Identifying Process Problems with the SAWO Functional Defect Classification Scheme

Tanja Toroi, Anu Raninen, Hannu Vainio, and Lauri Väätäinen

University of Eastern Finland, School of Computing, Kuopio, Finland
{tanja.toroi,anu.raninen,hannu.vainio,lauri.vaatainen}@uef.fi

Abstract. In this paper we present the first official version of SAWO, a functional defect classification scheme developed to enable the usage of defect data for Software Process Improvement (SPI) purposes. Defect data is one of the most important, although nowadays perhaps least discussed management information sources for SPI decisions. Applying our scheme, defects can be classified with accuracy needed to generate practical and targeted process improvement suggestions. The SAWO scheme classifies defects on three levels. On the first level, the focus is on software defects in general. The second level focuses on functional defects and the third level brings more detail to the functional level. Further, we present the validation results of SAWO with three software companies' defect data consisting of 6363 defects.

Keywords: SAWO defect classification scheme, defect data analysis, process improvement.

1 Introduction

Defect data is rarely utilized in Software Process Improvement (SPI) [1]. Defect data is neglected even though defect data analysis is recognized as an effective and important approach for process improvement [2]. In addition, defect analysis, tracking and removing the major sources of defects offer the greatest short-term potential for improvements [3]. Previous research has shown that the defect classifications can be used to identify product and process problems [4] and to improve the testing and/or inspection activities [5]. In this paper we present a way of making defect data applicable as an input for practical process improvement.

We've developed the SAWO defect classification scheme to enable defect data analysis based process improvement. SAWO is a functional defect classification scheme, which classifies defects on three levels. The first level classifies defects at a general level. On the second level, the focus is on functional defects and the third level deepens the functional level, i.e. defects in control flow, processing, calculations and/or functional logic. In addition, missing duplicated, or overlapped features are considered functional defects. The SAWO scheme is based on an analysis of 11879 defects, see [6, 7]. Initial version of SAWO has been first published in [7]. The scheme presented in this paper is an improved version and is considered here as the official version. The representation style and descriptions of defect types have been evolved based on feedback from our case organizations. In addition, the scheme has now been validated with three software companies defect data.

In this paper we show how the SAWO scheme can be applied in software process improvement. The results presented are based on applying the SAWO scheme for 6363 defects of three software companies.

Our preliminary analysis [6] showed that defect distributions are surprisingly similar in different case organizations. The preliminary analysis was based on defect classification derived and combined from existing defect classification schemes [8, 9]. In addition, we noticed that 65% of the defects stored in the companies' databases are functional defects [6]. Hence, it appeared that the existing classification schemes [8, 9] were not detailed enough to produce meaningful results to be applied in SPI. To be able to truly benefit from the defect data, a more detailed defect classification was needed. However, such defect classification schemes are practically nonexistent in literature. The SAWO scheme was created to fill this gap.

Applying the SAWO classification scheme has been encouraging: the result of the classification is detailed enough to enable the identification of practical inputs for process improvement.

In comparison to our initial literature based scheme [6], the result of the new SAWO scheme is able to show the differences in software companies. The defect distributions reported in [6] were fairly similar as opposed to those generated applying the SAWO scheme. The differences in software companies defect data seem to lie on the functional level. The result of SAWO would appear to show the real problematic areas of the process in question. Hence, it would appear that by applying the SAWO scheme on the functional defect level we are able to learn something from the defects that we have not been able to make visible before.

The overall structure of this paper is: Research setting is described in section 2. In section 3, we present our defect classification scheme, SAWO. Section 4 describes the results of applying the SAWO scheme. Section 5 presents process improvement suggestions based on SAWO defect data analysis. The results are discussed in section 6 and section 7 provides the conclusion.

2 Research Setting

In this section we present the research problem and the case organizations. The research method used was case study [10].

2.1 Research Problem

It is shown that software defect data is one of the most important available management information sources for software process improvement decisions [3]. We conducted a preliminary study in spring in 2011 to find out what the most common defect types are and how this information can be used in process improvement [6]. The study was conducted using defect data from three software companies consisting of 11879 defects in total. Based on the results of the preliminary study it was noticed that further research was needed. The defect classification scheme applied was too general to provide detailed information to be applied for process improvement purposes.

In order to utilize the defect data for process improvement purposes, defects had to be understood in more detail. Hence, the research problem of the study is: Does the SAWO defect classification scheme provide practical inputs for software process improvement?

