2014 Sixth International Conference on Computational Intelligence, Communication Systems and Networks

Use of Method for Elicitation, Documentation and Validation of Software User Requirements (MEDoV) in Agile Software Development Projects

Srdjana Dragicevic
Split Airport,
Kastela, Croatia
srdjana.dragicevic@split-airport.hr

Stipe Celar
University of Split, FESB,
Split, Croatia
stipe.celar@fesb.hr

Luksa Novak
Split Airport,
Kastela, Croatia
luksa.novak@split-airport.hr

***Abstract*—Agile and lean software development methodologies appear as a popular alternative to plan-driven methodologies, but these methodologies have no structure in the process of user requirements specification. This paper shows how Method for Elicitation, Documentation and Validation of Software User Requirements (MEDoV) supports agile and lean software development methodologies. The MEDoV helps stakeholders in their everyday work with minimal impact on agility. The method also ensures an active role of business users, a wide-picture what usually is the problem for agile development, and definition of non-functional requirements, what the even bigger problem is. The MEDoV ensures that no unnecessary features are produced so that no extra code is created, and maintenance is easier, as well as code correction and improvement. Using models enables agility in product maintenance, especially for integrated systems where one change can have multiple impacts on different parts of system.**

Keywords — Requirements Engineering, Agile Methodologies, Lean Software Development, Business Process Management, UML Diagrams, Event-driven Process Chain (EPC) Method

I. INTRODUCTION

Plan-driven software development is suitable for projects with well-known technology and for projects where requirements remain relatively stable [1].

Effort to manage change of requirements, as well as to create greater value for customer, led to development of agile software development methodologies.

Lean principles were adapted to software development process in 2003. The authors state that lean principles are the basis on which agile practices can be built [2]. In practice, development teams combine agile and lean principles to achieve both economic efficiency and flexibility [3].

Agile development methodology (including lean principles) strives to ensure user satisfaction and delivers only necessary functionalities, for what it is indispensable to ensure correctness and completeness of the requirement. This is achieved by greater involvement of user, but techniques of requirements elicitation are practically not much different than in plan-driven software development [4]. This means that problem of communication between user and development team continues to exist, because the user often does not know how to explain what he actually needs, or the purchaser neglects demands of the final user, but stipulates the implementation by his acceptance. Agile development is based upon prioritisation of requirements, but, without understanding user needs, prioritisation becomes impossible. Particular problems are non-functional requirements which can jeopardise the entire project. Most often the user implies them, but does not understand their importance, and agile methods implicitly recognise such requirements when eliciting requirements for individual iterations [4].

Agile and lean methodologies have no structure in the process of user requirements specification [5], so the main goal of this research is to provide method which ensures correctness and completeness of requirements and enables requirements change, with minimal impact on agility.

Integrated business processes are the basis of competitive advantage and their efficiency is a prerequisite for survival of the company [6] [7]. Business process management (BPM) is a discipline which treats business processes as assets that directly improve business performance through operational excellence and business agility [8]. BPM enables implementation of business processes in such a way that implemented software supports business goals and can be managed from the business process perspective [9].

Experience has shown that requirements are often defined independently of, or even completely ignoring, business processes [10]. Developers are concentrated on data and expect that business stakeholders understand technical terminology, what usually is not a case.

Business process models which are designed to help document, communicate or improve business processes can also be used for requirements elicitation. Using models for requirements documentation enables agile maintenance, especially in large and complex software systems.

Basic concept of Method for Elicitation, Documentation and Validation of Software User Requirements (MEDoV) is developed in [11] and [12]. In this paper, the method is further refined to support agile and lean software development methodologies with minimal impact on agility.

Agile and lean software developments are briefly introduced in Chapter II and III. The MEDoV method is described in Chapters IV – VII. Case study is described in Chapter VIII. Chapter IX concludes the paper.

II. AGILE METHODOLOGIES

Agile development is not a methodology in itself. It is a term which describes several independently developed agile methodologies, all sharing the same values [13][14].

Agile methods are principle-based [13]. Practices are activities used to implement agile principles and values to

978-1-4799-5076-8/14 $31.00 © 2014 IEEE
DOI 10.1109/CICSyN.2014.27

specific situation and specific project. Numerous practices are created by applying these principles, such as refactoring, small release cycles, pair programming, and so on.

