# Working with Users to Ensure Quality of Innovative Software Product despite Uncertainties

Barbara Begier

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

**Abstract.** Problems with uncertainty refer to various aspects in knowledge engineering. An open list of uncertain items in expert system development is given and their sources are listed, too. The periodical assessment of a software product by its users is recommended to reduce uncertainties. If symptoms of 'disease' are learnt then the proper diagnosis and therapy may be specified. User involvement is needed to ensure quality of innovative software during its evolutional development. Users provide regular feedback on a product. The experience, related to the expert system assessed by its users, is described.

**Keywords:** expert system development, user-centeredness, uncertain data, feedback from users, software quality, software product assessment.

## 1 Introduction

The problem to develop an innovative software product, like an *expert system* or a knowledge-based system, which satisfies its users seems to be more difficult than in the case of a conventional software product. Development of an expert system is burdened by *uncertainty* [4]. This concept applies to various items in knowledge engineering. For example, an expert system may operate uncertain knowledge. Other examples of uncertain items and their sources are given in Section 2.

Probabilistic methods are usually applied to solve problems with uncertainty. But just the humans generate uncertainties. So user involvement and their feedback provided on a product seem to be the promising solution to overcome a problem of uncertainty and to ensure quality of an innovative software product. Many software producers declare their *user-oriented* approach. It means usually working *for* users and respecting their goals when developing a new product. But a point of view of software authors may substantially differ from the real user's point of view. Thus the term of *user-centeredness* remains the term applied previously. Users' involvement may help reducing possible uncertainties which burden innovative software products to build solutions that satisfy their users. Forms of cooperation with users' are presented in Section 3.

The applied life cycle model providing feedback from users on a developed product is presented in Section 4. The recommended here, basic form of users' involvement is a periodical product assessment by questionnaire survey. Real and/or potential users take part in it. The design of a questionnaire and results of the conducted

N.T. Nguyen, C.-G. Kim, and A. Janiak (Eds.): ACIIDS 2011, LNAI 6591, pp. 337–346, 2011.

surveys are described in the fourth section. The presented research refers to the real life experience with the unique software product AFIZ (this name is an acronym of the *Analyzer of Facts and Associations*, expressed in Polish). It has been developed for analysts working for the Police and/or the Border Guard. The described user involvement has had a form of regular meetings with the most involved specialists and also the most recommended form of a periodical software product assessment by available user representatives.

## 2 Uncertainties in an Innovative Product Development

The notion of *uncertainty* refers to many aspects in knowledge engineering. Also the development process of an expert system and information resources required for that purpose are burdened with uncertainty. An open list of uncertain items contains:

- Completeness of requirements.
- Availability of required documents and real life data.
- Definition of a development process – if it is proven in practice to ensure quality.
- Suitable selection of quality criteria and measures of developed software.
- Identification of quality factors and their real impact on an analyzed process.
- Way of assessment of *quality in use* (selected measures, available real users).
- Sense of automating specified activities – how to prove that the introduced solution is correct and welcome in user community in the real life.
- Assumption that product improvements bring the expected results.
- Suitability of the provided infrastructure to the innovative product development.

User involvement is needed to deal with the first eight from the listed above 9 items. There are many sources of an uncertainty (extended version of that in [12]):

- Lack of available experts in the given domain to take part in a collaborative design of ontology or contextual reasoning, for example.
- 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.
- Various experts become the stakeholders in the software process – each expert may represent his/her private and subjective point of view or interest.
- Analyzed data may not be representative for the most of real life cases.
- Some data may be false (at variance with the facts) for unidentified reasons.
- A set of observed and measured items may be insufficient for inference purposes.
- Human cognition is not normalized – this statement refers also to human observations, opinions and considered cases.
- Domain rules and data are noticed in the given moment of time; different values and phenomena may be observed in various periods of time; some rules and data may change before a knowledge-based system is completed.
- Domain experts may differ in specifying relationships between observed symptoms and based on them diagnosis (they may be even wrong).
- There are many possible conclusions and therapies applicable in the given case.

