RIGA TECHNICAL UNIVERSITY

Faculty of Computer Science and Information Technology

Institute of Applied Computer Systems

Gusts Linkevičs

Student of the Doctoral Study Program "Computer Systems"

Supporting Implementation of the Agile Paradigm in Software Development Companies

Summary of the Doctoral Thesis

Scientific Supervisor

Dr.sc.ing., Professor

U. SUKOVSKIS

RTU Press

Riga 2016

Linkevičs G. Supporting Implementation of the Agile Paradigm in Software Development Companies. Summary of the Doctoral Thesis. – R.: RTU Press, 2016. – 43 pp.

Printed according to the decision of the Board meeting of the Institute of Applied Computer Systems, Faculty of Computer Science and Information Technology, Riga Technical University on May 26, 2016, Minutes No. 12300-4.1/4.

ISBN 978-9934-10-902-7

THE DOCTORAL THESIS PROPOSED TO RIGA TECHNICAL UNIVERSITY FOR THE PROMOTION TO THE SCIENTIFIC DEGREE OF DOCTOR OF ENGINEERING SCIENCE

To be granted the scientific degree of Doctor of Engineering Sciences, the present Doctoral Thesis has been submitted for the defence at the open meeting at the Faculty of Computer Science and Information Technology, Riga Technical University, 1 Setas Street, auditorium 202, at 14^{30}, on February 06, 2017.

OFFICIAL REVIEWERS

Professor, Dr. sc. ing. Jānis Grabis

Riga Technical University, Latvia

Professor, Dr. sc. comp. Rudīte Čevere

Latvia University of Agriculture, Latvia

Professor, Dr. Albertas Čaplinskas

Vilnius University, Lithuania

DECLARATION OF ACADEMIC INTEGRITY

I hereby declare that I have developed this thesis submitted for the doctoral degree at Riga Technical University. I confirm that this Doctoral Thesis has not been submitted to any other university for the promotion of other scientific degree.

Gusts Linkevičs .....
signature

Date .....

The Doctoral Thesis is written in Latvian and includes introduction, 5 sections, result analysis and conclusions, bibliography, 6 appendices, 123 tables, 63 figures, overall – 227 pages. The bibliography contains 123 references.

Table of Contents

Introduction

Contemporary business environment is very dynamic and customers are forced to change software requirements frequently, so that they can meet new environmental and business conditions. Customers require frequent and quick software deliveries and upgrades. Traditional software development methods fail to provide the necessary flexibility, which is provided by agile methods.

The first reference to agile software development can be found in 1957, the approach being mentioned in an article by Craig Larman and Victor Basili [112].

In 1970, Dr. Winston Royce presented his article "Managing the Development of Large Software Systems" [106], where he criticizes sequential software development. In the same year, E. A. Edmonds prepared his article "A process for the Development of Software for Nontechnical Users as an Adaptive System", which he wanted to publish in the journal "Computer Aided Design", but publishing of article was refused with comments, "If you don't know what you will do before you start, don't start at all". The article was published later in 1974, in the journal "General Systems" [113].

Irrespective of resistance, agile methods continued developing, for example, "Scrum" (1986) [113], "Extreme Programming" (1999) [19], etc. The meeting of agile method experts and creation of "agile manifesto" in 2001 became the turning point in the agile software development [7].

Today, software development companies of different sizes partially or fully switch to agile software development. The number of successfully developed projects using the agile approach is the main reason for the switch, and this was confirmed by Standish Group research in 2012 [109]. Research shows that 42% of successful projects have been developed using agile methods and only 14% have been developed using a more traditional approach (Figure 0.1.).

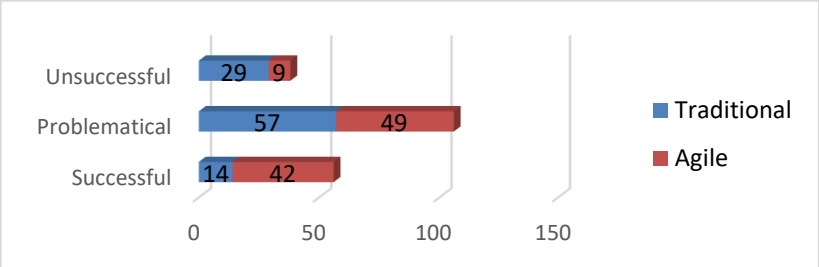

Figure 0.1. Standish Group research results 2012.

Similar tendencies regarding agile software development were noticed by Scott Ambler from AmbySoft [110], where 68 % of successful projects used iterative methods and

67 % used the agile approach, with about half of projects succeeding using traditional methods (Figure 0.2.).

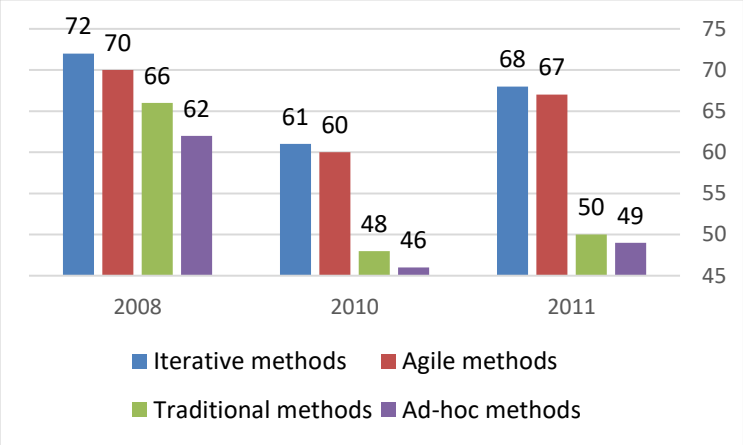

Figure 0.2. Successful projects based on AmbySoft research.

Forrester Research studies [111] found that the number of projects developed using the agile approach increased from 35.4 % in 2009 to 38.6 % in 2010. The number of traditionally developed projects decreased from 13.4 % to 13.0 % (Figure 0.3.).

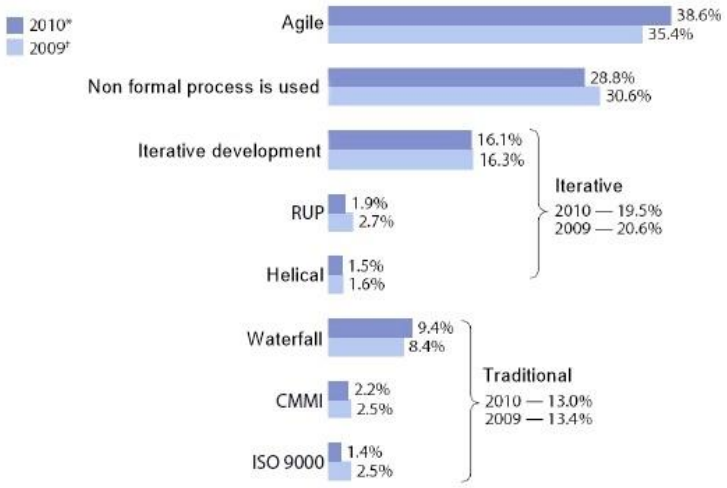

Figure 0.3. Results of research conducted by Forrester Research.

Research results analyzed in this Doctoral Thesis indicate that the usage of agile methods has increased and that agile projects have been more successful. However, there are also indications that not all agile projects have been successful. Standish Group research (Figure 0.1.) shows that 9 % of agile projects are unsuccessful, whilst 49 % had considerable problems.

1.1. Motivation of the Research

The motivation of each organization to implement the agile paradigm differs. In most cases, it is desirable to obtain more successful software development projects in a company portfolio. Some organizations have been successful in transforming into agile software development, but there are also organizations where the transformational process is unsuccessful.

There is a necessity to find a way to help software development companies successfully and quickly adopt the agile paradigm in order to make software development more successful.

One of the most significant motivations to develop this Doctoral Thesis about transition to agile software development is bad experience of the author within several projects aimed at transformation into agile software development. The author has spent more than 6 years researching this problem, with other organizations experiencing similar problems when transforming into agile software development and this is also reflected in conference reports [1][2][3][4][5].

The author of this Doctoral Thesis has worked in five software development companies, which have tried to implement the agile paradigm. In the author's opinion, only one transformation has been successful. For this very reason, the author has decided to find the best solution to this problem.

The term **"organizational agility"** in the context of this Doctoral Thesis has a narrowed meaning and applies only to the software development company's ability to develop software using agile methods and practices.

1.2. The Aim and Tasks of the Doctoral Thesis

The aim of this Doctoral Thesis is to create a method and a tool to support implementation of the agile paradigm in a software development company.

The proposed aim is based on the following **hypotheses**:

- organizations have low awareness of agile methods, and this creates problems in the implementation of the agile paradigm;
- by using methods and tools, it is possible to evaluate the agility level of an organization;
- knowledge regarding the current agility level of an organization helps to create appropriate improvement plans in order to improve their organizational agility level.

To achieve the given aim, the following tasks have been defined:

- to investigate the advantages and risks of agile methods, practices and their significant properties;
- to create a dictionary of regularly used terms. This glossary of terms is required to define the Topicality Index (TI) and is used to determine the TI value for agile methods, practices and keywords;
- to define organizational agility and create an Organizational Agility Model (OAM), identifying which practices influence particular OAM elements;
- to define Agility Impact Index (AII) and determine AII values for OAM elements;
- to create an Organization Domain Agility level determination method (ODA) and a question generation algorithm to be employed by the method;
- to approbate the algorithm FOIL (First-Order Inductive Learner) to determine top-level agility;
- to approbate the method and tool at one or more organizations.

1.3. Research Object and Methods

The **object** of this Doctoral Thesis is the usage of agile methods in software development companies.

The **subject** of this Doctoral Thesis is the development of a method and a tool for determination of organizational agility level.

Analysis of literature, conference materials and internet articles was used to identify significant features, advantages and risks of agile methods.

Analysis of dictionaries and literature was used to create a glossary of terms. The glossary will be visualized using Mind Map software.

Research of conference materials from 2008 to 2012 was used to define TI values for methods, practices and keywords. Information about methods, practices and keywords is stored in a database, exclusively created by the author for data analysis.

Analysis of literature and author's personal experience was used to define the OAM model, and it was verified by the agile method expert network.

Expert network and the DELPHI method were used to define Agility Influence Index (AII) values for each OAM model element.

Results of the method ODA were verified by evaluating the organizational agility level of three software development companies. The companies differed in size, they also work

on different types of projects. One of the companies used agile software development within only one specific project for a particular client.

1.4. Scientific Novelty and Practical Value

Scientific novelty:

- definition of TI has been provided based on term research and creation of glossary "Mind Map". TI values have been calculated for agile methods, practices and keywords;
- organizational agility definition has been given and an Organizational Agility Model (OAM) has been created. OAM has been used to develop the agility determination method ODA;
- AII definition has been provided and AII values have been defined for each OAM element using the agility expert network. Domain, Subdomain and Attribute value tree (DSA) has been developed to make graphical representations of AII values;
- the ODA method collects data from employees, and not to overburden employees with a large number of questions, a question generation algorithm has been created. The question generation algorithm generates small sets of questions for each employee for each survey iteration.

Practical value

The practical value of this Doctoral Thesis lies in the fact that it may help software development companies successfully transform from traditional software development models to the agile paradigm by highlighting problematic areas. Problematic areas are identified by using a developed ODA method and question generation algorithm, which are implemented in an organizational agility level determination software prototype.

1.5. Approbation of the Research Result

Research results are reflected in five publications:

1. Linkevics, G., Adopting to Agile Software Development, volume 16 "Applied Computer Systems", 2014. – Latvia, Riga, RTU, 2014 –64–71 pp. (EBSCO).
2. Linkevics, G., Sukovskis, U., Evaluation of the Agility Level of the Organization, volume 18 "Applied Computer Systems", 2015. – Latvia, Riga, RTU, 2015 –21-26 pp. (De Gruyter).

