ProVAL Data Format Specification for Pavement Profile

1. Scope

1.1. This document describes a data file format specification for pavement profile.

1.2. This file specification describes the variables and sizes of all data that will be stored in the file. The file is in binary format, and is fully documented in this specification.

1.3. This specification is designed to be independent of hardware platforms, computer languages, and Operating System (OS).

2. Referenced Documents

2.1. ASTM Standards

E867 – 97 Terminology Relating to Vehicle-Pavement Systems.

2.2. Institute of Electrical and Electronics Engineers (IEEE) Standards

IEEE Standards 754-1985 (Re-affirmed 1990): IEEE Standard for Binary Floating-Point Arithmetic.

3. Terminology

3.1. Definitions:

3.1.1. Terminology used in this practice conforms to the definitions included in Terminology E 867.

3.2. Definitions of terms specific to this standard:

3.2.1. *Int8* – Data type for an 8-bit, unsigned integer.

3.2.2. *Int32* – Data type for a 32-bit, signed integer.

3.2.3. *Single* – Data type for a 32-bit, signed real number, i.e. single precision IEEE floating point.

3.2.4. *4-byte String* – An ASCII string of 4 characters in length. No null character is included at the end of the string.

3.2.5. *8-byte String* – An ASCII string of 8 characters in length. No null character is included at the end of the string.

3.2.6. Array (numeric data type) – Sequence of data of the specified numeric data type. Only the values are stored, but no information about the array is stored.

3.2.7. Array (String) – ASCII strings separated by a tab. There is no tab after the last string.

3.3. Symbols

3.3.1. *n* - total channels of elevation data.

3.3.2. *m* - total number of test locations (i.e. data points).

4. Profile Data Specifications

4.1. File Structure

4.1.1. The general file structure is divided into five sections: (1) File Header, (2) Metadata, (3) Longitudinal Profile Data, (4) Transverse Profile Data, and (5) File Trailer. The five sections are stored sequentially (see Figure 1).

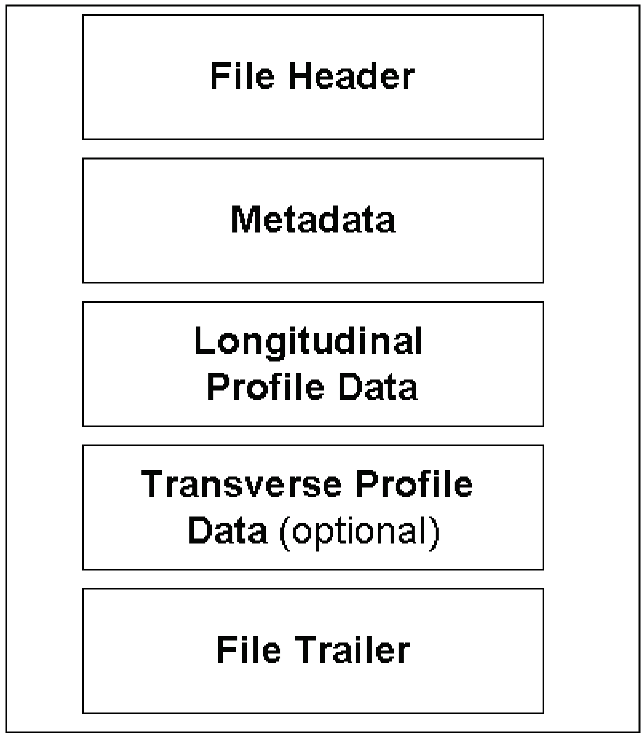

Figure 1: Layout of the File Structure

4.1.2. Each of these portions of the file is described in the following sections, as well as the data types and other descriptors that will be required by the file. The data will be written to the file sequentially, with the offsets listed in the file header as guides to find various portions of the file. It is important to note that all offsets are relative to the beginning of the file. Because offset values may not be known at the time of writing the file header, these values need not be written. However, spare space must still be reserved for the offsets so that the values can be updated when known.

