Building Dedicated Project Management Process Basing on Historical Experience

Cezary Orłowski[1(✉)], Tomasz Deręgowski[2], Miłosz Kurzawski[3], and Artur Ziółkowski[1]

[1] Department of Information Technology Management, WSB University, Gdańsk, Poland
{corlowski, aziolkowski}@wsb.gda.pl
[2] Acxiom Corporation, Gdańsk, Poland
Tomasz.Deregowski@acxiom.com
[3] Faculty of Management and Economics, Gdańsk University of Technology, Gdańsk, Poland
mkurzawski@zie.pg.gda.pl

Abstract. Project Management Process used to manage IT project could be a key aspect of project success. Existing knowledge does not provide a method, which enables IT Organizations to choose Project Management methodology and processes, which would be adjusted to their unique needs. As a result, IT Organization use processes which are not tailored to their specific and do not meet their basic needs. This paper is an attempt to fill this gap. It describes a method for selecting management methodologies, processes and engineering practices most adequate to project characteristic. Choices are made on the basis of organization's historical experience. Bespoken Project Management process covers technical and non-technical aspects of software development and is adjusted to unique project challenges and needs. Created process is a hybrid based on CMMI for Development Model, it derives from different sources, uses elements of waterfall and Agile approaches, different engineering practices and process improvement methods.

Keywords: Project Management · Standards · Methodologies · Waterfall approach · Agile · Case Based Reasoning

1 Introduction

Contemporary IT Organizations need to implement and support wide range of projects. Very often IT organization is responsible for providing simple services such as maintenance of existing solutions, where project activities are limited to bug fixing and making minor changes. At the same time the same IT organization actively develops existing systems and products, integrates products and system provided by external companies or work on rewriting existing solutions using new, more efficient technologies. In addition to project types listed above, the same IT organization is often engaged in developing new, unique products for internal and external clients.

DOI: 10.1007/978-3-319-54430-4_76

Not only the type of the projects can differ as a part of the same IT organization. Even if projects are of the same type, they parameters such as budget, schedule, requirements, quality expectations, project risks, used technology and team structure may vary. These parameters are also crucial factors which shape unique project characteristic. Each IT organization and its members have also unique experience resulting from previous projects. All these different factors should be taken into consideration when choosing the way in which project will be managed.

There is no single methodology which allows effectively manage a wide range of unique IT projects. Authors of this paper see the need to expand Agile mindset and use Agile approach not only to develop projects but also to develop processes used to manage these projects. The way project management process is build could be in many ways similar to the process of building a software. Project Management process should constantly adapt to changing requirements and environment. The choice of Project Management Methodology and Processes should be based on project characteristic and historical experience. All these principles are derived from Agile approach.

The shape of Project Management Process used to manage IT project is very often a key to project success. Unfortunately, existing knowledge does not provide a method, which enables IT organizations to choose proper project management methodology, adjusted to unique project specific.

This paper is an attempt to fill this gap. It describes a method of selecting management methodologies, processes and engineering practices most adequate to project needs. Decisions are based on organization's historical experience. Knowledge of processes, methodologies and tools which worked for similar projects in the past, helps design optimal process for newly started projects. The goal is to build bespoken Project Management process, which covers all different aspects of software development (technical and non-technical), adjusted to unique project challenges and needs. Newly created process is from definition a hybrid, it derives from different sources, uses elements of waterfall and Agile methodologies, different engineering practices and process improvement methodologies.

Building dedicated, hybrid Project Management processes, adjusted to the needs of particular project, is an important part of Agile Transformation. Agile Transformation is understood in the context of this paper as applying Agile mindset to the process of creation and adaptation of project management processes and methodologies. Agile mindset is understood as constant adaptation to changing requirements, circumstances and environment. Organizations that haven't yet begun Agile Transformation Process use same approach to manage all different types of projects or don't use any methodology at all.

