Error Reporting and User Interfaces for Post-collection Data Editing

Pavle Kozjek, Statistical Office of the Republic of Slovenia (SORS)

1. Introduction

At Statistical Office of Slovenia, Blaise is used for integrated data collection and editing of CAI surveys, as well as for editing of data from other sources (from high- speed data entry, secondary sources, follow-up of CAI surveys, etc.). The old interactive editing system (GVS-2000) running on a mainframe is no longer supported and is being replaced by Blaise. There are also some old Cobol and PL-1 data editing applications on the mainframe that also need to be replaced as soon as possible. They are not interactive and are organised as time consuming cyclic editing processes with printed lists of errors and re-entered corrections.

Although the GVS-2000 system is now outdated, it used to provide some good and functional solutions for data editing for many years, and users (data editing staff) were satisfied with it. In a new network environment, users expect at least all the functionality of the previous system. Since Blaise is mainly CAI oriented, some functionality needed for batch processing and post collection editing is not originally obtained by the system. The paper describes our approach how to add this functionality.

2. Data editing system on SORS LAN: some development issues

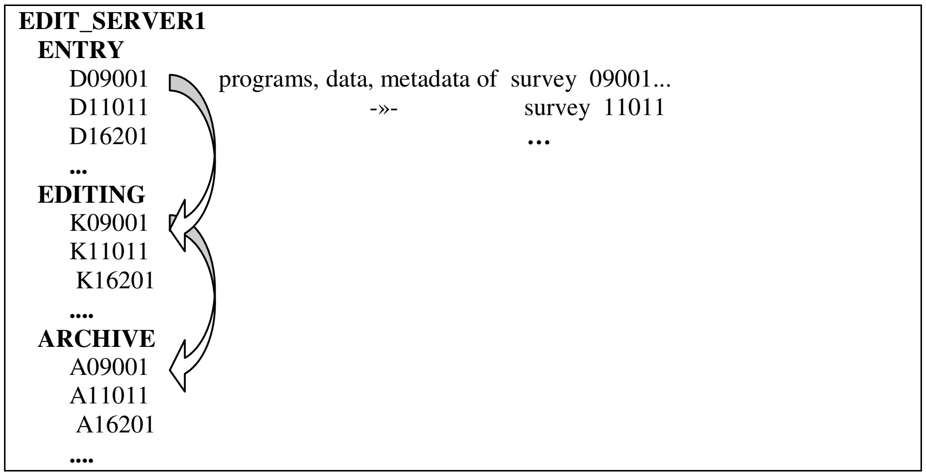
In the past, probably the most obvious problem of the data editing system at SORS LAN was the absence of standard generic procedures to obtain control and overview of the editing processes and to allow different types of users preparing, maintaining and using data editing applications in the same way. It was therefore absolutely necessary to define proper standards and conventions to avoid tailor-made solutions for each single survey. We were aware that for instance some CAI surveys (e.g. LFS, HES, etc.) can never completely “fit” into such a standardized system, due to their complexity and special needs, but the majority of other surveys can be placed there.

When developing a LAN data editing system, we followed the example and used the main principles of GEntry, a relatively successful in-house developed system for high-speed data entry. Blaise (version 4.4) was used for interactive and batch editing procedures, and VB for user interfaces. However, data editing is a much more complex process than high-speed data entry, so we were afraid to meet some unsolvable problems during development.

Some of the main expected characteristics of the new LAN editing system are:

- Standardized and transferable LAN environment for data editing (hierarchical system of folders on the server)
- Standard set of main tools (Blaise, VB, etc.)
- Generic and parameterized user interfaces
- Standard naming conventions of:
    - survey applications, based on a codelist of SORS surveys
    - generic programs (for instance; M*code* = DB of data for editing, A*code* = address register DB; *code* = survey code from SORS register of surveys)
    - fields in the datamodel. SORS’s metabase (under development) is expected to have a very important role here.

* Standardised flow of data and metadata:



```
EDIT_SERVER1
  ENTRY
    D09001      programs, data, metadata of  survey  09001...
    D11011                    -»-                survey  11011
    D16201                                       ...
    ...
  EDITING
    K09001
    K11011
     K16201
    ....
  ARCHIVE
    A09001
    A11011
     A16201
    ....
```

Figure 1: folder structure on server and basic data flow in the proces od fata editing

The ARCHIVE folder is intended for (on-line) storing data and metadata after editing on micro level. Data from this folder can be directly accessed by different users for further processing (statistical-macro editing, imputation, analysis, tabulation, etc.). In a similar way, raw data are archived before entering the editing process.

