Blaise for Post-collection Data Editing

Building general data editing system based on Blaise

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

1. Introduction

The Blaise system is mostly and traditionally used to support surveys, where data collection and editing are integrated in the process called computer assisted interviewing (CAI). Since the system is also able to support other modes of survey processing, SORS is trying to extend its usage to support administrative data editing in general. It means that data editing system based on Blaise should be ready to receive and process any data coming from the input, regardless of the data collection mode.

Compared to CAI surveys, processing of data collected on paper forms is usually organized in a different way. A typical process begins with high speed data entry as the first phase, and data editing (reviewing-checking and correction) as the second phase. This kind of 'separate' data editing requires a different approach when preparing survey applications. Some characteristics:

- more batch processes
- different administration and user interfaces
- sources of additional information for data editing not always available
- different (data editing) screen layouts
- needs for (many) generated reports, etc

In the past, different ad-hoc data editing applications were developed at SORS to cover the needs of individual surveys. But for continuous and efficient processing of a large number of different surveys a standard and generalized solution is necessary, with some of the key requirements:

- automation, efficiency and reliability
- integration into complete statistical process
- acceptability for survey methodologists, application developers and application users
- complete LAN infrastructure support (system and user administration, archiving etc)
- openness for upgrading
- long-term support

During the development of the new data editing system we tried to use best practices to fulfil these requirements as much as possible. Knowledge about existing data editing methods and tools was used as a basis for the new solutions.

2. Background and starting points

Statistical Office of the Republic of Slovenia (SORS) has been using Blaise for data entry and editing of CAI surveys for almost ten years. Since 2001 Blaise also supports all high speed data entry from paper forms, making use of GEntry, in-house developed generator of data entry applications, and using Blaise also for

editing of these data seems to be a natural next step in the process: at last (but not least) we don't need to bring another software (and license) into the office.

Data editing at SORS was traditionally supported by different tools and applications (Cobol, Pl-1, Godar) running on a mainframe platform. Many of them are still in production, organized as circulating processes, with printed error messages and corrections re-entered in each cycle. Other solutions (Godar) are more efficient, based on relational technology (Rapid), combining batch and interactive work. But there's a big problem about their support in the future.

The new system is bringing many changes to survey editing process: developers and users are migrating to a new platform (LAN with Windows XP as the operating system) and they are using the new software. And often, the survey contents and methodology are using the 'opportunity' to change when migrating to a new environment.

Our previous experience with GEntry was good and helpful from many aspects, and Blaise with VB interfaces was proven as a good combination to build systems, robust and easy to maintain. We learned how to:

- build efficient systems on LAN
- enforce standard solutions
- design simple and friendly user interfaces
- educate users
- organize maintenance of applications etc

But the data editing process is more complex: it requires much more interaction with 'environment': address registers, classifications, other reference files. There are needs for different formats of input and output data and metadata, there's a large variety of complexity of applications, needs of different users etc. There were also some doubts at the office: is Blaise, known as a CAI tool, capable to support large traditional 'paper form' surveys?

3. Preconditions for development of a new data editing system

The idea about the data editing system was to define it as a relatively independent module which can receive data (and as much as possible metadata) in Blaise or different ASCII formats, and produce clean (at administrative level) Blaise or ASCII data with basic technical metadata for further processing

Training of new developers and users was one of the first issues. For new and inexperienced developers of Blaise applications it was crucial to have standardized and user friendly environment for developing and testing applications. And for users of applications we need to prepare clear and easy access to all functions necessary for operational work – data editing. User interfaces - shell around both environments - were built by Visual Basic. Similar to Gentry, templates and automated generation of code were used wherever possible, to standardize work in development and production environment.