3. Linkevics, G., Evaluation of Agility in Software Development Company // Proceedings of the Joint International Conference on Engineering Education & International Conference on Information Technology (ICEE/ICIT 2014), Latvia, Riga, 2-6 June, 2014 -320-332 pp.
4. Linkevics, G., Sukovskis, U., Determining Agility Impact Index and generating employee based questions to assess organizational agility // Proceedings of the International Conference on Engineering Education (ICEE 2015), Croatia, Zagreb, 20-24 July, 2015.
5. Linkevics, G., Sukovskis, U., Using ODA Method and FOIL Algorithm to Determine Organizational Agility Level // Proceedings of the 10th International Multi-Conference on Computing in the Global Information Technology (ICCGI 2015), Malta, St. Julians, 11-16 October, 2015 -93-100 pp.

Research results were presented at five conferences:

1. Linkevics, G., Adopting to Agile Software Development. RTU 53rd International Scientific Conference, Riga, Latvia, 11-12 October, 2012.
2. Linkevics, G., Sukovskis, U., Evaluation of the Agility Level of the Organization. RTU 56th International Scientific Conference, Riga, Latvia, 14-16 October, 2015.
3. Linkevics, G., Evaluation of Agility in Software Development Company. Joint International Conference on Engineering Education & International Conference on Information Technology (ICEE/ICIT 2014), Riga, Latvia, 2-6 June, 2014.
4. Linkevics, G., Sukovskis, U., Determining Agility Impact Index and generating employee based questions to assess organizational agility. International Conference on Engineering Education (ICEE 2015), Croatia, Zagreb, 20-24 July, 2015.
5. Linkevics, G., Sukovskis, U., Using ODA Method and FOIL Algorithm to Determine Organizational Agility Level. 10th International Multi-Conference on Computing in the Global Information Technology (ICCGI 2015), Malta, St. Julians, 11-16 October, 2015.

1.6. Outline of the Doctoral Thesis

This Doctoral Thesis consists of an introduction, 5 chapters, a conclusion, bibliography and 6 appendices.

The introduction contains information about the topicality of the subject matter, motivation for the research, the list of aims and tasks, description of the research methods and information about scientific novelty and practical value of the Thesis. Information about

approbation and outline of this Doctoral Thesis are presented at the end of the introductory chapter.

The first chapter describes agile software development and analysis, its advantages and risks. The agile methodology terminology problem is examined and the terms used in this Doctoral Thesis are described. A great deal of attention in this chapter is dedicated to determination of the Topicality Index (TI) for agile methods, practices and keywords. Definition of dynamic environment is given at the end of the chapter.

The second chapter of the Doctoral Thesis describes agile methods and practices with the highest TI. It allows identification of the common and different features of various agile methods. Based on the analyzed data, the Organizational Agility Model (OAM) is built.

The focus of the third chapter is organizational agility. Organizational agility definition is provided and the OAM model developed. OAM model consists of several domains, subdomains and attributes. OAM top level domains are: Organization, Productivity, Quality, Process, Value and Project domains. Definition of the Agility Impact Index (AII) is provided and the detailed AII value determination process is developed. AII values are determined using the agility expert network and the DELPHI expert evaluation method.

The fourth chapter focuses on the conceptual model of agility level determination and organization domain agility level evaluation method ODA. The ODA method is developed and the question generation algorithm is used by the method created. Domain, Subdomain and Attribute value tree (DSA) is created to visualize expert evaluated AII values and Employee Evaluated (EEV) values. Approbation of the FOIL algorithm is performed to generate rules for DSA value tree processing. Data flow diagrams, architecture and tool prototype are developed to determine the organizational agility level.

The fifth chapter describes approbation of the ODA method by determining the agility level of three different organizations.

Brief conclusions are made at the end of each chapter, with key results and overall conclusions presented in the final section of this Doctoral Thesis.

This Doctoral Thesis has six appendices. Information about agile methods and their TI values are summarized in Appendix 1. Information about keywords used at the conferences and their TI values are summarized in Appendix 2. Appendix 3 contains information about agile practices. Detailed information about using the FOIL algorithm and rules generated by the algorithm for classes C_2 and C_3 are presented in Appendix 4. Appendix 5 contains expert evaluated AII values for all OAM elements, and Appendix 6 reports on approbation results of the ODA method.

2. Agile Software Development

There is no single definition of agile software development, various sources provide several definitions of the term ‘agile software development’. Some sources, e.g. [103][106], mention that agile software development is a methodology, whilst others [104][105] say that it is a common term, which combines agile methods and practices. Seven from eight sources make reference to the agile software development manifesto [7]. One of these definitions is as follows:

Agile Software Development is an umbrella term for a set of methods and practices based on the values and principles expressed in the Agile Manifesto [104].

2.1. Advantages and Risks of Agile Software Development

Agile method coach Dave Moran in his article mentioned 10 advantages of agile software development (and the author of this Doctoral Thesis agrees with the list) [8]. According to Moran, agile development delivers systems quicker, embraces business agility; reduces risks; increases productivity; creates a sustainable development environment; enables emergent innovation; builds trust and relationships; expects continuous improvement; is motivating and engaging; and addresses the realities of software development and business needs.

There are also some risks in using agile development methods. Organizations and teams should take these into account [9], including: unprofessional teams, bad communication with the customer or inside the team, poor specifications, unclear or unrealistic requirements, retrospect is not used or implemented [108], knowledge is not shared or stored, metrics are not used or used incorrectly, planned time is not precise, scope of project or iteration is unclear, complicated or erroneous contracts and an insufficient level of knowledge.

2.2. Terminology Problem

Different teams and individuals work in various organizations and their knowledge level varies considerably. Problems emerge, then different sources describe the same items using various definitions. To solve the problem, it is required to perform term analysis and achieve common denominators using the context of this Doctoral Thesis.

Term definitions from 28 various sources are used to build a “Mind Map” of terms and their descriptions. Mostly the terms “Methodology”, “Method”, “Principle”, “Practice” and “Procedure” are used (Figure 2.1.).

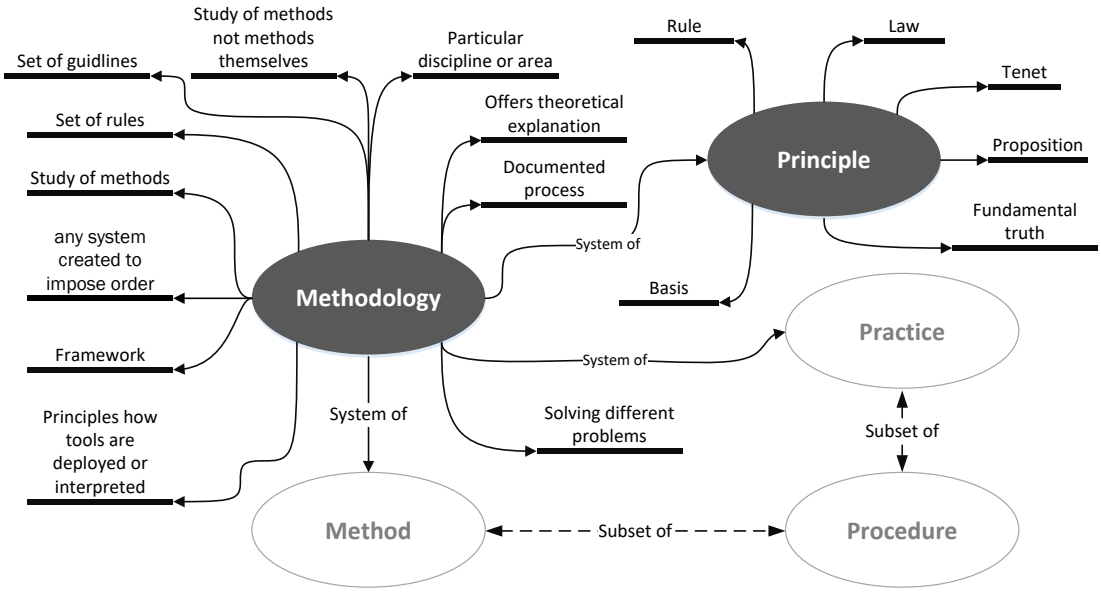

Figure 2.1. Terminology map (fragment).

The term “Methodology” is described as “particular discipline or area”, “Offers theoretical explanation”, “Framework”, “Documented process”, etc. In short, “Methodology” is a system of methods and practices.

The term “Method” is described as “Tool of scientific investigation”, “Way of solving problem”, “Logically ordered plan”, “Generalized concept”, “Established way”, etc. “Method” can consist of several procedures, which, when ordered and executed in particular way, can give a defined result.

The term “Practice” is well described as “Routine”, “Habit”, “Contrast to theory” and “Could not be documented”. “Practice” consists of various procedures, which are not generalized and are systematic.

In the context of this Doctoral Thesis, it was decided that the term “Methodology” is defined –as follows, “Methodology is a system of methods and principles which are used in a particular area” [51] and “Methodology is a study of methods, not the method itself” [52]. The term “Method” is defined as “Regular and systematic way to achieve a goal” [50].

2.3. Topicality Index

Topicality Index (TI) is an indicator, which defines to what extent organizations and teams are interested in a particular method, practice or problem. Definition of TI is based on the analysis of conference materials and is expressed with the equation:

$$IOA_{\mathrm{m}} = \frac{100 \cdot A_{\mathrm{m}}}{A_{\mathrm{y}}}, \quad (1)$$

where IOA_{m} – TI of the method;

A_m – number of articles where the method is mentioned;

A_y – total number of articles in a year.

In accordance with the terminology map, the methods are defined as “Scrum”, “Extreme Programming”, “Lean”, “Dynamic System Development”, “Crystal”, “Feature Driven Development” and “Agile Model Driven Development”.

The TI method values are determined from 2008 up to 2012 and are based on the analysis of those conference materials. Conferences are organized by “Agile Alliance”, which is a non-profit organization and one of the main players in the agile software development field. Conferences organized by “Agile Alliance” fully reflect the tendencies in the field.

Determination of TI values consists of five steps. The first step is to create an agile method list, the second step is to crosscheck each element on the list against the term map (Figure 2.1.). The third stage is to create a software tool and database for data analysis. The fourth step is to gather and save the conference data. The stored information contains the name of the article, description, year, article category or area and URL address (Uniform Resource Locator). Identification of the keywords is a manual process, keywords are identified by the context of the article and not the appearance of a particular keyword. The fifth and final stage is grouping, sorting and merging of the gathered data. At the end of this process, the agile method TI values are defined (Table 2.1).

Table 2.1.

TI of agile methods

Method	**2008**	**2009**	**2010**	**2011**	**2012**
Scrum	18.75	16.78	15.00	13.51	16.82
Extreme programming	8.16	5.77	3.26	1.93	1.66
Lean	7.91	5.77	7.39	8.69	11.14
Dynamic System Development Method	0.51	0.70	0.00	0.00	0.00
Crystal	1.02	0.35	0.00	0.00	0.00
Feature Driven Development	0.00	0.52	0.22	0.00	0.24
Agile Model Driven Development	0.26	0.00	0.43	0.00	0.00

“Scrum” is the method with the highest TI, followed by the “Lean” approach. Similar research results have been shown by “VersionOne”. Research results show that in 2011 “Scrum” was the most popular method (Figure 2.2.) and it was used by 52% of participants, whilst 14% of participants used “Scrum” hybrid methods [115].

The “Scrum” method displays the highest TI during the entire researched period from 2008 to 2012. The TI of the “Lean” approach increases every year, whilst the TI of the “Extreme Programming” decreases every year. The TI of other methods is considerably

lower and it would not be recommended to use them as a basis for transformation purposes in agile software development. Selecting an appropriate agile method is very important during the transformation process, but it is not the only important decision an organization and its team has to make. It is important to select appropriate agile practices, which in combination with the selected method could help achieve the desired results. Selection of an agile practice depends on the organization and the team. In most cases it is the responsibility of the team and it is influenced by different factors, for example, particular project, particular stakeholder (client) or some other factor.

