Informal approach based on user involvement to overcome uncertainties in a software project and to achieve high quality of an innovative product

Barbara Begier
Institute of Control and Information Engineering,
Poznan University of Technology,
pl. M. Sklodowskiej-Curie 5,
60-965 Poznan, Poland
E-mail: barbara.begier@put.poznan.pl

Abstract: There are serious problems with uncertainty referred to various aspects during evolutional development of an innovative software product. An open list of uncertain items in an expert system development is presented. In AI area some formal approaches to uncertainties are applied to build an expert system for medical diagnostics. Following the general idea applied in medicine, an informal approach to reduce uncertainties concerning product functions and handled data has been applied to improve software product quality. If symptoms of software ‘disease’ are learnt then the proper diagnosis and therapy may be specified. Various forms of user centeredness in a software process are discussed. Among other activities, the periodical assessment of a software product by its users is recommended to reduce some uncertainties. Users provide this way the regular feedback on a product. The experience, related to the developed non-trivial expert system assessed by its users, is described.

Keywords: expert system development; uncertain data; informal approach to reduce uncertainties; user-centeredness; feedback from users; software quality; software product assessment.

Reference to this paper should be made as follows: Begier, B. (xxxx) ‘Informal approach based on user involvement to overcome uncertainties in a software project and to achieve high quality of an innovative product’, Int. J. Intelligent Information and Database Systems, Vol. X, No. Y, pp.000–000.

Biographical notes:
1 Introduction

Despite many efforts and achievements in the area of artificial intelligence (Russel and Norvig, 2009) there is still a lack of useful methodologies to design an entire knowledge-based system. Quality is not in the centre of its authors’ attention. They concentrate their efforts on selected innovative mechanisms often based on formal methods. For example, probabilistic methods are usually applied to solve problems with uncertainty. User participation in a software development involving AI applications is ignored as a non-technical solution.

An innovative expert system as well as development of any knowledge-based system is often burdened by uncertainty related to requirements (Hsu et al., 2008). Other examples of uncertain items and their sources are given in Section 2. First of all, an expert system may operate uncertain knowledge. In addition, problems do not appear separately but they are interwoven in causal networks (Munk-Madsen, 2006). The most important domain for the treatment of uncertainties is diagnostics – medical or technical.

But in fact, just humans generate uncertainties. Human behaviour is not described formally and probably will never be. So, an interdisciplinary approach and semiformal or even informal methods are more and more applied in software production. For example, it refers to role-based stakeholder identification in a software process (Achtercamp and Vos, 2008). One of possible informal approaches is user involvement in software development (Begier, 2007). Feedback from users on a product seems to be the promising solution to overcome problems with uncertainty in software development and at the same time to ensure quality of an innovative software product. The informal approach with user involvement in the software process to reduce uncertainties is described in Section 3.

The concept of user-oriented approach is not new and it is often declared by software producers. In practice it has various meanings. Mainly it means working for users including formulating and respecting their goals during a new product development. But a point of view of software authors may substantially differ from the real user’s point of view. The user-centred design process starts with an envision phase (Quesenbery, 2001). The envision activities end with creation a vision of a product – its concept, target users, usability goals, and early requirements are specified. Then the analysis and design phases take place, as usually. The evaluation and refine phase is critical for the project. It aims to ensure that the product will meet user needs. After the implementation phase a special attention is paid on how users will be supported at their work when using the delivered product. Users participate in the design and evaluation phases. It helps anticipating problems with usability.

The author emphasises working with software users to obtain a regular feedback from them. In author’s opinion, users’ involvement helps creating solutions that satisfy their users by reducing uncertainties concerning software functions and handled data. Forms of cooperation with users’ are discussed in Section 4. The applied life cycle model providing feedback from users is presented in Section 5.