Organization which has begun Agile Transformation starts adapting project management processes to the needs of particular project. Only newly started projects are managed using dedicated, bespoken project management process, but organization is aware of the need to use dedicated solutions and starts collecting metrics on historical projects and projects in progress. It also starts improving ongoing projects and adapts their management processes to the specific needs of managed project.

Organization which has completed Agile Transformation is ready to implement various projects with different specific using dedicated, bespoken Project Management process. The course of Agile Transformation is summarized in the figure below (Fig. 1).

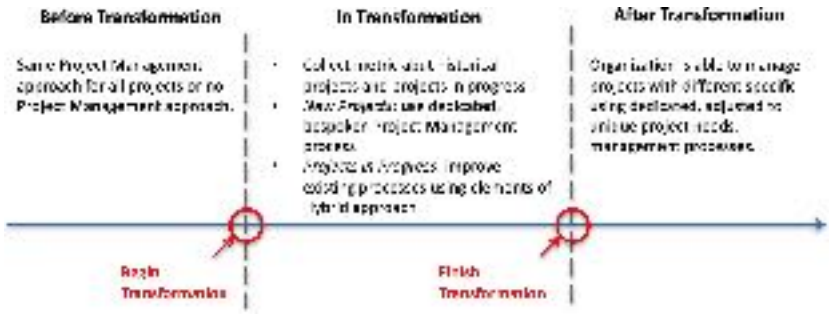

Fig. 1. Agile Transformation Process

A problem associated to lack of tools and methods, which allow choosing adequate, processes and methodologies, adjusted to unique needs of the project was described in one of earlier publications "Model for building Project Management processes as a way of increasing organization readiness for Agile transformation" (Deręgowski Tomasz 2016). Proposed in this article Model for Designing Hybrid Management Processes (DHMP) defines concept of building custom project management processes, which are adjusted to unique project needs. Processes are based on results of project, client and delivery organization analysis and they try to fill all the gaps in existing processes and address specific needs resulting from project characteristic.

This paper is a continuation of earlier research. It concentrates on the process of creating hybrid, bespoken project management process based on organization's historical experience. It also describes experience with an introduction of hybrid approach and how it impacted project performance and project result. Described in this paper method for defining project management process based on characteristic of the project and historical experience is an integral part of Model for Designing Hybrid Management Processes (DHMP) defined in previous publications.

2 Case Based Reasoning as a Tool for Constructing Project Management Process

Described in this paper method for selecting most appropriate, adjusted to unique project needs, Project Management methodology and management processes is based on Case Based Reasoning (CBR) method (Fig. 2).

CBR is a method for solving new problems based on the solutions of similar past cases. It has been chosen for two main reasons. Firstly, CBR method is based on team's practical knowledge and previous experience, not on theoretical models or general knowledge. CBR knowledge-based systems retrieve and reuse solutions that have

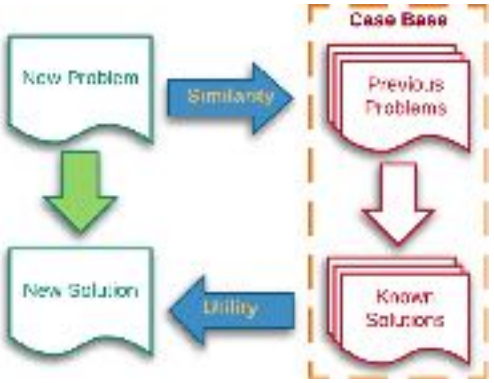

Fig. 2. Case Based Reasoning method

worked for similar cases in the past. A new problem is solved by finding a similar past cases and reusing them in the new problem situation.

Secondly, CBR enables incremental learning. New experience is retained each time a problem has been solved and is added to Case Base so it could be reused to solve future problems.

A widely accepted model of the CBR processing is the CBR cycle proposed by Aamondt and Plaza (1994) (see Fig. 3). There are four stages in CBR processing defined in this model: Retrieve, Reuse, Revise and Retain.