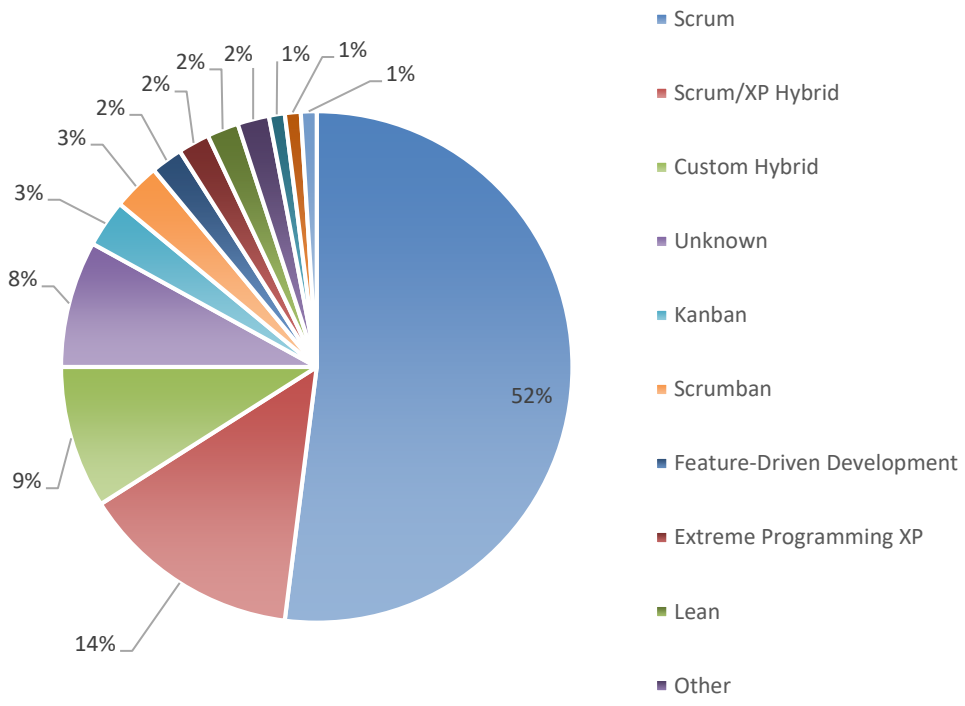

Figure 2.2. Usage of agile methods.

To determine TI values of agile practices information from previous steps is used. Determination of the practice TI value consists of several steps. The first step is to create a list of agile practices. The initial practice list consisted of 176 practices and was acquired based on the analysis of 22 sources. The second step is to remove duplicate practices from the list. As a result of these steps and merging the lists, only 111 practices were left. The third step is to check the list against keywords saved in the database. After cross checking the list, there are only 70 agile practices left in the final list. This Doctoral Thesis contains information about only 35 agile practices (Table 2.2.) which have the highest TI values in the research period.

Table 2.2.

TI of agile practices

Practice	2008	2009	2010	2011	2012
Test Driven Development	12.88	10.84	3.04	6.56	5.69
Retrospective	11.48	15.73	3.91	2.70	4.74
Review	8.55	8.22	0.87	1.74	2.61
INVEST	8.42	7.34	0.87	1.16	5.69
Code Refactoring	7.14	5.59	1.96	2.90	3.79
User stories	6.76	6.12	3.48	4.44	6.64
Continuous deployment	6.63	8.04	1.30	1.54	1.42
Backlog usage	6.38	7.34	4.13	1.93	6.64
Pair programming	6.12	5.77	2.61	3.47	3.08
Measurements	5.36	6.47	3.48	1.54	3.32
Automated testing	4.08	1.31	0.43	1.54	0.00
Automated unit testing	3.95	2.45	1.30	1.16	1.90
Continuous integration	3.83	6.99	0.87	1.93	1.42
Acceptance testing	3.57	2.27	1.30	2.32	0.95
Release Planning	3.57	2.27	0.00	0.39	0.95
Iteration Planning	3.32	2.80	1.30	0.00	0.47
Estimation	2.81	1.40	1.74	0.77	2.37
Daily Stand up meeting	2.81	1.40	1.30	0.39	0.47
Behavior Driven Development	2.81	3.50	1.74	1.35	1.42
Planning game	2.30	1.57	0.65	0.00	0.95
Working software	1.79	2.10	1.30	0.77	1.42
Source Control	1.53	1.40	0.00	0.77	0.47
Active Stakeholder Participation	1.28	2.80	0.43	0.39	1.90
Definition of Done	1.02	0.52	0.22	0.19	2.13
Emergent Design	1.02	1.57	0.00	0.19	1.66
Exploratory testing	1.02	0.70	0.43	0.77	0.47
Facilitation	0.89	2.10	0.22	0.39	1.18
Cross-functional team	0.26	1.40	0.00	0.77	3.32
Usability testing	0.38	1.05	1.30	0.39	0.00
Code review	0.26	1.05	0.43	0.00	0.47
Story mapping	0.26	0.00	0.43	0.39	0.95
Sustainable pace	0.26	0.70	0.43	0.00	0.47
Kanban board	0.26	0.70	0.43	0.00	0.95
Acceptance Test Driven Development	0.38	0.52	0.00	1.16	1.66
Automated build	0.38	0.87	0.43	0.00	1.42

Determination of keyword TI values is a good way to find information about trends of other organizations and interests of their teams. Keywords are the terms which identify a direction of interest, for example, “Learning” or “Coaching”.

Preparation of keyword data consisted of three steps. The first step is to manually identify keywords in the conference materials. Manual identification of keywords is based on the article's context, not just the instance of a particular keyword. Such an approach improves the quality of data. During research and analysis, 1,257 keywords were identified. Some keywords are synonyms and are merged during the next step. The third step is grouping of keywords by year. The final list consists of 37 keywords (Table 2.3.) with the highest TI values.

Table 2.3.

Keywords with highest TI

Keyword	2008	2009	2010	2011	2012
Agile adoption	16.58	21.33	13.48	20.08	23.70
Experience report	16.33	11.19	2.61	9.65	10.90
Agile team	16.07	6.29	13.91	11.58	18.01
Practices	14.80	14.69	31.74	20.08	21.33
Testing	14.80	14.34	9.13	7.34	9.00
Leadership	14.54	13.99	12.61	6.56	8.53
Organizational culture	14.29	13.64	8.26	10.04	13.74
Tools	11.99	11.54	9.57	15.83	8.53
Business value	10.97	8.74	3.04	3.47	9.95
Customer	10.46	6.64	2.61	6.95	11.85
Distributed agile	10.20	8.04	7.83	1.93	3.79
Development	9.69	11.54	9.13	6.18	8.53
Learning	8.93	7.69	3.48	12.74	6.16
Large scale agile	7.40	6.64	10.00	5.79	8.53
Transition	7.40	6.29	15.22	15.44	16.59
Quality	7.40	4.55	2.61	7.34	7.58
Collaboration	6.89	8.39	7.83	17.37	20.85
Communication	6.89	5.59	5.22	5.41	5.21
Organization	6.63	5.94	11.74	4.25	1.42
Coaching	4.08	11.54	5.22	10.04	12.32
Enterprise	3.57	6.99	16.96	9.27	12.32
User experience	5.10	6.99	6.96	5.41	8.06
Environment	4.85	6.64	7.83	5.79	7.58
Planning	3.06	6.29	3.04	3.47	4.27
Product management	0.77	5.94	6.09	0.39	0.47
Requirements	1.79	4.90	10.00	3.86	4.27
Teambuilding	1.02	2.45	6.96	1.93	2.84
Project management	2.81	2.45	6.96	6.56	0.95
Problems	4.08	4.20	6.52	5.79	5.69
Research	2.55	1.75	5.65	5.79	4.74
Hands on labs	0.26	2.10	3.04	8.49	6.64
Business	0.51	1.05	1.30	7.34	1.42
Mentoring	0.77	2.10	0.43	6.18	7.58
Innovation	1.53	4.55	2.17	5.79	5.21
Principles	0.00	0.70	2.17	1.93	7.11
Scaling agile	2.30	5.59	4.35	3.47	6.64

Not all organizations have the ability to hire agile method coaches and experts, as they are rather expensive, so there is a need for an alternative solution. It is proposed in this Doctoral Thesis to develop a new method for determination of organization's agility level, so it could be used during organization's transformation process and after it. The method should help determine problematic areas while transforming to the agile method, without hiring expensive experts each time.

3. Agile Methods and Practices

Software development companies usually use some of the agile methods as a basis for agile software development. Information about each researched agile method included in this Doctoral Thesis consists of basic information, information about process, main components and essential features. Emphasis is made on using the "**Scrum**" method, as it has demonstrated the highest TI value during the research period (Table 2.1).

Various organizations, projects and teams in addition to a particular method should use various agile practices, which help to finish development of a project successfully.

This Doctoral Thesis contains a short description, the English name, known synonyms, positive and negative features of each practice described in Table 2.2. Table 2.2. lists practices with the highest TI value in the research period. Detailed information about agile practices can be found in Appendix 3 of this Doctoral Thesis.

When an organization has determined its agility level and identified problematic areas, it can use information about these practices to create an appropriate improvement plan.

4. Organizational Agility

Organizational agility is the ability of a software development company to transform from traditional development models to the agile development model; to successfully develop various software development projects using agile methods and to be able to quickly react to issues and changes in their environment.

4.1. Organizational Agility Model (OAM)

Organizations that wish to use agile software development should consider their agility level determination. Determination of their agility level helps organizations to evaluate how well they would deal with agile software development and at what level of suitability they are. This will help organizations determine what extra knowledge is required and what actions should be taken to adapt to agile software development more successfully. Organizations are complex and sophisticated entities, and for this reason it is difficult to evaluate them directly. It is proposed that organizations are evaluated from different domain perspectives. The term **"domain"** in the context of this Doctoral Thesis describes a particular area of the organization. For example, "Development domain" describes, analyses and evaluates an area of the organization that is connected with the development of software. The Organizational Agility Model (OAM) (Figure 4.1.) consists of six domains.

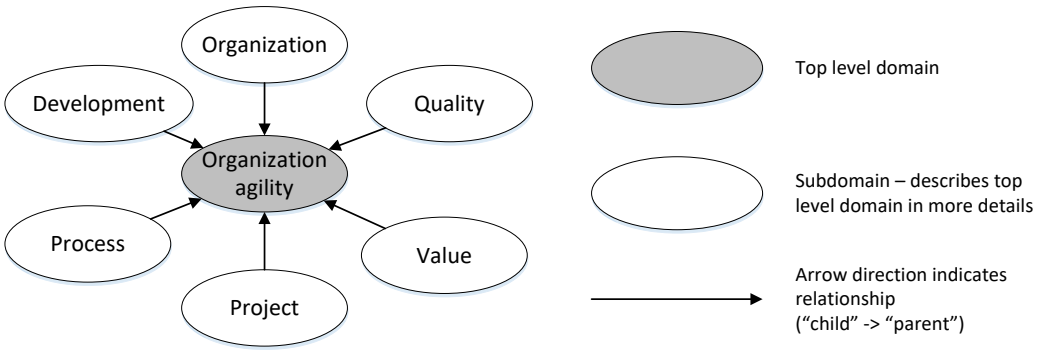

Figure 4.1. Organizational Agility Model (OAM).

The model developed in this Doctoral Thesis is based on the author's experience and has been considerably updated with information gathered from experts during determination of AII (Agility Impact Index) values. Research of literature has shown that studies from "Scrum.org" have a similar vision of top level domain structure [121].

"Organization domain" analyses and evaluates organizational areas, which are connected to overall organization attributes, for example, "Organization size" or "Organization experience with agile methods". "Development domain" evaluates and analyses the process of software development. The "Quality domain", as the name suggests, is concerned with ensuring the quality of the delivered software. The "Process domain" evaluates how well "Scrum" processes are working. The "Value domain" analyses and evaluates delivery of business value to customers. The "Project domain" deals with the analysis and evaluation of projects developed by the organization. The summary contains only some information about the "Organization domain". Detailed information about other domains is included in Chapter 3.1. of this Doctoral Thesis.

Organization domain describes an area of the organization where organization level subdomains and attributes are constantly evaluated and improved. (Figure 4.2.).