The speed and intensity of development was limited by the need to support regular statistical »production«, organize training and obtain support for the new staff working with Blaise: application developers and application users. Especially for some developers who were used to work in a mainframe environment, the changes were not easy: first, they had to adapt to environment and development on LAN, and then to study new tools. As the main (but not the first) data editing pilot project, the application was developed for the monthly Industrial Production Survey, which is a typical representative of a large group of surveys where some additional functionality was required. Some characteristics:

* Many and complex checks on a relative small number of fields
* Many reference files (often including data from more than one previous period)
* In some cases, mixing of microediting and statistical editing
* »Dynamic« reference files (derived from the last version of edited data)
* Adding new (additional) batches of raw data during the editing process

The Industrial Production Survey was prepared and tested for editing in Blaise already about five years ago, with Blaise III. But the production really started in January 2003, with Blaise 4 and VB interfaces, when environment on LAN at SORS became capable to support more complex processes.

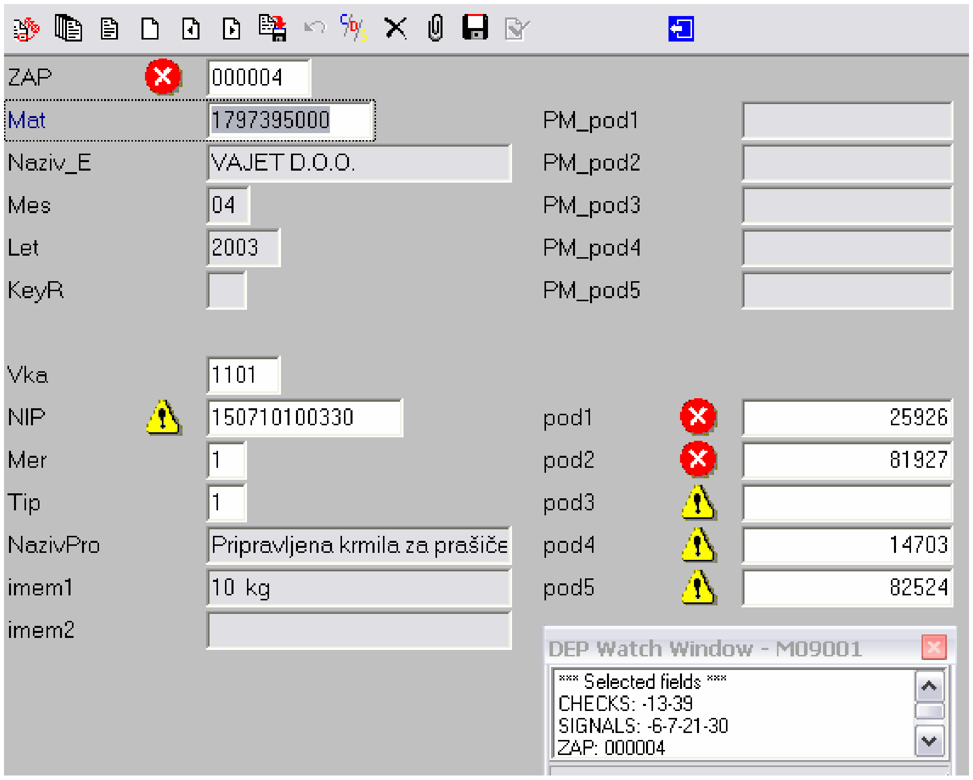
3. Error handling and reporting issues

The Blaise system is mostly and traditionally used to support surveys, where data collection and editing are integrated in the process of computer assisted interviewing (CAI). It is based on interactive work of an interviewer.