At Retrieve Stage CBR system looks for historical cases relevant to target problem which is to be solved. Similar cases are extracted from the system and transferred for further processing. Relevant cases are to some extend similar to target problem: most of their key aspects and parameters are similar to key aspects and parameters of target

Fig. 3. Case Based Reasoning cycle

case. At Reuse Stage experience described in the corresponding, historical cases, which were retrieved in the previous step of CBR processing, is used to solve target problem. In most cases experience and knowledge contained in historical cases need to be adapted to the needs of target case. During Reuse Stage the solution for target problem is defined. In the next step of CBR processing called Revise Stage, applicability of defined in previous step solution is revised and evaluated in practice. At this stage it is checked how solution created in Reuse stage impacted target case and if it helped with solving it. Last, fourth step of CBR processing is Retain. During Retain Stage solution generated and revised in previous stages is archived in CBR system so it could be reused to solve future problems.

Description defines the problem which was solved (past Cases stored in Case Base) or is about to be solved. It defines not only the problem, but also state of the world when the problem occurred. Problem discussed in this paper concerns building dedicated, adjusted to unique project needs, Project Management processes which would maximize the probability of project success. Project is characterized by seven factors, each of them is defined by the level of its complexity. Parameters used to describe project characteristic are budget, requirements, schedule, risk, technology and project team.

The last part of case definition determines outcome, effect of applying described solution to given problem. What was the reaction of the system after the solution was applied, if the problem was totally or partially solved. In case of problem discussed in the paper outcome is understand as project result. Project result can be defined in many different ways. Traditionally project success is defined as delivering project on time, within the budget, with all features which were defined at the beginning of the project. Project could be also considered as success if particular level of client satisfaction was achieved. The question of how to define the success of a project is not relevant to this paper and will not be discussed.

In order to be available for future reuse, cases are organized in a Case Base. Case Base contains of collection of historical cases which were solved in the past and for which the solution and outcome are known. Case Base contains organization's knowledge which result from its previous experience. It is also often the source of knowledge about business domain in which organization operates. To draw useful conclusions, Case Base needs to be initialized with set of past cases. It also contains a mechanism which enables effective comparison of historical cases and drawing useful conclusions.

The usage of theoretical model defined in this chapter as well as findings and conclusions resulting from the application of this model in production environment, are described in the following part of this paper.

3 Design and Implementation of Project Management Process Based on Historical Experience

Pilot study was carried out in a software development company. The company has more than 4,000 employees, distributed among different localizations in United States, South America, Australia, Asia and Europe, including Poland. Solutions and products

provided by company are used to support multichannel marketing (Arikan 2008), data mining (Witten 2011) and Big Data (Marz 2015).

The main goal of pilot study was to check if it is possible to improve team's performance and probability of project success by using dedicated, bespoken, project management processes. It was also important to prove, that such a process can be created on the basis of company's historical project experience.

Prior to the pilot study Case Base was initialized with information about eleven past projects which were carried out by described organization. Survey developed and described in "Building Project and Project Team Characteristic for creating Hybrid Management Process" (Deręgowski Tomasz 2016) paper was used to build characteristic of each of analyzed projects. Survey was used to gather information about methodologies and project management processes used to manage each of examined projects. Because all surveyed projects have been completed, it was possible to collect information about projects' results and their success or failure factors which impacted project's overall result.

In order to verify model for building project management processes based on historical experience one of the projects carried out by described organization has been chosen. Project which has been chosen for the model verification was carried out for one of the biggest and most important clients of European branch of the company. Cooperation with this client lasted for three years, delivery team consisted of 15 associates. Authors of pilot study decided to choose this particular project because of repeatability of project environment. Every four months project team delivered functional release which contained of similar number of Change Requests (CR). Each functional release was provided by the same project team, used the same technologies and had similar scope. Repeatability of experiment environment allowed to measure the impact of each change on project overall performance. Similarity between functional releases allowed to minimize the impact of external factors.

