Triple-S

The Triple-S survey interchange standard
The story so far

Stephen G Jenkins

1996

Abstract

The Triple-S standard defines a means by which both survey data and meta-data (variables) may be transferred between different survey programs running on different software and hardware platforms. As such it represents the first real attempt to resolve what should be a simple, but is frequently a complex, process.

This paper gives the background to the problem of survey transfer and charts the design of the standard including a rationale for the approach adopted. It then discusses issues raised by developers and users of the standard and finally outlines features currently under discussion for inclusion in later versions of the standard.

1. INTRODUCTION

Improvements in computer hardware and software technology have opened up more opportunities there are now software packages for all phases of survey work. Some specialise in particular stages of the process, for example, specialist CATI software; software for hand-held data collection and interviewing; and specialist software for the calculation of particular tables, charts and statistics.

In addition many users now have, or have access to, their own desk-top machines and are capable of running analyses in addition to those that may have been conducted by an external agency or bureau.

These factors have created an environment where the ability to transfer survey data between different software programs increasingly a necessity, rather than a nicety. Clearly, however the problem is not a new one but the typical solutions available and adopted so far have a number of failings. Current software implementations incorporate one or more of the following methods.

Import and export text based data files.

These are typically comma-, tab-, or space-delimited text files. A problem with this approach is that, by itself, it is only suitable for the exchange of survey respondent data. It does not allow for the exchange of metadata in the form of the survey questions or variables. Furthermore, it is either limited to representing surveys with a ‘flat’ structure or, for surveys

where a more elaborate data structure exists, the structure must be inferred from sources other than the data files themselves.

Write specific import and / or export functions to link with existing 'internal' formats.

A major failing of this approach is that import and export functions need changing when file formats used by the target software changes. For example, if software package A is able to read and write files for package B then A must undergo some form of revision every time that changes are made to the internal layout of files for B - to exacerbate the problem, A should ideally be able to support both the current and (all) previous versions of B's files thus complicating the process over time. As an added hindrance, it is not common for software writers to openly publish details of 'internal' file formats therefore some form of reverse engineering is required carrying with it the inherent risk that some of the knowledge derived or assumptions made may, in fact, be incorrect.

Individual packages devise their own published standard.

This avoids the problems associated with the reverse engineering aspects outlined above but still carries with it the risk for the importer / exporter of having to cope with apparently arbitrary changes to the interface for the target software. There also remains the issue of the number of versions to be supported increasing over time but that is likely to be moderated by a desire from the authors of the target software to avoid unnecessary changes.

Different software authors realise agreements to implement a common standard.

This is an approach that has worked well - the present author has been instrumental in the development of two such interfaces between **snap** software (from Mercator) and **Star** software (from Pulse Train Technology) which have worked effectively for a number of years. However, to be truly effective, this approach would involve a tremendous amount of work by software providers. In the extreme, **n** survey related software programs would require the specification and development of $\mathbf{n^2-1}$ different interchange formats. In practice, many of these specifications may come to resemble each other and thus it is possible, but by no means likely, that a standard comes about by evolution rather than by design.

Work with a number of other companies to develop a suitable standard.

This is the approach that has been adopted in the development of Triple-S. Of course, there is the danger with this approach that the proverbial camel is designed in place of a sleek race-horse. However, in this particular case we believe that by keeping the group down to a small size and setting ourselves clear and realistic objectives at each stage, we have at least set a path toward a universal standard.

THE CREATION OF Triple-S VERSION 1.0

At the 21st SGCSA Conference a paper was presented[(1)] which called for the development of a standard for the transfer of surveys between otherwise incompatible survey software.

Subsequently a working group, which included the present author, was formed and spent a year drafting a specification. Version 1.0 of the resulting Triple-S standard was published in

May 1995 and, since that date, almost 100 copies have been despatched by request to software developers and user organisations around the world.

Triple-S has been devised as a survey interchange format, not as a native survey definition format or a replacement for the many ?????(commercial / ad-hoc) survey definition languages currently in use. Provided that the chosen software supports the Triple-S standard, surveys can be ported between different hardware and software platforms using Triple-S format files on any convenient medium including diskettes, magnetic tape and so on.