The basic and confirmed in practice form of users’ involvement is a periodical product assessment by a questionnaire survey. The structure of a questionnaire and accepted forms of results are described in Section 6. The presented approach and its results refer to the real life experience with the unique software product Analyzer of Facts and Associations (AFIZ, expressed in Polish) developed for analysts working for the Police and the Border Guard. The described user involvement has had two forms: regular
Informal approach based on user involvement to overcome uncertainties  

meetings with the involved specialists and then a periodical software product assessment by available user representatives. The last one is the most recommended here form of cooperation with users. The presented paper is an extended version of that presented during the ACIIDS conference (Begier, 2011).

2 Problems of uncertainties in an innovative product development

There is observed an amazing progress of software technology including an area of artificial intelligence. After negative lessons learnt there are numerous successful applications in medical diagnostics, for example. Various tools and design environments are available on the market. The IBM Ilog (2011), and using the clips notation the Java Expert System Shell (Jess, 2011) are their good and leading examples. The substantial number of this kind of tools may generate problems with an exchange of models and data or knowledge items.

There are undertaken efforts to specify and to provide an engineering approach to build experts systems. They include visualisation of developed processes referred to the area of application, definition of a product life cycle, and verification methods of obtained results like it is in the case of conventional software. On the other hand, knowledge acquisition still requires a lot of work to do. It concerns the required number of introduced facts and rules which are sufficient for useful solutions. It often meets problems with uncertainty.

Uncertainties appear in many situations in real life, including business management or ordering medical treatment, for example. Dew to new software technologies and fields of their applications, the notion of uncertainty has been introduced in reference to an intelligent system development. This notion is helpful when there is a need to consider situations, events, and first of all the inference process in which not all assumptions and facts represent certain data like it takes place in the case of data known to the doctor, for example. The uncertainty factors may be derived from representative statistics or may be estimated by experts, other humans or built in software.

Notion of causality is broadly applied in medicine from among other domains. The diagnosis is based on symptoms observed in the given real life case. At the beginning the probability of each symptom is specified a priori. Then the therapy, based on the expert knowledge and the observed symptoms, is specified. There is only some probability that the observed symptom points out the given disease because it appears at some patients but not at all of them. So, there is no certainty that the ordered therapy will work. It happens in conventional medical practice and should be considered also in the development of an expert system supporting doctors. In general, not only in medical applications, there are many reasons and at the same time various sources of an uncertainty related to uncertain data and uncertain knowledge in problem solving (Puppe, 1993):

- A set of observed and measured items may be insufficient for inference purposes.
- Knowledge about the relationships between the observed symptoms and the diagnosis is uncertain so the obtained conclusions may not work in each case.
- It is hard to specify the plausibility of some assumptions being used in the inference process.
• Analysed data and related rules may not be representative for the most of real life cases of the considered kind.
• Some data may be false for various or even unidentified reasons.
• Human cognition is not normalised so observations made by humans may differ in the considered cases because his/her observation is uncertain; thus the input may be uncertain.
• Some observations and then the subjective conclusions may represent the specified hypothesis or even a wishful thinking of an expert but not the proven evidence; then they are not confirmed later when more real life cases are analysed. It may happen because the field of observation is too small (the number of observed and considered cases is insufficient) and/or the considered cases are related to the inventive therapies.
• Domain experts may differ in specifying relationships between observed symptoms and based on them diagnosis (some of them may be wrong).
• There are many possible conclusions and therapies applicable in the given case.
• Domain rules and data are noticed in the given moment of time but some events and data are time-dependent and different values and phenomena may be observed in various periods of time.

If data are uncertain then the uncertain reasoning takes place. The problem is how to manage uncertain data and to reduce uncertainties in information processing and the inference process in real life cases. One solution, recommended in an expert system development, is to build the truth maintenance system (TMS) (Doyle, 1979) or reason maintenance system (RMS) which is the belief revision system. It is responsible for updating the knowledge base according to new situations, data, and provided evidence. The need to update the given knowledge base is required to preserve consistency. It appears in the case of user’s correction of a fact, replacement of an assumption or justification due to new information. Certainty factors (also called evidences, scores or probabilities) are estimated by experts.

