include function templates. Function templates are a variation of the standard OPD element. Instead of being a single parameter with defining attributes, a function template is an equation of one or more standard OPD parameters. Function Templates allow the user to perform mathematical, time series, and logical operations between Parameters from different input streams and dictionaries.

## CONCLUSION

CAPS 2.0 has succeeded in its goal to provide an extensible, easy-to-use generic data reduction tool available on a low-cost platform. It serves as a bridge from digital recording to data analysis using a modern Windows95/NT-compliant GUI. The object oriented development approach on this software effort enables CAPS 2.0 to host future enhancements and modifications with less system trauma, insuring lower development costs. As telemetry processes are re-engineered, the reusability of CAPS 2.0 software will become invaluable. Continued support and development will enhance its capabilities, making CAPS an even more powerful, generic instrumentation data tool well into the future.

## ACKNOWLEDGEMENTS

The author wishes to acknowledge the outstanding work and diligence of the CAPS 2.0 development team: Michael J. Paulick, Mike Dunn, Ben-z Lawrence and Tim Thomas. Also, gratitude is expressed to Dennis Hart, Lori Smailes, Jerry Sparks and Christopher D. Edwards for their contributions in developing a quality software product.

## REFERENCES

Hart, Dennis, Digital Data Standard (DDS) , Department of the Air Force, Eglin AFB, Florida, 1996.

Hunt, Trent W. with Paulick, Mike and Hart, Dennis, Common Airborne Processing System (CAPS) 2.0 Software Design Document, Department of the Air Force, Eglin AFB, Florida, 1997.

Rarick, Michael J. and Lawrence, Ben-z, Common Airborne Processing System (CAPS) 1.3 User's Guide, Department of the Air Force, Eglin AFB, Florida, 1994.