2.2 The Case Organizations

In this section we describe the case organizations of the study. The case organizations are presented in table 1.

Table 1. Case organizations

	Company A	**Company B**	**Company C**
Market	Farming	Metal industry	Telecommunications
Size	Small	Large	Medium
Employees in development / system testing	9 / 4-6	24 / 2	30 / 1
Country	Finland	Multinational	Finland
Defect tracking system	Mantis	Jira	HP Quality Center
Language of the defect descriptions	Finnish	English	Finnish
Analyzed defects, in total	2938	554	2871
Functional defects	1826	185	1788
Coding language	Delphi	C#	Java

The case organizations in this study are dissimilar in many ways. For example, they produce software products for very different business domains. In addition, the companies differ in size. The study presented in this paper consists of 6363 defects in total, 2938 defects in company A, 554 defects in company B and 2871 defects in company C. The amount of defects is fewer than in our earlier studies [6, 7] because in this study we chose the newest defects (from 2008 to 2011) from the defect databases. The total amount of functional defects is 3799, 1826 defects in company A, 185 in company B and 1788 in company C.

3 The SAWO Defect Classification Scheme

In this section we present the SAWO defect classification scheme. The initial version of the scheme was developed in 2012 and is presented in [7]. In this paper we present an improved version of the scheme. In addition, the scheme has been validated with two new companies' defect data. The main difference to the one presented in [7] is that the scheme classifies defects on three levels presented in three different tables. This makes the scheme easier to understand and apply. In addition to improving the representation style, we have clarified the defect descriptions and changed one defect type's title (i.e. 6.6 Printing) and description to be more descriptive.

3.1 The SAWO Defect Classification Scheme Explained

The SAWO defect classification scheme classifies defects on three levels. The first level of the scheme is a combination of the schemes by Beizer [8] and Humphrey [9] and it divides defects in ten types. The second level of the scheme is applied to classify the functional defects, i.e. defect type Function is divided into sub-types. The second level adapts Beizer's functional defect classification. It consists of six defect types. The third level of the scheme classifies Feature/Function correctness defects in more detail. The third level has been developed and validated with our case organizations. There are six defect types in the third level. The structure of the SAWO defect classification scheme is depicted in figure 1.

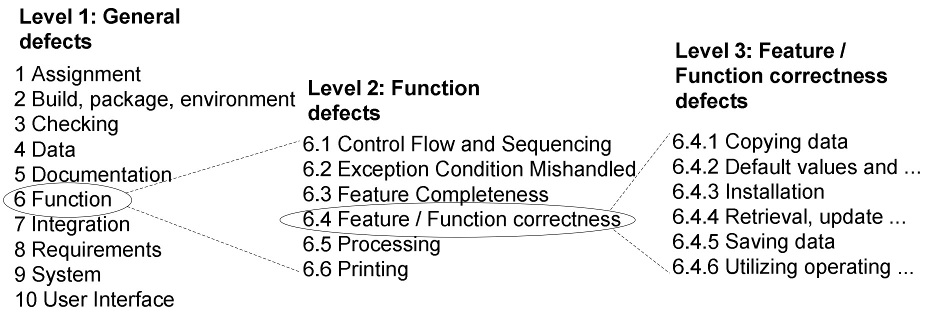
Level 1: General defects
1 Assignment
2 Build, package, environment
3 Checking
4 Data
5 Documentation
6 Function
7 Integration
8 Requirements
9 System
10 User Interface
Level 2: Function defects
6.1 Control Flow and Sequencing
6.2 Exception Condition Mishandled
6.3 Feature Completeness
6.4 Feature / Function correctness
6.5 Processing
6.6 Printing
Level 3: Feature / Function correctness defects
6.4.1 Copying data
6.4.2 Default values and ...
6.4.3 Installation
6.4.4 Retrieval, update ...
6.4.5 Saving data
6.4.6 Utilizing operating ...

Fig. 1. The structure of the SAWO defect classification scheme

The three levels of the SAWO defect classification scheme are described in detail in tables 2-4, levels 1-3 respectively.

