Statistical Data Processing with R – Metadata Driven Approach

Rudi SELJAK (rudi.seljak@gov.si)
Statistical Office of the Republic of Slovenia
Jerneja PIKELJ (jerneja.pikelj@gov.si)
Statistical Office of the Republic of Slovenia

ABSTRACT

In recent years the Statistical Office of the Republic of Slovenia has put a lot of effort into re-designing its statistical process. We replaced the classical stove-pipe oriented production system with general software solutions, based on the metadata driven approach. This means that one general program code, which is parametrized with process metadata, is used for data processing for a particular survey. Currently, the general program code is entirely based on SAS macros, but in the future we would like to explore how successfully statistical software R can be used for this approach.

Paper describes the metadata driven principle for data validation, generic software solution and main issues connected with the use of statistical software R for this approach.

Keywords: *Statistical software R, metadata driven systems, data validation*
JEL: *C10, C18, C80, C88*

INTRODUCTION

In 2011 the Statistical Office of the Republic of Slovenia (hereinafter SURS) started an internal project Standardization of Statistical Data Processing in order to find a more rational solution that would replace classical stove-pipe oriented production system of data processing. The main goal of the project is automatization of the data production process, ensuring repeatability and traceability and consequently making it more transparent.

The central part of the paper is dedicated to the description of the main features of the data validation process by the metadata driven approach with statistical software R. Some conclusions and the main issues connected with the use of R are stated in the last part of the paper.

GENERAL SOLUTIONS – MAIN CHARACTERISTICS

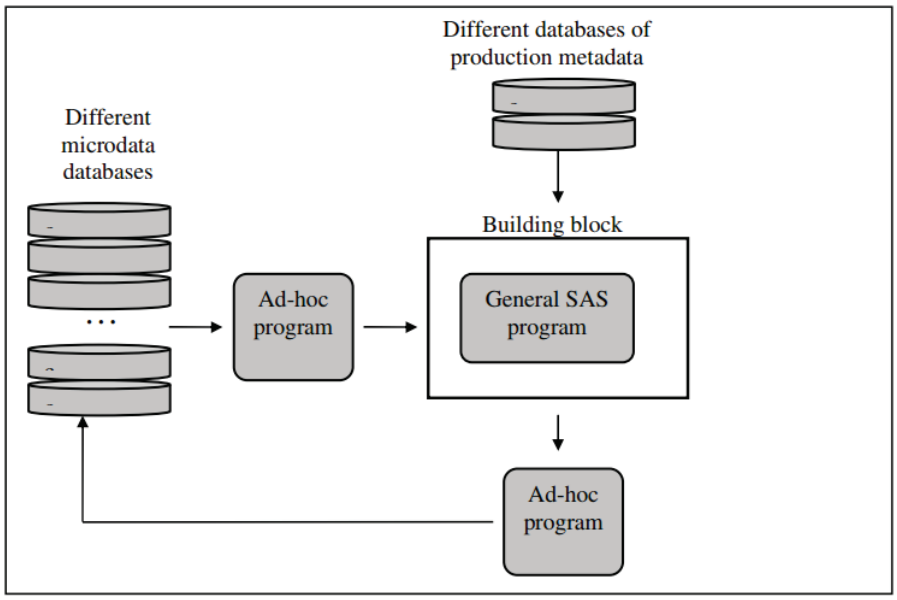
Modernization and standardization of the system of statistical production at SURS are based on the development of small generic solutions. These small generic software solutions are often called building blocks. They are designed in such a way, that they enable easy and flexible linking of inputs and outputs of the individual components to the whole statistical process.

These small generic software solutions are based on the metadata driven approach. This means that all the parameters for the particular survey and the particular reference period are stored and maintained in a special database, called metadata database. The second part of the application is an ad-hoc program, where the table of microdata, the main input of the building block, is prepared. This program is different for every survey, because we allow microdata to be stored in different types of databases, and usually we need to merge microdata from different locations and databases to get the final input table. The third part of the application is a general program, which is never changed for the needs of a particular survey.