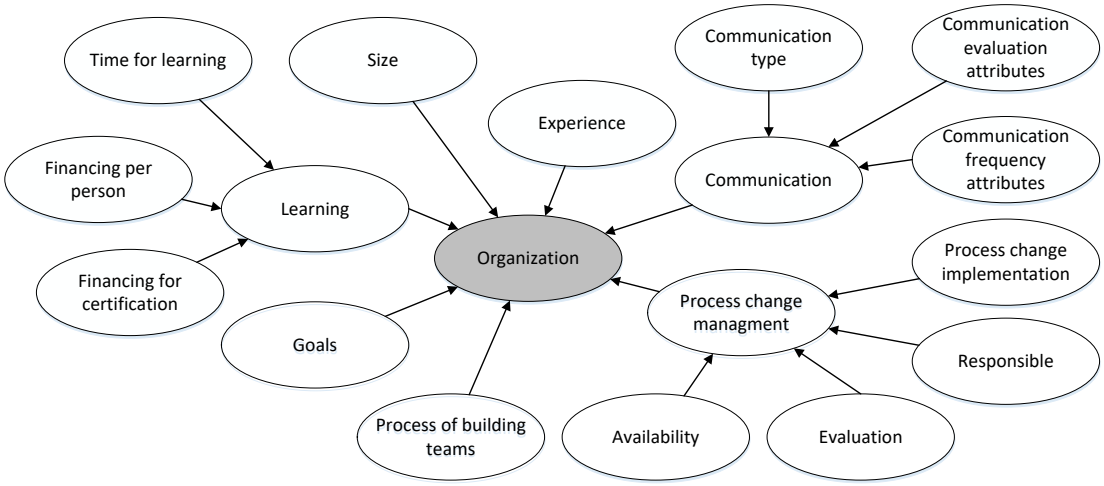

Figure 4.2. Organization domain.

Organization domain consists of eight subdomains and can be extended:

- process change management is required in order to evaluate how processes are used and implemented in an organization. Process change management requires sensibility and responsibility to be undertaken;
- communication plays an important role in agile methods. It is important to evaluate internal and external communication, for example, communication with distributed teams differs from communication between team members located in one room;
- the learning subdomain characterizes organization's ability to learn and is tightly connected with the knowledge subdomain;
- organizational size has to be taken into account, as different agile practices should be used in certain cases;
- organization's experience plays an important role in an organization's agility level, as some organizations have worked with agile methods for some time and have already found practices that work. Sometimes, these established practices may not work well under particular circumstances. Some organizations may have just started working with agile methods and have not tried and tested other methods and practices;
- the team building subdomain describes and evaluates how teams are built.

Each domain is described by subdomains and each subdomain is described by their attributes, for example, "Size". Organizational size attributes vary according to their physical locations. Size information described in this Doctoral Thesis is valid within the Europe Union (Table 4.1.) [95].

Table 4.1.

The number of people in an organization

Name	**Value 1**	**Value 2**
Big	x >250	>50 million EUR
Average	50 < x < 250	<=50 million EUR
Small	11 < x < 50	<=10 million EUR
Micro	x < 10	<=2 million EUR

The developed attributes and their values take up approximately 40 pages of this Doctoral Thesis, for this reason the summary reports only on "Organizational size" attributes. Other attributes and their values can be found in Chapter 3.1. of this Doctoral Thesis, where all domains, subdomains and attributes are described in more detail.

Organizations can modify the OAM model to suite their individual needs. In case of changes to the OAM model, organizations have to reevaluate AII values of the changed items (reevaluation is done by experts). The Doctoral Thesis contains information on how to create new attribute values in order to extend the OAM model.

Various practices influence different parts of the OAM model. Usage of particular practice can increase or decrease agility level for a particular OAM element. Practice compliance to a particular domain is created based on practice description analysis, author's experience and expert opinion. In order to shorten domain names in Table 4.2. The following abbreviations are used: Organization Domain (OD), Development Domain (PD), Quality Domain (KD), Process Domain (PRD), Value Domain (VD) and Project Domain (PRJD).

Table 4.2.

Agile practices and their influence domains

Practice	**OD**	**PD**	**KD**	**PRD**	**VD**	**PRJD**
Test Driven Development		x	x	x		
Retrospective	x	x		x		
Review		x		x		
INVEST		x			x	
Code Refactoring		x	x			
User Stories		x			x	
Continuous Deployment		x		x	x	
Backlog Usage		x			x	
Pair Programming		x	x			
Measurements	x	x	x	x	x	x
Automated Testing			x			
Automated Unit Testing			x			
Continuous Integration		x			x	
Acceptance Testing			x			
Release Planning					x	x
Iteration Planning		x		x		
Estimation		x		x	x	
Daily Stand Up Meeting		x		x		
Behavior Driven Development		x		x		
Planning Game		x		x		
Working Software		x	x		x	
Source Control	x	x		x		x
Active Stakeholder Participation	x	x	x	x	x	
Definition of Done		x				
Emergent Design		x		x		
Exploratory Testing			x			
Facilitation				x		
Cross-Functional Team		x				
Usability Testing			x			
Code Review		x	x			
Story Mapping		x		x		
Sustainable Pace		x	x		x	
Kanban Board		x	x	x		
Acceptance Test Driven Development			x			
Automated Build		x	x	x		

4.2. Agility Impact Index (AII)

Determination of the AII value is an important part of this Doctoral Thesis, as AII values for various domains, subdomains and attributes are not the same. The AII value determines how a particular OAM element influences organizational agility level. The scale from 0 to 10 is used and it is a modified Likert scale [122], where a value of 5 means that an element is not improving or worsening the agility level. Values above 5 improve agility level and values below 5 worsen the agility level. The modified scale with 0 is used, because of "Primary Intelligence" research [123] where it has been determined that 0 allows identification of the direction of the scale more quickly.

The expert questionnaire DELPHI method [99][100] is used to identify weight value of domains, subdomains and attributes. The formation of an expert group is very challenging, as it is necessary to gather a group of 10-20 experts [99] and they need to have the same level of expertise. Creation of an expert group is complicated by the fact that all experts should be available during the research. Taking into account that agile experts are busy, the questionnaire process was organized on the Internet.

Expert group is defined with the equation:

$$EG = \{S, K, M\}, \tag{2}$$

where EG – group of experts;

S – experts who worked or work as a "Scrum Master";

K – experts who are part of an agile team;

M – experts with extensive knowledge of agile software development. These experts use agile software development on a daily basis and have participated in transformations to agile software development in several cases.

To find experts, the author used the network of contacts from his professional life and their contact networks. All involved experts have experience from 5 to 15 years.

Experts evaluated 141 domains/subdomains and 578 attributes. A full list of the evaluated items can be found in Appendix 5 of this Doctoral Thesis. Results of each iteration are processed and experts can see them on a survey website. The survey is repeated until expert coherence is reached. Expert coherence is reached, when the calculated Coherence Coefficient *(CC)* for each element is smaller than 1 and this was reached after 2 iterations. Expert coherence coefficient is calculated with the equation:

$$CC = \max\{E_i\} - \text{avg}\{E_i\}, \quad (14)$$

where CC – Coherence Coefficient;

E_i – Expert evaluation value for element i.

The summary of this Doctoral Thesis includes only a small part of the defined AII values (Table 4.3.). Full results can be found in Appendix 5 to this Doctoral Thesis.

Table 4.3.

Expert determined AII values of the DSA elements (fragment)

Code	Name	AII
1	Organization domain	7
...	...	...
2	Productivity domain	8
2.1	Communication	9
2.1.1	Communication type	8
2.1.1.1	Face-to-face	8
2.1.1.2	By phone	6
2.1.1.3	Written	5
2.1.1.4	Skype or some alternative tool	6
...	...	...
3	Quality domain	8
...	...	...
4	Process domain	8
...	...	...
5	Value domain	7
...	...	...
6	Project domain	7
...	...	...

5. Determination of Agility Level

A systematic approach is required for agility level determination. The aims of this Doctoral Thesis will be reached by using an iterative approach for agility level determination.

5.1. The ODA Method

The ODA method is intended for regular evaluation of organization, project or team's agility level. In order to achieve the desired agility level, it is necessary to provide regular usage of the method. The method gives the opportunity to identify problematic areas and improve the agility level in those areas. The method can be used by organizations which plan to use agile software development. The ODA method is intended to be used in combination with the "Scrum" method, but can also be adapted to work with other agile methods.

The process of the ODA method consists of several sub-processes. The most essential are “OAM element evaluation” and “Question generation” (Figure 5.1.).

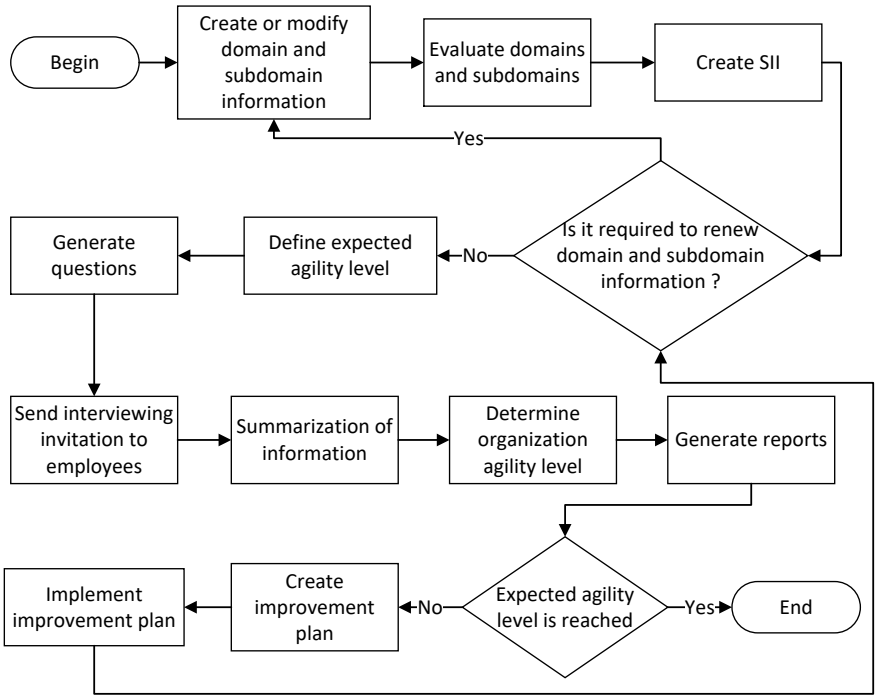

Figure 5.1. Process of the ODA method.

Method consists of 11 sub-processes (Table 5.1.)

Table 5.1.

Processes of the ODA method

Process	**Description**
Create or modify domain and subdomain information	Domains, subdomains and attributes are created or modified. Initial OAM structure is already provided
Evaluate domains and subdomains	OAM elements are evaluated or reevaluated by experts. DELPHI method is used to get evaluation values
Create AII	As a result of expert survey AII values of OAM elements are created
Define expected agility level	Organization or team defines expected agility level
Question generation	AII value is used to generate question set for each employee
Send interviewing invitation to employees	Each participant of evaluation receives the link to the survey with the generated questions
Summarization of information	Survey results are summarized
Determine organizational agility level	Summarized information and defined AII values are used to determine agility level of the organization, project or team
Generate reports	Based on the defined agility level and desired agility level reports are created, so users can analyze the situation and its dynamics
Create improvement plan	In case if the desired agility level is not achieved, organization or team creates improvement plan to improve agility level
Implement improvement plan	Based on improvement plan some actions are made to increase agility level

The agility improvement process is repeated frequently, with frequency of repetitions dependent on an organization and its team. The number of questions in each iteration is

determined by an organization. It is important to take into consideration that a large number of questions is usually met with reluctance and the quality of answers decreases. The recommended number of questions is 10 and questionnaires should require 5 to 10 minutes to fill out [102].

5.2. Question Set Generation and Visualization of the Gathered Information

The question generation component is an important part of the ODA method. This component is needed to generate a small question set for each employee during each questioning iteration. The question generator takes this into account and generates questions based on AII value (Figure 5.2.).

A question set generator generates a subset of questions for each employee from the full question set, which is defined by the equation:

$$Q = \{q_1, q_2, q_3 \dots q_m\}, \quad (3)$$

where Q – set of all questions;

$q_{1\dots m}$ – questions, where m is the total number of questions.

Employee question set is a subset of all questions and is defined by the equation:

$$A_{1\dots n} \in Q, \quad (4)$$

where Q – set of all questions;

$A_{1\dots n}$ – subset of employee questions, where n is the total number of employees, who participate in the questionnaire.

