Entry data validation in citizen oriented applications

Ion IVAN, Bucharest University of Economics
Cristian CIUREA, Bucharest University of Economics

Abstract. This paper defines the concept of citizen oriented computer science. It presents the characteristics of applications built for the citizens and defines the procedures for entry data building. It determines the types of entry data in citizen oriented software applications and procedures for their validation. It analyzes the effects developed by the efficient validation procedures.
Keywords: computer science for citizen, software applications, data validation, validation procedures.

1. CITIZEN ORIENTED COMPUTER SCIENCE

The knowledge-based society suppose the growth at a very high level of human-computer interactions in order to solve the problems of citizens, such as:

- tax payments, to the state budget and local budgets;
- freight payments, ordered over the Internet or not;
- obtaining approvals;
- carrying out knowledge tests;
- training using e-learning software;
- book tickets and acquisitions;
- appointments in the audience;
- commands clothes, launched on the Internet;
- medical tests, to establish the group of blood or health status.

The researches on the development of information society are based on the results obtained in the following areas:

- building hardware characterized by speed processing really big, memory capacity, too, and a huge variety of ways to purchase data that eliminates any restriction related to the loading of databases, in terms of quantity, but particularly in terms of quality;
- developing software by the existence of tools that allow the design and development of program products very complex by highly skilled specialists, who master the requirements of work as a team, in the industrial flows;
- a shift towards the use of computer science in order to solve the problems completely, either directly by the users, because the program products work independently by the developers, providing complete and accurate results for entry data, without requiring assistance elements of specialty; the access to all resources is on-line;
- information security at the level of users protection, the level of databases, software and the performance level processes;
- changing the concept of data collection, processing large volumes of data and models for defining which aims to build optimization algorithms;
- integration of databases to manage redundancy applications generating continue processing flows; if a new software application, citizen-oriented, requires elements already existing in the databases currently in use, it will

be referenced those databases in compulsory, the integration of databases being condition of existence of the application.

The purpose of citizen-oriented computer science is to develop very different applications which close the cycle with a specific action. In this context, lending a book from a library is done in two ways, as follows:

- the reader enter the library site, they create a user account, consult the list of available books, select a specific book that will be delivered at home;
- the reader go to the library, access the database with available books, select the book that interest him, and this book is made available automatically through a hole.

For reservation and purchase of plane tickets, the steps forward are as follows: enter the site, choose and pay tickets online and, ultimately, obtain an electronic confirmation with the same value as a ticket purchased from a Agency.

Obtaining a particular type of permit is made as follows: access the site and select the category requested authorization, the payment will be made online, enter the cadastral database, analyzes the process in electronic way and receives authorization.

Banks made available to customers a service by which they conduct electronic confirmation, automatically, of the customer payments for certain customs offices in order to obtain the free of customs. For the electronic payments made by the client to customs offices, the bank deliver an electronic file, which is for the customs offices, the proof of the debt end by the customer. In this way, there is no need for submission by the customer, a copy of the green payment order on paper.

The advantages of such applications are as follows:

- the software applications work independently of developer and owner;
- the software applications do not produce mistakes;
- people who use these applications work with them without effort and without documentation in advance;
- the software applications use databases that interact; database with the country's population interacts with the database of educational diplomas and with the businesses database;
- transparency is very high, because are made available to the user all the information it needs.

The knowledge-based society implies:

- the existence of compatible databases, independently realized, for subsets of collectivities, which by concatenation without essential changes, allow to making processes relating to the whole community; in the case where shall develop new applications that will be subject to aggregation processes, it would be defined templates and structures which all developers must observe;
- defining the mechanisms and vocabulary for building the acronims from strings made with the words in vocabulary, achieving it in this way, a very effective way of building keys search of information required for solving the problems of the citizens; the words list contains names of countries, names of cities; to this list is added words corresponding professions, very

common objects, common actions and types of commonly used locations (hospital, station, train, school, museum, town hall, police, weather, map);

