# An Object-Oriented Telemetry Format Management (TFM) System

# An Object-Oriented Telemetry Format Management (TFM) System

Tientien Li, Ph. D.
TACT Inc.
4911 Angeles Crest Cir.
La Canada, CA 91011

ABSTRACT

The telemetry format is a key piece of information utilized by both the flight segment and the ground segment of a mission. During the evolution of a mission, the telemetry format is usually going through many changes and refinements. Sometimes, a format may even evolve from mission to mission. The conventional Relational Data Base Management Systems (RDBMS) do not work well with telemetry formats because of the multi-dimensional nature of most telemetry formats.

To reduce the complexity of managing dynamic telemetry formats, an innovative Telemetry Format Management (TFM) system has been designed. The TFM system utilizes new object-oriented concepts in managing the creation, the evolution, and the utilization of telemetry formats. It supports common telemetry formats including: Time-Division Multiplexed (TDM) telemetry formats and packet telemetry formats. By using the TFM system, one can greatly simplify most tasks associated with the development of telemetry formats. This paper describes the architecture, design concepts, and operational philosophy of the TFM system.

## 1. Introduction

As the sophistication and data rates of airborne and space-based instruments increased, telemetry formats used in these missions became more and more complex and difficult to maintain. Today, the cost of developing telemetry formats and maintaining the format consistency has increased substantially. Software tools such as Relational Data Base Management System (RDBMS) may provide some help in maintaining some well defined telemetry formats. Unfortunately, RDBMS does not work well with complex telemetry formats. This is because the multi-dimensional structure of most complex telemetry formats does not not map well onto the 2 dimensional row-and-column model used in conventional RDBMS. Some higher level structural information are usually lost when a complex telemetry format is represented in the 2 dimensional RDBMS model.

The Telemetry Format Management (TFM) system is designed based on a multi-dimensional structural model of telemetry formats. The model is very adaptable and can be used to represent some very complex telemetry formats. After an overview of design concepts used in the system, the telemetry format model and system tools for supporting format development are described in some detail.

## 2. Design Concepts

A very brief introduction of Object-Oriented Design (OOD) concepts is presented here. More detailed discussion on OOD concepts can be found in [Cox 86]. There are three key OOD concepts: abstraction, encapsulation, and inheritance.

- Abstraction is the ability to specify generic attributes and necessary operations required for modelling a class of objects with respect to a problem domain. A model defined by a set of representing attributes and operations is often called an abstract data type. For example, let RADAR be an abstract data type for a class of radars. If we are working on a ranging problem, we may define RADAR to have attributes: ANTENNA, TRANSMITTER, RECEIVER, and DISPLAY, and operations: TRANSMIT, RECEIVE, CALCULATE-RANGE, and DISPLAY-ECHOES.

- Encapsulation is the ability to hide non-essential and implementation dependent information from the user of abstract data types. This feature allows one to use an abstract data type without worrying about its implementation details such as the data structure representing attributes, temporary accounting variables, and algorithms used in implementing various operations.

- Inheritance is the ability to defining subtypes by inheriting type specifications, i.e., attributes and operations, from a parent type. This feature allows one to build new data types upon existing ones. For example, we may define a subtype RADAR-ALTIMETER for a subclass of radar altimeters. This type may be defined by inheriting attributes and operations of its parent type RADAR and by overloading the range calculation algorithm used in the CALCULATE-RANGE operation.

The advantages of OOD concepts can be summarized as follows: Abstraction allows one to think at a higher level; Encapsulation allows one to work at a higher level; and Inheritance allows one to evolve new and more advanced data types.

## 3. System Architecture

The TFM system was designed utilizing OOD concepts. The system architecture consists of two parts: an abstract data type definition for telemetry formats, and a set of integrated development tools for creating formats, modifying formats, and extracting information from formats.

### 3.1 Data Types

The TFM system defines an abstract data type, FORMAT, representing a class of telemetry formats. In this section, we will outline the specification of the FORMAT data type and its subtypes.

#### 3.1.1 Basic Attributes

In the TFM system, there are over 30 attributes defined for the FORMAT data type. Only a few key attributes will be described here. Conceptually, a format defines the layout of elements within a spatial object. The following are some key attributes that define various exterior properties of a format.