An additional data structure is required for storing direct justifications of a conclusion, like it is in the justification-based TMS. So, the expert system should remember all justifications, then their modifications and extensions. This semi-formal approach may be applied in medical expert systems (Jankowska, 2005; Puppe, 1993), for example. The TMS updates the given knowledge base – it should remove facts which have lost their justification, introduce new facts, and remove facts which are in contradiction. But there are a lot of problems with treatment of loops, efficiency, and comparison of different solutions (Puppe, 1993). The guiding idea of uncertain data management including uncertain reasoning is applied on a high level of abstraction and proven usually in simplified conditions referred to limited and sometimes even trivial data.

Problem with uncertainties refers to many aspects in knowledge engineering. Also, the entire development process of an expert system and its information resources are burdened with uncertainty. The given above list of sources of uncertainty may be extended by the following, related to availability of domain experts in a software process:
Informal approach based on user involvement to overcome uncertainties

- There is no consensus if domain expert should be involved in a software process; a software purchaser is not interested to delegate his employees to cooperate with software developers because of cost.
- There is a lack of available experts in the given domain to take part in a collaborative design of ontology or contextual reasoning, for example.
- Available experts underestimate their role in a software process.
- Domain experts may not be interested to share their knowledge with software developers and to supply them real samples of data – their knowledge has been gathered year by year and becomes the valuable good desired on a market; in practice, it refers but is not limited to doctors, for example.
- Various experts become the stakeholders in the software process – each expert may represent his/her private and subjective point of view or interest.
- Software developers are not accustomed and often not skilled enough to cooperate with domain experts.

As a result many uncertain items appear in a development process of an innovative software system. An open list of uncertain items contains:
- completeness of requirements
- sense of automating specified activities – there is no agreement that the introduced solution is correct and welcome in user community in the real life
- availability of required documents and real life data
- suitable selection of quality criteria and their measures of software under development
- definition of a development process – if it is proven in practice to ensure quality
- suitability of the provided infrastructure to the innovative product development
- identification of quality factors and their real impact on an analysed process
- assessment of quality in use (selected measures, available real users)
- assumption that product improvements will ensure the expected results.

The problem is how to consider all the mentioned above items in a software process and how to improve it in advance?

3 Informal approach with user involvement to reduce uncertainties

In the development of any expert system most of problems are generated by humans. And, in author’s opinion, only humans may help to overcome the identified problems. Thus the described approach, followed that applied in medicine, may be also adapted and applied in a software process to improve an innovative software product quality by reducing some uncertainties. Such informal approach requires to:
Specify a set of symptoms $s_1, \ldots, s_n$ of inappropriate functioning of a software system to describe its ‘illness’ $x$. Specify also a set of symptoms of any undesired software behaviour.

Identify, in cooperation with users and domain experts, their sources.

Gather data coming from observations and tests and also those expressed by software users.

Develop a set of therapies $t_1, \ldots, t_k$ in a form of required software improvements specified due to the observed symptoms, obtained notes, and suggestions given by users.

But who is the right person to assess described symptoms and how to manage them to specify and order suitable therapies? The answer is that real software users are domain experts although an involvement of external experts is not excluded. User involvement is recommended here to deal with the most of the listed in the previous section sources of uncertainty. Recommended forms of cooperation with users in a software process are described in the next section. But the most recommended and not expensive form of user involvement is a periodical assessment of a given product version. In practice, it refers to an expert system.

In turn, software developers should be skilled enough for introducing the specified methods and recommendations based on cooperation with users into the software process. The following steps are recommended to improve quality of an expert system:

- Periodical assessment of product quality by its users is planned and performed according to the plan.
- Quality measures are derived from quality criteria and included in a questionnaire of a survey; poor notes assigned to these measures during the survey are potential symptoms of ‘disease’ of the assessed software product.
- Poor values given by users for the specified measures confirm the ‘disease’.
- Importance of each symptom is expressed and emphasised by its weight.
- The diagnosis, based on an analysis of obtained values and remarks given by respondents, points out improper solutions and other weak points of a product. In particular, it may find out also a lack of some features and facilities.
- The set of therapies referred to the noticed symptoms and their weights is specified in a form of required product improvements planned for the next iteration of software development.