- building rules for the design of software interfaces, so that all applications that require the same entry data structures or who offers the same digital content structures to implement interfaces that do not differ significantly by keywords, by their position and, especially, the size of texts used; shall ensure, in this way, the continuity in all software applications; a citizen familiar with a particular application, commonly referred to, will carry out without any effort the pass to a new application, if the new application take in very large proportion the interface content from the old application;
- the orientation to natural requirements, formulated by users, of the whole stream processing; if the preinformation society contained flows defined by organizations, and the citizen respect the rules which benefit exclusively the team from the organisation, the new context created comes to radically alter the position of the citizen, what makes the structuring applications to be made strictly on the requirements of citizens; the levels description of the characteristics regarding the citizen are dispersed in several databases; any application, which covers the characteristics of existing levels in these databases, is bound to work with existing databases, which means a radical change of the databases owners behaviour, as well as databases users; the structures for the databases records regarding collectivities become public, and the creators of new citizen-oriented applications select, depending on the requirements of the processing algorithms, databases necessary, pay the access to these and eliminate the data reinput;
- harmonizing the interests of those who produce software with the requirements, as various, of the development of the citizen-oriented applications, to cover the vast diversity of requirements for settling the problems of citizens; now there are many applications for enterprise accounting, for the calculation of salaries, for management of stocks, for contracts and for supplies; are not covered applications by interest on training on-line, access to resources, temporary allocation of resources, the information on fields, making behavior calculations, conduct transactions; dispersed, with great difficulties retrieval, with partial approaches, there are partial solutions to some of these problems; the citizen, remained on a second plan, has solutions that not take into account the requirements by the fact that the sites names were related to poor content, the acronims used are based on unknown mechanisms for achieving and that is not covered throughout the whole collectivity; all unsuccessfully references made by citizens accelerates the decrease of their satisfaction degree in front of software applications; the harmonization imply the creation of programs based on projects, so that funding must be managed strictly to the creation and implementation of citizen-oriented applications, covering the areas for which there are no solutions in the computer science field.

The specialty literature contains papers, books for software engineering like [GAUS04] and [PRES88], books for quality management software, such as [WEIN07], which develops, in the theoretically and practically plan, the main requirements to design, realize and implement performant software applications. Are

unilateral approaches, enabling the allocation and the reequipment of resources strictly from the viewpoint of those who build applications.

2. THE CHARACTERISTICS OF THE CITIZEN ORIENTED SOFTWARE APPLICATIONS

They differ from the others by the fact that are developed starting from the target group. The other applications started from the owner requirements. If a bank wants to develop a specific software application, then the application will be built on the basis of specifications developed by the bank. The new generation of software applications are built on the basis of the wishes of citizens. Are defined specifications that satisfy the citizen, the user, but not the owner.

The cycle of development of an citizen-oriented application is as follows:

- the owner and the developer defines the target group of the application;
- are determined the objectives of the target group, in order to define the problem to solve;
- are developed specifications in order to meet the wishes of the target group;
- is writen code for the application development;
- is tested the application by the target group.

To the distributed software applications design is analyzed the target group, whereas the quality characteristics of those depend by the people who access defined resources, and which, by the satisfaction degree obtained after settling on-line problems, determine the further development of other applications.

The citizen-oriented software applications have the following features as:

- the software complexity is a feature highlighted only by comparison between two or more software programs; a system may be viewed as a graph in which nodes are represented by subsystems, and directed edges by the links between subsystems; the McCabbe software complexity involves associating each programme a graph in which nodes correspond to the instructions, and directed edges meaning marks the execution transition from an instruction to the other;
- reliability, which lies in the ability of applications to maintain their level of performance in certain situations for a specified period of time; reliability is considered as confidence in the design and the ability of a software product to function properly in all the conditions envisaged in mind from the begining;
- friendly interfaces, involving easy navigations in the application menus and which do not require documentation of use.

The stages of development cycle should include:

- the analysis of the target group, taking into account the estimated number of people who access the application resources, the structure of sex, age, level of qualifications of people who make up the target group; will consider the abilities which influence the people to address the access to the resources of the software application; all of these allow the development of a comprehensive study on the behaviour of the target

group subdivisions in different situations; are estimated the representative sample size and are scheduled all confirmation operations of the application behavior reported at the sample members accessing of the application resources;

