
The Triple-S

Survey Interchange Format Standard

A standard for porting surveys between survey packages

on various hardware and software platforms

Version 1.0 ***September 1994***

Introduction

This document describes the Triple-S format for survey data and variables.

Background

The aim of the Triple-S standard is to define a means of transferring the key elements of entire surveys between different survey software packages across various hardware and software platforms.

This initial version of the Triple-S standard has been devised by Keith Hughes, Stephen Jenkins and Geoff Wright. The impetus was a paper by Peter Wills(1).

Summary

A Triple-S survey is described in two ASCII text files. One, the Definition File, contains version and general information of the survey together with definitions of the survey variables. This is used to interpret the contents of the Data file. By default the Definition File has a file extension of 'SSS' and the corresponding Data File has the same name but with the extension 'ASC'.

The format of each of the files has been designed to enable software read/write routines to be easy to implement. To further aid the development process the files are relatively simple to read by eye.

1. **Data Use and Reuse**, P Wills, SGCSA, Bristol England September 1992.

The Definition File

Outline

The definition file is comprised of a number of keywords some of which have associated parameters. These describe two aspects:

a. *the file itself* in terms of version number, date and time of creation etc.

b. *the survey* in terms of the survey variables.

The following shows an outline of the contents of the definition file.

```
SSS
    VERSION 1.0
    DATE "date_text"
    TIME "time_text"
    ORIGIN "origin_text"
    USER "user_text"

    SURVEY
        TITLE "survey_title_text"

        RECORD record_id
            VARIABLE variable_id

                variable_details

            END VARIABLE


            VARIABLE variable_id

                variable_details

                END VARIABLE
        END RECORD
    END SURVEY
END SSS
```

The items shown in uppercase are keywords. Those shown in lower case or as numeric digits are keyword parameters.

Basic Formatting Rules

1. General
    a. Only characters in the ASCII range decimal 32 to decimal 255 are considered valid - all others are ignored when read.

2. Keywords
    a. The case of letters in keywords is insignificant, so a is equivalent to A, b to B and so on.

    b. Any number of space characters (decimal 32), including none, can appear between keywords and parameters. Thus VERSION 1.0 could be written VERSION1.0 and END SSS could appear as ENDSSS

    c. Individual keywords cannot contain embedded space characters, thus VER SION is considered invalid.

3. Text String Parameters
    a. The double quote character, decimal 34, is used to introduce and end text strings.

    b. To represent the double quote character *in* a string, surround the character with braces: {"}
    Alternatively use the two-digit hexadecimal ASCII equivalent (surrounded by braces): {22}

    c. To represent characters in the ASCII range decimal 0 to decimal 31 *within* a string, use the two-digit hexadecimal ASCII equivalent surrounded by braces. Any number of consecutive characters can be surrounded by just one pair of braces, for example to represent a CR/LF sequence within a text string specify: {0D0A}

    d. To represent a newline *instruction* within a string use the special mnemonic: {NL}

    e. To represent the open brace character *within* a string, outside its special uses described in rules b, c and d above, surround it with braces, thus: {{}
    Alternatively, use the two-digit hexadecimal ASCII equivalent surrounded by braces: {7B}

4. Other Parameters
    a. The case of letters in other parameters is insignificant, so that a is equivalent to A, b to B and so on.

    b. Any number of space characters (decimal 32), including none, can appear between parameters and keywords. Thus VERSION 1.0 could be written VERSION1.0

    c. Individual (non string) parameters cannot contain embedded space characters, thus 1 .0 is considered invalid.

Formatting Recommendations

In order to improve the readability of the definition file, it is recommended that:

1. The file is organised into lines using CR,LF (decimal 13,decimal 10) combinations. These are ignored when read (according to Basic Formatting Rule 1a).

2. At most one keyword, or keyword phrase, appears on one line.

3. Lines are indented with space or tab characters to reflect the structure inherent in the file. An indent is applied after every keyword which has a corresponding END keyword.