Negative symptoms are observed by users and expressed during software assessment by values assigned to the specified measures. It is assumed that after therapy $t_i$ the presence of a symptom $s_i$, should decrease. After release of each software version its assessment by potential users takes place. All notes and suggestions are carefully stored. Then related improvements of the product are performed and recorded as a history of the evolitional product development.
Informal approach based on user involvement to overcome uncertainties

4 Various interpretation and forms of user-centred design

Users’ involvement in software development is recommended starting from the Nygaard’s participatory design initiated in early seventies to emphasise social aspects of innovative software product. Designers and programmers use different language and show other preferences than users. So, cooperation with users is required to ensure a high level of users’ satisfaction from an innovative software product.

The rapid development of the electronic government concept is recently observed and a lot of effort is spend on it. E-government allows the public and organisations to receive specified services at any place and any time. There are many open questions at this area. Does the offered technology fit to the specified task characteristics? How does it affect the work of employees? Some of these kinds of questions have been answered in an empirical study performed on Taiwan where e-government services were ranked second among 198 countries in 2008 (Luarn and Huang, 2009). But do citizens trust the available tools and methods of e-government? Do they like to use these services?

The ISO 13407 standard, first published in 1999 and revised in 2010, defines a general human-centred design processes for interactive systems. Human-centred activities may be included throughout a software development cycle. But this standard does not specify exact methods. Most methodologies suggest specifying the key stakeholders which are helpful to set the software product vision. Users are no longer assumed as the monolith group – the real user profiles are to be recognised and developed. There are various forms of user involvement during the design phase: brainstorm sessions, doing walkthroughs of design concepts, meetings using paper and pencil to design, creating and testing early prototypes, and others.

The term of user-centeredness meets various interpretations related to its various dimensions (Iivari and Iivari, 2006) although all of them are intended to ensure software quality. At first, the declared user focus is limited to product goals specified from a fictional user’s perspective because no real user takes part in this activity. In turn, the term of work-centeredness refers to activities and efforts oriented to effective work of an average but still abstract user whose actions are supported by the developed system. Then a concept of user participation in the software process means working with users although this idea is often limited to small software projects during their design and then testing referred mainly to the user interface. The most advanced is the term of a personalisation of a product according to his/her preferences including personalised forms of required information retrieval. In such interpretation, the personalisation requires involving the diversity of real users in the design process and learning user’s behaviour when using the system.

The user centred design (UCD) may be also related to the web-based software development where the human-computer interaction is emphasised like it takes place in the described case of an interactive map of the underground train and a navigation of train routes (Waralak, 2011). Human-computer interaction (Greenberg, 1997) consists of four main parts which are, respectively, responsible for: context of use, human characteristics describing how people are expected to interact with machines at their daily work, interface architectures, and the development process of the interface. In general, the study of interaction between people and computers is an interdisciplinary subject because people cannot be expected to act and react always in the same manner as machines. And more, the interactive media, including those applied in medical
diagnostics, integrate digital devices, combination of electronic text, graphics, moving images, and sound.

The key problem of an expert system is to make the developed tool useful and at the same time easy to navigate it. The recommended inexpensive and acceptable technique is to design and apply questionnaires for specified users. Users’ satisfaction with a software product refers first of all to its functionality and usability. The designed interface makes product pleasant (or unpleasant) to use and view.

Early user involvement is related to higher quality of requirements – the analysis of real process shows that involving users is also related to the project final success (Kujala et al., 2005). It has been emphasised that the direct contact with users helps analysing the real-world examples and the context of software use. When users are not involved, the developers tend to observe them as one big faceless mass. Software developers’ isolation from users may lead to serious problems to satisfy them.