One of the often-asked questions was: can Blaise obtain all data editing functionalities of the old mainframe system? If something already used suddenly becomes unavailable, the users won't be very satisfied. The mainframe Godar system, based on Rapid RDBMS was a kind of reference, with the relatively automated covering of complete survey data editing process. Needs were identified, and some additional solutions were developed and implemented:

- a survey administration system was developed, supporting user access to editing system. Access is based on parameters (password, survey code and period...) and user rights (Windows XP/NT user groups).
- each error in the survey editing application is numbered with its own error code, to optimize searching and to report data correctness
- set of standard fields was added to each editing data model:
- array of Boolean values to handle errors
- alternative key (statistical ID)
- some status fields about survey process and contents
- a number of standard batch templates were prepared: ASCII to Blaise and vice-versa, restructuring data (record to form and vice-versa), comparing between data models, importing and exporting data etc.

With these additions we were ready to prepare and test applications in the new system. Developer's and user's environment were defined as separate shares on the data editing server, and survey applications were located in sub-folders.

4. The system - developer's perspective

As mentioned before, most of developers had no previous experience with Blaise or development on a network environment. So training was necessary as the first step. A short developer's manual was prepared, and three-day in-house course for developers was organized in February 2002. Contents of the manual and the course were limited to the needs of data editing, and practical work on examples was emphasized. It would be wise to continue with the real work immediately, but it took another six months before the new data editing system became operational.

It was not surprising that migration to a new platform was the main problem for most of application developers. Blaise itself was well accepted: syntax is not much different from other programming languages, and some key features of Blaise (modularity, re-use of code etc.) were immediately visible in practice. It is very encouraging for developer when the extent of application code is reduced up to ten times… Templates (for data models and Manipula setups) were designed to guide developers and to support some in-house conventions about developing applications.

To help beginners on a LAN environment to concentrate on applications, a developer's user interface was prepared. It supports:

- direct access to development environment, based on parameters (code of survey, developer etc.)
- copying all templates and standard setups (modelib, depmenu…) to survey development folder, to begin the new application
- import and export data
- implementation - sending complete application to production environment

Perhaps this interface will become obsolete (or less important) when developers became familiar with the new environment. But it is proved as a good start for new developers.

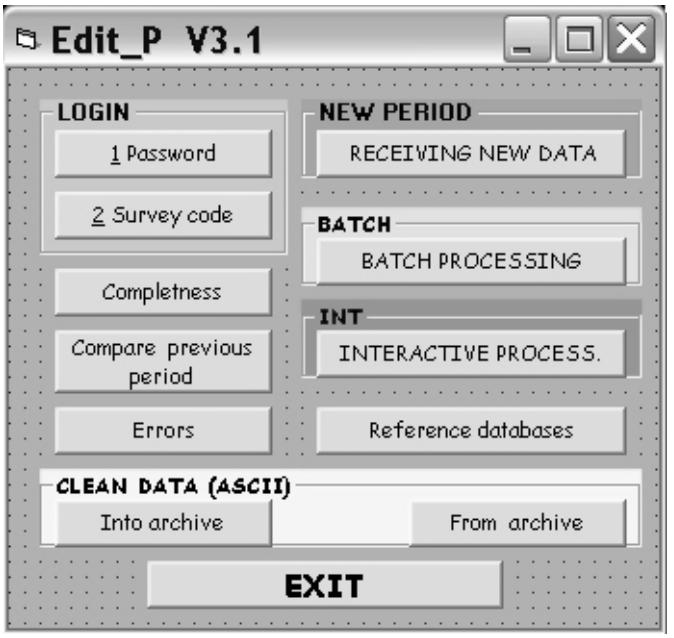
One of the main questions was how to design data models in the editing system: form-based or record-based. Old mainframe applications were record-based, but we decided to use (with exceptions allowed) form as object of observation: it makes editing much more effective and organized, and it's also better supported by Blaise. But our high speed data entry is record based as well, so we need to convert data first. This was solved by Manipula template record to form; another template

(form to record) is applied where record structure is needed for further processing. Despite templates, restructuring is time consuming and error-prone, and in the future we will try to tend towards single (form based) data model in the process of survey data entry and editing.

5. The system - user's perspective

Data editing user environment is physically defined as a share on the data editing server, and survey applications are located in sub-folders. Access to applications is enabled through general VB user interface. A short manual was prepared to explain users how to use interface and how to edit data, and a course was organised before the first application was implemented.