Notion of *causality* is broadly applied in medicine. The diagnosis is based on the observed symptoms which are real life data. Then the therapy is specified. 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 even trivial medical data. Then this idea has been referred to any knowledge-based system [12]. In the author's opinion, it may be also applied to improve software quality by reducing some uncertainties. It requires to:

- Specify a set of symptoms $s_1, \ldots, s_n$ of 'illness' $x$ (inappropriate functioning).
- Identify, in cooperation with users, their sources and specify their importance.
- Gather data coming from observations and tests (expressed by notes and opinions given by software users, for example).
- Propose a set of therapies $t_1, \ldots, t_k$ (specification of software improvements).

Various people, including domain experts, may judge symptoms (measures) differently. But the greater number of participants, the better statistical results. 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 criteria and their measures are specified as potential *symptoms* of a software product '*disease*' (poor values given by users) and included in a questionnaire of a survey.
- 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 *therapy* referred to the noticed symptoms is specified in a form as required product improvements.

After therapy $t_j$ the presence of a symptom $s_i$, (observed by users and expressed by the given values of measures) should decrease. Notes and suggestions given by users are stored and related product improvements are recorded. Thus a history of an evolutional product development is maintained. In addition, problems do not usually appear separately but they interwoven in causal networks [8]. In the development of an expert system most of problems are generated by humans. And in author's opinion, only humans may help to overcome the identified problems.

## 3 User-Centeredness and a Scope of User Involvement

Designers and programmers used so far to work in isolation from users. They use different language and show other preferences than users. Thus *users' involvement* in the development process is recommended to emphasize social aspects of innovative software product [3] and to ensure a high level of users' satisfaction from it. This idea derives from the Nygaard's *participatory design* initiated in early seventies.

The *user-centeredness* is intended to ensure software quality. However, the meaning of this term may differ representing its various dimensions [6]. The declared *user focus* is usually limited to specifying goals from user's perspective but a fictional user substitutes for a real one. Another dimension refers to *work-centeredness* – there is a focus on effective work of an average but still abstract user whose actions are

supported by the intended system. Next, a concept of *user participation* in the software process means working *with* users instead of working for them. But this idea is often limited to small software project during its design and then testing user interface. Finally, to provide an adaptable *personalization* of a product according to his/her preferences requires involving the diversity of real users in the design process and learning user's behavior when using the system. The author' emphasizes *working with users* to obtain a regular feedback from 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. Principles expressed in the Agile Manifesto [11] and applied in agile methodologies emphasize *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.

Software developers are often prejudiced and afraid of users' negative impact on project performance which can become more time-consuming and less effective than previously. So there is the question of when user participation is actually helpful. The described survey data from 117 software projects [13] 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. In turn, users were most happy by engaging minimal time in the development process.

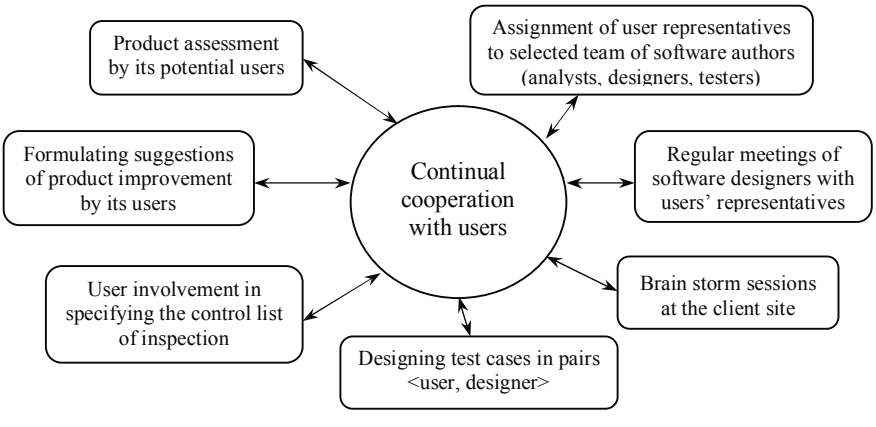


**Fig. 1.** Some recommended forms of cooperation with users

In some agile approaches users participate in brain storm sessions to build a vision of a future software product and specify its features because software product solution and its applications implicate their daily work. Forms of user participation in a software process, recommended by the author, are shown in the Figure 1. The experiments described in this paper are focused on a *product assessment by its users*.