4.2. File Header

4.2.1. File header contains the information pertaining to the data file type, software version information, and information about the data contained (Table 1).

Table 1 - File Header

Variable name	Data Type	Data	Default Value
Signature	4-byte String	Identifies file as being written in the Standard Pavement Profile Format	"SPPF"
Version	4-byte String	Identifies the version number of the file format	"1.04"
SW Version	8-byte String	Identifier of the software that produced the file	e.g. "TGPA1.00"
Metadata Offset	*Int32*	Offset in byes to the beginning of the Metadata	NA
Longitudinal Data Offset	*Int32*	Offset in bytes to the beginning of the longitudinal profile data	NA
Transverse Data Offset	*Int32*	Offset in bytes to the beginning of the transverse profile data	NA

4.3. Metadata

4.3.1. Metadata is structured, descriptive information about a resource, or data about data. Using metadata in the binary file format will allow generic operating on the data information about which the reader software has no prior knowledge. Also, metadata will allow scalable evolution of the data description without requiring simultaneous upgrades to all reader software.

4.3.2. The first value in the metadata portion will provide the number of Metadata Entries (*MDE*s) and the information regarding each of them. Table 2 shows the information required to construct and appropriate *MDE*.

4.3.3. The metadata tags are listed in Table 5, and can be used in any number or order. If no metadata tags exist, *Number of MDEs* = 0.

4.3.4. The names of the *standard* metadata entries (see Table 3) are not stored in the metadata entry to conserve space and more importantly, to allow for localization, i.e., the file is not tied to one written language. User-defined metadata entries cannot be arrays and the data type is always String.

4.3.5. Previous versions of this specification did not specify what data should be stored for empty arrays. The current convention has been to store a one byte value of the same data type as the array. For example, an array of singles with no elements would store a value of "0".

Table 2 - Metadata

Variable name	Data Type	Data
Number of MDEs	*Int32*	Number of MDEs

Table 3 - Metadata Entries

Variable name	Data Type	Data
Tag of MDE	*Int32*	Metadata Tag (see Table 5)
Data Type of MDE	*Int32*	Data type index of MDE (see Table 9)
Array Size	*Int32*	"-1" if not an array. "0" if the array is empty. Numbers greater than 0 specify the number of elements in the array. Even though arrays of strings are stored differently than other types of array, an Array Size should still be specified here.
Count	*Int32*	For Data Type "String" "and Array (String)", Count = the number of bytes in the string. For other data types, Count = 1.
Name Length	*Int32*	For Metadata entries listed in Table 5, this is 0. For user-defined entries, this value is the length of the Name.
Name	String	Name of the Metadata.
MDE	varies	Information associated with Tag of MDE.

4.4. Longitudinal Profile Data

4.4.1. There are two ways to store the profile data: "location-wise" and "array-wise". The first method is appropriate for data recording during profile data collection to prevent data loss, while the latter is appropriate for post-processing to speed up software reading and writing.

4.4.2. If the Data Storage Format, from Metadata tag #522 in Table 10, equals 1 (i.e. "location-wise"), the longitudinal data will be stored as a sequence of current longitudinal distance followed by corresponding elevations of longitudinal sensors at this location, beginning at left side of vehicle. The next block of storage will store longitudinal distance and all elevation data for the next location, and so on. However, the location may not need to be stored if a specific data interval is given. (see Figure 2)