A simplified schematic presentation of functioning of such a building block is presented in the following figure:

Schematic presentation of functioning of the building block

Figure 1



One building block usually covers only a small part of the statistical production chain. As already mentioned above, only data validation process will be presented later on in this paper. This is in fact a very important part of statistical process; because the following processes (e.g. data editing and imputation) are to a large extent based on the results of the data validation.

DATA VALIDATION - PROCESS METADATA

In this section we will provide a short illustrative example of the usage of process metadata for data validation. We have chosen the Monthly Statistical Survey on Earnings Paid by Legal Persons to show how to define the metadata table, present the general programing code in R and the final outputs of this process.

About the data

The statistical survey Monthly Report on Earnings Paid out by Legal Persons provides an insight into the amount of average monthly earnings and their changes in the Republic of Slovenia. Observation units are legal persons of the public and private sector, or their units, registered for performing activity in the Republic of Slovenia.

Variables which are the subject of our interest are:

- LETO – year
- MESEC – month
- MAT_ST – id variable
- PL_13PL - gross paid 13th wage
- BP_13PL_KP - gross paid 13th salary on the basis of collective agreement
- BP_13PL_IP - gross paid 13th salary on the basis of individual agreement
- BRUTO_PLACA – gross wage
- NETO_PLACA – net wage
- MES_ZI - the number of months with overdue payments
- BP_NIZ - the minimum gross wage
- ZAP_MIN – the number of employees who received minimum wage
- ZAP_BP – the number of employees who received a wage based on the hours worked
- ZAP_BP_KP - the number of employees who received a wage based on the hours worked on the basis of collective agreement
- ZAP_BP_IP - the number of employees who received a wage based on the hours worked on the basis of individual agreement
- ST_NADURE - the number of paid overtime hours
- ZAP_PL - the number of employees who received wages

The main purpose of the data validation process is to perform an overview over the microdata and to lookup for some errors that need to be corrected in the next step of statistical process, which is data editing. In our illustrative example we will define six logical checks, which will determine three different types of errors: error, warning and other. We usually use the option other in cases, when we just want to check something in data and it is not necessary that this is an error in our microdata.

- ❖ OTHER: How many units received the 13rd wage.
- ❖ ERROR: Error, if the net wage is greater than gross wage.
- ❖ WARNING: How many units delayed the payment for at least 6 months.
- ❖ OTHER: How many companies have a min of gross wage higher than €1.500.
- ❖ WARNING: Check whether more than half of units have the min wage.
- ❖ WARNING: Check if units have on average been for more than 40 hours of overtime work.

Process metadata

There is only one table, which provides all the metadata that we need for data validation. Suppose that we want to check if the net wage is higher than gross wage. This would clearly be an error in the microdata that have to be corrected. The structure of the table must have the following structure:

Metadata for logical checks

Table 1

TABLE	LC_LABEL	ERROR_ DESCRIPTION	CONDITION	ERROR_TYPE	VALIDITY
DATA	LK002	Error, if the net wage is greater than gross wage.	BRUTO_PLACA<NETO_ PLACA	ERROR	1

Short description of the fields:

LC_LABEL	Label of the logical check. It has to begin with LK.
ERROR_DESCRIPTION	Description of the error.
CONDITION	Condition which determines our check.
ERROR_TYPE	Type of an error e.g. error, warning or other options.
VALIDITY	Validity for the specific check. If the value is zero, the check will not be executed.

In our case the metadata table is constructed in Excel, but in general it can be located in any other database, which can be connected to R, for example MS Access or ORACLE.