3.1 Retrieving from Case Base Projects with Similar Characteristic

In the first step of CBR processing, cases which are similar to the test case for which project management process is to be designed, are extracted from Case Base. There are various different approaches to determine the level of similarity between two projects. One of them are numeric measures. In context of described experiment, to describe the characteristic of the project, seven key parameters of the project were identified. Each parameter describes different aspect of the project. Projects aspects that are used to determine the degree of similarity between two different cases are budget, schedule, requirements, quality, risk, technology and project team.

Each parameter can be one of three values: Low Complexity (1), Medium Complexity (2) and High Complexity (3). The comparison result of two projects is the result of a comparison of individual parameters that make up the characteristics of the project. There are three possible results of parameters comparison:

– Parameters are equal; both of them have either high, medium or low complexity. When both parameters are equal, the result of the comparison is 1.

– Parameters are similar; it means that there are not equal, but at the same time they are also not significantly different. Two parameters are similar when one of them have low complexity and the second have medium complexity, or when one of the parameters have medium complexity when complexity of the second parameter is high. When two parameters are similar, the result of the comparison is 0.5.
– Parameters are different when their complexity is significantly different. It means that one parameter has low and second high complexity. The result of comparison of two parameters which are different is 0.

The overall result of comparison of individual parameters defines the level of similarity between two projects. Dedicated weights can be assigned to each of the parameters. The higher the weigh, the greater the impact of particular parameter on the summary result of comparison. The degree of similarity between two projects is brought to a value which is the average similarity of individual parameters.

For a numeric representation of similarity, where each aspect which makes up the total result has attribute-value representation, a similarity measure is the generalized Hamming measure. It combines the importance of each attribute of project characteristic with its local similarity value. Global similarity for two projects P_1 and P_2 is the sum of local similarities $sim_j(P_1, P_2)$ multiplied by relevance factor w_j.

$$sim(P_1, P_2) = \sum_{j=0}^{n} w_j \cdot sim_j(P_1, P_2)$$

sim_j - local similarity for aspect j
w_j - relevance (weight) of aspect j of the project
n - number of aspects describing project.

In the context of described study four projects with characteristic similar to the characteristic of Test Project P_T were retrieved from Case Base. Authors of this study assumed, that similar project is a project for which the level of similarity is greater than 66%. The degree of similarity differed from project to project. The characteristic of first project (P_A) was identical to the characteristic of Test Project P_T and the degree of similarity was 100%. The degree of similarity for two next projects (P_B and P_C) was 69%. The degree of similarity for last projects (P_D) was 75%.

Characteristic of each project and detailed results of comparison are described in the table below (Table 1).

The experience gained during the implementation of projects P_A, P_B, P_C and P_D have been used to design Project Management Process for project P_T which was about to begin.

3.2 Reuse of Proven Management Solutions

In the course of the survey, interviewed expert listed Process Areas which had the most significant impact on project result. List of processes was based on Processes Areas

Table 1. The result of comparison of analogical projects

Process Area	w_j	Test Proj. P_T	Project P_A		Project P_B		Project P_C		Project P_D	
			Comp.	sim_j	Comp.	sim_j	Comp.	sim_j	Comp.	sim_j
Budget	0.125	Med.	Med.	1,0	Low	0,5	Med.	1,0	Med.	1,0
Schedule	0.125	Med.	Med.	1,0	Med.	1,0	Med.	1,0	High	0,5
Requirements	0.125	Med.	Med.	1,0	Med.	1,0	High	0,5	High	0,5
Quality	0.125	Low	Low	1,0	Low	1,0	Low	1,0	Low	1,0
Risk	0.125	Med.	Med.	1,0	Med.	1,0	Med.	1,0	Med.	1,0
Technology	0.125	Low	Low	1,0	High	0,0	Low	1,0	Low	1,0
Team Comp.	0.25	Low	Low	1,0	Med.	0,5	High	0,0	Med.	0,5
***sim* (P_T, P_A)**			**100%**		**69%**		**69%**		**75%**	

defined in CMMI for Development model (CMMI-DEV) (Chrissis et al. CMMI for Development: Guidelines for Process Integration and Product Improvement 2011).