- defining the problem to solve is made from actual requirements formulated by members of the target group; the beneficiary of the application needs some final results; users citizens have the capacity to provide certain entry data; if there are differences between the input data provided by the citizens, naturally, and entry data required for results obtaining they need beneficiaries of the software application, then the new concept of software engineering, citizen-oriented, requires that the recipient organization to pay the application access to existing databases to fill the information deficit; when things are reversed, the application beneficiary has a responsibility to solve the problem without transferring at the citizen's level his information minuses, turning it the citizen on a element dependent by the organization; the applications developer are passive role in this stage, recording all the requirements of the target group, requiring to the application recipient to solve access problems to other databases in order to get complete and correct input information of the problem to solve; the passive role move forward, namely the idea that the developer redirect the entire set of techniques, methods, language, personnel, equipment, data sets test to resolve the requirements of the citizen, but not to satisfy the beneficiary or, worse, get him profit at the expense of the beneficiary, through the transfer of negative effects, generated by an inadequate solution, to the many citizens who turns on the inadequate resources of the application, repeatedly;
- the specifications developing reflects, first, the requirements of the target group members; by using a rigorous language proceed to the description of problem to solve that problem of the target group members;
- the application design is based on the citizens requirements to carry out a communication, naturally, and to develop processing streams as short; it examines how the target group members want to resolve problems, without they impose the restrictions of the recipient organization or technical restrictions specific to the computer science field; the application structure should reflect the degree of flexibility that see the members of the target group; starting from the requirements of the target group, from the interfaces designing, from the used vocabularies, from the processing modules, from the results presentation modules, is done so that all the processes to coincide with how people want to result the human-computer interaction; they should notice the ease and naturalness with which it comes to resources, so that the success rate of interaction is possible;
- the encoding is done while respecting the quality levels required for the procedure, including the vocabularies accepted by the citizens, generating strings very close to natural language to facilitate understanding of the significance of results; assisting the coding process is intended to take over the code all the existing information in the specifications in order to achieve a best possible overlap of what is doing every written module in way with what the citizens want;
- the modules testing and the testing of the entire citizen-oriented application is made using data test created on the basis of information from

members of the target group; the application beneficiary and application developers create sets of additional test data that addresses the technical side of the application and which, in comparison with sets of specifications initiated by citizens bring a global plus quality to the application, without being inconsistent with the requirements of citizens.

3. ENTRY DATA IN CITIZEN ORIENTED SOFTWARE APPLICATIONS

Crossing the stages defined by technology guarantee obtaining citizen-oriented applications, characterized by efficiency, operationality and, especially, by maximizing the satisfaction degree of the target group members.

The citizen-oriented applications are available on the Internet or into a network of computers used by the target group. These applications have a number of forms which allow the introduction of data and the selection of some values or options. In the selection of values or options, the list of selected items must be complete, that is to cover all the existing values for the selected field. For example, selecting the type of credit card is made from the following list: Visa, Mastercard, Maestro. There must be no situation in which a user hold one card other than those mentioned in the list of selected items.

The holder of a credit card is completed on a form through two fields, one in which introduces the last name, and the other which introduces the first name of the person. The values of the two fields are not comutative. Validation of these fields must include the introduction of three or more consecutive identical letters, in which case that name or surname is considered invalid.

Validation fields in which introduced free text, such as field for comments, involves the definition of a vocabulary of prohibited words. The text entered must not contain words or substrings from that vocabulary. Thus, it avoids filling obscene words in fields that contain free text.

A symposium contains sections themes, each section having a name and a list of keywords. Validation classification of an article in a given section requires that the title of the article contains words from the list of keywords such sections. In the field on the form for the abstract completion of that article shall be represented as a limit number of characters or number of words completed. It achieved thus validating the length of the entered text.

The forms are required and optional fields. The entry data are represented by strings of letters, calendar data, codes or numeric values. These data, with the completion of their, falling into the following categories:

- are correct and complete: on a form is $C_1, C_2, ..., C_n$ fields to fill; but each C_i field has a V_i field of values; the situation of correctly and completely appear when all C_i, $i=1..n$, fields belong from V_i, $i=1..n$ vocabularies;
- is correct, but incomplete: in which case all $C_1, C_2, ...,C_{i-1}, C_{i+1}, ..., C_n$ fields belong from the V_i, $i=1..n$, vocabularies, but is missing the C_i field; in this case, the application generates messages that indicate what fields were not completed on the form, as in Figure 1:

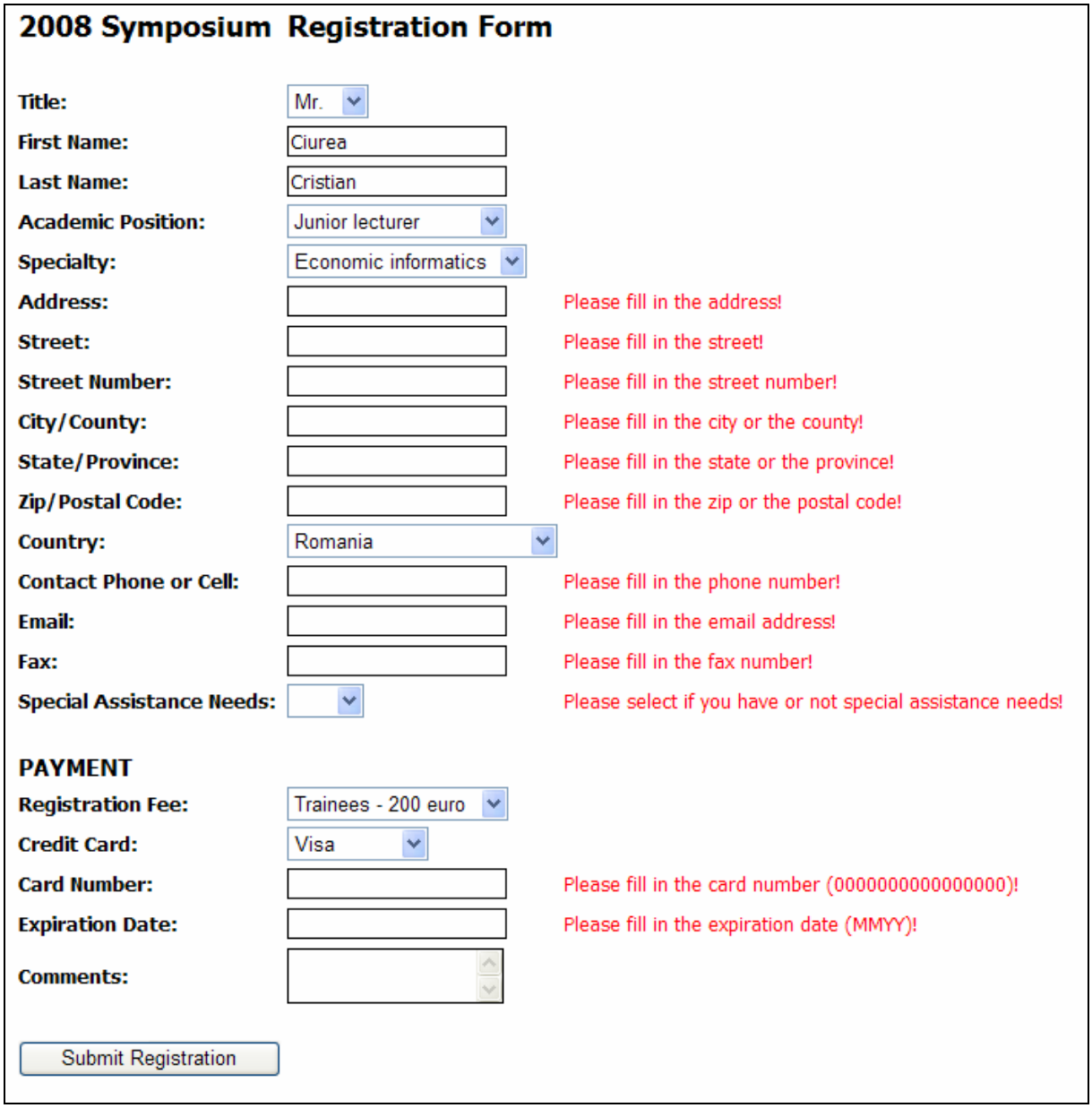

Fig. 1. Incomplete entry data

- are complete, but incorrect: a situation in which all the fields are filled, but one or more fields do not belong the domain values; in this case, at every field is specified what is wrong filled, as in Figure 2:

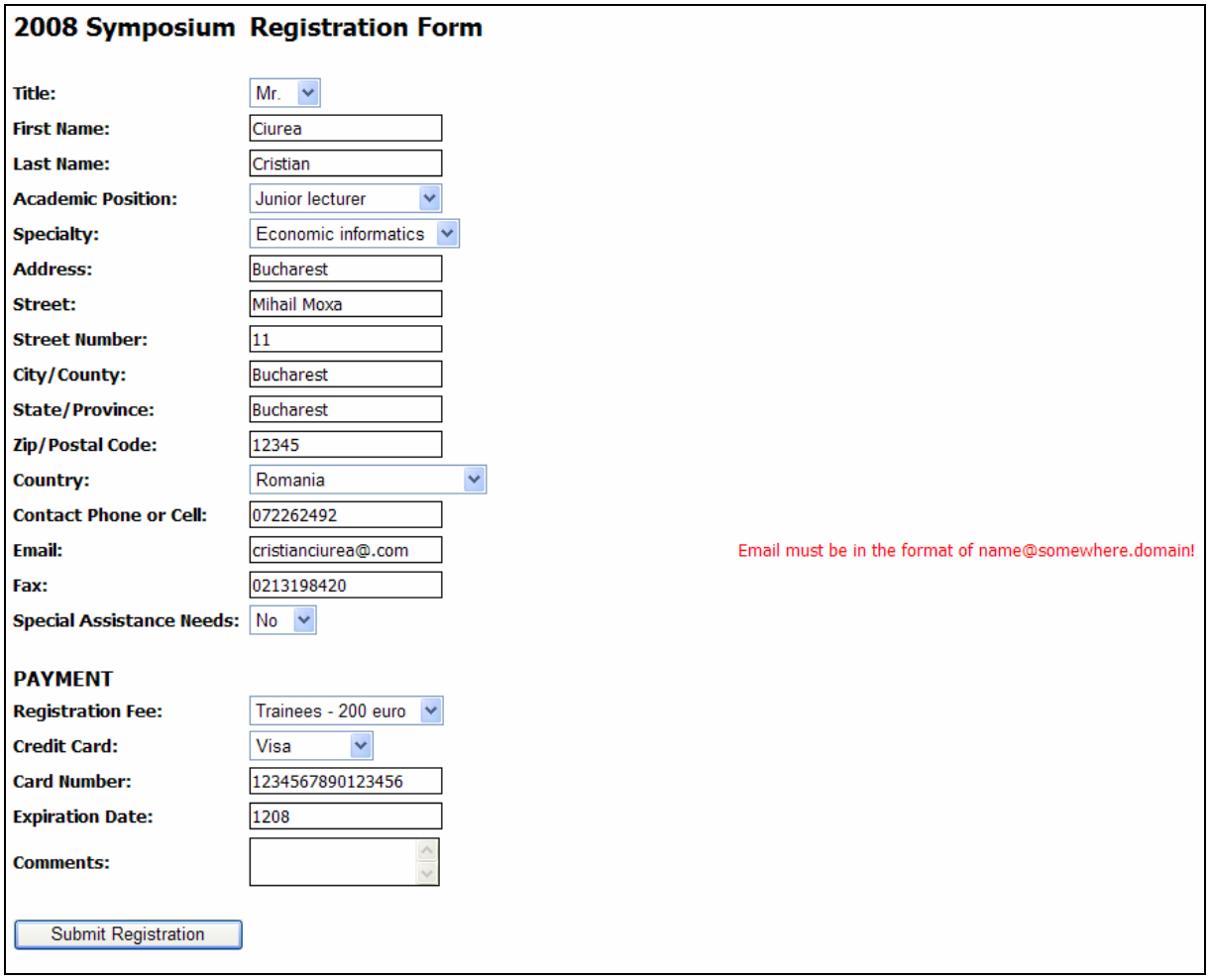

Fig. 2. Incorrect entry data

At every test data recorder, if they were not placed correctly and in full, appears a list of errors and the data are not transmitted to the database.

The introduction of some fields into the form and the possibility of transmitting data, even if they are not complete, there must not be in a citizen-oriented software application.

4. VALIDATION PROCEDURES

Procedures are builded for validating data: alphabetic, numeric and correlations between fields. For each application are used standard procedures and the error messages should be very nuanced so as to help troubleshooting a breeze. These error messages must show to the user where and what is wrong. Such applications, in which is communicate exactly what the user has entered wrong, are easily implemented in a situation in which fields contain complete verifiable information that structure, such as the personal identification number, telephone number, email address, date calendar or bank card number. In the case in which the user has to complete free information, then the accuracy of error messages is more difficult to implement.