Definition File Keywords

This section describes the syntax and function of each of the keywords and keyword phrases used to form a Triple-S definition file. The keywords are shown in the order they are expected in the file.

SSS
Mandatory. Should be the first (not ignored) characters in the file.

VERSION 1.0
Mandatory. The current version is 1.0.

The following four keywords (DATE, TIME, ORIGIN and USER) can appear in any order and are optional. Each should appear at most once.

DATE "date_text"
Optional between VERSION 1.0 and SURVEY. The parameter should represent the date the file was created. For example: DATE "7th July 1994"

TIME "time_text"
Optional between VERSION 1.0 and SURVEY. The parameter should represent the time the file was created. For example: TIME "11:15"

ORIGIN "origin_text"
Optional between VERSION 1.0 and SURVEY. The parameter should represent the originating system (program and operating system). For example: ORIGIN "MyProg v2, DOS 5.00"

USER "user_text"
Optional between VERSION 1.0 and SURVEY. The parameter should represent the user who created the file. For example: USER "A Smith"

SURVEY
Mandatory. Introduces details of the survey.

TITLE "survey_title_text"
Optional but if present appears between SURVEY and RECORD. The text should represent the survey title. For example: TITLE "The Fitness Centre Survey"

RECORD record_id
Mandatory. One RECORD keyword appears after SURVEY or after TITLE (after SURVEY). It is used to introduce the definition of the variables. The record_id can be used in conjunction with the variable_id (see the VARIABLE keyword) to generate a unique variable name on import. The record_id is any single character A to Z or a to z. For example: RECORD A

Definition File Keywords (continued)

For each variable being described there should be a block comprising:

VARIABLE variable_id
Mandatory. The variable_id is an integer number of up to four digits, in the range 1 to 9999, with or without leading zeroes. Each variable_id must be unique within a RECORD ... END RECORD block.
For example: VARIABLE 10

NAME "name_text"
Mandatory. The name_text should represent the name the variable had in the original survey.
For example: NAME "Q1a"

LABEL "label_text"
Mandatory. The label_text should represent the label of the original variable.
For example: LABEL "First visited"

TYPE type_name
Mandatory. The type of the variable as represented by type_name which must be one of:

SINGLE
MULTIPLE
QUANTITY
CHARACTER
LOGICAL

For example: TYPE SINGLE

The details following the TYPE specification vary according to the type_name parameter.

For *SINGLE* and *MULTIPLE* variables:

One of two methods can be used to specify the categories for variables of type SINGLE and MULTIPLE, either the VALUES method or the SIZE method:

VALUES Method
The VALUES method requires the definition of label text for each value category. This is done using the VALUES specification which consists of the VALUES keyword, followed by one or more occurrences of a category_value number (optional) and some category_label_text, followed by END VALUES:

VALUES
category_value "category_label_text"
category_value "category_label_text"
END VALUES

Definition File Keywords (continued)

The first category_value is always 1. Subsequent category_values are incremented by one, thus the final category_value is always the same as the number of items in the list. For example, a simple Yes/No variable might be recorded as:

```
TYPE SINGLE
VALUES
        1 "Yes"
        2 "No"
END VALUES
```

Or alternatively, missing out the optional category_value numbers, by:

```
TYPE SINGLE
VALUES
        "Yes"
        "No"
END VALUES
```

Where omitted, category_value numbers will be assumed from the position of the corresponding category_label_text in the list.

SIZE Method

The SIZE method requires the definition of the number of values but without details. It takes the form of a SIZE specification:

SIZE size_specification

A SIZE specification is mandatory if no VALUES specification has been given. It defines the number of category values (that is, the number of the highest value).
For example the above Yes/No variable could alternatively be described by:

```
TYPE SINGLE
SIZE 2
```

It is the responsibility of the importer to generate appropriate value label text as required.