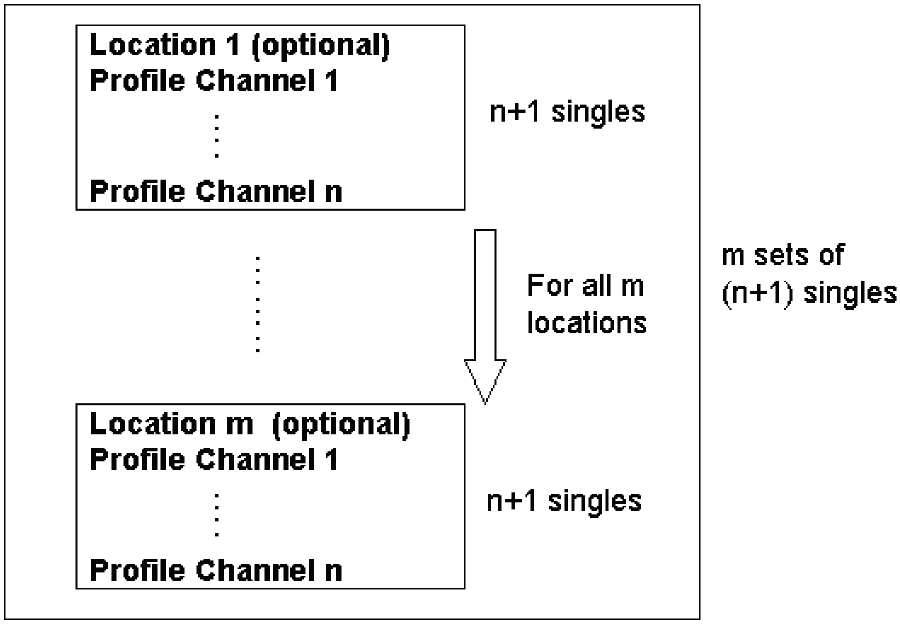

Figure 2: Location-wised Storage

4.4.2.1. In general, if a location and elevation channels are recorded for each test location, every set of *n*+1 Singles (one distance data and *n* channels of elevation data) will be read as one profile location. If a specific data interval is included in the metadata, only *n* Singles will be read for each location, with distance being calculated from the beginning of the test location. For example, if a standard interval exists and a single channel of profile data is present, only one Single will be read for each location. If two are present, then two Singles will be read per point.

4.4.2.2. The location-wise format is recommended for profiler data acquisition software. Storing the data after every sampling location allows for immediate writing to protect against data loss and reduce memory requirements.

4.4.3. If the Data Storage Format, from Metadata tag #522 in Table 10, equals 2 (i.e. "array-wise"), then the longitudinal data will be stored as a sequence of the longitudinal distance array followed by the elevation array of each longitudinal sensor, beginning at left side of vehicle for all locations. (see Figure 3)

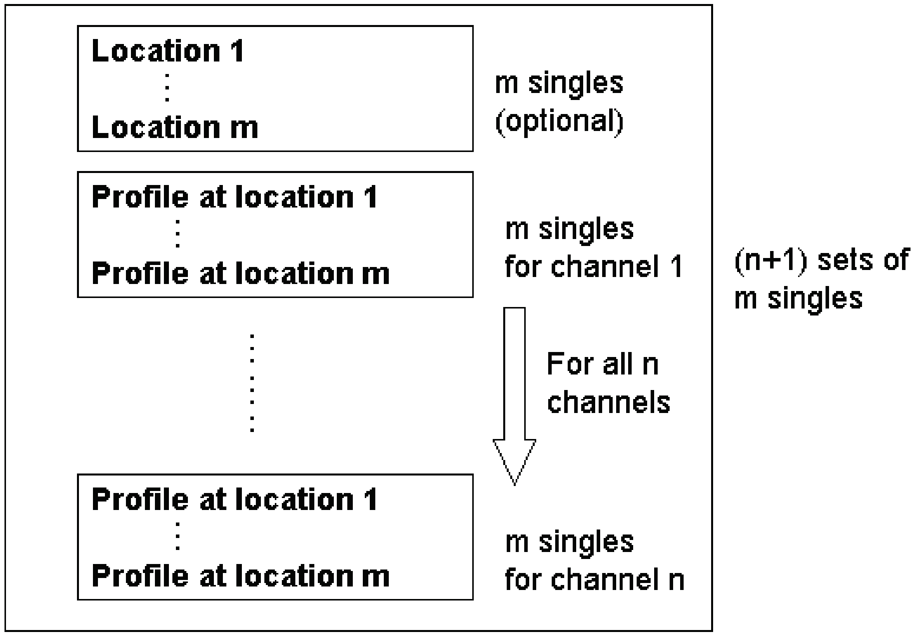

Figure 3: Array-wise Storage

4.4.3.1. In general, if a distance and elevation channels are recorded for each test location, *n*+1 sets (one distance data and *n* channels of elevation data) of *m* Singles (*m*: total number of test locations) will be stored in sequence. If a specific data interval is included in the metadata, only *n* sets of *m* Singles will be stored sequentially, with distance being calculated from the beginning of the test location by the software. For example, if a standard interval exists and a single channel of profile data is present, only one set of m Singles will be stored. If two are present, then two sets of *m* Singles will be stored sequentially.