Our primary design goals for the standard were that:

- It encapsulates the definitions and data of a large number of common surveys.
- It is reasonably simple to implement for both export and import.
- It does not prejudice further development by incorporating more than is needed. It was never an intention to make this the first and last attempt but rather to make it a good basis for further development with the benefit of experiences of both users and implementors.

We wanted to produce a standard which 'works' (in the sense that software users and developers would use it) and to produce it in a reasonable timescale. We also wanted to avoid adopting a 'lowest common denominator' solution to the problems raised. Triple-S therefore had to deal with realistically large problem sizes.

In practice, when attempting to devise such a standard, there are inevitably some comprises to be made

The approach adopted was to use ASCII text based files. A text based layout avoids the problems associated with transferring binary files - clearly this is an important point since these files are *designed* to be moved and transferred between different computer systems. The following additional benefits were perceived:

1. The files can be inspected using a suitable text editor which speeds the development of both import and export programs.
2. Import functions could be written before equivalent export functions (if any) by hand-coding suitable Triple-S files.
3. Users could introduce corrections if problems arose with particular implementations 'in the field'.

The only real drawback to using text-based files is that they are not renowned for their efficient use of storage space. However, that is not so much an issue now, and is becoming less so. Where it is, a variety of proprietary compression methods and techniques could be used to reduce storage requirements.

It was important that the syntax used would make it a relatively simple job to write a suitable export module and import parser. Anyone able to write an import module should also be able to write an equivalent export module thus placing no obstacles in the way of implementing both (where appropriate) rather than just implementing an import module - we certainly did

not want to see a multitude of programs importing Triple-S files but none offering an equivalent export.

It was essential to include a description of both the survey data and the survey metadata as well - after all, the inability to transfer metadata is what makes many of the 'standard' techniques, described previously, cumbersome to use. On the basis that we wanted to provide at least the basics, we based Version 1.0 of the standard on surveys which contained Single and Multiple response variables (for tick-box questions); Quantity variables (for open-ended numeric questions); Character variables (for open-ended text response questions) and Logical variables (for filtered sub-sets to be represented).

DESCRIPTION OF Triple-S VERSION 1.0

Description of V1

Two files

One is used to interpret the other

Example survey (use previous examples etc.)

Example of a Single variable with description of its various parts (what does 'size' mean etc.)

Include a critique of the limitations etc in here as well)

`VARIABLE 1`	*Introduction to the definition of a new variable.*
`NAME "Q1"`	*The name the variable had in the exporting survey.*
`LABEL "Number of visits"`	*A (typically short) description of the data represented by the variable*
`TYPE SINGLE`	*Type definition enables remaining specification to be interpreted.*
`VALUES`	*Introduction to the list of labels and corresponding data values*
`1 "First Visit"`	*The labels and corresponding data values*
`2 "Visited before within the year"`	*stated explicitly - the data values must*
`3 "Visited before that"`	*correspond exactly with the*
`END VALUES`	*The end of the list of values*
`END VARIABLE`	*The end of the definition of the current variable*
`VARIABLE 2`	*Introduction to the definition of a new variable.*

NAME "Q2"	*The name the variable had in the exporting survey.*
LABEL "Attractions visited"	*A (typically short) description of the data represented by the variable*
TYPE MULTIPLE	*Type definition enables remaining specification to be interpreted.*
VALUES	*Introduction to the list of labels and corresponding data values*
1 "Sherwood Forest"	*The labels and corresponding data values*
2 "Nottingham Castle"	*stated explicitly - the data values must*
3 "{"}Friar Tuck Restaurant"	*correspond exactly with the*
4 "Other"	
END VALUES	*The end of the list of values*
END VARIABLE	*The end of the definition of the current variable*
VARIABLE 3	*Introduction to the definition of a new variable.*
NAME "Q3"	*The name the variable had in the exporting survey.*
LABEL "Other attractions"	*A (typically short) description of the data represented by the variable*
TYPE CHARACTER	*Type definition enables remaining specification to be interpreted.*
SIZE 30	*Introduction to the list of labels and corresponding data values*
END VARIABLE	*The end of the definition of the current variable*
VARIABLE 4	*Introduction to the definition of a new variable.*
NAME "Q4"	*The name the variable had in the exporting survey.*
LABEL "Miles travelled"	*A (typically short) description of the data represented by the variable*
TYPE QUANTITY	*Type definition enables remaining specification to be interpreted.*
SIZE 1 TO 999	*Introduction to the list of labels and corresponding data values*
END VARIABLE	*The end of the definition of the current variable*

VARIABLE 5	*Introduction to the definition of a new variable.*
NAME "Q5"	*The name the variable had in the exporting survey.*
LABEL "Enjoyed visit"	*A (typically short) description of the data represented by the variable*
TYPE LOGICAL	*Type definition enables remaining specification to be interpreted.*
END VARIABLE	*The end of the definition of the current variable*

BNF definition ??