The presentation manner of the results of validation are as follows:

- displays a confusing message which specifies that are errors in data;

- on a different form than they have completed the data shows a list of errors stating the field and the nature of the error;
- the form in which data is completed, at the right of the fields wrong filled appears the error messages, colored in red.

The process of recording data in the database should contain stages of Figure 3:

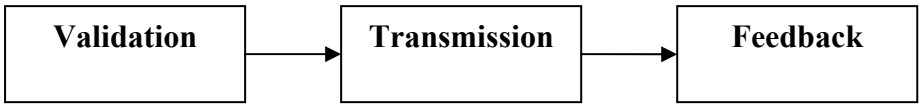

Fig. 3. The stages of recording data process

The validation stage ensure the proper completion of information to be recorded and processed, eliminating the aberrant data or inconsistent with reality. The transmission stage assumes the achievement connection with the database and the actual recording of values in the database. The receiving feedback stage assumes the reception by the user of a complete response regarding the finality for its action, if the payment was made successfully, or if requested document was issued.

In the case of an Internet banking applications, after the user completes the data required to make a payment, when pressing the button "Approve", the payment status shall be transformed into "Approved by client". At an interval of several seconds, when the payment enter in the processing phase, its status changes to "Received by the bank". When the payment was made successfully, its status becomes "Approved by the bank", confirming to the user the finality of his action.

The data validation is classified in two types: simple validation and complex validation. The simple validation is useful for simple testing of a single isolated control located on a form. In most cases, it validates more controls placed on the same form and whose contained information is intercorelate. In this situation, it is about a cross or complex validation. An example of cross-validation is shown in Figure 4 and refers the correlation between the average obtained by a student at a particular school exam with his school situation:

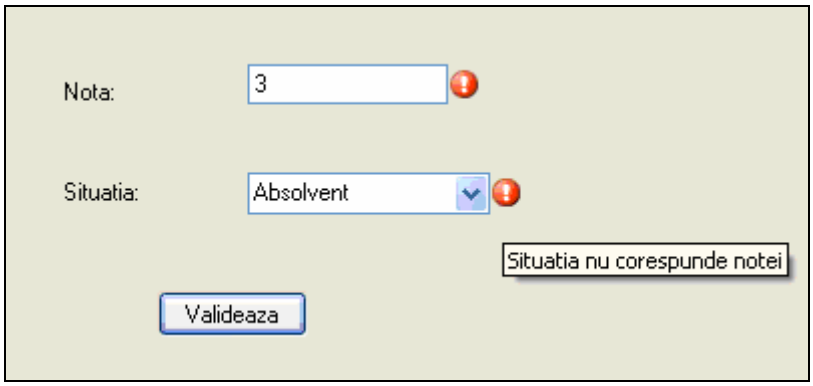

Fig. 4. Example of cross-validation

The C# source code, to simple validate the average in the interval [1, 10] and for the cross-validation of correlation between the average obtained with the school situation is as follows:

RETURN

```
private void button1_Click(object sender, EventArgs e)
{
int nota = Convert.ToInt32(textBox1.Text);
if (nota < 1 || nota > 10)
errorProvider1.SetError(textBox1,"Average outside the range 1-10");
else
if (nota<5 && comboBox1.Text=="Graduate")
{
errorProvider1.SetError(textBox1,"The average not correspond with the
school situation");
errorProvider1.SetError(comboBox1,"The school situation not
correspond with the average");
}
else
if (nota>=5 && comboBox1.Text=="Restant")
{
errorProvider1.SetError(textBox1," The average not correspond with
the school situation ");
errorProvider1.SetError(comboBox1," The school situation not
correspond with the average ");
}
else
{
errorProvider1.SetError(textBox1,"");
errorProvider1.SetError(comboBox1,"");
}
}
```

In the C libraries are offering a series of functions to validate a character introduced by the keyboard, such as *isalpha*, *isalnum* and *isdigit* used as follows:

```
char c;
scanf( "%c", &c );
if( isalpha(c) ) printf( "You entered a letter of the alphabet\n" );
if( isalnum(c) ) printf( "You entered the alphanumeric character
%c\n", c);   if( isdigit(c) ) printf( "You entered the digit %c\n", c
);
```