There are various forms of user involvement in a software process. Most of them refer to requirements specification and software usability testing. In many projects users are stakeholders in software development although their real impact on a process is discussible – regular users are not the decision-makers. Software developers emphasise an expected cost of user involvement although the future savings on product improvement cannot be ignored. Principles behind the Agile Manifesto (Principles, 2001) and applied in agile methodologies emphasise the direct communication between developers and software users. An aim of these activities is to learn who users are, to know their expectations, and their point of view on quality of software under development.

There is the question − when and which forms of user participation are actually helpful in software development? Software developers may be prejudiced and afraid of users’ negative impact on project performance. At the beginning, cooperation with users may seem to be time-consuming and less effective than previously. However, in a long perspective the number of needed iterations decreases because of the lower number of detected errors and higher than before quality assessed by product users. The survey related to data concerning 117 software projects (Subramanyam et al., 2010) confirm that the highest level of software authors’ satisfaction results from a high user involvement in new software projects although at the same time users’ expectations are excessively growing. It is not surprising that users prefer engaging their minimal time in the development process.

A practice to build a vision of a future software product is one of the most recommended in agile approaches. Users participate in brainstorm sessions to create the product vision and to specify software features because applications of the developed software solution will implicate their daily work. Forms of user participation in a software process, recommended by the author, are shown in the Figure 1. The most recommended here and confirmed by real life experiments is a periodical product assessment by its users.
Informal approach based on user involvement to overcome uncertainties

Figure 1  Some recommended forms of cooperation with users

There is a question how many user representatives should be involved in a software process to ensure software quality referred at least to product usability. Its evaluation is essential to make sure that the developed software product is easy to learn and to use for real users (Hwang and Salvendy, 2010). Users are not identical – they represent various personality types and differ in age, gender, level of education, profession, life conditions, needs, and their life experience. Thus, the individual representative engaged in the development process of an expert system is not enough to express various users’ expectations and suggestions. There is no consensus regarding the optimal sample size of user representatives – from $4 \pm 1$ (it means four or five users are needed) to detect 80% of usability problems (Nielsen, 1994) up to the entire population of potential users.

When analysing user community in practice, one may observe that software users share many common features. First of all they are working for the same organisation where the similar competencies are required and tested during the recruiting process. So, users usually show almost the same educational level, place of living, type of work, acquired skills, trainings, and professional experience. The data related to 102 usability evaluation experiments allow forming a general rule for optimal sample size: it would be $10 \pm 2$ (Hwang and Salvendy, 2010), although the cited numbers refer only to usability evaluation. If the number of respondents is insufficient in the real case then one can dispute their suggestions.

5 Users involvement in an innovative product development cycle

Software product is addressed to the specified community of its direct and indirect users – clerks, doctors, engineers, applicants, etc. All of them are experts in the given domain of an application. Software designers may misinterpret user needs and then the implemented facilities do not meet user expectations. So, the first step is to learn who users are. In other words, software authors should learn user profiles.
Evolutional development of an expert system refers to its *shell* responsible for communication with users, its *kernel* consisting of various data (knowledge base), and provided mechanisms including an inference engine. Users’ involvement is recommended here to emphasise social and quality aspects of an innovative product development – to identify and overcome problems with uncertainties as much as possible. The direct relationships may cause that users have confidence in software authors and are inclined to entrust them with making use of results of their own professional experience. Otherwise designers cannot expect to learn user expectations and study the real life cases. Users’ involvement may help avoiding problems with reliable real data for testing purposes.

An evolutional product development supported by a continuous feedback from users is presented in the Figure 2. Users’ representatives take part in software development starting from its early phases. Users’ involvement is not limited to the requirements specification although it’s been proved that project is successful when users play an active role in this phase (Kujala, 2008). The general vision of an expert system should be created in cooperation with users. Then potential sources of uncertainty are pointed out during direct meetings of software designers and its potential users. Users’ involvement is needed to include required data verification in software product design and its testing – these procedures should consider specific conditions for the application domain.