PROBLEMS ENCOUNTERED / LIMITATIONS IMPOSED / Triple-S IN USE

As already been discussed, a number of limitations were necessarily introduced, and there are a number of solutions to the problems they create.

Flat file data structure

The majority of surveys conducted either naturally conform to a flat structure or are forced to by limitations in the software used for definition, entry of data or subsequent analysis.

However, where more complex structures do exist, it is always possible to construct a 'flat' view for a particular aspect. For example, a survey of persons within households could be exported with a household perspective or a person perspective:

- In the household view export, each data record would represent an individual household and possibly contain summary data for the persons within (such as perhaps the number in total, the numbers by gender, the numbers by age categories etc).

- In the person view export each data record would register details of an individual together with corresponding data relating to the household in which they live. If an indication of the number of the person within the household was also included then analyses of household-only data could be performed by selecting only those records where the person number was 1. If there were any households with no persons within then the corresponding person data would be recorded as missing and be given a person number of 0 - in that case, household-only analyses would be performed on data where the person number was 0 *or* 1 and person-only or person-household analyses would be performed on data where person number was 1 or greater.
- Although more flexible, the latter form does not, by itself, allow a complete analysis of the original data - it does not, for example, allow for calculations of the average number of persons per household. Hence both forms would be required to perform a full analysis (and even then, knowledge of the anticipated analyses would be required by the exporter).
- Generally, the more complex the structure to be represented, the more the scope of the analysis needs to be anticipated by the exporter.

Data types modelled

There are no special facilities for representing and maintaining the semantics of constructed values such as dates, times and special coding schemes. The typical solution is to export such data as a *CHARACTER* type variable - this deals with the representation issue but leaves the interpretation for (hand) reconstruction in the importing software.

With continuous data values an alternative would be for the exporter to generate a suitable numeric representation and export a *QUANTITY* variable. For example, data representing dates could be exported in a*YYYYMMDD* format and have an associated *QUANTITY* variable with a specification of *SIZE 19960101 TO 19961231* (to indicate that any date in 1996 is acceptable).

No limitation on problem sizes

Application programs (for survey work) invariably impose some size limitations on the problems they will cope with, for example there may be a limit to the number of variables, or the number of cases (data records), or both. There may be some limit to the number of codes or values a particular variable could represent or a limit to the (maximum *or* minimum) length of the text label associated with each variable code or value.

By avoiding specifying any size limitations, the importer is given the onus of resolving such problems where they arise. So that if, for example, an importer is reading a Triple-S definition file which describes a variable with 100 codes and the importing software only allows 50 codes per variable then it could choose to:

- Create a number of variables, each of which has the maximum number of allowable codes defined (50 in the case cited) and which, together, are capable of analysing the original data.

- Create one variable of up to the maximum allowable number of codes, any remaining codes are then ‘lost’ from the import.

- Do not attempt to import ‘oversize’ variables at all.

- Abandon the attempt to import the entire survey if an ‘oversize’ variable is encountered.

The conventional solution to the problem of oversize text labels is to truncate them, perhaps using ellipses to indicate that truncation has taken place.

No constraint on naming conventions

This is similar to the above problem in that the variable names found may exceed the maximum length allowed by the importer. However, they may also be syntactically or semantically incorrect to the importing program for a number of reasons:

- Names may contain illegal characters. Some programs allow the use of underscore characters, ‘_’, others do not.

- Names may be composed incorrectly. Many programs expect names to begin with a letter (A to Z) but this is not a universal requirement.

- Some names may be reserved for a special meaning within the importing software.