Validation of an e-mail is done in the C#.NET language through the following source code:

```
protected void Button1_Click(object sender, EventArgs e)
{
bool ok = true;
string[] splitemail = (tb_email.Text).Split(new Char[] { '@' });
int ce = 0;
foreach (string s in splitemail) if (s.Trim() != "") ce++;
if (ce != 2)
{
v_email.Text = "The e-mail address is not valid!"; ok = false;
}
}
```

The Visual Studio has implemented for ASP.NET a series of controls to facilitate writing source code for validation of fields whose format is the default entry. One of these is called `RegularExpressionValidator`, which enables the validation of e-

RETURN

mail address, telephone numbers, internet address, postal code and the social security number. The validation of the e-mail address with this control is done by following ASP.NET source code:

```
<asp:RegularExpressionValidator ID="v1_email" runat="server"
ControlToValidate="tb_email" ValidationExpression =
"\w+\x40{1}\w{2,}\.{1}\w{2,}" ErrorMessage = "Email must be in the
format of name@somewhere.domain!" Display = "Dynamic">
</asp:RegularExpressionValidator>
```

Instead of `RegularExpressionValidator`, the Visual Studio environment contains for ASP.NET and other controls for entry data validation, namely: `RequiredFieldValidator`, `RangeValidator`, `CompareValidator` and `CustomValidator`.

The expiration date of bank card is inserted in the form of a number composed of four digits, the first two representing the month, and the last two the expiration year. The form field in which introduced these four figures must contain a validation that verifies whether the first two digits represents a number less than or equal to 12, and the last two figures represent a number greater with no more than two units than the current year. Overall, it validates that the number consisting of four digits to represent a calendar date greater than the current date. Source code in C#.NET which implement this validation is:

```
protected void Button1_Click(object sender, EventArgs e)
{
string month = tb_expiration.Text.Substring(0, 2);
string year = tb_expiration.Text.Substring(2, 2);
if (Convert.ToInt32(month) > 12) Label2.Text = "The month must be not
greater than 12!";
else if (Convert.ToInt32(year) >
(Convert.ToInt32(Convert.ToString(DateTime.Now.Year).Substring(2, 2))
+ 2)) Label2.Text = "The card is valid only 2 years!";
else if ((Convert.ToInt32(year) <=
Convert.ToInt32(Convert.ToString(DateTime.Now.Year).Substring(2, 2)))
&& (Convert.ToInt32(month) <= DateTime.Now.Month)) Label2.Text = "The
card has already expired!";
}
```

There are more options for the data validation, namely:

- *option 1:* rigid validation, meaning that the data entered are validated field with the field and the user can not move forward until it completes correctly previous fields. This validation is accompanied by a reminder regarding the nature of the error in the form, like a text message or a sound signal;
- *option 2:* flexible validation on the form, displaying messages erroneous data;
- *option 3:* the form fields erroneous become red;
- *option 4:* on the form, after sending the data appear a list of errors.

Any of the four variants validation would apply, the application must solve a single problem: the information recorded in the database must be accurate and complete.

RETURN

5. THE EFFICIENCE OF THE INTERACTION

From issue to issue, from user requirements and from the financial resources of the owner's application are defined:

- the duration of making the application, defined as the period between the stage of designing the specifications until the end stage of testing and launching the product on the market;
- the complexity level, measured in the components number of the application, connections to databases and interaction with the users;
- the quality level, resulting after the tests realized and appreciations given by the users;
- the diversity of available resources, represented by the options that the application offer to the users;
- the degree of solution coverage, measured as the difference between what was desirable to be implemented and what was actually achieved;
- the connection with other applications, which shows the integrability degree.

The efficiency of the interaction between citizen-oriented applications and their users are measured with some indicators such as:

- the necessary time for data introduction in a form: is measured in seconds and it is measured starting the selection or completion of the first field on the form by pressing the button for data transmission;
- the number of data reinput, because of errors: every press of the button for data transmission, a counter is automatically incremented;
- the costs of development and use.

An application is efficient if the user ends with the data reinput in a few iterations and in a very short time.