When SORS decided to use Blaise as the main LAN data editing system, one of the frequently asked questions was if Blaise can obtain all data editing functionality of the old mainframe system. If something already used suddenly becomes unavailable, the users will not be very happy. The mainframe GVS-2000 system, based on Rapid RDBMS, was a kind of reference, with the relatively good and automated coverage of complete survey data editing process. Improved error handling and reporting for post-collection data editing on LAN was one of the first and most important needs. Blaise itself is very much oriented to interactive work, so we added some solutions developed fot the needs of post-collection data editing:

- An array of Boolean values was added to each editing data model to save and handle each single error
- Each error in the survey editing application was numbered with its unique error code to optimize searching, maintenance and documentation, and to report data correctness at any moment
- A list of errors (checks and signals) is reported on screen constantly and independently of cursor position, using DEP watch window
- A number of standard batch templates were prepared, to generate and show different reports about errors

With error codes added (as an array of fields) into the survey datamodel, batch reports and analyses of errors in data are quite easy. Some more time was needed to find a good solution for error reporting in interactive mode. Here is an example of the Blaise screen with a list of errors in DEP watch window for survey M09001 – Industrial Production:



ZAP	✖	000004			
Mat		1797395000	PM_pod1		
Naziv_E		VAJET D.O.O.	PM_pod2		
Mes		04	PM_pod3		
Let		2003	PM_pod4		
KeyR			PM_pod5		
Vka		1101			
NIP	⚠	150710100330	pod1	✖	25926
Mer		1	pod2	✖	81927
Tip		1	pod3	⚠	
NazivPro		Pripravljena krmila za prašiče	pod4	⚠	14703
imem1		10 kg	pod5	⚠	82524
imem2					

DEP Watch Window - M09001

```
*** Selected fields ***
CHECKS: -13-39
SIGNALS: -6-7-21-30
ZAP: 000004
```

Figure 2: Data editing screen with list of errors

Values of auxfields CHECKS and SIGNALS (list of error codes) and field ZAP (primary key) in DEP watch window are calculated with every press of Enter, so a data analyst can see the effect of corrections immediately and at one place. There are also other settings and options (layout, hot keys, etc.) adapted to this mode of data editing.

With these additions, error reporting was already much closer to user's needs. Nevertheless, we still could not find an easy way to search the database for specific errors in an interactive way. Instead of this, a generic batch module was prepared. A user enters a selected error code into interface and Manipula setup produces a list of units with this error, and brings the report to the screen. Users adapted to it, but from the point of view of functionality it is not completely satisfying, and we are still looking for a simple interactive solution.

4. User interfaces

Generic user interfaces were found as one of the most important components of the new SORS data editing system. They are key for access and control of the complete editing process. They were designed in permanent cooperation between their developers and users. We followed the principles that the interface should be self-descriptive and as simple as possible for use and maintenance.

A special **interface for application developers** is usually not a part of the system, since developers are supposed to use tools directly from their environment. But in our case, when developers are still learning their new tools and new environment, the interface for development was found extremely helpful and useful. Through the menu system it supports:

- Direct access to development environment, based on parameters (code of survey, developer, etc.)
- Copying all templates and standard setups (modelib, depmenu, etc.) to survey development folder, to begin the new application
- Generated import and export of data
- Implementation - including the application into the production environment

Probably this interface will become obsolete (or less important) when developers become more familiar with the new environment. But it has proved to be a good start for new developers.

Different is the generic **interface for application users** (analysts-data editing staff) which is a normal part of the production system. The current version already covers almost all needs of users of about 40 surveys. Almost - because with few surveys there are still some very special processes, which can not be triggered by buttons in the interface. But at the end, with complete overview of task and functions in editing processes they will be included as well.