Moreover the usability evaluation is essential to make sure that the developed software product is easy to learn and to use for real users [5]. There are several methods for usability evaluation: Think Aloud (testing including thinking aloud at

user's workplace instead of at labs), Heuristic Evaluation (usability inspection), and Cognitive Walkthrough (a theory-based method in which every step required to perform a scenario-based task is evaluated to detect potential mismatches).

User's community is not homogeneous – users represent various personality types and differ in age, gender, level of education, profession, and their life experience. Thus the individual representative engaged in the development process of an expert system is not enough to express various expectations and suggestions. The question is how many software quality evaluators need to be involved to present opinion of user community. There is no consensus regarding the optimal sample size in this case – from 4±1 (four or five users are needed to detect 80% of usability problems) [9] up to all population of potential users. In practice, software users working for the same organization share many features, like their educational level, place of living, type of work, acquired skills, trainings, and professional experience. The analysis of data with 102 usability evaluation experiments allows forming a general rule for optimal sample size: it would be 10±2 [5] instead of 4±1. If it is so with usability evaluation then a similar number of users may be sufficient to assess other software quality features.

## 4 Feedback from Users in a Product Life Cycle. A Case Study

### 4.1 User Involvement in an Innovative Product Development

Quality problems in an expert system development refer to both its *shell* responsible for communication with users and its *kernel* consisting of various data (knowledge base) and provided mechanisms including an inference engine. Software designers may misinterpret user needs and then the implemented facilities do not meet user expectations. There are also a lot of problems with reliable data for testing purposes.

Any 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. So the first step is to learn who users are. Users' involvement in software development is recommended here to emphasize social and quality aspects of an innovative product – 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 including the required test data.

General aim is to obtain a feedback from users on a developed product to make it acceptable by a wide spectrum of its users. An evolutional product development supported by a continuous feedback from users is recommended here as shown in the Figure 2. Users' representatives take part in software development starting from its early phases. In the described approach the most valuable feedback is obtained in the product periodical assessment by user community. Such approach has been applied by the author successfully to the expert system developed for civil engineers [2].

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 [7]. 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.