One might anyway expect the names to be unique but that is not guaranteed. The names ‘Q1’ and ‘M1’ are obviously distinct but what about the names ‘Q1’ and ‘q1’ ? Some, but not all software, would determine that ‘Q1’ and ‘q1’ amount to the same name. The issue is not just one of the case of characters used, either, for example some programs will treat ‘Q01’ (zero between Q and 1) as being equivalent to ‘Q1’.

The typical solution to this problem is to ignore the defined names and generate new names by appending the variable number to the *RECORD* identifier character. This guarantees unique names comprised of a letter followed by from one to four digits and it would be an unusual program that found such names unacceptable. The problem with this approach is that frequently all of the variables are renamed thus making it harder for a user of an exported survey to equate with its imported version.

An alternative solution is to attempt to use the names as they were defined. Where problems arise, that is where the importer considers names illegal, alternatives can be generated perhaps using the technique described above. However, generating unique names that might be memorised by a user is not an easy task and, anyway, it is always possible that a generated name might duplicate a name given later in the import.

Triple-S IN USE

Version 1.0 of the Triple-S standard was published in September 1994 and, since then, copies have been issued by request to over 80 developers and users of survey software around the world. In February 1996 a survey was conducted of those who had requested information in an attempt to determine the

Standard produced in 1994 ??

Sent to some 80 software developers and users around the world by request

Some implementors using

Survey of implementors

Those exporting

Those importing

Some ‘interesting’ facts from those replying:

There is a strong feeling that an attempt at a standard was long overdue

Most of those implementing Triple-S implemented a Triple-S export -

not surprising when one considers that many of those are concerned with the data-collection end of the survey process

A number of <Survey Houses> identified a requirement to supply survey data to clients without dictating the software that those clients might use to

perform their own analysis.

Developers of software for hand-held data collection units, desktop CATI and

CAPI systems, questionnaire design software, scanning systems.

Linking bespoke in-house software with bought-in package software

Linking different software packages together.

Comments from users

Massey Ferguson using SNAP on a PC running Windows to do their own, in-house

analysis of a survey originally conducted by ??? using Merlin on a ??? machine.

FUTURE DIRECTIONS

Possible Future Additions

Following publication of the original, version 1.0 of the Triple-S standard, and with the benefit of comments from implementors, users and other interested parties, work began in January 1996 to draw up a list of enhancements and modifications for consideration.

It was felt that some of the proposed changes, although reasonably simple to implement, would greatly increase the flexibility and utility of the current standard. These have been (and, at the time of writing, are being) considered for a Version 1.1 of Triple-S. Other, more complex requirements have been put under an umbrella Version 2.0.

It is anticipated that version 1.1 of the Triple-S standard will be published in advance of the ASC Conference in September 1996.

The following items have been put forward for consideration for Version 1.1:

Compatibility with earlier (1.0) version

All version 1.0 definition files must be readable by a Version 1.1 importing program, and be interpreted identically to a version 1.0 program.

Comments and Notes will enable descriptions and explanations to be incorporated.

The introduction of Comments and Notes implies that more hand-crafting of the contents of the specification file is anticipated. Additionally, since the body of a comment would be ignored by an importer, comments could be used to ‘hide’ sections of a Triple-S file if

required - maybe to avoid importing unnecessary variables (and associated data); or to avoid sections that the importing software is having problems with for some reason..

Optional parameter section

The exporting software frequently 'knows' much more about the survey being exported than the elements of the Triple-S definition and data files can portray. An optional section located toward the start of the definition file would enable some of this extra information to be passed on by the exporter and used at the discretion (or abilities of) the importer. For example, the exporter may know that the names of the objects defined begin with a letter, contain only letters and numeric digits, are between 1 and 8 characters in length and are unique within the scope of the survey. By expressing information such as this an importer can make much more informed decisions about the import process.

Position and length information for variables

Currently, the position of the data field corresponding to a particular variable is implied by the location of the variable relative to all others in the definition file. Introducing this feature brings two benefits:

- It enables the data file to be interpreted much more readily by eye than is currently the case. To estimate the position of the data field for a particular variable in a Version 1.0 data file, the observer would have to inspect all variables prior to the one of interest and evaluate and sum the lengths of their corresponding data fields.

- It provides for sections of the definition file to be omitted (possibly using the comment feature outlined above) for whatever reason.