**Figure 2** Evolutional development of an expert system and its improvement based on continuous feedback from users (see online version for colours)
Informal approach based on user involvement to overcome uncertainties

The chosen technique to obtain a feedback from users on a developed product must be acceptable by a wide spectrum of its users including the managerial staff of a given organisation. In the described approach the most valuable feedback is obtained in the product periodical assessment by user community. Such approach has been previously applied by the author to the expert system developed for civil engineers. It was the very successful experience (Begier, 2010).

The reported case study refers to the product AFIZ, which is a part of software developed under the scientific grant entitled Polish Platform for Homeland Security (PPBW, 2005–2011), the unique scientific initiative beyond European dimension. The undertaken activities are aimed at creating integrated computer tools to improve public security. Those tools based on modern technologies should support police and other security services, like the border guards. Due to the sensitive nature of data and project topics, a main part of research work within the platform has the classified status. The aim of the considered product is to facilitate analysis of telephone and International Mobile Equipment Identity (IMEI) billings. The analysis of telephone billings helps finding a valid telephone number of a wanted person. Location of masts base transceiver station (BTS) involved in an analysed communication may help tracking his/her steps.

Individual experts working for the mentioned above public services, especially investigative ones, participated in early phases of the considered software development cycle. Product functional features should be identified and planned for each iteration – an iterative-incremental model of software life cycle has been applied. The concept of a feature refers to that introduced in the Feature Driven Development (FDD), one of agile methodologies (Agile ... FDD). The FDD and its practices rely on granularity of product functionality. Significance of each specified functional feature for an entire product is considered. The feature hierarchy is specified according to the assigned priorities and then implemented.

The very important form of users’ involvement is software product assessment in a survey by questionnaire. Each working software version is assessed by its users who assign values to the specified quality measures of the considered product. Respondents assess this way to what degree the provided software tool meets their needs. Users are also supposed to add also their suggestions on how to improve the product. The obtained results are not limited to easiness of product use and user interface. They can indicate some missing requirements, point out the proper interpretation of requirements, and implicate the way of their implementation. Results of a product assessment are the basis of further product improvements and have real impact on an entire development process. Details of the performed software product assessment by its potential users are described in the next two sections.

6 Questionnaire survey to assess software product by its users

The first step to arrange the quality assessment of a given expert system is to plan all required activities. Before any undertaken action it is necessary and not obvious to get an agreement of senior managers of the customer organisation. In the reported case, due to the character of an application, senior officers of the respondents have hesitated for some time to agree on a design of the questionnaire content. After necessary explanations the agreement became true.
The described questionnaire starts with an initial part intended to learn who respondents are. It contains the following phrases:

1. I am a person in the age bracket: A (21 to 35), B (36 to 49), C (over 50)
2. My level of education is: A (master degree), B (bachelor), C (high school)
3. My position at work is: A (managerial), B (ordinary), C (specialist)
4. Self-assessment of my skills in computing is … (<1, 5> using school marks)
5. Frequency of using computer by me in last six months can be estimated as: not at all, once a month, once a week, every 2 days, every day
6. My self-assessment of using other professional software is … (<1, 5>)
7. My place of work is located in … (<region of the country>).

Quality criteria and their measures have to be specified to assess a product. They should include provided data verification and data exchange mechanisms. Maintained incorrect data are the main source of improper system behaviour and its poor usefulness from the users’ point of view. Developers are erroneously convinced that these are problems of marginal importance and just users should take care of it. In practice, incorrect data project on the users’ bad opinion of a product. In the presented case, there are 55 questions in the main part of the designed questionnaire divided into subparts referred to the selected quality attributes of the considered software:

- functionality
- navigation as a result of a lucid construction (or not)
- easy way of data input and then results presentation
- ease of learning to use
- ease of using
- data security
- reliability and efficiency
- portability
- general assessment of a product and user satisfaction with it.