Figure 1: Data editing user interface



Access to survey applications is defined by parameters entered into the user interface: password, survey code, period of survey, names of reference databases etc. The password also defines the level of user: survey administrator or data editor. Some buttons (e.g. receiving 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 receiving ASCII files (data, references etc) into Blaise, runs batch checking and then goes to interactive review and correction. Blaise (and other Windows) executables are called through VB Shell function, using Blaise command line parameters. Example:

Figure 2: Interactive editing

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

Figure 2 shows commands behind *Interactive editing* button: call of survey data editing application in user environment, where *potDEP* is path to DEP, *potLK* is path to survey data editing folder, and *SCode* is 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. With some surveys, batch checking process is relatively slow and we need to discover reasons: hardware, applications (performance issues), network processes etc. But this does not seem to be a critical point of the system.

6. The system - statistician's perspective

In the old system some statisticians were lacking insight into clean data after data editing. Now the access is no longer a problem any more: user just need to be added into the correct group of users by Windows XP system administrator. Another question is if they really need to access Blaise databases: in most cases a generated ASCII (CSV) file would be quite enough to make a quick overview of final data in Excel (or some other tool, where a short macro analysis is possible).

Other data formats (e.g. ASCII-relational) and various generated reports are also available now for each survey. The contents can be adapted to the user's needs.

7. Implementation

The first survey was implemented into the new system in June 2002. Real work - development and implementation - started in September and October. New Blaise users and developers were supported by experienced colleagues when developing their first applications. Templates and user interfaces were (and still are) developed parallel with applications, considering user needs. Periodically they are replaced with the newer versions.

By the end of March 2003, 16 survey applications were implemented into the new system, including some comprehensive monthly surveys (Monthly Industrial Report, Report on Construction Activities, Report on Purchase of Agricultural Products etc.). No serious problems were perceived and users accepted them well. They are satisfied with better overview of the data editing process (forms instead of records), screen designs, accessibility to applications and flexibility of work (users - data editors – are no longer 'fixed' to their physical part of data). There are some comments on slow batch checking processes and we are working on solution of the problem. From developer's side, a total control and access to the user environment is an excellent way to improve applications, remove eventual errors and help users when necessary. Step by step we are also making improvements on organizational aspects of survey data processing.

Although the system is new (with parts still under development), there are already many lessons learned from the experience. The feedback form developers and users is in general positive. Successful implementation means:

- a step forward in building and implementing standards for application developers (modularity, re-use of components, common templates, using metadata) and application users (common interface, editing screens, commands, etc.)
- automation of processes
- efficient and uncomplicated administration: overview and control of processes, data flow and users involved
- intuitive usability

- ability to include (and combine) multiple modes of data entry and editing
- increased production efficiency – concentration on quality
- easier maintenance

Administration of some processes (e.g. preparing everything necessary to start new survey period) is not yet completely automated, and with complex surveys some auxiliary tasks will probably remain 'manual'. For such cases good documentation of complete procedure is even more important.

8. Conclusions

About half a year after implementation it seems that the main objective is reached: the new system successfully supports data editing of surveys that were traditionally running on a mainframe, supported by various tools. The system also covers CADI surveys: integrated data entry and editing from paper forms, and - probably most important: it brings common in-house standards of development and usage of data editing applications. Some advantages of new approach are already visible now; the others are expected later, when users and developers get more experience. The system can be considered as a part (or module) of general system for data collection and editing at SORS.

With successful start we should not consider the job as finished: new users and developers will need a lot of help and support, and it will take some time and efforts until all the components of the system will be completely harmonized. We should also analyse and find the way of connection between editing system and Metis - SORS metadata base, which is under development.

We believe that lessons learned with the data editing system will also help us to approach to some other important issues and challenges of data collection at SORS. One is building an effective system for administration and automation of survey processes with special requirements - most complex and demanding surveys (e.g. Labour Force Survey, Monthly report on Earnings, etc.) where 'all possible' modes of data collection (including internet) should be enabled and combined. New features of Blaise (version 4.5 or higher) should probably be included to get the proper results.

9. 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

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 Information Technology, Geneva, 15-17 February 1999