4.4.3.2. This data format is recommended for software that reads and writes the data during post-processing. Data stored as one continuous array ("array-wise") can be read and processed much faster than the "location-wise" storage format.

4.5. Transverse Profile Data

4.5.1. The transverse elevation readings are treated the same as the longitudinal data.

4.6. File Trailer

4.6.1. The File Trailer is used to signal the end of the file (see Table 4).

Table 4 - File Trailer

Variable name	Data Type	Data	Default Value
End of File	3-byte String	Indicates the end of the file	"@@@"

4.7. Sections

4.7.1. A section is defined by the use of two event markers. The first event marker is the start location of the section and the second event marker is the stop location. Special attention should be paid to *lead-in* and *lead-out* event markers. These two markers define the section that is bounded by the lead-in and lead-out. Please note, they do *not* define the lead-in and lead-out, but the section between them. Here's an example:

A 1000-point profile has a lead-in of 50 points and a lead-out of 40 points. Points 0-49 will constitute the lead-in, so the event marker index for lead-in will be 50. Points 960-999 constitute the lead-out, so the event marker index for lead-out will be 959.

Tags 526 and 527 were defined before the use of event markers to define sections. These tags are now considered obsolete.

4.8. Predefined Tags

Table 5 - Metadata Tags and Descriptions

Tag	Name	Data Type	Notes
256 – 511	*General Profiler and Location Information*		
258	Title	String	required
259	Profiler Trade Name and Model Number	String	
260	Vehicle Identification	String	
261	Date Data Was Collected – (yyyymmdd)	String	
262	Time Data Was Collected – (HHMMSS)	String	
263	Profiler Operator Name	String	
264	Average Vehicle Speed Associated with Data	Single	
265	Original Filename before Import	String	Read-only
271	Agency District Name	String	
272	Agency District Number	Int32	
273	County Name	String	
274	County Number	Int32	
275	Nearby City Name	String	
281	Roadway Designation	String	
282	Lane Identification	String	
283	Station Number of Beginning Point	String	
284	Reference Marker or Milepost of Beginning Point	String	deprecated
285	Pavement Surface Type (See Table 7)	Int32	
286	Direction of Travel	String	
287	Station Number of Ending Point	String	

288	Reference Marker or Milepost of Ending Point	String	deprecated
291	Ambient Temperature	Single	
292	Surface Temperature	Single	
293	Climatic Conditions (see Table 8)	Int32	
294	Data History	String	Read-only
295	Date File Last Modified – (yyyymmdd)	String	Read-only
296	Time File Last Modified – (HHMMSS)	String	Read-only
297	Date File Imported From Original File Format – (yyyymmdd)	String	Read-only
298	Time File Imported From Original File Format – (HHMMSS)	String	Read-only
299	Run Number (multiple runs - same location on the same day)	Int32	
300	Profiler Type (see Table 11)	Int32	
301	Country Name	String	
302	State/Province Name	String	
303	Wind Speed (mph)	Single	
304	Wind Direction	String	
305	Thumbnail Image	Array (Byte)	Read-only
306	Start Milepost	Single	
307	Stop Milepost	Single	
308	Profiler Direction (see Table 14)	Int32	
309	File Key	String	Read-only
310	Profile Keys	Array (String)	Read-only
311	Section Keys	Array (String)	Read-only
312	Section Names	Array (String)	Read-only
512 – 767	*Longitudinal and Transverse Profile Information*		
512	Number of Longitudinal Elevation Channels	Int32	required
513	Number of Transverse Elevation Channels	Int32	required
514	Number of Longitudinal Data Points	Int32	required
515	Number of Transverse Profiles Data Points	Int32	required
516	Longitudinal Distance Between Longitudinal Data Points	Single	
517	Longitudinal Distance Between Transverse Profiles	Single	

