How to Make Osiris More Interoperable While Waiting for the Ideal System

by Lennart Brantgarde and Leo Rubinstein[1],

The concept of interoperability will in this paper be understood as a quality that makes a system flexible over time and adaptable in relation to other systems. It stands for high portability but also for high usability for different purposes.

The Osiris format was long ago sentenced to death. Its death-struggle has now been streched into an extremely painful decennial process and I would guess that there will be several years more to come before it is over. The reason for this extended process is not so much dependent upon an abundance of virtues of Osiris but on the lack of alternatives. Osiris is amazingly enough still the only format that easily can reach a high interoperability versus a variety of functions that you have to operate at a modern dataoriented archive.

Two contrary principles for a data archive are:

1. Archive and store what you get and disseminate what you have

2. Transform what you get into a format that enables you to disseminate what is wanted in terms of formats by the market.

We suppose no one operates exactly on one of these extremes - we are located on the continuum in between, some more to the first position, some more to the second. This paper deals with how to achieve a position close to the second while still using Osiris as the basic archival format for preservation and documentation.

As you know one of the basics of Osiris is its oldfashioned fixed format. This makes it easy to program utilities for since every single unit of information always is found at the same spot. One way of making Osiris more interoperable is to create utilities. And we would guess that a whole set of utilities have emerged at various data libraries over the years but unfortunately they have never been collected and distributed to the benefit of all. At our site we have therefore been forced into this kind of utility production. We are going to present five such utilities and show how you can get hold of the programs.

From Osiris type 1 to Osiris type 3

In the first place we had to convert an Osiris-type-1-codebook into an Osiris-type-3-codebook. The binaries in the type-1-codebook were of no use in the long run at least if you want full control of the file. We created a program that is called ***123.***

This program comprises two things

1. Conversions of binary numbers to decimal codes.

2. Conversion of character codes, usually from EBCDIC to ASCII

As far as the binary numbers are concerned a complicating fact is that some data entries in a type-1 codebook are stored binary in 2 bytes and can have values between zero and 65336. The same information in a type 3 codebook has only four digits reserved and they are supposed to be stored decimally. Consequently no higher numbers than 9999 can be generated.

Our solution to this was the gv-convention. This comes into consideration only for numbers above 9999 and consists of a hexadecimal kind of system where zero thru nine and capital A thru capital F have been replaced by the minor letters g thru v where these letters stands for zero thru 15. In other words base 10 has been replaced by base 16 which allow us to take care of all binary numbers without losing any information and at the same time preserve the basic format of type 3.

The second problem - moving from character code representation to another - was solved by using, as a second parameter, a name of a file containing a translation matrix, a table with 256 lines. It is up to the user to define this table or modify existing ones. In such a table there are just two columns where the left one has entries equal to the line number -1 and the right one tells what character should replace the former. Value -1 in the right column means that the code is undefined. When encountering such undefined values the program shows the partially converted line and asks for a character to put in.

The program can be initiated by the command:

123 codebook [table]

where codebook is the stem of name of the codebook file and table the name of the translation table file. The program assumes that infile is codebook.os1 and writes the outfile codebook.os3. If the optional parameter table is left out the program assumes that the existing internal table named ascii

should be used.

Making an eyefriendly document out of an Osiris-type-3-codebook.

The original basic fixed formatted Osiris codebook file is not very useful for having as a readable document on the bookshelf. You have to get rid of all abundant figures and rearrange the information into a userfriendly textbook. This is done in our environment by a program called

kbl

The program is initiated by the command:

kbl infile outfile [dumpfile]

where infile defines your osiris-type-3-codebook-file and outfile defines what file you want to get out. In a unix environment the outfile can be browsed using the unix utility more and printed using lpr .

Moving an Osiris-type-3-codebook into World Wide Web or AUIS.

A modern Data library or a modern Archive has to consider desktop browsing of the archival holdings. This is absolutely necessary for an archive that has to deal with a scattered academic community located miles and hours from the office. To our help facilities like WWW, Netscape and Mosaic have emerged like benevolent fairies out of the dark.

Like the original fixedformatted Osiris the modern WWW is carrying an heritage from the childhood of computing. To get a text into good order in a web you have to tag it into HTML which is a sample of SGML. The next utilityprogram to be presented here tranforms an Osiris-typ-3-codebook into an html-tagged document that, put out on the server, not only makes use of bold characters and italics but also automatically creates internal weblinks between the table of contents and the rest of the codebook document. In one stroke a stereotype Osiris codebook is turned into a decent modern eyefriendly document and a fullyfledged hypertext document.

The program is called htkbl and is started by the command

htkbl infile outfile [dumpfile]

exactly repeating the moments of the former program. Dumpfile is the name of a file to where wrong lines in a codebook (Infile) will be copied if any. For the AUIS-environment a utility program called **atkkbl** can be used the same way.

Getting from Osiris to SPSS and further out

So far we have just talked about utilities facilitating the infomative functions of an archive - to get Osiris ready for printing and make it exposable through modern networks. Fundamental to an archive is to disseminate both data AND documentation.

In order to facilitate distribution of data in a variety of formats we have created a utility that enable us to go automatically from Osiris-type-3 to a portable systems file in SPSS.

The program is called **otosp** and is started thru command

otosp codebook setup datafile exportfile

where codebook is the name of your Osiris codebookfile (Type 3), where setup is the name of file including all Spss control cards -including value labels - , where datafile is the name of your Osiris datafile and where exportfile is the name of your outfile created by SPSS.

The program can also be started by its mere name

otosp

in which case the program will be prompting questions for all other parameters.

Once in SPSS there are other conversion programs that can bring you further out into the djungle of statistical packages.

How do I get hold of these utilities?

All of them are stored under /pub/programs in

ftp://oden.ssd.gu.se

You are welcome to pick what you want. Source codes to 123, otosp, kbl, atkkbl and htkbl are free to be copied , compiled and used. But it is to be noted that these programs were created to suit the needs and requirements at the SSD and so far they have been tested only here. You should pay some attention to readme-files in subdirectories for improving your chances to install and customize the programs for your environment.

For those of you who wou ld like to have a training example we have also made available a codebook of the oldfashioned ICPSR type, the codebook to Almond and Verba The Civic Culture. This exist in a type-1 binary version and is supposed to be converted to a typ-3 ascii version and further on to WWW and SPSS.

For installing copy a complete subdirectory respectively to your computer read the read-me file modify the make file: change the dir-string to the name of an appropriate directory on your computer (in search path for executable files). run the make file.

Further questions should be addressed to:
Leo.Rubinstein@ssd.gu.se

Good Luck!

1. This paper has been presented at the CSS96/IASSIST conference at Minneapolis, University of Minnesota, May 12 - 19, 1996. It describes a set of programs that have been built around a major documentation system called A-SIDE developed by Stephen Greene for Swedish Social Science Data Service and presented in Stephan Greene: A Functional approach to Documentation and Metadata, IASSIST Quaterly vol 19 no 1, spring 1995 pp18. Support for this presentation has been received from the Swedish Institute, Stockholm, grant 989/301/40