Table 2. SAWO defect classification scheme, level 1

ID	Defect Type	Description
1	Assignment	Declaration, duplicate names, scope, limits
2	Build, package, environment	Change management, library, version control
3	Checking	Error messages, inadequate checks
4	Data	Database structure and content
5	Documentation	Comments and messages
6	Function	Logic, pointers, loops, recursion, computation, function defects
7	Integration	Integration problems, component interface errors
8	Requirements	Misunderstood customer requirements
9	System	Configuration, timing, memory, hardware
10	User Interface	Procedure calls and references, I/O, user formats

Table 3. SAWO defect classification scheme, level 2

ID	Defect type	Description
6.1	Control Flow and Sequencing	Defects in control flow (e.g. path left out, unreachable code, improper nesting loops, loop termination criteria incorrect).
6.2	Exception Condition Mishandled	Defects in exception handling. Exception conditions are not correctly handled, wrong exception-handling mechanisms used.
6.3	Feature Completeness	Feature is executed inadequately. Missing feature, duplicated, overlapped feature.
6.4	Feature/Function correctness	Implementation of feature / function is incorrect. Feature not understood, feature interaction.
6.5	Processing	Defects in processing, calculations. Algorithmic, arithmetic expressions, initialization, cleanup, precision.
6.6	Printing	User messages are incorrect. Printing on screen / paper, defects in reports.

Table 4. SAWO defect classification scheme, level 3

6.4	Defect type	Description
6.4.1	Copying data	Defects in copying data between systems / databases. Difficulties in making backups.
6.4.2	Default values and initial states	Defects in programs default values e.g. programs default selection causes failures in software.
6.4.3	Installation	Problems during installation of the developed program.
6.4.4	Retrieval, update and removal of data	Relates to refreshing the screen. Data inputs from user doesn't update properly to the screen.
6.4.5	Saving data	Data doesn't save to system. Data can't be saved when it should be possible or it can be saved when it shouldn't be able.
6.4.6	Utilizing operating system services	Problems related to operating systems (e.g. Windows), e.g. mouse commands, tab order, and other features provided by the OS.

4 Applying The SAWO Scheme

In this section we present the results of the defect classification applying the SAWO scheme. Data from the years 2008-2011 is classified.

4.1 SAWO Defect Distribution, Level 1

We applied the SAWO defect classification scheme level 1, presented in table 2, for three software companies defect data consisting of 6363 defects. The result of the defect classification is presented in figure 2.

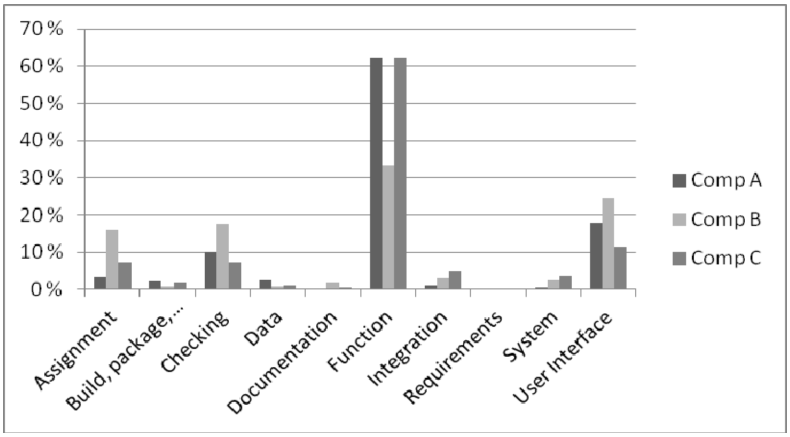

Fig. 2. SAWO defect distribution level 1

From figure 2, it can be seen that by far the most common defect type in every company is “Function” defect type (total of 3799, 59.7%). The second most common defect types are “User Interface” (total of 991 defects, 15.6%), and “Checking” (total of 601 defects, 9.4%). “Requirements” (total of 14 defects, 0.2%) and “Documentation” (total of 27 defects, 0.4%) are the rarest defect types.

4.2 SAWO Defect Distribution, Level 2

Despite the SAWO Level 1 made the problem points of the software processes visible, it is reasonable to study functional defects in more detail due to their large amount. Hence, we applied the SAWO defect classification scheme level 2 (see table 3) to classify the defects in a more precise manner. The classification was conducted for the functional defect data consisting of 3799 functional defects (see section 4.1). The defect distribution is presented in figure 3.