Name — A name up to 64 characters long may be defined for a format. The system uses the name to identify the format and to refer elements or locations within the format.

Dimension — Each format occupies an N-dimensional space. An example of a 3 dimensional major frame format is shown in Figure 1. Table 1 contains additional examples of various dimensions. The dimension of a format defines a coordinate system for identifying elements within the format. The TFM system defines a coordinate to be a tuple with the $i^{th}$ element of the tuple identifying a (N-I) dimensional coordinate. For example, (5) identifies the $5^{th}$ minor frame; and (1, 2, 3) identifies the location corresponding to the $3^{rd}$ bit of the $2^{nd}$ byte of the $1^{th}$ minor frame of a major frame format.

Size — If an N-dimensional format consists of M copies of (N-1) dimensional elements, the size of the format is determined by the number M and the unit size of the (N-1) dimensional element. For example, assuming a 128 bits minor frame size, the format shown in Figure 1 has size 128 x 8, i.e., 1024 bits. Variable size formats are allowed in TFM. However, it is assumed that a length indicator field must be located within the format.

| Dimension | Format | Elements | Possible Element Values |
|---|---|---|---|
| 1 | bytes, words | bits | 0/1, ON/OFF, TRUE/FALSE |
| 2 | minor frames | bytes, characters, words | ASCII, bit string, short int |
| 3 | major frames | minor frames | numbers, strings, minor frames |
| 4 | format families | major frames | paragraphs, major frames |

Table 1. Examples of 1 to 4 dimensional formats

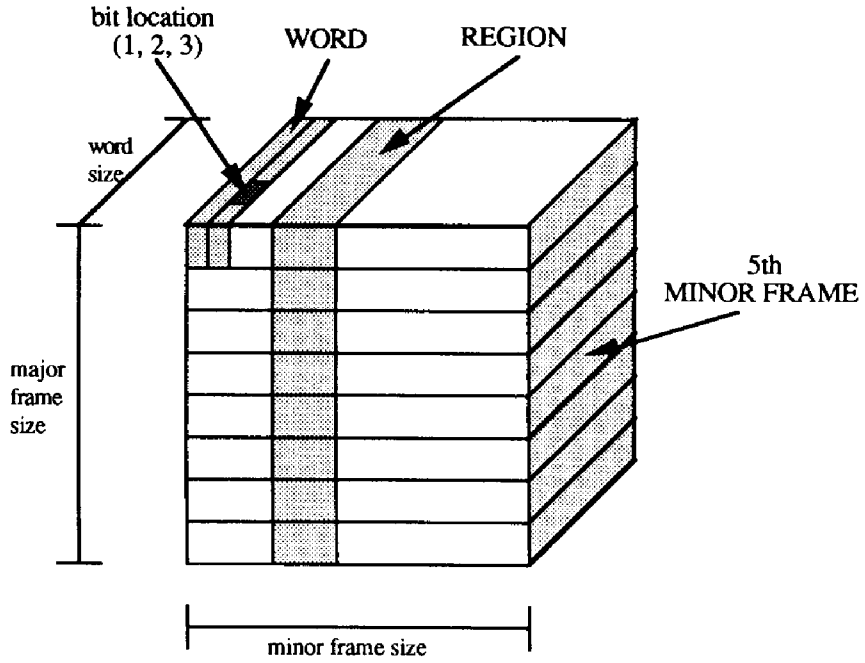


Figure 1 An example of a 3 dimensional format.

Addressing Elements of a format may be specified in sequential mode or random mode. In the sequential mode, the address of an element is assumed. In the random mode, element addresses have to be explicitly specified.

Another set of key attributes are those defining the layout structure of a format and characteristics of elements.

Regions Contiguous or non-contiguous sub-space, called Regions, may be defined for a format. Regions are used to specify fixed columns, subcommutated words, and virtual channels for embedded telemetry or packet telemetry. The system allows the user to specify locations either in the coordinate system of the format or the coordinate system of a region within the format.