518	Longitudinal Sensor Spacing From Vehicle Center (negative values to the left of vehicle center, positive to the right).	Array (Single)	required
519	Transverse Sensor Spacing, From Vehicle Center, (negative values to the left of vehicle center, positive to the right).	Array (Single)	
520	Names for Longitudinal Sensors	Array (String)	required
521	Names for Transverse Sensors	Array (String)	
522	Longitudinal Data Storage Format (see Table 10)	Int32	required
523	Channel Type for each Longitudinal Profile (see Table 12)	Array (Int32)	
525	Profile Offset (if linear distance adjustment or correlation is performed)	Single	
526	Profile Start Index (for use in lead-in)	Int32	deprecated
527	Profile Stop Index (for use in lead-out)	Int32	deprecated
528	Event Marker Index	Array (Int32)	
529	Event Marker Text	Array (String)	
530	Event Marker Type (see Table 13)	Array (Int32)	
768 – 1023	*Measurement Units Information*		
768	Units for Longitudinal Distances (see Table 6)	Int32	required
769	Units for Elevation Data (see Table 6)	Int32	required
770	Units of Speed (see Table 6)	Int32	
771	Units of Temperature (see Table 6)	Int32	
772	Units of Sensor Spacing (see Table 6)	Int32	
1024 - 2047	*Reserved for User Defined Metadata Entries*		

Note: Any tags not listed below 1024 are reserved for future use.

Note: “Read-only” indicates the user should not be allowed to edit the value. Only the software or profiler should be allowed to change the value.

Note: deprecated items are no longer used.

4.9. Appendix

Table 6 : Units

Value of Tags 768 - 1023	Unit
Distance and Elevation	
73	Mils
1	Inches
2	Feet
4	Miles
5	Millimeters
6	Centimeters
7	Meters
8	Kilometers
Speed	
24	Feet / Second
28	Miles / Hour
27	Meters / Second
26	Kilometers / Hour
Temperature	
35	Degrees Fahrenheit
33	Degrees Centigrade
Time	
36	Sec

Table 7 : Pavement Surface Types

Value of Tag 285	Pavement Surface Type
0	Undefined
1	Portland Cement Concrete
2	Hot-Mix Asphalt
3	Unpaved

Table 8 : Current Climatic Conditions

Value of Tag 293	Climatic Conditions
0	Undefined
1	Sunny
2	Hazy / Fog
3	Partly Cloudy
4	Mostly Cloudy
5	Overcast
6	Light Rain / Snow
7	Moderate Rain
8	Heavy Rain

Table 9 : Data Types

Index*	Data Type	Size (bytes)	Description
8	String	Varies	One byte ASCII (no Unicode support)
17	Byte	1	8-bit unsigned integer
3	Int32	4	32-bit signed integer
4	Single	4	Single precision IEEE floating point

*Data type index values follow Microsoft programming conventions.

Table 10 : Data Storage Format

Value of Tag 522	Data Storage Format
1	Location-wise
2	Array-wise

Table 11: Profiler Type

Value of Tag 300	Description
1	High speed
2	Light weight
3	Manual

Table 12: Channel Location

Value of Tag 523	Description
1	Left Wheel Path
2	Right Wheel Path
3	Centerline

Table 13: Event Marker Type

Value of Tag 530	Description
1	Generic Marker
2	Section - Start
3	Section - Stop
4	Ignore - Start
5	Ignore - Stop
6	Lead-In (First point after the lead-in stops)
7	Lead-Out (Last point before the lead-out starts)

Table 14: Direction of Travel

Value of Tag 286	Description
1	Forward (in traffic flow)
2	Reverse (against traffic flow)