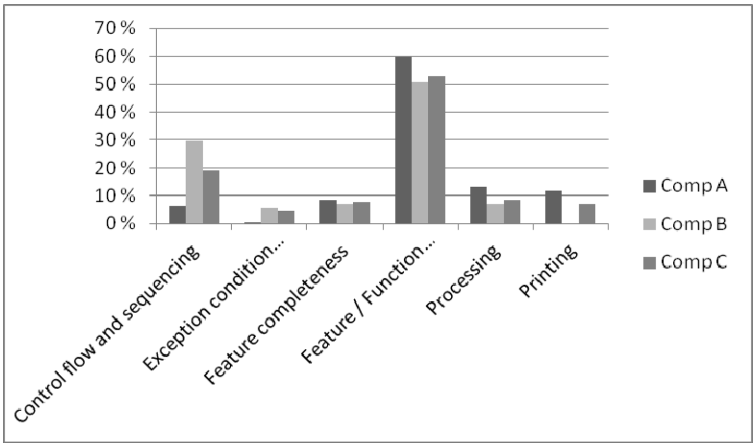

Fig. 3. SAWO defect distribution level 2 (Functional defects)

From figure 3, it can be seen that the defect type “Feature/Function correctness” is remarkably more common than the other defect types. “Feature/Function correctness” includes 56.3% of the defects (2139 defects in total). The second most common defect types are “Control flow and sequencing” (total of 512 defects, 13.5%) and “Processing” (total of 406 defects, 10.7%). “Exception condition mishandled” is the most uncommon defect type (2.6% of the defects).

4.3 SAWO Defect Distribution, Level 3

The case organizations wanted to find out what the “Feature/Function correctness” issues are, in order to improve their development and testing processes. In order to figure this out, we applied SAWO level 3 (see table 4) for the “Feature/Function correctness” defects, 2139 defects in total. The results can be seen in figure 4.

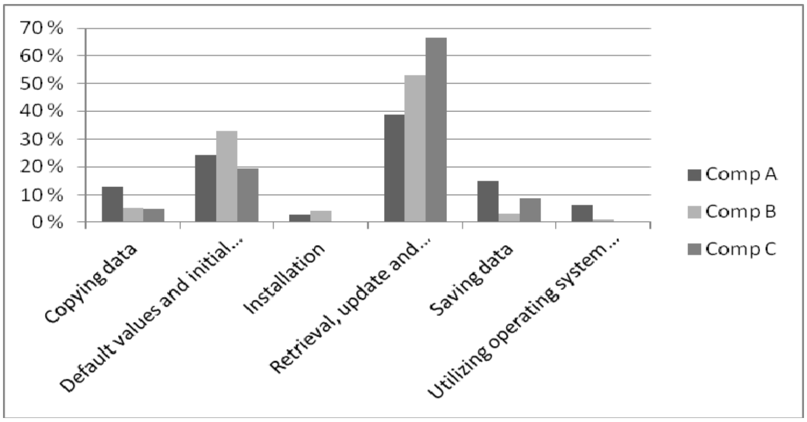

Fig. 4. SAWO defect distribution level 3 (Feature/Function correctness defects)

The most common defect type is “Retrieval, update and removal of data” (total of 1107 defects, 51.8%). The second most common defect types are “Default values and initial states” (total of 480 of defects, 22.4%) and “Saving data” (total of 249 of defects, 11.6%). “Installation” is the most uncommon defect type (only 1.6% of the defects).

5 Process Improvement Suggestions Based on SAWO

Applying SAWO, it was possible to provide practical and targeted improvement suggestions for the case organizations. The defect distributions of the case organizations vary, most likely because of the special characteristics of the organizations. Hence, the improvement suggestions are company-specific and offer solutions to current problems.

Based on the defect data classification, the case organizations are able to see their software engineering problem points from the defect point of view. The classification shows that the most common defects in all the case organizations are functional defects. In addition, user interface defects and checking defects are also common. When the functional defects are studied in more detail, it can be noticed that the most troublesome functional issues are related to retrieving, updating and removing data, default values of the variables and forms, and control flow and sequencing. The most common defect types of the case organizations are summarized in table 5. The amount of Function defects and Feature/Function correctness defects have been divided into sub defect types in SAWO levels 2 and 3.