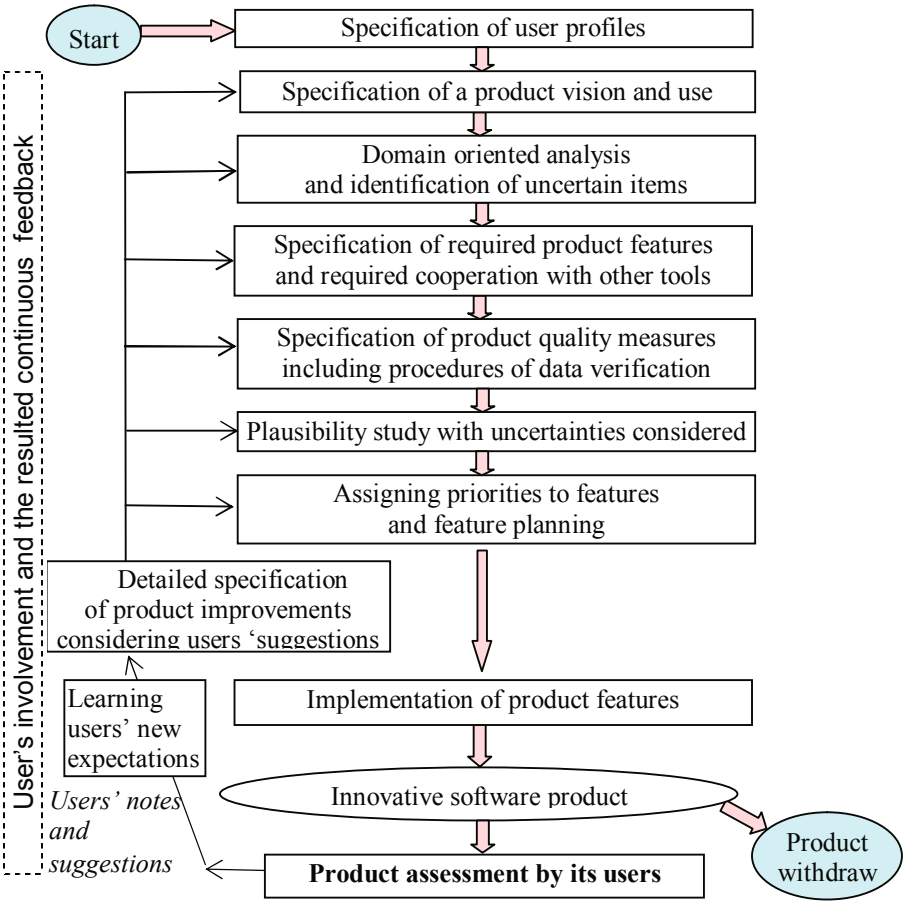


**Fig. 2.** Evolutional development of an expert system and its improvement based on continuous feedback from users

The described research refers to the product AFIZ (*Analyzer of Facts and Relations* expressed in Polish), which is a part of software developed under the scientific grant entitled *Polish Platform for Homeland Security* [10], the unique scientific initiative beyond European dimension. The undertaken activities are aimed at creating integrated computer tools to support specified efforts to improve public security. The main focus is to support police and other security services, like the border guards, with tools based on modern technologies. 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 IMEI (*International Mobile Equipment Identity*) billings. The analysis helps finding a valid telephone number of a wanted person. Location of masts BTS (*Base Transceiver Station*) involved in an analyzed communication may help tracking his/her steps.

Individual experts working for the mentioned above public services, especially investigative ones, participated in all phases of the considered software development

cycle. Applying an iterative-incremental model requires to identify product functional features to be implemented in each iteration. The concept of a *feature* refers to that introduced in the FDD (Feature Driven Development) [1], one of agile methodologies. The FDD and its practices rely on granularity of product functionality. Significance of each specified feature for an entire product is considered. The feature hierarchy is specified according to the assigned priorities and then implemented.

In the described research, the very important form of users' involvement is software product assessment in a survey by questionnaire (Section 4.2). 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 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, as described in Section 4.3, are the basis of product further improvements and have real impact on an entire development process.

### 4.2 Design of a Questionnaire to Assess Software Product by Its Users

Due to the character of an application, senior officers of the respondents have to agree on the design of questionnaire content. The described questionnaire starts with an initial part intended to learn who respondents are. It contains the following phrases:

I. I am a person in the age bracket: A (21 to 35), B (36 to 49), C (over 50)
II. My level of education is: A (master degree), B (bachelor), C (high school)
III. My position at work is: A (managerial), B (ordinary), C (specialist)
IV. Self-assessment of my skills in computing is (<1, 5> using school marks)
V. 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
VI. My self-assessment of using other professional software (<1, 5>)
VII. My place of work is located (<region of the country>).

To assess a product the quality criteria and their measures have to be specified. They should include provided data verification and data exchange mechanisms. Maintained incorrect data are the main source of improper system behavior and its poor usefulness from the users' point of view. Developers are erroneously convinced that these are problems of marginal importance. 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),
- 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.

The psychometric scale devised by Rensis Likert has been applied to express each respondent's answer. 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*. Then a respondent is asked to indicate his/her degree of agreement with this 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*.

Such scale is applied at schools in Poland where "5" is the highest mark equivalent to the American "A" and "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 improvement of a product.

### 4.3 Results of the Questionnaire Survey

The first assessment of the considered expert system AFIZ took place at the end of training of a large group of its potential users. The paper questionnaires were applied then. The second assessment took place in an electronic form after 2 months. Senior officers gave their agreement on participation of interested officers in the survey but the questionnaires were fulfilled anonymously to tell the truth.

***User profile***

An average respondent is a relatively young and high educated man who works as a specialist, uses computer daily and knows specific software applied in the profession. 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% is an individual specialist; the others occupy ordinary positions. All respondents 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 this item because they did not want to be identified on its basis.

***First edition of a product assessment***

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 considered survey more than 30% respondents omitted 6 items ─ many users skipped 2 items concerning ease of learning to use a program, 2 items concerning program security, one question about program installation, and one question about the percentage of daily work supported by the assessed product. Probably more experience with a considered tool is required to answer these items. So 47 items of the questionnaire could be statistically processed and then analyzed.