A question set of each employee consists of three types of questions and is defined by the equation:

$$A_{1\dots n} = \{P, N_n, O_n\}, \quad (5)$$

where $A_{1\dots n}$ – subset of employee questions, where n is the total number of employees, who participate in the questionnaire;

P – set of priority questions – initiator of the questioning marks some priority questions, answer to which is required. Priority questions are added to every employee question set and form 20% of all questions;

N_n – unanswered questions are sorted by AII, where n is a particular employee. Unanswered questions with high AII value are added to the set, after priority questions. These questions form 60% of the question set;

O_n – previously answered questions are ordered by AII, where n is a particular employee. There are often important questions that require a compulsory answer. These questions form the remaining 20 % of the question set.

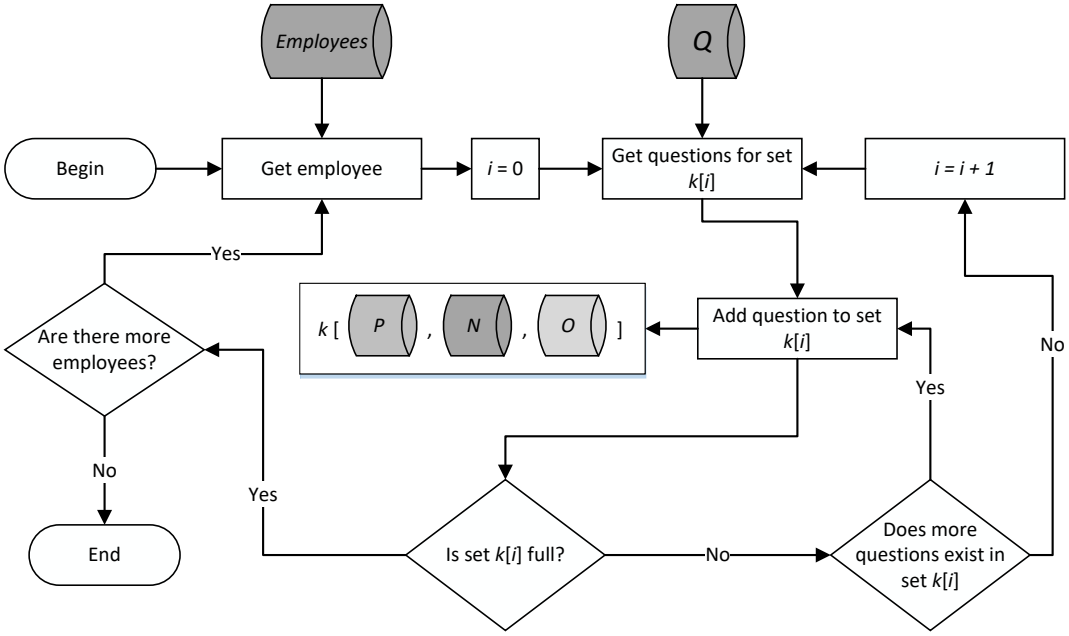

Figure 5.2. Question set generation process.

After questions are generated, they are sent to an appropriate employee. After interviewing results are ready, the DSA value tree is constructed.

The DSA value tree is a convenient way to represent the gathered information and identify problematic areas. Expert evaluated AII values are viewed alongside the processed employee evaluation values (Figure 5.3.).

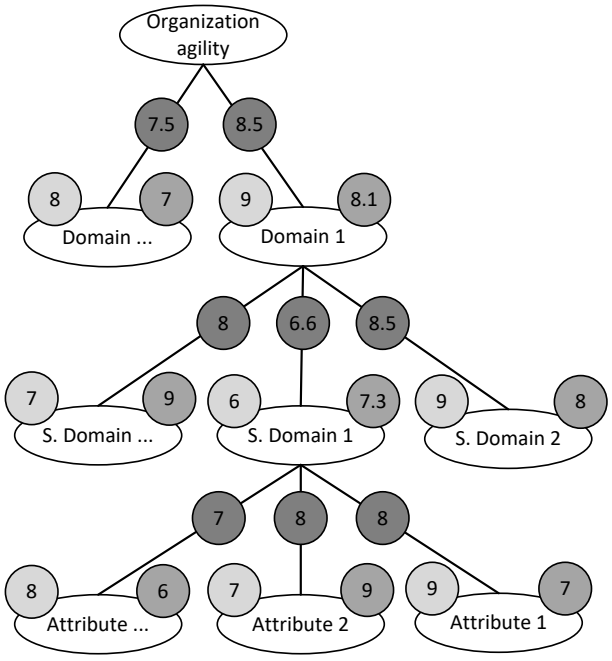

Figure 5.3. DSA Value Tree.

Each element of the DSA value tree has two values at each node. The expert evaluated AII value is on the left side of the node and the employee evaluated value (EEV) is on the right side.

The EEV value of each DSA value tree node is calculated by the equation:

$$\overline{x} = \frac{\sum_{i=1}^{n} x_i}{n}, \tag{6}$$

where $\overline{x}$ – attribute EEV value;

x_i – attribute evaluation is given by employee i;

n – number of employees who have evaluated a particular attribute.

The value of each EEV element is calculated by the equation:

$$\overline{y} = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i}, \tag{7}$$

where $\overline{y}$ – EEV value of a domain or subdomain;

x_i – EEV value of an attribute or subdomain;

w_i – AII value of an attribute or subdomain (weight).

Only on rear occasions there is just one project and team. If an organization is working on several projects with several teams, it is necessary to filter the DSA value tree at different levels. It is possible to determine agility at different levels: organization, project and team levels (Figure 5.4.).

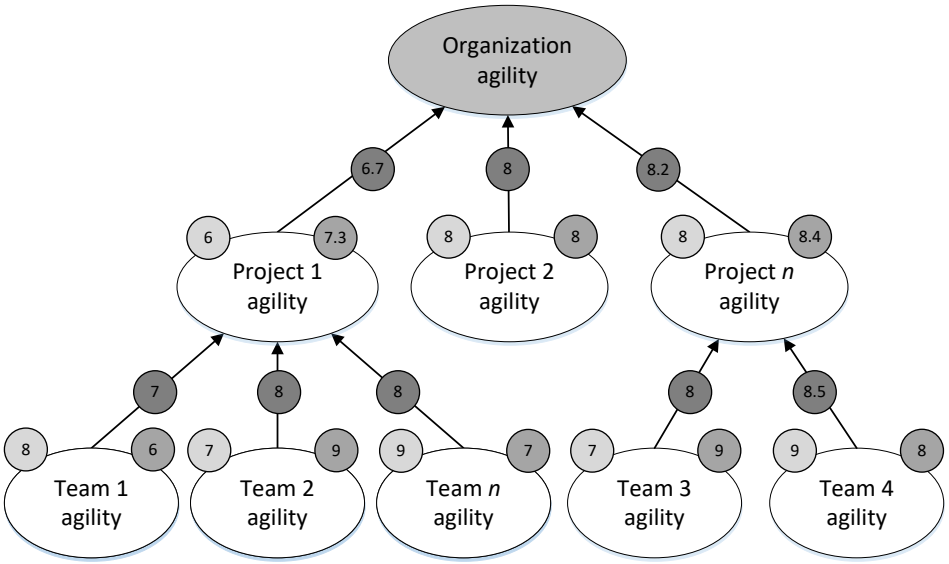

Figure 5.4. Agility grouping levels.

In case of several projects and teams there are some more steps required for agility evaluation. One of the steps is the identification of the Team Impact Index (TII) and the second step is a Project Impact Index (PII) determination.

Team Impact Index (TII) is a way to identify how influential a particular team is in the context of a particular project. Determination of the TII value is the responsibility of project management and the teams.

Project Impact Index (PII) is a way to identify how influential a particular project is in the context of an organization. For example, if most people in an organization work on project A, and project A brings in most of the revenue. At the same time, project B may be less influential, it can be a smaller internal project with only a small team. Thus, the influence of both projects on an organization's agility will differ. Determination of PII value is the responsibility of the top management.

PII and TII values can be reevaluated if such a situation occurs.

5.3. Using the FOIL (First-Order Inductive Learner) Algorithm

FOIL is a rule-based learning algorithm which can be used to solve classification tasks [89]. The FOIL algorithm uses a learning data set to generate rules. A learning data set contains verified information (expert evaluation) which identifies mapping from domain values to agility classes. A larger learning data set allows for the generation of more precise rules. The learning data set included in this Doctoral Thesis is test data and is included only to demonstrate how the algorithm works. Creation of real learning data is a time-consuming process and it is not planned to create it in the context of this Doctoral Thesis. In order to create high quality data, it is recommended that a list of 20 enterprises is prepared to participate in the evaluation process by the expert network, which should consist of 10 to 20 experts. The test learning data set contains information about three agility classes C_1=Not agile, C_2=Partly agile, C_3=Agile and three classes are used for simplification purposes. The FOIL learning data set contains information about 24 cases, where C_1 class has 15 samples, C_2 class has 5 samples and C_3 class has 4 samples. Using the FOIL algorithm on the learning data set produced 9 rules (Table 5.2.).

Table 5.2.

Rules generated by FOIL algorithm

C_1	• $C_1 \leftarrow D4(X, 1)$ [1] • $C_1 \leftarrow D1(X,1) \wedge D4(X,2) \wedge D3(X, 2) \wedge D2(X, 1)$ [2] • $C_1 \leftarrow D1(X, 1) \wedge D4(X,2) \wedge D3(X,1) \wedge D2(X, 2)$ [3] • $C_1 \leftarrow D1(X,2) \wedge D3(X,2) \wedge D2(X,1)$ [4]

C_2	• $C_2 \leftarrow D4(X, 2) \wedge D3(X, 1) \wedge D2(X, 1)$ [5] • $C_2 \leftarrow D4(X, 2) \wedge D3(X,1) \wedge D2(X, 2) \wedge D1(X, 3)$ [6] • $C_2 \leftarrow D3(X, 1) \wedge D4(X, 2) \wedge D1(X, 2)$ [7]
C_3	• $C_3 \leftarrow D4(X, 2) \wedge D3(X, 2) \wedge D1(X, 3)$ [8] • $C_3 \leftarrow D4(X, 2) \wedge D3(X, 2) \wedge D2(X, 2)$ [9]

5.4. Agility Determination Tool

Determination of an agility level is iterative and there is a necessity for a tool, which can be used by experts for AII value determination and by employees for EEV value determination. The tool should provide a convenient way to represent the gathered information.

There are five user stories, which describe how the tool will be used: “Answering of questions” (Figure 5.5.), “Report generation”, “Configuration of tool”, “AII value determination”, and “Creation of improvement plan”.

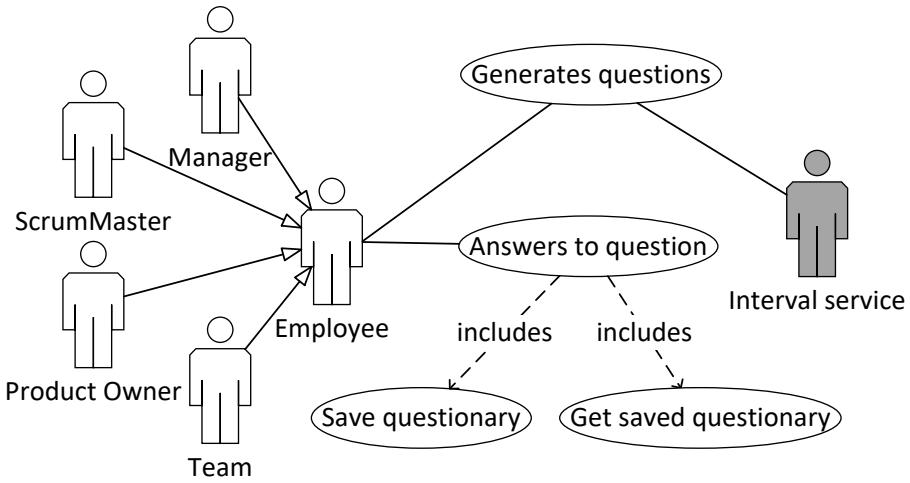

Figure 5.5. Question answering user story.

Question generation can be initiated by some employee or automatically by “Interval service”. “Interval service” can be configured to different intervals and it is an organization or team’s decision, how often they want to execute it. This and other user stories help create an appropriate architecture for the tool.

Tool consists of seven modules, where each module is responsible for a specific functionality (Table 5.3.).