CMMI is a process improvement method, which integrates all process and procedures existing in IT Organization into single coherent model and is often used to assess the quality and maturity of processes implemented in IT organisations. CMMI is also used to identify potential gaps and shortcomings of existing processes and procedures. CMMI provides holistic approach, processes described in CMMI model cover full Software Development Lifecycle, from gathering requirements through software development to maintenance and support.

Another important feature of CMMI is the fact that it is method and tool agnostic. It does not concentrate on specific methodologies or tolls. CMMI defines which process should be implemented in an organization and why, but it doesn't define how they should be implemented. Thanks to that it can be used with all types of methodologies, models and techniques, both Agile and traditional.

Within the CMMI-DEV model 22 process areas are defined. Each Process Area describes a key aspects related to particular area of software delivery. CMMI processes are grouped into 4 categories, each category relates to different level of organisation maturity. Seven first Process Areas are related to the Managed Maturity Level. On Managed Level projects are planned and managed in accordance with policy, however the policy might vary between different projects carried out by the same organisation. On Managed Level polices are limited to the most important aspects of Software Development Process such as Configuration Management, Project Planning and Monitoring, Quality Assurance and Requirements Management. For the purpose of this study only first seven Process Areas associated with Managed Maturity Level were used. Analyzing all twenty-two Process Areas would be too time-consuming and would require significant expertise from people participating in pilot study.

As mentioned in previous chapter, four projects had characteristic similar to the characteristic of Test Project P_T. Projects Areas and their impact on the success of each of the projects are listed in Table 2. Two Process Areas which had the biggest impact on project result in all four cases were Configuration Management and Process and Product Quality Assurance. Team responsible for delivering Test Project P_T had already decided to implement Process and Product Quality Assurance Requirements.

Table 2. Project Areas and their impact on project result

	Success factor				
Process Area	Project A	Project B	Project C	Project D	Avg.
Configuration Management	High	Med.	High	High	**High**
Measurement and Analysis	Low	Low	Low	Low	Low
Project Monitoring and Control	Med.	Med.	Med.	Med.	Med.
Project Planning	Med.	Med.	Med.	Med.	Med.
Process and Product Quality Assurance	High	Med.	High	Med.	**High**
Requirements Management	Med.	Med.	Med.	Med.	Med.
Supplier Agreement Management	Low	Med.	High	Low	Med.

Based on survey results it has been decided to also implement additional Process Area Configuration Management. The introduction of this process wasn't originally planned.

Other Process Areas were of minor importance and weren't taken into consideration when designing project management process for Test Project P_T.

Establishing Configuration Management Process requires several actions which are described in detail in CMMI-DEV model. The first action involves identifying and documenting the characteristics of objects which configuration will be managed. In described case Project Team decided to formally manage versions of application code and application environments. To do it effectively, it was important to prevent any unauthorised changes in application and its runtime environments. Every new release of code and change in environment configuration was done through Release Management Team which used a set of tools for release automation and configuration management. It helped to prevent any unauthorised changes made outside of the process, without being tracked, reviewed and formally approved. Before any change was migrated to new environment it was audited and the possibility of negative impact on the system behaviour was eliminated. Right after migration full set of automated regression and performance tests was run. Every issue introduced by new release was immediately discovered, faulty release was withdrawn and issue was immediately fixed. New processes and procedures significantly increased the quality of code and let the team discover potential issues right after they were introduced, when they were easier and cheaper to fix.

To implement requirements related to Configuration Management Process Area, team decided to use practices and tools related to DevOps movement (Kim, 2016) such as Infrastructure as a Code (Humble 2010), Virtualisation (Humble 2010), Continuous Integration (Duvall 2007), Continuous Delivery (Humble 2010) and Test Automation (Bisht 2013). Team also adapted some of Extreme Programming techniques such as Unit Testing (Beck 2004), Code Reviews (Beck 2004) and Static Code Analyze (Chess 2007).