The author is not entitled to present precisely the statements of the questionnaire and the detailed data concerning assessment of its separate items. But even statistical results may be interesting. 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. Program functionality was highly assessed but with some exceptions. Items concerning ease of learning to use a program 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 program 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 program is not stressful – the mean value of related marks was 4.33. The level of respondents' satisfaction from a program had a mean value 3.9.

The most important and valuable for the product authors were suggestions of program improvements given by respondents at the end of the questionnaire. Precisely every second respondent gave suggestions and/or remarks. Many opinions were shared by several respondents. After the careful analysis, the list of 31 proposals of program improvements was established and transferred to software authors.

***Second edition of a product assessment***

During second edition of the survey respondents assessed the unchanged product. They applied the same questionnaire in both surveys of the AFIZ tool in 2010. Again, two items concerning product portability were to be omitted. Questionnaires were fulfilled by almost the same users (only 3 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).

Features of the AFIZ program were assessed highly again – 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.

Almost every respondent (90.5%) gave his/her remarks and suggestions of program improvement. They usually wrote several sentences or a list containing 5 items on average. Again, there were repetitions in the suggested proposals.

## 5 Conclusions

The problem with uncertainty when referred to the expert systems seems to be more serious than in the case of a conventional software product. User involvement is

recommended and applied to solve problems with uncertainties. Its valuable form, proven in practice is software product assessment by its real and potential users.

The assessment indicates clearly strong and weak points of a product and thus becomes the leading activity in cooperation with users. It breaks off the product authors' isolation from users but it does not interrupt their work decidedly and thus may be accepted by developers' community. Various suggestions on how to improve a product are introduced. This approach does not eliminate other forms of cooperation with users like joint teams, regular meetings, brain storm sessions, and others.

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 after their basic needs are satisfied.

The notion of a software project success from the perspective of developers is at the first sight different than that from software users' point of view. But in a long time perspective the satisfied customer is the primary measure of a project success.

## References

1. Agile Software Development using Feature Driven Development (FDD), http://www.nebulon.com/fdd/
2. Begier, B.: Evolutionally Improved Quality of Intelligent Systems Following Their Users' Point of View. In: Nguyen, N.T., Katarzyniak, R., Chen, S.-M. (eds.) Advances in Intelligent Information and Database Systems. SCI, vol. 283, pp. 191–203. Springer, Heidelberg (2010)
3. Begier, B.: Users' involvement help respect social and ethical values and improve software quality. Information Systems Frontiers 12, 389–397 (2010)
4. Hsu, J.S.-C., Chan, C.-L., Liu, J.Y.-C., Chen, H.-G.: The impacts of user review on software responsiveness: Moderating requirements uncertainty. Information & Management 45, 203–210 (2008)
5. Hwang, W., Salvendy, G.: Number of People Required for Usability Evaluation: The 10±2 Rule. Communications of the ACM 53, 130–133 (2010)
6. Iivari, J., Iivari, N.: Varieties of User-Centeredness. In: Proceedings of the 39th Hawaii Conference on System Sciences. IEEE, Los Alamitos (2006)
7. Kujala, S.: Effective user involvement in product development by improving the analysis of user needs. Behaviour & Information Technology 27, 457–473 (2008)
8. Munk-Madsen, A.: Classifying IS Project Problems: An Essay on Meta-Theory, Dept. of Computer Science. Aalborg University, Denmark (2000-2006)
9. Nielsen, J.: Estimating the number of subjects needed for a thinking aloud test. Int. Journal of Human-Computer Studies 41, 395–397 (1994)
10. Polish Platform for Homeland Security, http://www.ppbw.pl/en/p_strona_glowna.html
11. Principles behind the Agile Manifesto (2001), http://agilemanifesto.org/principles.html
12. Puppe, F.: Systematic Introduction to Expert Systems. Springer, Heidelberg (1993)
13. Subramanyam, R., Weisstein, F.L., Krishnan, M.S.: User Participation in Software Development Projects. Communications of the ACM 53, 137–141 (2010)