Retain original coding

There are many instances where the data is represented by its *value* rather than by a substituted code. For example, it would be commonplace to see characters '1' and '2' used to record the responses to a question with 'Yes / No' choices. However, where the data represented such things as postcodes, US State codes the data value itself is often used as its own code. In these instances users (researchers) often know that the data is recorded in exactly that way and adding this feature enables the semantics of these values to be retained.

Data recorded as 'spread fields'

Spread fields provide a way of recording multiple-response data in contiguous space-delimited sub-fields. The benefits of this approach are that both the original coding scheme and 'order of mention' are preserved.

As an example, suppose that a question 'Which instruments do you generally use for writing ? ' was asked and that the following choices were given: Fountain Pen (code 1), Ballpoint Pen (2), Pencil (code 3), and Other (code 4). A respondent giving replies 'Pencil' and 'Fountain Pen' would have data recorded in a Triple-S Version 1.0 data file as:

1010

But in a spread field format the same data might be recorded as:

31__ ('_' represents space)

Extended 'missing' values

Many survey programs define more than one category of 'missing' data. They might for example include such categories as 'Don't know' and 'Can't remember' as well as 'No Reply' and 'Not Asked'. The current, version 1.0, standard only allows for the specification of exactly one 'missing' category for each variable.

Rating scales

Many studies such as employee surveys and consumer surveys use rating scales to provide a single measure for attitude scales. For example, questions with a response scale Very Good / Good / Ok / Poor / Very Poor might be analysed by calculating the mean of scores of 2 / 1 / 0 / -1 / -2; thus, increasingly strong negative attitudes (towards Very Poor) result in correspondingly negative mean scores, strong positive response (towards Very Good) result in positive mean score and middling opinions (around Ok) result in a near zero score.

Rating scales provide such a facility without necessarily being 'tied' to any one particular variable.

Dates and Times

A method has been given earlier in this paper for dealing with Date type data within the confines of the Version 1.0 specification. Providing in-built types for Date and Time will simplify the process for those programs able to manipulate data and time values directly. The downside is that the import process will be made more complex for software with no specific representation of Dates or Times.

At present, four broad topics have been put forward for consideration for a Version 2.0 of Triple-S and these are: Structured datasets; Shared answer lists; Routing information; and definition of analyses.

Structured datasets

This feature would enable at least two-level hierarchical datasets to be described. This would be sufficient for representing simple household / person surveys for example. However, the scope may extend much further to hierarchies with more than two levels (for example, Household / Person / Trip structures) or, indeed, to the ability to describe any dataset which can be represented by a directed acyclic graph.

Shared Answer lists

The ability to describe shared code lists enables a potentially large reduction in the size of a Triple-S definition file where many variables use identical answer lists. This often occurs in attitude surveys where there are frequently many questions asked to which the response is one of Very Good / Good / Ok / Poor / Very Poor or some similar scale. This feature is

similar to, but distinct from, the Rating Scales feature currently under consideration for Version 1.1 and described above.

Routing information

The inclusion of routing information would enable some form of interview flow-of-control to be described. At the time of writing, no discussions have taken place as to the, even approximate, form this might take.

Definition of analyses

The inclusion of analysis definitions, probably in the form of tables, would not only enable standard analyses to be transferred but also provide another check on the correctness of the import - that is by comparing tables built by the exporter with those constructed by the importer. Again, no discussions have so far taken place as to the form the definition of analyses might take - only that they be considered.

The inclusion of specific derivation expressions is not considered feasible at this time due to the large number of differences between specification languages in host software and the consequent complexity involved in translating to and from an intermediate, portable form. However, who knows what lies beyond ?

CONCLUSIONS

In general, as the features supported by Triple-S become richer, so more emphasis is put on the importing side of the transfer equation. Put simply, a software program which does not support a particular feature will obviously have no problem exporting its own surveys because none of them (the surveys) will exhibit that property. But, when importing, these features may well be encountered and thus demand some course of action to be taken - whether that is to refuse to import those surveys or to implement some form of strategic conversion or ‘fixup’.

Making the import process complex at the expense of the export process directly endorses one of our original goals and should make the exchange and interchange of data and metadata between survey software packages more commonplace as time goes on.