R code and results

Packages, which have been used for metadata driven approach are:

- ❖ openxlsx
- ❖ ROracle

First we read the metadata from Excel

```
checks<-read.xlsx(„Metadata.xlsx“, sheet = 1, startRow = 1,
colNames = TRUE,skipEmptyRows = TRUE, rowNames = FALSE)
  > checks

    TABLE LC_LABEL ERROR_DESCRIPTION
[1] data  LK001   How many units received the 13rd wage.
[2] data  LK002   Error, if the net wage is greater than gorss
                  wage.
[3] data  LK003   How many units delayed the payment for at
                  least 6 months.
[4] data  LK004   How many units have a min of gross wage
                  higher than €1.500.
[5] data  LK005   Check whether more than half of units have
                  the min wage
[6] data  LK006   Check if units have on average paid for more
                  than 40 hours of overtime work.

CONDITION                                          ERROR_TYPE VALIDITY
PL_13PL > 0 || BP_13PL_KP > 0 || BP_13PL_IP > 0     OTHER       1
BRUTO_PLACA<NETO_PLACA                              ERROR       1
MES_ZI>=6                                          WARNING      1
BP_NIZ>1500                                         OTHER       1
ZAP_MIN/(ZAP_BP+ZAP_BP_KP+ZAP_BP_IP)>0.5           WARNING      1
ST_NADURE/ZAP_PL>40                                WARNING      1
```

At the second step we create new variable R_st, where the programing code for logical checks is defined.

```
R_st <- paste(checks$TABLE,“$“,checks$LC_LABEL,“<- ifelse(„,
checks$CONDITION,“,1,0)“,sep= „“)

  > R_st
[1] „data$LK001<- ifelse(PL_13PL>0||BP_13PL_KP>0||BP_13PL_
    IP>0,1,0)“
[2] „data$LK002<- ifelse(BRUTO_PLACA<NETO_PLACA,1,0)“
[3] „data$LK003<- ifelse(MES_ZI>=6,1,0)“
[4] „data$LK004<- ifelse(BP_NIZ>1500,1,0)“
[5] „data$LK005<- ifelse(ZAP_MIN/(ZAP_BP+ZAP_BP_KP+ZAP_BP_
    IP)>0.5,1,0)“
[6] „data$LK006<- ifelse(ST_NADURE/ZAP_PL>40,1,0)“
```

Let us say, that `data` is the name of the table with microdata. Then every element of variable `R_st` defines new "dummy variable" in the microdata table and each of these dummy variables represent one of the logical checks, because it is defined under the condition given in the metadata table. For example, variable `LK001` in the microdata table will have value 1, if the condition will hold for a specific unit and 0 otherwise.

In the next step we read the microdata from the database and save it in the table named `data` as mentioned earlier. The table with microdata is usually prepared in the so called ad-hoc program, where microdata from different environments and surveys are combined. In our case all the variables that we need are in the Oracle data base – table `ZAPM_MAT` on schema `ZAPM`. Below is an example of how to access the data, located in the Oracle database.

1. Define parameters, specific for the database:

```
driver <- dbDriver(„Oracle")
host <- host
port <- port
sid <- sid
user <- user
pwd <- pass
```

2. Make the connection to the database:

```
connect.string <- paste(
  „(DESCRIPTION=",
  „(ADDRESS=(PROTOCOL=tcp)(HOST=", host, „)(PORT=", port, „))",
  „(CONNECT_DATA=(SID=", sid, „)))", sep = „")
con <- dbConnect(driver, username=user, password = pwd, dbname
= connect.string)
```

3. Read the microdata and attach the variables in the workspace

```
data<-dbReadTable(con, „ZAPM_MAT", schema = „ZAPM")
data<-data[(LETO==2012 & MESEC==01)]
attach(data)
```

The main trick in the whole process is how the code in character variable `R_st` is executed. We can do this by using functions eval and parse in the variable `R_st`.

```
for (i in 1:length(R_st)){
  eval(parse(text=R_st[i]))
}
```

In this step all the new dummy variables are not just defined, but they are created in the table of microdata. And as we have all these variables, we

are able to make some summary statistics on them; to count how many "failed units" corresponds to the given conditions on each of the logical check.