A coordinate transformation function is provided to translate format coordinates to/from coordinates within a region.

Layout — A format layout is defined by a list of element definitions. Elements or regions defined in the layout are called Fields. Since (N-1) dimensional elements and N dimensional subspace, i.e., regions, are themselves format objects, a format may be defined recursively by defining its fields.

### 3.1.2 Standard Operators

There are over 50 standard operators defined for the FORMAT data type. Most users will probably never use these operators directly. Instead, they will use tools provided by the system to manipulate format objects. Only system tools utilize these operators.

The 50+ operators can be classified into the following four categories:

- constructor and destructor used to create and delete formats
- operators for setting format attributes
- operators for extracting attribute information
- utility operators, e.g., coordinate system translator

### 3.1.3 Subtypes

Commonly used formats can be defined as subtypes of FORMAT. The system provides a set of standard 1 and 2 dimensional format subtypes including:

- logical of various sizes (1-8, 12, 16, 18, 24, 32, 36bits)
- characters (7 and 8-bit ASCII)
- unsigned numbers (8, 12, 16, 24, 32, and 64-bit big-endian or little-endian)
- signed numbers (1's and 2's complement)
- floating point numbers (32 and 64-bit IEEE)

In addition, the TFM system supports inheritance. The user can define his/her own subtypes and derived types. Formats of any system/user subtypes can be instantiated and be used in the definition of other formats.

## 3.2 Tools

The system provides a set of integrated tools that work with the format data base or with individual formats. Figure 2 shows the data flow of the system.

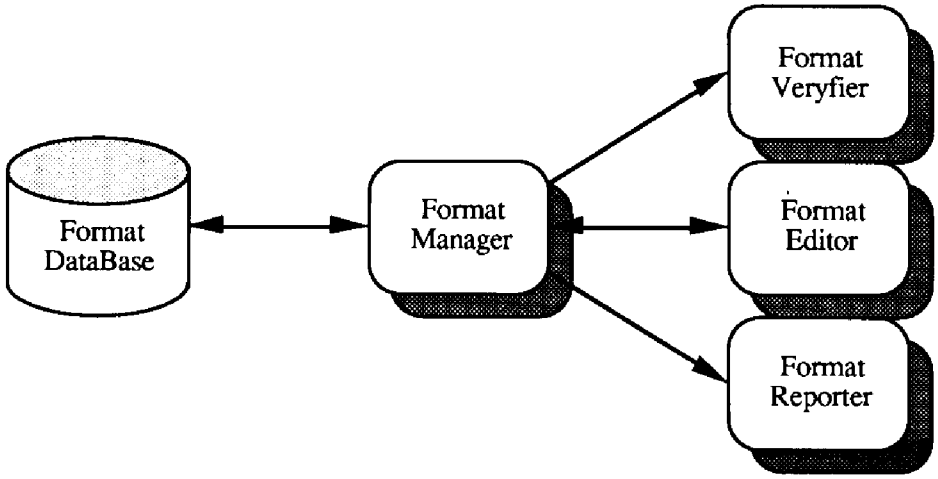


Figure 2. TFM System Data Flow

### 3.2.1 Format Manager

The TFM system has a simple object-oriented data base manager. It manages multiple formats and multiple versions of a format. Each format is identified by the format name. Each version of a format is identified by its creation date and the last modification date. The system does not automatically create new versions of a format unless it is requested by the user.

The format manager allows the user to create new format data base and move formats between data base. The system also manages subtypes and allows the addition and deletion of subtypes within a data base. A browser is provided for viewing attributes and information about selected formats or for viewing summaries of all formats stored in the data base. This tool provides quick access to information stored in the format data base.

### 3.2.2 Format Editor

The editor is the most important tool of the system. A sample display of the format editor is shown in Figure 3 which contains data extracted from [GE 83]. The editor supports four methods for defining format layout and field attributes:

- direct key in
- copying from another format
- instantiation of a predefined format subtype
- linking to another format

The first two editing methods are self-explanatory. Instantiation of a predefined subtype creates a format with its layout and field attribute values as defined by the subtype. Unlike

copying from another format, the value instantiated from a subtype changes as the definition of the subtype changes. When the change of subtype definition causing errors in a format definition, the user has to resolve these errors and maintain the format consistency. However, subtypes are supposed to be well defined and rarely changed.