Limitations:

1. VALUES and SIZE expressions are mutually exclusive. If a VALUES expression is used then there must be no SIZE expression given. If a SIZE expression is used then there must be no VALUES expression given.

2. At least one value category must be defined (either through the VALUES specification or through the SIZE specification).

3. There is no upper limit to the number of category values.

Definition File Keywords (continued)

For QUANTITY variables:

SIZE lowest_value **TO** highest_value

Mandatory. The SIZE specification explicitly defines the valid range, and implicitly defines the format and physical size of data for the variable. For example, if the data represented any amount between 0 and 500 to an accuracy of two decimal places:

SIZE 0.00 TO 500.00

Limitations:

1. The valid range can include positive or negative values. Negative values are identified by a single leading minus sign, '-'. Positive values are identified by the absence of a sign.

2. The number of decimal places must be the same for both the lowest_value and the highest_value parameters. The number of decimal places must be identical to the number of decimal places used to represent the data in the corresponding data file.

3. Values must contain at least one digit. The use of a decimal point is optional for integer values. The following table gives examples of correct and incorrect representations:

Value	
1.0	Correct
+1.0	Incorrect - 'plus'sign not allowed
-1.0	Correct
- 1.0	Incorrect - contains embedded spaces
1.	Correct
.1	Correct
-.1	Correct
-.	Incorrect - no numeric digits present

4. There is no upper or lower limit to the values that may be assigned to a quantity variable.

Definition File Keywords (continued)

For CHARACTER variables:

SIZE size_specification

Mandatory. Defines the maximum length of data for the variable.
For example: SIZE 20

Limitations:

1. The size_specification must be at least 1.

2. There is no upper limit to the size_specification.

For LOGICAL variables:

No keywords or parameters are used to amplify the specification of LOGICAL type variables.

Finally, for all variable types:

END VARIABLE
Mandatory. Completes definition of the variable.

Then either the definition of another variable (introduced by VARIABLE variable_id), or:

END RECORD
Mandatory. Finishes the definition for the set of variables.

END SURVEY
Mandatory. Finishes the definition for the survey.

END SSS
Mandatory. Finishes the file. Anything after this will be ignored.

The Data File

Overview

The data file is comprised of fixed-length records. Each record registers the responses for each of the variables in the corresponding definition file given by one respondent.

Data is recorded in fields of fixed length and arranged in the order defined by the variables in the definition file. The length of each field is determined from the type and size of the corresponding variable.

Basic Formatting Rules

1. Other than the record terminator (see below), only characters in the range decimal 32 to decimal 255 are considered valid - any others are considered an error when read.

2. The length of each record is determined by the corresponding definition file. The length is actually the sum of the lengths determined by the variables in the definition file (see below). There is no maximum record length.

3. Each record is terminated by either CR/LF, LF/CR, CR or LF, where CR is the carriage return character (decimal 13) and LF is the line feed character (decimal 10). Whichever terminator is used must be employed consistently - that is the same terminator must be used throughout the file.

4. The number of respondents is determined by the number of records in the file. There is no maximum number of records (and hence respondents) in the file.

5. There is no specific end-of-file character. The end of the file is determined by its physical size.

Individual Data Items

The following describe the methods used to represent data for each type of variable. In all cases, a field comprised of space characters represents missing data.

Variables of type *SINGLE*

Data is recorded as an integer number ranging from 1 to the number of values specified for the corresponding variable. The number 0 can be used to represent missing data.

The length is the minimum number of characters required to represent the largest value. Thus, variables with up to 9 values have a data field one character long; variables with from 10 to 99 values have a data field length of 2, and so on. If a particular number is smaller than the maximum for the field, it should be right justified using leading space or zero characters as padding.

For example, the seventh category value of a variable with 20 values would appear as: 07

Variables of type *MULTIPLE*

Data is recorded with one character per value of the corresponding variable. A character 1 is used to signify that a value has been selected, a character 0 signifies that a value is not selected.