The impact of implementation of new process was measured in the next step of CBR processing called Revise. Results are described in the next chapter.

3.3 The Impact of Changes in Project Management Processes on the Course of the Project

Project Management Process that has been used to manage Test Project P_T was based on CBR model and historical experience. One of the key recommendations concerned the formal implementation of Configuration Management Process Area. Configuration Management processes were adapted to the needs of P_T Project. Impact of changes in project performance which were caused by changes in Project Management process are described in this chapter.

Before establishing proper Configuration Management process four engineers of different specialisation were performing new software releases. Each engineer was responsible for releasing changes in particular technology, all releases were performed manually. Java releases were performed by Java developers, .NET releases by .NET developer and PLSQL releases by DBA. Release Process was coordinated by Release Engineer. After Transformation releases for all three technologies were automated and performed by Release Engineer. Human cost of software release was reduced by 75%.

Implementation of proper, fully automated Configuration Management process had also impact on time needed to perform release. In case of Production Releases, release time was reduced by 90%. In case of System Test and QA release time was reduced by 96%. Number of releases performed each week was increased by 1000%. Impact of implementing proper Configuration Management was summarised in Table 3.

Table 3. Impact of changes in Configuration Management on Release Management performance

	Before change	After change	Improvement rate
Number of engineers performing release	4	1	75%
Time to perform Production Release [days]	5	0,5	90%
Time to perform System Test/QA Release [days]	3	0,125	96%
Number of release per week	1	10	1000%

To test the quality of each release, project team introduced test automation. After every code commit full set of Unit Tests and Smoke Tests were run. In couple of minutes developer who committed the code got feedback from Continuous Integration system if his or her change didn't break the build and if all tests are passing. Even if one of the tests didn't pass, fixing the code and code build was main responsibility of developer.

Automated tests were also a key step in release process. Creation of new software package was preceded by the launch of full set of integration, regression and performance tests. To automate test process team used Robot Framework. This new approach guaranteed that only high quality code which passed all types of tests was migrated to test and production environments. Thanks to test automation, the time needed to perform full set of integration, regression and performance tests was reduced significantly.

Prior to test automation, the team needed five days to run full set of tests. After test automation was introduction application could be tested in four hours. This means that time and effort required to perform full set of tests was reduced by 90%.

To improve the quality of code, project team introduced Code Reviews. Every time developer finished working on new feature, he or she was obligated to organise Code Review. During Code Review meeting all new and updated code was reviewed by other developer and architect. Thanks to this new practice, many potential defects and issues were discovered and eliminated even before code was passed to testers. During Code Review developer also presented the result of Static Code Analyse. SonarQube was a tool used to perform static code analyse. Thanks to SonarQube, in couple of minutest tens of thousands of lines of code were checked against hundreds of rules. It let the team discover and fix many potential security and performance issues long before code was passed for testing.

All these new practices had significant impact on quality of the code and number of defects. When compared to analogical project from the past, significantly less defects were discovered during development, testing and in production. Number of defects found during Development and System Test was reduced by 42%. During User Acceptance Tests 75% less defects were found. The team was particularly proud of the fact, that no defects were found after the system was migrated into production. Changes in the number of defects and their distribution were presented in the Fig. 4.