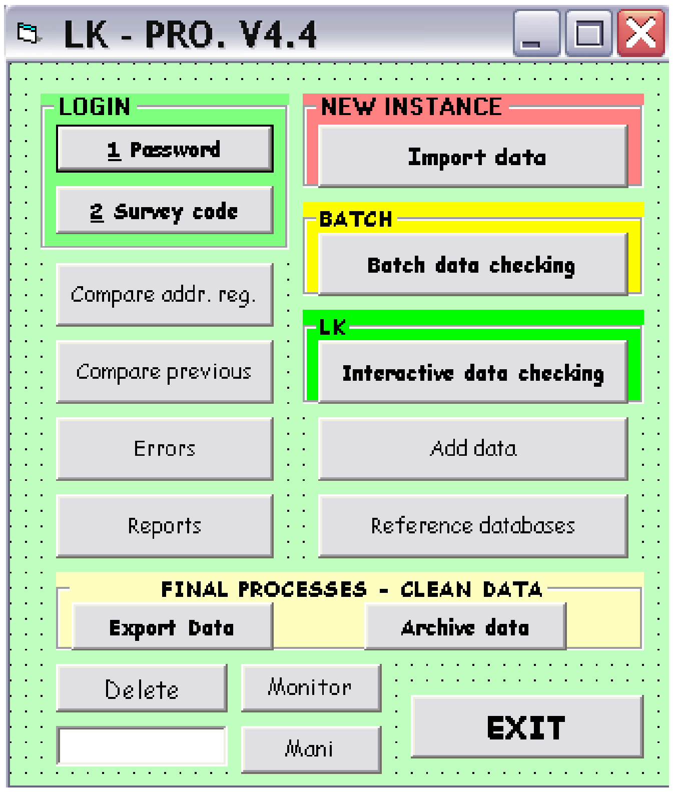


Figure 3: Data editing user interface

Access to survey applications is defined by parameters entered into the user interface: password, survey code, period (instance) of survey, names of reference databases, etc. The password also defines the level of user: survey administrator or data editor-analyst. Some buttons (e.g. import data, archiving) are active for administrator only. Other parameters (e.g. /g - get form; /P2 – data editing page layout) are always fixed. After the login procedure, the user begins with importing ASCII files (data, references, etc.) into Blaise, runs batch checking and then continues with interactive review and correction. Blaise (and other Windows) executables are called through the VB Shell function, using Blaise command line parameters. Example:

```
Shell (potDEP & potLK & "M" & SCode & " /g /p2 /t2 /Ydsn  /e /!" & potLK & "
/m" & potLK & "depmenu.bwm"), 2
```

Figure 4: Commands behind Interactive data editing button

Figure 4 shows commands behind the *Interactive data editing* button: call of survey data editing application in user environment. Parameters: *potDEP* is path to DEP, *potLK* is path to the survey data editing folder, and *SCode* is the numeric survey code. Get form mode, data editing page layout and data editing - dynamic checking (/g /p2 /t2) are default settings in the data editing system, and clean records are not brought on the screen (Ydsn). Depmenu and modelib files are designed for the common needs of data editing and can be adapted for the single survey if necessary.

In practice up to 20 users (data editors) work interactively on a survey database, editing their own part of data defined by key values. We tried to avoid mixing batch and interactive processes as much as possible, so batch checking is performed on complete database at the beginning, and later only when necessary.

It is very important that with this user interface we are able to cover different combinations of data entry and editing, manipulate and transform between different types of data models (record is form or record is row in a table: transformation is generated) and (only by entering different parameters) support data editing on different servers and locations.
This is enabled since the complete editing procedure is defined as a transferable module, which can be used at any place in the statistical information system.

5. Conclusions

The system for post collection data editing at SORS is successfully replacing the old mainframe data editing system. Error reporting and smooth access to the system, obtained by user interfaces, are important components of the integrity, and both were the result of in-house improvements, enabled by knowledge and experience from the past years. I have to say that some hints from our Blaise colleagues from other countries were also very helpful.
At the moment most important functions are covered and new surveys are “added” mostly without the need for any changes in the system. This functionality allows us to choose between modes of data editing in some surveys: integrated or separately, after data capture. This also means better flexibility and better availability of human resources: a job can be spread over different data entry and editing departments. However, what is most important is that the system succeeded to make the first steps to establish common in-house standards of development and usage of data editing applications.
Parallel with this, we try to build a similar system for CAI surveys, which is not an easy job, since it is hard to put so different surveys in a shell of standards and conventions. And with both systems, the question of efficient and automated use of metadata is still opened.

6. References

- Edgar, M., and Turner, M. J., Software Quality Framework, Meeting on the Management of Statistical Information Technology, Geneva, 17-19 February 2003
- Katic, M., Klanjscek, M., and Kozjek, P., Management and Monitoring of the Statistical Process and Impact on Data Quality, Meeting on the Management of Statistical Information Technology, Geneva, 17-19 February 2003
- Kozjek, P., Blaise Generator for High Speed Data Entry Applications, Proceedings of the 6th International Blaise Users' Conference, Cork, 2000
- Kozjek, P., Blaise for Post-collection Data Editing - Building general data editing system based on Blaise, Proceedings of the 8th International Blaise Users' Conference, Copenhagen, 2003
- Pierzschala, M., and Manners, T., Revolutionary Paradigms of Blaise, Proceedings of the 7th Blaise User’s Conference, Washington D.C., 2001
- SORS and Statistics Sweden, Feasibility study on the architecture of information systems and related equipment issues, Study implementation and Hardware/Software Specification for Tendering, September 1997
- Sundgren, B., An Information Systems Architecture for National and International Statistical Organizations, CES/AC.71/1999/4, Meeting on the Management of Statistical