The length is the number of values of the corresponding variable.

For example, a multiple variable with 5 values where the second and fifth values are selected would be recorded as: 01001

Variables of type *QUANTITY*

Data is recorded as a number with the same number of decimal places as were used in the SIZE specification of the corresponding (QUANTITY type) variable. A decimal point should always appear if one was used in the corresponding SIZE specification.

Data is as long as required to accomodate the *longest* allowable value defined by the SIZE specification of the corresponding variable. When calculating the physical size of data for the variable, an allowance should be made for the sign of negative values. Negative numbers are represented with a leading minus sign, '-'. No such allowance should be made for (the sign of) positive values. If a particular value can be represented in a smaller length then leading spaces are used as padding.

For example, a data value of 1 for a quantity variable of: SIZE 0.5 TO 1.5 would be recorded as: 1.0

Individual Data Items (continued)

Variables of type *CHARACTER*

Data is recorded as the original character string.

The length of the field is simply the value defined by the SIZE parameter of the corresponding variable. Data shorter than the maximum length is left justified and padded to the maximum length with trailing spaces.

For example a character variable of: SIZE 10 and data as the word 'CHARACTER' would be recorded as: CHARACTER

Variables of type LOGICAL

Data is recorded such that character 0 represents FALSE and character 1 represents TRUE.

The length of the field is always one character.

For example, a value of true would be represented as: 1

Examples

Example Triple-S Definition File

The example defines a survey with five variables (one each of types SINGLE, MULTIPLE, CHARACTER, QUANTITY and LOGICAL).

```
SSS
    VERSION 1.0

    DATE "06/06/94"
    TIME "16:55"
    ORIGIN "MyProg v2, DOS 5.00"
    USER "Ann Analyst"

    SURVEY
        TITLE "Historic House Exit Survey"
        RECORD V

            VARIABLE 1
                NAME "Q1"
                LABEL "Number of visits"
                TYPE SINGLE
                VALUES
                    1 "First visit"
                    2 "Visited before within the year"
                    3 "Visited before that"
                END VALUES
            END VARIABLE

            VARIABLE 2
                NAME "Q2"
                LABEL "Attractions visited"
                TYPE MULTIPLE
                VALUES
                    1 "Sherwood Forest"
                    2 "Nottingham Castle"
                    3 "{"}Friar Tuck{"} Restaurant"
                    4 "Other"
                END VALUES
            END VARIABLE

            VARIABLE 3
                NAME "Q3"
                LABEL "Other attractions"
                TYPE CHARACTER
                SIZE 30
            END VARIABLE
```

Example Triple-S Definition File (continued)

```
            VARIABLE 4
                NAME "Q4"
                LABEL "Miles travelled"
                TYPE QUANTITY
                SIZE 1 TO 999
            END VARIABLE

            VARIABLE 5
                NAME "Q5"
                LABEL "Enjoyed visit"
                TYPE LOGICAL
            END VARIABLE

        END RECORD

    END SURVEY

END SSS
```

Example Triple-S Data File

```
21011Amusement Park          0121
30100                          0031
21001"Marco's" Restaurant    0580
```

The above data corresponds to respondent data as follows:

Respondent 1:

Number of visits:	Visited before within the year	[2]	
Attractions visited:	Sherwood Forest	[1]	
	"Friar Tuck" Restaurant	[3]	
	Other		[4]
Other attractions:	Amusement Park		
Miles travelled:	12		
Enjoyed visit:	TRUE		

Respondent 2:

Number of visits:	Visited before that	[3]	
Attractions visited:	Nottingham Castle		[2]
Other attractions:			
Miles travelled:	3		
Enjoyed visit:	TRUE		

Respondent 3:

Number of visits:	Visited before within the year	[2]	
Attractions visited:	Sherwood Forest	[1]	
	Other		[4]
Other attractions:	"Marco's" Restaurant		
Miles travelled:	58		
Enjoyed visit:	FALSE		