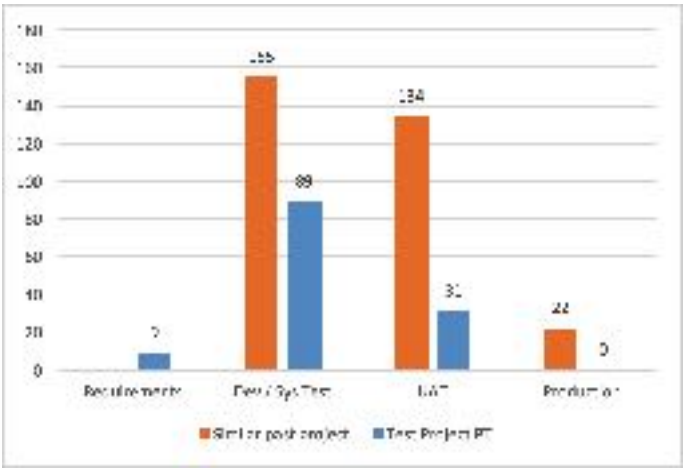

Fig. 4. Reduction in number of defects as the result of Configuration Management Process Area introduction

After the solution has been successfully adapted to the target problem, processes definition and information on its impact on project performance were stored as a new case in Case Base. Experience gained during pilot study has become available for future projects.

The impact of introducing Configuration Management Process Area was so significant, that it was decided to introduce it in all projects conducted by organization in which the pilot study was carried out.

4 Conclusions

Example of Test Project P_T shows that changes in Project Management processes could significantly impact project performance. Thanks to introduction of Configuration Management Process Area, the quality of software was much higher than the quality of software created in analogical projects which didn't manage configuration in the formal manner. It's also visible, that changes which caused such a significant improvement in project performance have been initiated by the usage of CBR model. Therefore, it is worth to continue research on adapting management processes to the specific of project characteristic.

The selection of analytical tools is a matter to be decided. Case Based Reasoning seems to be effective tool when it is used to analyse relatively small and simple set of data. Such solution seems to be sufficient, when IT organisations build recommendations sets based on their own, limited experience. Expanding knowledge base and using experience of other organisations will require more sophisticated and effective analytical tools. Another limitation of CBR method is inability to learn and make predictions on data.

The usage of selected elements of Machine Learning (Kelleher et al. 2015) might be the right direction for future research. Introduction of Pattern Recognition and use of algorithms that can learn from historical data and make data-driven predictions or decisions could bring discussed solution to the next level. Authors of this paper intend to address these issues in their further studies and develop described in this paper method.

References

Aamodt, A., Plaza, E.: Case-based reasoning: foundational issues, methodological variations and system approaches. AI Commun. **17**(1), 39–59 (1994)

Arikan, A.: Multichannel Marketing: Metrics and Methods for On and Offline Success. Sybex, Indianapolis (2008)

Beck, K.A.: Extreme Programming Explained: Embrace Change, 2nd edn. Addison-Wesley, Boston (2004)

Bisht, S.: Robot Framework Test Automation. Packt Publishing, Birmingham (2013)

Chess, B.: Secure Programming with Static Analysis. Addison-Wesley Professional, Boston (2007)

Chrissis, M.B., Konrad, M., Shrum, S.: CMMI for Development: Guidelines for Process Integration and Product Improvement. Addison-Wesley Professional, Boston (2011)

Deręgowski Tomasz, O.C.: Building Project and Project Team Characteristic for creating Hybrid Management Process. KKIO, Wrocław (2016)

Deręgowski Tomasz, O.C.: Model for building project management processes as a way of increasing organization readiness for Agile transformation. Konferencja Innowacje w Zarządzaniu i Inżynierii Produkcj, Tom II (2016)
Duvall, P.M.: Continuous Integration. Improving Software Quality and Reducing Risk. Pearson Education, Upper Saddle River (2007)
Humble, J.F.: Continuous Delivery. Reliable Software Releases through Build, Test, and Deployment Automation. Pearson Education, Upper Saddle River (2010)
Kelleher, J.D., Namee, B.M., D'Arcy, A.: Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples, and Case Studies. The MIT Press, Cambridge (2015)
Kim, G.: The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations. IT Revolution Press, Singapore (2016)
Marz, N.: Big Data: Principles and Best Practices of Scalable Realtime Data Systems. Manning Publications, Greenwich (2015)
Witten, I.H.: Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann, Burlington (2011)