Each item of the questionnaire has a form of a statement, for example: 7. Program maintains and reconstructs steps of data processing making all history of the considered matter available because the psychometric scale devised by Rensis Likert has been applied to express each respondent’s answer – a respondent is asked to indicate his/her degree of agreement with the given statement, using a five-point scale:

5 – I fully agree
4 – I rather agree
3 – I have doubts
2 – I rather disagree
1 – I completely disagree.
Informal approach based on user involvement to overcome uncertainties

The scale $<1, 5>$ is applied at schools in Poland where ‘5’ is the highest mark equivalent to the American ‘A’ and, respectively, ‘1’ is equivalent to the ‘E’.

Free space has been left at the end of the questionnaire for user’s suggestions and remarks concerning the expected improvements of the considered software product. The experience shows that it helps obtaining valuable information from users.

At the first time the paper questionnaires were applied. First assessment of the considered expert system AFIZ took place at the end of training of a large group of its potential users. The structure of the questionnaire and then respondents’ anonymous answers were known to senior officers in the reported case. The second assessment took place in an electronic form after two months. Senior officers gave their agreement on participation of interested officers in the survey but the questionnaires were fulfilled anonymously to be free to tell the truth.

At first, the user profile of respondents has been recognised. An average respondent is a relatively young and high educated man who works as a specialist, uses computer daily and applies specific professional software. As much as 54% of respondents are people in the 21 to 35 age bracket, 42% are 36 – 49 years of age and only 4% are above 49. Almost two thirds of respondents have got the master degree. Every fourth respondent occupies a managerial position, 30% are individual specialists; the others occupy ordinary positions. All of them use computer every day although only 30% assessed their computing skills as the very good, 61% as good and 9% as sufficient. Most of them (82%) declare their proficiency in using other professional software systems. Only one half fulfilled an item concerning their work location – respondents omitted it because they did not want to be identified on its basis.

7 Results of two editions of the considered product assessment

Each time respondents had to assess 53 measures from 55 items in the questionnaire; two items concerning product portability were postponed to further assessments. According to directives applied in marketing, the qualified minimum for each item is 70% of obtained answers. In the first edition of a survey more than 30% respondents omitted six items — many users skipped two items concerning ease of learning to use a programme, two items concerning programme security, one question about programme installation, and one question about the percentage of daily work supported by the assessed product. Probably it is required more experience with a considered tool to answer all questions. So, 47 items of the questionnaire could be statistically processed and then analysed.

Unfortunately, the author is not entitled to present precisely the statements of the questionnaire and the detailed data concerning assessment of its separate items. But also statistical results may be still interesting. As mentioned earlier, poor notes assigned by respondents to specified measures are the symptoms of a product disease. As much as 33 product characteristics (on 47) got the mark not less than 4.0, 13 got a mean value from the bracket $<3.5, 3.99>$, one got a mean value from the bracket $<3.00, 3.5>$, and only one less than 3.0. The last item is related to the possibly required help of an experienced user of the assessed expert system. Programme functionality was highly assessed but with some exceptions.

From among others, items concerning ease of learning to use a programme were assessed worse. Four items placed in the last part of the questionnaire (general assessment of a product and user satisfaction with it) were assessed quite well. The
conformity of programme functions with user expectations got the mean value 3.77 of all given marks. Respondents admitted that the considered expert system makes their work easier (mean value 4.0). Using a programme is not stressful – the mean value of related marks was 4.33. The level of respondents’ satisfaction from a programme had a mean value 3.9.

Suggestions of programme improvements given by respondents at the end of the questionnaire were the most important and valuable for the product authors. Precisely every second respondent gave suggestions and/or remarks. For example, users have made demands to implement visualisation of relationships between a group of telephone connections and BTS masts, to replace BTS code with a name of a place (town or village), to identify and mark BTS masts located near the border posts, to ensure import of many billings coming from various operators at the same time, to facilitate using information concerning activation and then deactivation of a telephone number given by an operator, etc.