There are two final outputs that are important for the user. The first one is the table with information on the "failed units" for each logical check. In this table we can see at which logical checks the unit passed its condition and on which failed.

```
  > down<-podatki[grep(„^[LK,MAT_ST]", names(data), value=TRUE)]
  > head(down)

      MAT_ST    LK001 LK002 LK003 LK004 LK005 LK006
1 xxxxxxxxx1    1     0     1     1     1     1
2 xxxxxxxxx2    0     0     1     0     0     1
3 xxxxxxxxx3    1     0     0     0     1     1
4 xxxxxxxxx4    1     0     1     0     1     1
5 xxxxxxxxx5    1     0     1     0     1     1
6 xxxxxxxxx6    0     0     1     1     1     0
```

It can be clearly seen that summary statistics is easier if the labels of all logical checks have something in common. That is why we have decided that all labels have to begin with letters LK.

In the second output we summarize the first output and count how many units failed to the conditions for each of the logical check.

```
 > nr_down<-t(down[grep(„^[LK]", names(data), value=TRUE)])
 > nr_down<-apply(nr_down,1,sum)
 > nr_down<-data.frame(nr_down)
 > nr_down

      nr_down ERROR_DESCRIPTION
LK001 4842    How many units received the 13rd wage.
LK002 0       Error, if the net wage is greater than gorss wage.
LK003 5236    How many units delayed the payment for at least 6 months.
LK004 321     How many units have a min of gross wage higher than
              €1.500.
LK005 5464    Check whether more than half of units have the min wage.
LK006 786     Check if units have on average paid for more than 40
              hours of overtime work.
```

We usually export these final tables into an Excel file for needs of the the users.

```
 wb <- createWorkbook()
 addWorksheet(wb, „down")
 writeData(wb,"down",down,startCol=1,startRow=1,rowNames=FALSE)
```

```
addWorksheet(wb,"nr_down")
writeData(wb,"nr_down",nr_down,startCol=1,startRow=1,rowNames=TRUE)
saveWorkbook(wb,"checks.xlsx",overwrite=TRUE)
```

CONCLUSIONS

The paper presents the general concepts of the generic solution and its implementation with R programing language. If we compare the code in R with the one which has already been in use and is made in SAS programing language, we must admit that there is no big difference in the length of it or its difficulty. The benefit of R is that the transformation of the character variable into a programing code is made in one single step. The same thing is made in SAS in two steps, because we have to export the character variable into txt file and then import it back to SAS. But on the other hand R has some other disadvantages. If we want to write metadata conditions without a prefix table_name$ we need to attach the variables of the microdata table in the process. The main issue with this is, if we are dealing with more than one table of microdata and the names of variables are the same.

This application is a great benefit for SURS. So far, several modules covering different parts of the statistical process (e.g. data validation, deterministic corrections, imputations, aggregation, standard error estimation, tabulation…) have already been developed and are also already introduced into the statistical production.

References

1. **Seljak, R., Pikelj, J., Malešič, K.**, 2016, *"Statistical data processing – new approaches at Statistical Office of the Republic of Slovenia"*, paper will be presented at the International Statistical Conference in Croatia- ISCCRO'16, Croatia (Zagreb)
2. **Dolenc, D., Krek, M., Seljak, R.,** 2011, *"Editing Process in the Case of Slovenian Register-based Census"*, paper presented at the UNECE Work Session on Statistical Data Editing, Slovenia (Ljubljana)
3. **Seljak, R., Blazic, P.**, 2011, *"Sampling error estimation – SORS practice"*, Presented at the 2nd European Establishment Statistics Workshop, Neuchatel, Switzerland, 12-14 September, 2011
4. **Seljak, R.**, 2009, *"Integrated statistical systems and their flexibility – How to find the balance?"*, Presented at the NTTS conference, Brussels, Belgium, 5-7 March, 2013
5. **Seljak R.**, 2014, *"Metadata driven application for data processing – from local toward global solution"*, paper presented at the UNECE Work Session on Statistical Data Editing, France (Paris)
6. **M. Matek**, *ANNUAL QUALITY REPORT FOR THE SURVEY,* Monthly Report on Earnings Paid out by Legal Persons, available at: ***http://www.stat.si/doc/metodologija/kakovost/LPK_ZAPM_2013_eng.pdf***
7. **R Core Team** 2015, **R: A language and environment for statistical computing**. R Foundation for Statistical Computing, Vienna, Austria. ***https://www.R-project.org/***.