Linking is a special mechanism provided to support concurrent development of different parts of a format. Linked format definitions may be incomplete and inconsistent. The automatic format consistency check may be turned on or off by the user. The user can continue to work on linked formats with known errors.

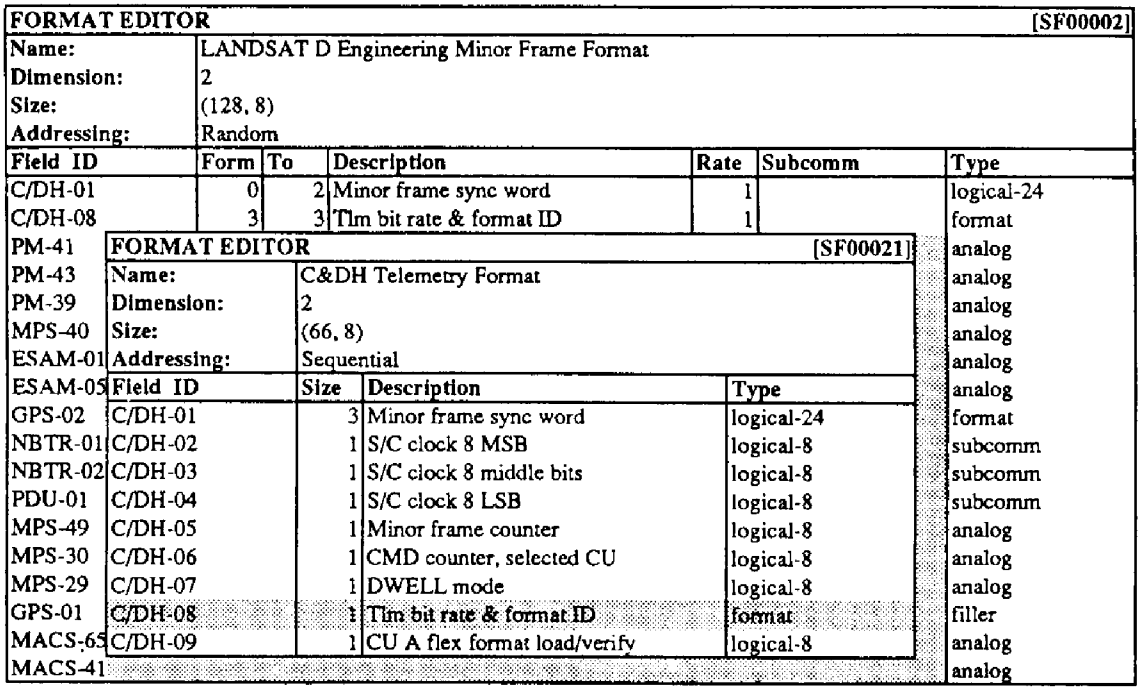


Figure 3. Format Editor Display

### 3.2.3 Format Verifier

The format verifier allows the user to check a format against a set of design rules for possible design violations. This is a very useful tool for checking format consistency when developing large and complex telemetry formats. Currently, the tool will check for following errors:

- missing or incomplete attribute definitions
- dimension and size inconsistency
- duplicate names
- missing or undefined subtype references
- locations with multiple definitions
- incomplete field definitions

### 3.2.4 Format Reporter

The system provides a report tool which can generate several standard reports for selected formats in the data base. Each report supports several user configurable parameters for report customization. For example, the layout report can display the layout of a format in units of bits, words, or minor frames for any selected range of the format.

Reports may be routed to a printer or a disk file. After a report has been routed to a disk file, format information may be extracted for exporting to other development and analysis tools.

## 4. Theory of Operation

This section describes how to use the TFM system to support various development methodologies.

### 4.1 Top-down Development

The TFM system supports the top-down methodology by allowing one to define a format starting at the top level specification and recursively down to lower level details. A format will be completely defined when all its lowest level fields are defined. This approach provides one a much stronger control over the structural and organizational consistency of a format definition.