Research over is intended to radically change the approach to software development cycle, because the knowledge based society involves the citizens cultivation of who interact with applications for settlement of as many and various problems of daily life.

In the meantime, the menus structure of the citizen-oriented applications will be stable, and therefore, when building a new application:

- is taken a stable structure of the menu;
- is made all fields validations;
- is minimized the data volume to be introduced or reintroduced from the keyboard.

In a banking computer science application, made available to citizens for electronic payments, there must be mandatory:

- all the data identifying the payer;
- all the data identifying the predefined beneficiaries;
- the list of accounts from which payments are made and the list of associated currencies;
- all forms of payment.

RETURN

Such an application requires the development of a procedure which is respected by all users, which presents the steps of acceptance and confirmation of electronic payments by the bank.

Citizen-oriented applications aimed at minimizing the discomfort of the citizen. Stages for a fine movement for speeding laws should be the following:

- the police officer has the device calibrated with valid certificate of calibration, posted on the Internet;
- draw up the document which has clearly filled in all fields;
- the document is issued on police car computer and has a unique code;
- the document is accepted by the driver;
- the fine is paid as follows: the police officer has a card acceptance device and the payment is made immediately or the payment is made later, to any bank or electronically. The payment made to the bank in the town hall account or in the police account, is reflected to the beneficiary automatically, regardless of what the bank is made or how. At the time of payment, the information reaches the city hall which belongs the payer, based on the personal identification number, and the police, based on the document code.

The transformation of information in knowledge becomes a reality if and only if they produce the widespread of automatic process for collecting data regarding the behaviour of all actors from the society and data processing with specific algorithms from the data mining area.

New technology that will be defined will include:

- cycle stages of development;
- tools to take the new requirements;
- quality metrics;
- models for costs;
- systems of collecting data regarding the behavior;
- components for adapting the structure of the application;
- components for ensuring security;
- evaluating options for each stage of the product;
- local component optimization;
- refining components;
- action subordination to the target group requirements.

With this new technology will develop applications that will complement the citizen, the user becoming part of the system. The Human-Computer interaction is more powerful, the system is more efficient.

6. CONCLUSIONS

The idea of citizen-oriented applications refers to an effective collaborative system in which people and equipment cooperate in order to achieve certain objectives. This is about Human-Human Interaction and Human-Computer Interaction. On-line citizen-oriented applications are accessed by a very large number of unhomogeneous users, which must spend as little time interaction with the application. Major problems arise from entering data errors. Therefore, applications must provide highly accurate

RETURN

information regarding the occurrence of mistakes and resolve them. The basic idea is that the user should end with the introduction of data in a few iterations.

REFERENCES

Ciurea, C. (2007). *Metrici ale sistemelor colaborative financiare*. Sesiunea cercurilor ştiinţifice studenţeşti din Academia de Studii Economice, Bucureşti.

Connolly, T., Begg, C. (2001). *Baze de date: proiectare, implementare, gestionare*. Editura Teora, Bucureşti.

Cooper, A. (1997). *Proiectarea interfeţelor utilizator*. Editura Tehnică, Bucureşti.

David, B., Delotte, O., Chalon, R., Tarpin-Bernard, F., Saikali, K. (2003). *Patterns in Collaborative System Design, Development and Use.* Laboratoire ICTT, Ecole Centrale de Lyon, France.

Fraizer, C., Bond, J. (1998). *Java API: manualul interfeţei de programare a aplicaţilor.* Editura Teora, Bucureşti.

Gause, D., Weinberg, G. (2004). *Exploring requirements quality before design.* Dorset House Publishing, New York.

Ivan, I., Boja, C., Ciurea, C. (2007). *Metrici ale sistemelor colaborative.* Editura ASE, Bucureşti.

Ivan, I., Doinea, M. (2008). *Aspecte privind optimizarea proceselor de autentificare în aplicaţii distribuite.* Economia – Teorie şi aplicaţii.

Pressman, R. (1988). *Software Engineering: A beginner's guide.* McGraw-Hill Publisher, New York.

Smeureanu, I., Dardala, M., Reveiu, A. (2004). *Visual C#.NET.* Editura CISON, Bucureşti.

Weinberg, G. (2007). *Quality Software Management: Systems Thinking.* Dorset House Publishing, New York.

RETURN