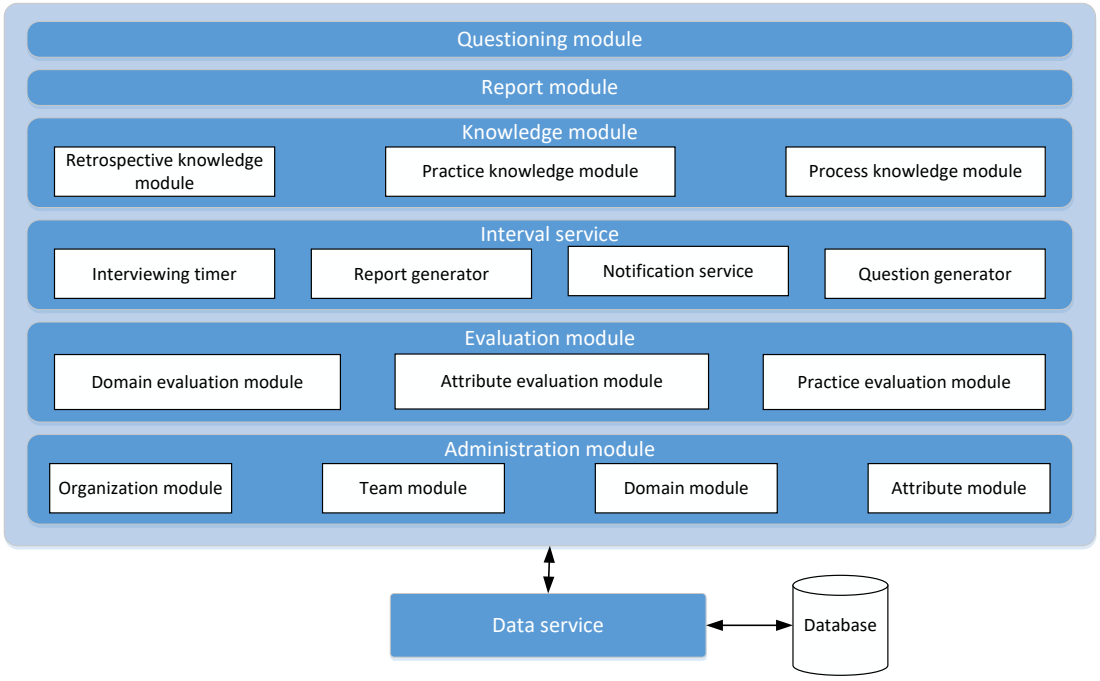

Figure. 5.6. Architecture of the ODA tool.

Table 5.3.

Description of architecture modules

Name	**Description**
Questioning module	Module provides survey process. In the survey module employees see their questions and mark their answers
Report module	Provides report generation initialization and report viewing
Knowledge module	Gathers and stores knowledge so it could be used later
Interval service	Wraps up processes which can be executed in the determined intervals: questioning timer, report generator, notification service and question generation service
Evaluation module	Provides experts with ability to evaluate domain, subdomain and attribute AII values
Administration module	Gives the opportunity to administrate the tool
Data service	Tools data layer, provides data storage channel

A tool created for method verification was developed using Microsoft ASP.NET MVC 5, and hosted on Microsoft developed platform Microsoft Azure. Database is based on the Microsoft SQL Server engine. Program code is developed using C# development language and uses HTML (Hypertext Markup Language), CSS (Cascading Style Sheets), JavaScript, jQuery and jQuery UI library to provide the required functionality.

The tool provides the opportunity to identify problematic domains of an organization in a convenient way (Figure 5.7.).

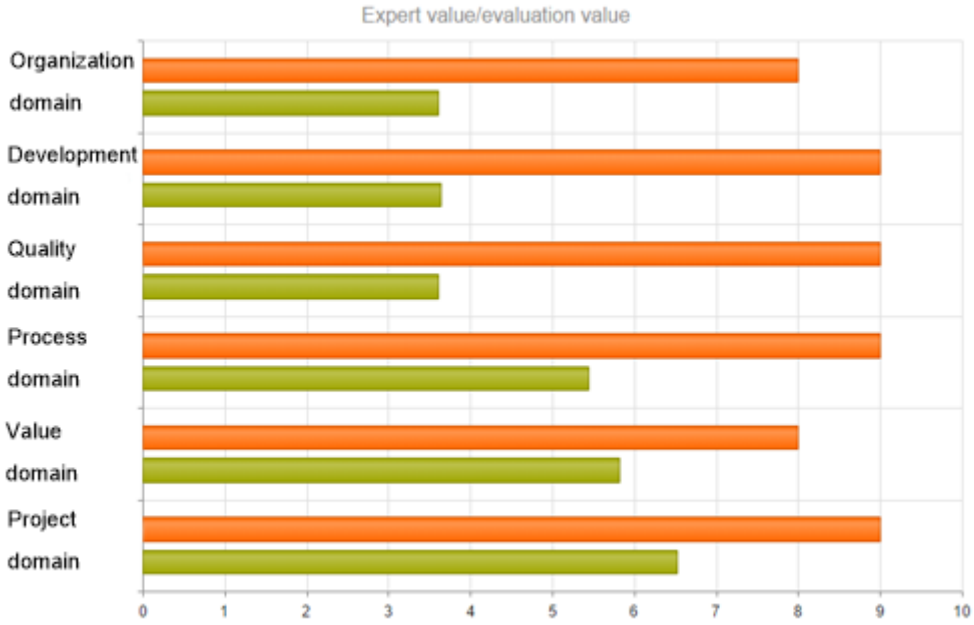

Figure 5.7. Graphical representation of the results in top level domain context.

Experts can connect to the tool and perform evaluation of OAM elements. Employees in their module can answer the generated questions. Employee with appropriate access rights can create and view generated reports, for example, top level domain report (Figure 5.7.).

6. Verification of the ODA Method

Verification of the method is done by analyzing three organizations, which have transformed to agile software development.

6.1. Verification Organizations

This chapter contains descriptions of verification organizations. These organizations have been selected by the author, because the author worked there during transformation processes (Table 6.4.).

Table 6.4.

Verification organizations

Name	Description	Reason for transformation
Organization 1	More than 10 years works with one client. Work with particular client takes ~90 % of company's resources. Development is done using "Waterfall" model.	There are problems with delivering on time and number of bugs. The client proposes to switch development to "Scrum". This offer is accepted and development continues with "Scrum" method.
Organization 2	Had not been working with software development prior to transformation. All software development had been done by third parties using "Waterfall" model.	There was bad experience with previous 3rd party development company. Organization decides to increase delivery times and quality of the project by creating internal software development department. It is decided to use

Name	Description	Reason for transformation
		"Scrum" method as this method was recommended by an external expert.
Organization 3	For more than 20 years develops software using "Waterfall" model.	Organization wants to add agile software development competence to its portfolio. Such opportunity appears when there is project with short development time and superficial documentation. It is decided to use "Scrum" method.

6.2. Verification Results

Agility level of all three organizations is determined by gathering answers to all questions from the question set *Q*. In cases where answers are not available for all questions, agility level is determined using only the available answers. Information about how many questions have been answered is indicated in the agility level report. The agility level determination tool prototype developed for this Doctoral Thesis is used to process the answers to the questions. The summary of this Doctoral Thesis includes information only about top level domains (Table 6.2.). Full information with all DSA tree elements is available in Appendix 6 to this Doctoral Thesis.

Table 6.2.

Results of organizational agility levels

			Organization 1	Organization 2	Organization 3
Code	Name	AII	EEV	EEV	EEV
1	Organization domain	7	3.71	7.40	5.73
2	Development domain	8	3.84	6.69	3.99
3	Quality domain	8	3.67	8.00	5.67
4	Process domain	8	5.28	8.35	7.08
5	Value domain	7	5.58	8.19	6.30
6	Project domain	7	5.60	5.94	6.13

By analyzing information in the six domain context, it can be noticed that Organization 2 has the highest agility level (Table 6.2.) and EEV values are the closest to the expert defined AII values. By using the developed prototype, it is possible to view data in more details for each domain and subdomains. Such an approach allows identification of problematic areas more precisely and facilitates development of an appropriate improvement plan.

The summary of this Doctoral Thesis contains information only about "Organization domain" (Figure 6.1.), detailed information is available in Chapter 5 of this Doctoral Thesis. By viewing the data in Figure 6.1., it is possible to conclude that the agility level of

Organization 1 in the organization domain context is lower than the agility level of Organization 2 and Organization 3.

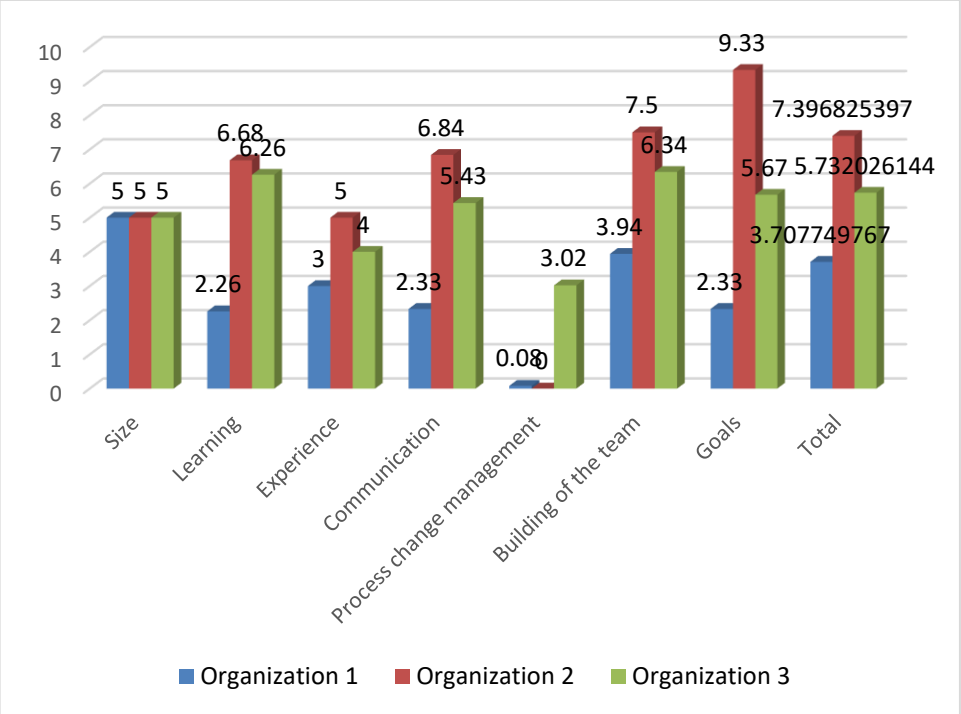

Figure 6.1. Comparison of verification organizations in organization domain context.

Organization 2 has the highest agility level in the context of organization domain. Organization 2 is one of the verified organizations where top level management decided that all software development projects should be developed by using the agile approach. For this method to work, experts were invited and additional employee training was held.

After viewing DSA data and analyzing it, it is possible to make conclusions and start creation of an improvement plan. The data indicate which domains and subdomains have to be offset. This helps organizations and teams to select appropriate practices described in the chapter of this Doctoral Thesis. As an additional help, information from Table 4.2. can be used. It contains information about practices and their influence areas on the OAM model. Organizations and teams can decide which practices from Table 4.2. will be used or tried during implementation of the next improvement plan.

After viewing and analyzing the research results, the author has determined that further research is required to make additional changes to the OAM model in order to add a more detailed breakdown of some elements. From the author's point of view, such necessity could

arise after investigating other organizations and their processes, which may not be fully explained by the existing OAM model.

Key Results and Conclusion

The aim of this Doctoral Thesis was to create a method, the concept of the tool, architecture for the tool and a tool prototype for determination of organizational agility level. As additional aims, it was decided to approbate the created tool prototype by evaluating the organizational agility level in several organizations. The following tasks were completed to achieve the proposed aim:

- the history of agile methods has been researched. Advantages and risks of agile software development have been identified. The significant features of agile methods and practices have been described;
- a terminology map with most often used terms has been created. Definition of the term TI has been provided and the TI values of agile methods, practices and keywords have been identified;
- organizational agility level definition has been provided and an OAM model has been created;
- the term AII has been defined and its value for each OAM element has been determined;
- the organizational agility level evaluation ODA method has been created and a question generation algorithm has been developed;
- a conceptual model and architecture of the agility level determination tool has been created;
- implementation of the proposed architecture has been completed;
- experimental verification of the developed prototype has been performed by evaluating the agility level of three organizations.