Table 5. The summary of the most common defect types of all the defects in each company

	Company A	**Company B**	**Company C**
1	6 - Function	6 - Function	6 - Function
2	6.4 - Feature/ Function correctness	10 - User interface	6.4 - Feature/Function correctness
3	10 - User interface	3 - Checking	6.4.4 - Retrieval, update and removal of data
4	6.4.4 - Retrieval, update and removal of data	6.4 - Feature/ Function correctness	6.1 - Control flow and sequencing
5	3 - Checking	1 - Assignment	10 - User interface
6	6.4.2 - Default values and initial states	6.1 - Control flow and sequencing	3 - Checking

Based on the results of the SAWO defect classification detailed improvement suggestions were given to each case organization. The suggestions related to the most common defect types are presented in table 6.

Table 6. Improvement suggestions related to the most common defect types

SAWO level 1		Company		
Defect type	**Improvement suggestions**	**A**	**B**	**C**
Function	See SAWO level 2.	x	x	x
Feature/ Function correctness	See SAWO level 3.	x	x	x
User Interface	Conduct more thorough **user interface design and testing**. In addition, create **instructions for the user interface design,** review user interfaces, and provide developers with **check lists** for the important issues which must always be checked. Further, make use of **product family engineering approach** in which new interfaces/systems reuse common product family components.	x	x	x
Checking	**Error messages** should be more **accurate**. It must be checked in user interface testing if the error messages are relevant in all the situations. Use **uniform error messages** throughout the system.	x	x	x
Assignment	Conduct **code inspections** in order to reduce the amount of bugs due to carelessness. **Review also the requirements** and/or the specifications where the values of the parameters are derived from (see below Default values and initial states).		x	
SAWO level 2 (Function defects)		**A**	**B**	**C**
Control flow and sequencing	Conduct **code inspection**. Code coverage analysis and control flow tracking can be assisted with an **automated tool**. Complex systems should be **documented more properly**. Documentation helps in recalling the functionality when new version will be developed or new developers take charge of the project.		x	x
SAWO level 3 (Feature/Function correctness defects)		**A**	**B**	**C**
Retrieval, update and removal of data	**Stress the importance of unit testing**. Data retrieval, updating and deletion defects could be detected already in the unit testing phase during which it would be cheaper to fix them. **Conduct pair programming**. Previous research has found that programmers working in pairs produce fewer bugs, than programmers working alone [11]. Establish more precise **naming practice of the fields of the database** and improve the **database design process**. Check the content of the database after insert/delete/update operations.	x		x

Table 6. (*continued*)

Default values and initial states	Conduct **code inspections and pair programming**. **Take test automation in use.** Use automation tool for the management of the parameters. **Initial states and conditions must be defined in design phase** and they must be reviewed. If the **values of the parameters** are received outside the company (e.g. by legislation) they must be **transformed electronically** to the phase in which they are used (e.g. design, coding, inspection).	x		

6 Discussion

Our preliminary analysis with three software companies' defect data (11879 defects) showed that 65% of the defects were functional defects [6]. Even though the practical improvement suggestions could be given based on the preliminary analysis, a more detailed classification was needed because of the huge mass of functional defects. We wanted to find out what the real problems are behind these functional defects in order to enable process improvement based on defect data. Defect data is one of the most important available management information sources for SPI decisions [3]. Yet, defect data is rarely utilized properly in process improvement efforts [1].

In addition to the SAWO scheme, there are not many functional defect classification schemes available in the literature. The example of one can be found in [8]. Further, applying the defect taxonomies is somewhat challenging because the schemes are not detailed enough to enable the identification of tangible targets for process improvement.

The SAWO defect classification scheme provides a more detailed classification on the functional level than any of its predecessors. It has been developed to more accurately identify the problem areas of software process and to help software companies allocate improvement resources to justifiable targets. The first two levels, general level and the first functional level of the SAWO scheme are based on those presented by Humphrey and Beizer [9, 8]. We developed the third level of the SAWO based on our defect data analyses [6, 7]. We have applied our scheme for three software company's defect data and learned that the three levels of classification show differences between the defect distributions of the case organizations.

Based on the classification results presented, practical process improvement suggestions could be provided to the case organizations. The defect distributions clearly show that certain areas are more error prone than others. Hence, it is reasonable to allocate improvement resources to those areas. For example, all of the case organizations were suggested to conduct code inspections. This was suggested due to the fairly large amount of Assignment (6.1%), Control Flow and Sequencing (8.0%), and Default values and initial states (7.5%) defects. These defect types suggest that the programmers are a bit careless and might benefit from inspections where the quality of the code was monitored.