### 4.2 Bottom-up Development

The TFM system also supports the bottom-up methodology. The system allows one to work on details of lower level formats first. A higher level format definition can be specified by using lower level formats as fields of the top-level format. This approach allows one to design or experiment with building blocks used in defining the high level format.

### 4.3 Multi-methodology Development

During the life cycle of a development, one may have to utilize multiple methodologies and to be able to move from one methodology to another depending on the focus of the design at that time. The TFM system supports multi-methodology development.

For example, let's consider the format shown in Figure 4. A hybrid telemetry format is used in this example. Within the Time Division Multiplexed top level format, region 2 defines an embeded virtual channel used for transfer packetized telemetry data from two instruments.

The format consists of three concurrently developed components: the top level, region 1, and packet formats used in region 2. The top level format and region 2 format are developed using the top down methodology in one development activity. The region 1 format is linked to the top level format and is developed concurrently using the bottom up methodology. Region 2 instrument packet formats are developed in the third development activity. The TFM system can support all three development activities concurrently and also support the integration of formats generated from the three activities.

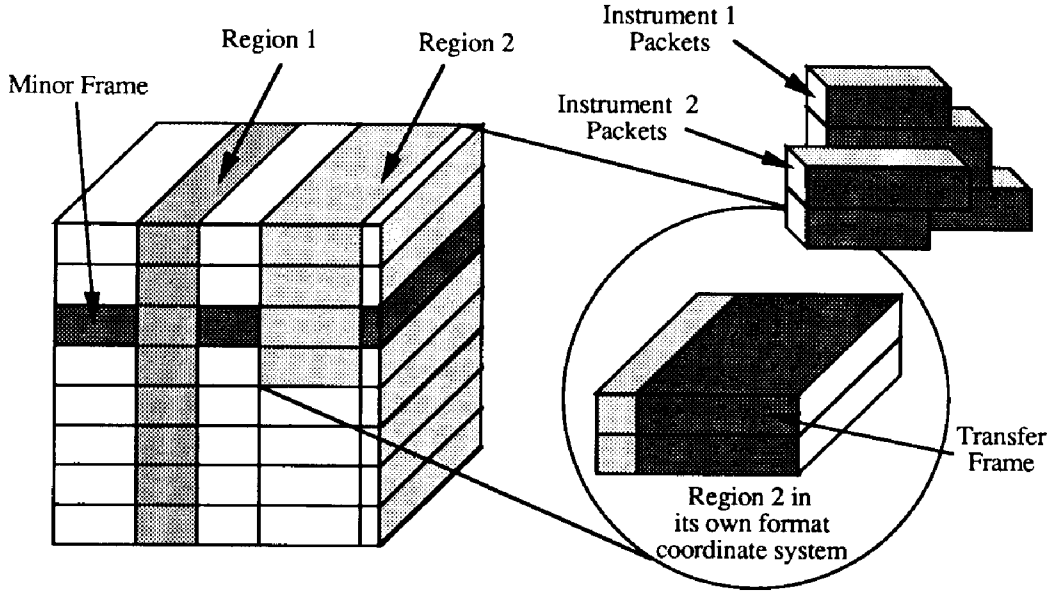


Figure 4 Multi-methodology Development of a Hybrid Format.

## 5. Summary

The TFM system is designed utilizing advanced OOD concepts. The system defines an abstract data type for telemetry formats and provides a set of integrated tools for automating key tasks associated with the development of telemetry formats.

A prototype of the TFM system is implemented in C++. The system runs on IBM/PCS under MS-DOS and will be ported to run on Sun workstations under UNIX. The system can be interfaced to other telemetry systems. For example, an interface to Eidel's EE315 PC/AT based PCM decoder system for generating signal definition files, i.e., SIG files [EE 90], has been developed.

## 6. References


[Cox 86] Object-Oriented Programming: An Evolutionary Approach, 1986, Addison-Wesley.

[EE 90] EE 315 PCM Decoder User Manual, Version 2.3, March 90, Eidel.

[GE 83] LANDSAT-D Data Format Control Book, Volume II (Telemetry), November 1983, General Electric.