All aims and tasks defined in this Doctoral Thesis have been reached. The following main theoretical and practical results have been achieved:

- the term TI has been coined, it is used to denote the level of interest shown in a particular agile method or practice by an organization or team. TI values of agile methods, practices and keywords have been defined using the research period from 2008 to 2012;

- organizational agility level definition has been developed and agility level of three verification organizations has been evaluated;
- the OAM model has been created to evaluate organization's agility level. The OAM model describes an organization using 6 domains, subdomains and an attribute system;
- definition of the term AII has been given; it defines how a particular OAM element influences organizational agility level. AII values are defined for all OAM elements using an expert network;
- organization agility level determination method ODA has been created based on the developed OAM model. The process and steps of the ODA method have been defined and described;
- a question generation algorithm has been created; it provides a small set of questions for each employee during each evaluation iteration;
- a conceptual model and architecture of the tool have been developed to verify the ODA method. Agility level determination tool prototype has been created based on the developed conceptual model and architecture. Verification has been done by evaluating the agility level of three organizations using the created prototype;

All three hypotheses proposed at the beginning of the Doctoral Thesis have been validated:

- organizations have low awareness of agile methods, and this creates problems in the implementation of the agile paradigm – the ODA method was used to evaluate three organizations. Each organization used a different approach to implement agile software development. Analysis of evaluation showed that the agility level of organizations, which did not use external experts and did not perform additional training of employees, was considerably lower;
- by using methods and tools it is possible to evaluate the agility level of an organization – the created ODA method and developed agility level determination tool prototype helped determine the agility level of domains of the three verification organizations;
- knowledge regarding the current agility level of an organization helps create appropriate improvement plans in order to improve its organizational agility level – using evaluated domain agility level values, it is possible to create an appropriate improvement plan, as values indicate which areas are problematic. By using appropriate practices, the agility level of those areas can be improved.

Following conclusions have been made within this research:

- the interest in agile software development demonstrated by software development companies has increased, as agile methods can help solve some of their problems. Agile methods are strongly oriented to the satisfying the client's requirements as quickly as possible, and provide the possibility to change requirements late in the project development process;
- agile methods have various advantages over traditional software development methods, but there are also risks connected to agile software development. Those risks have to be taken into account by organizations and teams, which have to consider the usage of appropriate agile practices to decrease the impact of those risks;
- a wide range of literature is available regarding agile methods and practices, but it is important to use it correctly, as novice users often interpret information differently, and this can lead to additional problems;
- most organizations are using or plan to use "Scrum" as their agile software development method and it is confirmed by the determined TI values and other sources. Method "Scrum" in combination with agile practices provides a better chance of achieving successful results. Various organizations and teams can use different agile practices to achieve the desired result. All agile practices described in this Doctoral Thesis have their advantages and disadvantages. It is the team's responsibility to select the appropriate practice after analysis of all advantages and disadvantages;
- the determined AII values in combination with OAM model and agility level determination method ODA provide opportunity to evaluate an organization's agility level in the context of the six created domains;
- the DELPHI method and expert network have been used to determined AII values, but is has to be remembered that experts are not easily available and not always forthcoming with answers, which complicates AII value determination;
- comprehensive agility level determination requires the usage of FOIL or a similar algorithm and a learning data set. It has to be taken into account that the gathering of comprehensive learning data is a complicated and time-consuming process, and analysis of enterprise data just in Latvia is insufficient. As a matter of fact,

the work load is so high that additional research is required to accomplish this, and this task lies beyond the scope of this Doctoral Thesis;

- the obtained domain agility level information is sufficient to create an improvement plan and the lack of learning data does not hinder determination of problematic areas;
- by making some adjustments to the OAM model, it is possible to use the ODA method with other agile methods.

These research results can be used by organizations, which plan to transform or have already transformed to use agile software development. The developed method and tool will help organizations to determine problematic areas and help develop an improvement plan. The created terminology map in combination with the information about methods and practices will help beginners to better understand the available literature and master the agile approach.

Potential directions for further research:

- to analyze more enterprises, which use agile methods, in order to create a more complete set of learning data to be used by FOIL or similar algorithm;
- to investigate the opportunity to improve the developed prototype in order to use it as a service for organizational agility level determination;
- by extending the expert network, the OAM model can also be extended within further research.

Bibliography

[1] Agile Alliance Conference 2008. Agile Alliance. http://agile2008.agilealliance.org/ (retrieved: 17.05.2012).

[2] Agile Alliance Conference 2009. Agile Alliance. http://agile2009.agilealliance.org/ (retrieved: 19.05.2012).

[3] Agile Alliance Conference 2010. Agile Alliance. http://agile2010.agilealliance.org/ (retrieved: 25.05.2012).

[4] Agile Alliance Conference 2011. Agile Alliance. http://agile2011.agilealliance.org/ (retrieved: 30.05.2012).

[5] Agile Alliance Conference 2012. Agile Alliance. http://agile2012.agilealliance.org/ (retrieved: 10.06.2012).

[6] Agile Software Development. Wikipedia, free encyclopedia. http://en.wikipedia.org/wiki/Agile_software_development (retrieved: 11.06.2012).

[7] Manifesto for Agile Software Development. Agile Alliance. http://agilemanifesto.org/ (retrieved: 11.06.2012).

[8] D. Moran, Top 10 Reasons to Use Agile Development. http://www.devx.com/enterprise/Article/44619/0/page/1 (retrieved: 15.06.2012).

[9] R. Levine and M. McDonough. Why Do Agile Projects Fail? http://www.brighthub.com/office/project-management/articles/55778.aspx (retrieved: 20.06.2012).
[10] R. Coffin and D. Lane. A practical Guide to Seven Agile Methodologies, Part 1, 2006. http://www.devx.com/architect/Article/32761 (retrieved: 26.06.2012).
[11] R. Coffin and D. Lane. A practical Guide to Seven Agile Methodologies, Part 2, 2006. http://www.devx.com/architect/Article/32836 (retrieved: 20.06.2012).
[12] L. Williams. A Survey of Agile Development Methodologies, 2007. http://agile.csc.ncsu.edu/SEMaterials/AgileMethods.pdf (retrieved: 22.06.2012).
[13] Agile Courses. University Of Oxford, 2011. http://www.softeng.ox.ac.uk/subjects/AGM.html (retrieved: 15.07.2012).
[14] Agile Delivery Methods. Agilier. http://www.agilier.com/agile-business-change-techniques/agile-delivery-methods.html (retrieved: 11.08.2012).
[15] Agile Software Development. Enotes.com, 2011. http://www.enotes.com/topic/Agile_software_development#Agile_methods (retrieved: 11.06.2013).
[16] Agile Software Development Site. Seapine Software. http://www.devagile.com/ (retrieved: 11.06.2013).
[17] Extreme Programming. Wikipedia The Free Encyclopedia. http://en.wikipedia.org/wiki/Extreme_Programming (retrieved: 11.06.2013).
[18] G. Lenz and T. Moeller, NET-A Complete Development Cycle. Addison-Wesley Professional, ISBN-10: 0321168828, ISBN-13: 978-0321168825, 2003.
[19] K. Beck and C. Andres, Extreme Programming Explained: Embrace Change (2nd Edition), ISBN-10: 0321278658, ISBN-13: 978-0321278654, 2004.
[20] S. Warden, Extreme Programming Pocket Guide (1st Edition). O'Reilly Media, ISBN-10: 0596004850, ISBN-13: 978-0596004859, 2003.
[21] M. Cohn, Succeeding with Agile: Software Development Using Scrum. Addison-Wesley Professional, ISBN-10: 0-321-57936-4, ISBN-13: 978-0-321-57936-2, 2009.
[22] Scrum (development). Wikipedia The Free encyclopedia. http://en.wikipedia.org/wiki/Scrum_(development) (retrieved: 15.07.2013).
[23] K. Schwaber, Agile Project Management with Scrum. Microsoft Press, ISBN-10: 073561993X, ISBN-13: 978-0735619937, 2004.
[24] Feature Driven Development. Wikipedia The Free Encyclopedia. http://en.wikipedia.org/wiki/Feature_Driven_Development (retrieved: 14.06.2013).
[25] S. R. Palmer and J. M. Felsing, A Practical Guide to Feature-Driven Development. Prentice Hall, ISBN: 0130676152 / 0-13-067615-2, 2002.
[26] FDD process model. Feature Driven Development. http://www.featuredrivendevelopment.com/files/FDD%20Process%20Model%20Diagram.pdf (retrieved: 11.06.2012).
[27] A. Carmichael and D. Haywood, Better Software Faster. Prentice Hall, ISBN-10: 0130087521, ISBN-13: 978-0130087522, 2002.
[28] Test-driven development. Wikipedia The Free Encyclopedia. http://en.wikipedia.org/wiki/Test-driven_development (retrieved: 19.03.2016).
[29] K. Beck, Test Driven Development: By Example. Addison-Wesley Longman, ISBN-10: 0321146530, ISBN-13: 978-0321146533, 2002.
[30] J. W. Newkirk and A. A. Vorontsov, Test-Driven Development in Microsoft NET. Microsoft Press, ISBN-10: 0735619484, ISBN-13: 978-0735619487, 2004.

[31] E. Hakan and M.Torchiano, On the Effectiveness of Test-first Approach to Programming. Proceedings of the IEEE Transactions on Software Engineering, 31(1), January 2005.
[32] N. Llopis, Stepping Through the Looking Glass: Test-Driven Game Development (Part 1). Games from Within, 2007. http://www.gamesfromwithin.com/articles/0502/000073.html (retrieved: 15.06.2012).
[33] M. M. Matthias and F. Padberg, About the Return on Investment of Test-Driven Development. Universität Karlsruhe, Germany. pp. 6, 2007. http://www.ipd.uka.de/mitarbeiter/muellerm/publications/edser03.pdf (retrieved: 05.06.2012).
[34] M. Fowler, K. Beck, J. Brant, et al., Refactoring: Improving the Design of Existing Code. Addison-Wesley Professional, ISBN-10: 0201485672, ISBN-13: 978-0201485677, 1999.
[35] Dependency inversion principle. Wikipedia The Free Encyclopedia. http://en.wikipedia.org/wiki/Dependency_inversion_principle (retrieved: 22.06.2012).
[36] Unified Process. Wikipedia The Free Encyclopedia. http://en.wikipedia.org/wiki/Unified_Process (retrieved: 11.01.2016).
[37] S. Kendall, The Unified Process Explained. Addison-Wesley Professional, ISBN-13: 978-0201742046, 2001.
[38] S. Sousa, The Advantages and Disadvantages / Best Practices of RUP Software Development. http://www.my-project-management-expert.com/the-advantages-and-disadvantages-of-rup-software-development.html (retrieved: 04.02.2016).
[39] M. Broomé, Rational Unified Process - an overview. http://www.it.uu.se/edu/course/homepage/acsd/vt09/RUP-slides.pdf (retrieved: 05.09.2013).
[40] Microsoft Solutions Framework. Wikipedia The Free Encyclopedia. http://en.wikipedia.org/wiki/Microsoft_Solutions_Framework (retrieved: 17.10.2013).
[41] M. Keeton, Microsoft Solutions Framework (MSF): A Pocket Guide. Van Haren Publishing, ISBN-10: 9077212167, ISBN-13: 978-9077212165, 2004.
[42] M. S. V. Turner, Microsoft Solutions Framework Essentials: Building Successful Technology Solutions. Microsoft Press; 1 edition, ISBN-10: 0735623538, ISBN-13: 978-0735623538, 2006.
[43] S. Anderson, Collins English Dictionary – Complete and Unabridged. HarperCollins Publishers, New Yourk, NY, ISBN-10: 0007191537, ISBN-13: 978-0007191536, 2003.
[44] Kanban. Wikipedia The Free Encyclopedia. http://en.wikipedia.org/wiki/Kanban (retrieved: 17.10.2013).
[45] Lean Software Development. Wikipedia The Free Encyclopedia. http://en.wikipedia.org/wiki/Lean_software_development (retrieved: 17.10.2013).
[46] Specification by Example. Wikipedia The Free Encyclopedia. http://en.wikipedia.org/wiki/Specification_by_example (retrieved: 17.10.2013).
[47] Kanban (development). Wikipedia The Free Encyclopedia. http://en.wikipedia.org/wiki/Kanban_(development) (retrieved: 17.10.2013).
[48] Guide to Agile Practices. Agile Alliance. http://guide.agilealliance.org/ (retrieved: 19.03.2016).