Further, there were also company-specific differences in the defect distributions. Hence, there were also differences in the improvement suggestions provided. For example, company A was suggested to conduct more thorough user interface design and testing and create instructions for the user interface design due to the large amount of User Interface defects (18.0%). Company B was suggested to check in user interface testing if the error messages are relevant in all situations. In addition, they were suggested to use uniform error messages throughout the system. This was suggested due to large amount of Checking defects (17.7%). Further, company C was suggested to improve the database design process and pay attention to the database naming conventions. The suggestion is related to the Retrieval, update and removal of data defects (22.0%).

There may be some limitations to this study. Firstly, we may have misinterpreted some defect descriptions due to lack of domain knowledge and language of the defect descriptions. In one case company the language of the defect descriptions is English while others used Finnish. English text of a non-native English-speaker may sometimes be difficult to understand and may cause misunderstandings. However, the amount of the defects in this study is so large that single misclassified defects do not influence the reliability of the results. Secondly, we do not know yet if the proposed improvement suggestions improve processes in the long run. However, it appears that case companies gained from the improvement suggestions: two companies initiated unit testing improvements and one company started to implement a test automation tool.

Based on the results of the study, SAWO would appear to provide practical input for SPI efforts by making the most problematic areas of software products visible. Further, SAWO enables the utilization of defect data, one of the most important data sources on which to base SPI decisions on. The three levels of SAWO enable us to understand software defects on a level of detail that appears not to have been possible with the existing defect classification schemes. SAWO is especially beneficial at the early stages of SPI projects when visible results are needed quickly to maintain motivation for SPI [12].

7 Conclusion

In this paper we present how defect classification can be applied as an input for software process improvement. The main contribution of the paper is the first official version of the SAWO classification scheme, initial version of which is first introduced in [7]. In addition, the SAWO scheme is validated via classifying three software companies' defect data. Further, based on the results of the defect data classification practical, company-specific process improvement suggestions are provided. Applying the SAWO scheme, the problem areas of software development and testing processes can be identified. Further, process improvement actions can be targeted to the real problem areas identified based on the defect data classification. The SAWO scheme enables software companies to utilize defect data, one of their most important, and nowadays perhaps least used management information sources for SPI decisions.

Acknowledgements. This research was funded by the Finnish Funding Agency for Technology and Innovation (Tekes) with grant 70030/10 for METRI (Metrics Based Failure Prevention in Software Engineering) project.

References

1. Fredericks, M., Basili, V.: Using Defect Tracking and Analysis to Improve Software Quality. In: DoD Data & Analysis Center for Software, DACS (1998)
2. Vinter, O.: Experience-Based Approaches to Process Improvement. In: Proceedings of the 13th International Software Quality Week, San Francisco, USA (2000)
3. Grady, R.B.: Practical software metrics for project management and process improvement. Prentice Hall, New Jersey (1992)
4. Bhandari, I., Halliday, M.J., Chaar, J., Chillarege, R., Jones, K., Atkinson, J.S., Lepori-Costello, C., Jasper, P.Y., Tarver, E.D., Lewis, C.C., Yonezawa, M.: In-process improvement through defect data interpretation. IBM Systems Journal 33(1), 182–214 (1994)
5. Freimut, B.: Developing and using defect classification schemes. Fraunhofer IESE IESE-Report No, 72 (2001)
6. Raninen, A., Toroi, T., Vainio, H., Ahonen, J.J.: Defect Data Analysis as Input for Software Process Improvement. In: Dieste, O., Jedlitschka, A., Juristo, N. (eds.) PROFES 2012. LNCS, vol. 7343, pp. 3–16. Springer, Heidelberg (2012)
7. Toroi, T., Raninen, A., Vainio, H.: Using Functional Defect Analysis as an Input for Software Process Improvement: Initial Results. Communications in Computer and Information Science 301, 181–192 (2012)
8. Beizer, B.: Software Testing Techniques. International Thomson Computer Press (1990)
9. Humphrey, W.: A discipline for software engineering. Addison-Wesley (2007)
10. Yin, R.K.: Case study research: Design and methods. Sage publications, INC. (2009)
11. Cockburn, A.: Williams. L.: The Costs and Benefits of Pair Programming. In: Succi, G., Marchesi, M. (eds.) Extreme Programming Examined, pp. 223–243 (2001)
12. Zahran, S.: Software process improvement: practical guidelines for business success. Addison-Wesley, Reading (1998)