Many opinions were shared by several respondents. After the careful analysis, the list of 31 proposals of programme improvements was established and transferred to software authors. These proposals constitute a set of the specified therapies. Their effectiveness will be examined during the next product assessment which takes place when the new software version is developed.

The second edition of a product assessment took place after two months. Respondents assessed the unchanged product and applied the same questionnaire. Again, two items concerning product portability were to be omitted. Questionnaires were fulfilled by almost the same users (only three persons were different) so their profile was also almost the same as previously. In 52 cases on all 53 items of the questionnaire the answers were qualified to their statistical processing (minimum 70% answered).

Again, features of the AFIZ programme were assessed highly – 27 items got the mean value higher than or equal to 4.0, 20 got their mean value from 3.5 to 3.99 bracket and only four had mean values less than 3.5 but more than 3.0. But a lot of measures got a bit worse values than before including 9 measures of functionality (on 12 in total), 7 of interface friendliness (on 10), and only 3 on 7 measures of ease of use. This fact confirms a conjecture that an unchanged product may get with time worse notes than before because user expectations are growing and growing. The percentage of very good and good marks was in 33 cases higher in the first edition, one was equal in both editions, and 19 measures got lower notes at the first time. The essential dispersion of answers related to a given item was observed in many cases.

It is worth emphasising that during the second edition of the survey almost every respondent (90.5%) gave his/her remarks and suggestions of programme improvement. They usually wrote several sentences or a list containing five items on average. Again, there were repetitions in the suggested proposals. This time respondents have made demands concerning expected cooperation with other software products applied by telephone operators in a home country and abroad. Other demands were concerned with filtering data and transforming various formats of applied data. There were suggestions to extend product functionality to track the route of an analysed person (more precisely, his/her cellular telephone), to use names of towns and roads during these activities, to use many kinds of data identifying a wanted person or one connection, to implement graphical support of telephone connections applied in one analysed affair, to use specified colours to mark some interesting data, etc.
8 Conclusions

Development of an expert system is burdened with uncertainty more seriously than it is in the case of a conventional software product. Considered uncertainties are mostly related to software product functions and their results expected by software users. User involvement in the form of software product assessment by its real and potential users is recommended and applied to solve problems concerned with uncertainties related to: completeness of requirements, justification of automating specified activities, availability of required documents and real life data, design and implementation of specified functions, a need to cooperate with other software tools, selection of quality criteria and measures of software, and others. This form of users' involvement has been proven in practice.

The questionnaire survey, repeated after each iteration, indicates strong and weak points of the developed product version. Poor notes given by users during the software assessment are the symptoms of a product disease. Then a set of therapies is specified in the form of a product desired improvements. Thus, it becomes the leading activity in cooperation with users throughout the software development cycle. It decidedly does not interrupt the developers' work and thus has been accepted by their community. Many valuable suggestions on how to improve a product have been introduced during each survey. On the other hand, the described experience shows that the same product after a relatively short period of time may be assessed a bit worse by its users than before. So, the conclusion is that each software product has to be continuously improved because user needs and expectations are growing quickly after their basic needs are satisfied.

Measures applied in the conducted surveys referred also to user satisfaction from a software product. The higher user notes given to particular measures and the greater number of user suggestions mean the lower level of uncertainty in software development. This approach may be recommended in many applications. It does not eliminate other forms of cooperation with users like joint teams, regular meetings, brain storm sessions, and others.

And also from the software developers' point of view the described approach makes the desired success of the software project possible although all differences between their own and the users’ points of view. The satisfied customer is the primary measure of a project success in a long time perspective.

References


Comment [t3]: Author: Please provide the issue number.

Comment [t4]: Author: Please provide the access details (date when the site was accessed/visited).