[49] Methodology. Business Dictionary. http://www.businessdictionary.com/definition/methodology.html (retrieved: 14.07.2013).
[50] Method. The Free Dictionary. http://www.thefreedictionary.com/method (retrieved: 14.07.2013).
[51] The American Heritage Dictionary of the English Language. Houghton Mifflin Harcourt, Boston, MA, ISBN-10: 0395825172, ISBN-13: 978-0395825174, 2000
[52] Methodology. Wikipedia The Free Encyclopedia. http://en.wikipedia.org/wiki/Methodology (retrieved: 14.07.2013).
[53] Atdd. Guide to Agile Practices. Agile Alliance. http://guide.agilealliance.org/guide/atdd.html (retrieved: 14.07.2013).
[54] Acceptance Testing. Guide to Agile Practices. Agile Alliance. http://guide.agilealliance.org/guide/acceptance.html (retrieved: 14.07.2013).
[55] Automated Build. Guide to Agile Practices. Agile Alliance. http://guide.agilealliance.org/guide/autobuild.html (retrieved: 14.07.2013).
[56] A. Kolawa, Automated Defect Prevention: Best Practices in Software Management. Wiley-IEEE Computer. Society Press, ISBN 0-470-04212-5, 2007.
[57] S. Butt and R. Badger, Benefits of Automated Testing. http://red-badger.com/blog/2013/02/01/benefits-of-automated-testing/ (retrieved: 14.07.2013).
[58] S. Vaaraniemi, Benefits of Automated Testing. Codeproject. http://www.codeproject.com/Articles/5404/The-benefits-of-automated-unit-testing (retrieved: 15.07.2013).
[59] BDD. Guide to Agile Practices. Agile Alliance. http://guide.agilealliance.org/guide/bdd.html (retrieved: 15.07.2013).
[60] Pragmatic BDD for NET. SpecFlow. http://www.specflow.org/ (retrieved: 16.07.2013).
[61] Refactoring. Guide to Agile Practices. Agile Alliance. http://guide.agilealliance.org/guide/refactoring.html (retrieved: 16.07.2013).
[62] Code review. Wikipedia The Free Encyclopedia. http://en.wikipedia.org/wiki/Code_review (retrieved: 20.07.2013).
[63] Continuous Deployment. Guide to Agile Practices. Agile Alliance. http://guide.agilealliance.org/guide/cd.html (retrieved: 20.07.2013).
[64] Cross-functional team. Wikipedia The Free Encyclopedia. http://en.wikipedia.org/wiki/Cross-functional_team (retrieved: 21.07.2013).
[65] Cross-Functional teams. Reference for Business, Encylopedia of Business, 2nd edition. http://www.referenceforbusiness.com/small/Co-Di/Cross-Functional-Teams.html (retrieved: 16.07.2013).
[66] Daily meeting. Guide to Agile Practices. Agile Alliance. http://guide.agilealliance.org/guide/daily.html (retrieved: 16.07.2013).
[67] Definition of done. Guide to Agile Practices. Agile Alliance. http://guide.agilealliance.org/guide/definition-of-done.html (retrieved: 16.07.2013).
[68] Emergent Design. Wikipedia The Free Encyclopedia. http://en.wikipedia.org/wiki/Emergent_Design (retrieved: 16.07.2013).
[69] Estimation. Guide to Agile Practices. Agile Alliance. http://guide.agilealliance.org/guide/estimation.html (retrieved: 17.07.2013).
[70] Facilitation. Guide to Agile Practices. Agile Alliance. http://guide.agilealliance.org/guide/facilitation.html (retrieved: 17.07.2013).
[71] Invest. Guide to Agile Practices. Agile Alliance. http://guide.agilealliance.org/guide/invest.html (retrieved: 17.07.2013).

[72] Iteration Planning. Version One. http://www.versionone.com/Agile101/Agile-Development-Iteration-Planning/ (retrieved: 22.07.2013).
[73] Kanban board. Guide to Agile Practices. Agile Alliance. http://guide.agilealliance.org/guide/kanban.html (retrieved: 22.07.2013).
[74] T. Javdani, H. Zulzalil, A. Ghani, On the Current Measurement Practices in Agile Software Development. University Putra, Malaysia. http://arxiv.org/ftp/arxiv/papers/1301/1301.5964.pdf (retrieved: 23.07.2013).
[75] Pair Programming. Guide to Agile Practices. Agile Alliance. http://guide.agilealliance.org/guide/pairing.html (retrieved: 23.07.2013).
[76] Planning Poker. Guide to Agile Practices. Agile Alliance. http://guide.agilealliance.org/guide/poker.html (retrieved: 25.07.2013).
[77] Story Mapping. Guide to Agile Practices. Agile Alliance. http://guide.agilealliance.org/guide/storymap.html (retrieved: 25.07.2013).
[78] Sustainable Pace. Guide to Agile Practices. Agile Alliance. http://guide.agilealliance.org/guide/sustainable.html (retrieved: 25.07.2013).
[79] Tdd. Guide to Agile Practices. Agile Alliance. http://guide.agilealliance.org/guide/tdd.html (retrieved: 27.07.2013).
[80] Usability Testing. Guide to Agile Practices. Agile Alliance. http://guide.agilealliance.org/guide/usability.html (retrieved: 28.07.2013).
[81] User Stories. Guide to Agile Practices. Agile Alliance. http://guide.agilealliance.org/guide/user-stories.html (retrieved: 29.07.2013).
[82] Dynamic Environment. Artificial Intelligence Lab. http://ai.eecs.umich.edu/cogarch4/toc_defs/defs_env/defs_dyn_env.html (retrieved: 30.07.2013).
[83] Dynamic Environment. Ask.com. http://www.ask.com/question/what-is-a-dynamic-environment-in-an-organization (retrieved: 30.07.2013).
[84] D. Weyns and H. Van Dyke Parunak, F. Michel, Environments for Multi-Agent Systems II. Springer, ISBN-10: 3-540-32614-6, ISBN-13: 978-3-540-32614-4, 2005.
[85] K. S. Rubin, Essential Scrum: A Practical Guide to Most Popular Agile Process. Addision-Wesley Professional, 2012.
[86] J. Beaver, The Agile Team Handbook. CreateSpace Independent Publishing Platform, 2013.
[87] J. Humble and D. Farley, Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation. Addision-Wesley Professional, 2010.
[88] R. Pichler, Agile Product Management with Scrum: Creating Products that Customers Love. Addision-Wesley Professional, 2010.
[89] T.M. Mitchell, Machine Learning (10.4 and 10.5 Chapter). pp. 283-291. 1997.
[90] A. Borisovs, Izdales materiāli mācību kursā „Intelektuālās datorsistēmas", 2012.
[91] S. Parshutin, G. Kuleshova, A. Barisov, Application of first order rules to reconstructing link damages in logistics net. 2005.
[92] A. Cockburn, Crystal Clear, A Human-Powered Methodology For Small Teams, including The Seven Properties of Effective Software Projects. Addision-Wesley Professional, ISBN-10: 0201699478, ISBN-13: 978-0201699470, 2004.
[93] Dynamic Systems Development Method. Wikidot. http://dsdmofagilemethodology.wikidot.com/ (retrieved: 30.03.2014).
[94] M. Poppendieck and T. Poppendieck, Lean Software Development: An Agile Toolkit. Addison-Wesley Professional, ISBN 0-321-15078-3, 2003.
[95] European Commission, The new SME definition. Official Journal of the European Union L 124, May 2003.

[96] H. Rubin, A Metrics View of Software Engineering Performance Across Industries. IT Metrics Strategies V:9:3, September 1999.
[97] Communication. Oxford Dictionaries. http://www.oxforddictionaries.com/definition/english/communication (retrieved: 15.03.2014).
[98] Collaboration. Oxford Dictionaries. http://www.oxforddictionaries.com/definition/english/collaboration (retrieved: 15.03.2014).
[99] М. Саркисян, Теория прогнозирования и принятия решений. Высшая школа, 1977.
[100] Л.В. Ницецкий и Л.П. Новицкий, “Применение методов экспертного опроса для оценки качества диалоговых обучающих систем”. Методы и средства кибернетики в управлении учебным процессом высшей школы. Сборник научных трудов, Рига РПИ, 1986.
[101] N. Slocum, Participatory methods toolkit, A practitioner’s manual. King Baudouin Foundation, ISBN 90-5130-506-0, 2005.
[102] SurveyMonkey, How Much Time are Respondents Willing to Spend on Your Survey? https://www.surveymonkey.com/blog/2011/02/14/survey_completion_times/ (retrieved: 01.08.2015).
[103] Agile Software Development Definition. TechTarget. http://searchsoftwarequality.techtarget.com/definition/agile-software-development (retrieved: 01.05.2016).
[104] What is Agile Software Development. Agile Alliance. https://www.agilealliance.org/agile101/what-is-agile/ (retrieved: 01.05.2016).
[105] Agile Software Development definition. PC Magazine, Encyclopedia. http://www.pcmag.com/encyclopedia/term/37607/agile-software-development (retrieved: 01.05.2016).
[106] What Is Agile? Agile Methodology. http://agilemethodology.org/ (retrieved: 01.05.2016).
[107] Qumer., & Sellers, An evaluation of the degree of agility in six agile methods and its implacability for method engineering. Information and Software Technology, pp. 280-295, 2008.
[108] E. Derby and D. Larsen, Agile Retrospectives, Making Good Teams Great. Pragmatic Bookshelf, 2006.
[109] The CHAOS Manifesto. The Standish Group, 2012. http://blog.standishgroup.com/ (retrieved: 01.10.2016).
[110] S. Ambler. Agile projects success rate. AmbySoft. http://www.ambysoft.com/surveys/ (retrieved: 01.10.2016).
[111] Agile Adoption rates. Forester Research. http://www.forrester.com (retrieved: 01.10.2016).
[112] C. Larman and V. Basili, Iterative and Incremental Development: A Brief History, 2003. http://www.craiglarman.com/wiki/downloads/misc/history-of-iterative-larman-and-basili-ieee-computer.pdf (retrieved: 20.03.2016).
[113] E. A. Edmonds, A Process for the Development of Software for Nontechnical Users as an Adaptive System. General Systems, 1974.
[114] H. Takeuchi and I. Nonaka, New Product Development Game. Harvard Business Review 86116, p. 137-146, 1986.
[115] State of Agile Survey Results. VersionOne Inc, 2011. http://www.versionone.com/ (retrieved: 01.17.2016).

[116] H. C. Maurya, A. Khatoon, N. Chaudhary, Metrics for Software Project Size Estimation. International Journal of Advanced Research in Computer Science and Software Engineering, Volume 5, Issue 1, January, 2015.
[117] Project Sizes. Method 123 Project Management Methodology. http://www.mpmm.com/project-sizes.php (retrieved: 01.24.2016).
[118] Information Technologies, Project Classification. The University of New Mexico. https://it.unm.edu/projects/projectdefined.html (retrieved: 01.24.2016).
[119] Ten Tips for Agile Leaders. The Pragmatic Bookshelf. https://pragprog.com/magazines/2012-02/ten-tips-for-agile-leaders (retrieved: 01.31.2016).
[120] The Role of Leaders on a Self-Organizing Team. Mountain Goat Software. https://www.mountaingoatsoftware.com/blog/the-role-of-leaders-on-a-self-organizing-team (retrieved: 01.31.2016).
[121] Agility Path. Scrum.org. https://www.scrum.org/Portals/0/Documents/Agility-Path/Agility-Path_Executive-Summary.pdf (retrieved: 17.05.2012).
[122] Likert scale. Wikipedia The Free Encyclopedia. https://en.wikipedia.org/wiki/Likert_scale (retrieved: 16.07.2016).
[123] J. Bledsoe. The Magic in a 0-to-10 Rating Scale. Primary Intelligence. https://en.wikipedia.org/wiki/Likert_scale (retrieved: 16.07.2016).