Development teams usually do not strictly stick to all practices of selected methodology, but choose the subset of agile practices suitable for the specific project [14].

Agile methods are based on iterative and evolutionary methods. Flexible and fast response to changes is ensured by short development iterations, minimal planning and incremental improvements. Iteration length is fixed, usually from one week to one month. Each iteration is project in itself. It includes all activities of development cycle, from requirements analysis to customer acceptance, and ends with an (internal) application release. Unnecessary documentation is eliminated [14].

Reality is that requirements change during project, and detailed documentation early in the project can be a waste of time and efforts. Therefore, in agile development requirements are defined at the beginning of each iteration. But, this approach has some weakness [4] [5] [15]:

- Lack of the wider picture; connections between requirements may be missing.
- Non-functional requirements are elicited implicitly; no eliciting and managing techniques are provided.
- Limited documentation causes problems in product maintenance.
- Lack of specification requirements engineering activities.

III. LEAN SOFTWARE DEVELOPMENT

The term "Lean Production" was first used in 1990 to describe organization system used in Toyota Production System. Lean philosophy is based on understanding value from customer point of view.

The Poppendiecks apply five principles of lean production [14] to software development and synthesize them with agile practices. Results are seven principles of lean software development [16].

IV. THE MEDOV METHOD

The quality of requirements elicitation activities has a great impact on software development [17]. Ambiguity is a major problem in requirements specification, but not the only one. Stakeholders have to deal with following types of decisions [18]:

- Quality decisions, e.g.: Is the requirement non-redundant, concrete and understandable?
- Preference decisions, e.g.: Which requirements should be considered for the next release?
- Classification decisions, e.g.: Which topic, component, or team does particular requirement belong to?
- Property decisions, e.g.: Is the effort estimation for particular requirement realistic?

Increasing size and complexity of software projects make these decisions all the more difficult. The use of business models can facilitate these decisions for business stakeholders, but, the most popular BPM notations still lack the constructs to be easily used in requirements engineering [19].

The MEDoV is designed to help stakeholders make right decisions, and to document these decisions in the right way. The developed method uses recommendations of ISO standards (ISO 9001:2008 and ISO 25010:2011) to improve the quality of software.

The MEDoV method consists of three phases (Figure 1).

V. MEDOV – PHASE I

Project success correlates with fulfilment of business goal(s). It is necessary to identify project objectives and link them with specific business processes (functions), and to indicate known risk factors. At the end of the project, it must be clearly shown whether the objective has been fulfilled. In other words, the objectives must be measurable. Therefore, one or more Key Performance Indicators (KPIs) must be defined for each project goal and critical factor. In this way, development team receives information about priorities. Based on the resulting KPI development, team has a good knowledge where to focus its efforts and resources in order to provide maximum business benefit.

Failure to meet non-functional requirements may result in dysfunctional (useless) system, especially if requirements are related to system performance and reliability. Non-functional requirements also require measurements, so at least one KPI should be defined for every requirement.

Phase I enables deep understanding of customer values and expectations. This is in line with lean principles (Table I), and provides a wider picture and definition of non-functional requirements, what is a problem often encountered in agile development.

Phase I is performed only at the beginning of the project, while Phases II and III are repeated in each iteration.

VI. MEDOV – PHASE II

Phase II includes elicitation and documentation of requirements, and requirements analysis.

The MEDoV provides use of business models and process approach, and participants in RE are elected to the role in the process. Process owners are responsible for accuracy and completeness of elicited business data.

Event-driven Process Chain (EPC) models and Unified Modelling Language (UML) diagrams are used for requirements documentation. The main reason for EPC selection is acceptance by business users, because of user-friendliness and easiness of modelling. On the other hand, UML diagrams are widely accepted by development teams.

The MEDoV recommends top-down approach, beginning with high-level processes and then refining them to middle-level processes. High-level process description is modelled only once, at the beginning of the project. Iteration scope is continuously refined. Consequently, steps in Phases II and III are repeated in each iteration.

All information about business processes are stored in business process repository. The repository is the basis for their automation, too. Repositories can range from passive containers to sophisticated ones.

A. Conventions

Using models for documentation of business process offers a number of advantages: uniformity, reduced complexity, reuse, ease of evaluation, integrity... It is necessary to define conventions of the project in order to take full advantage of listed benefits. These conventions include modelling conventions, graphics conventions, roles and privileges, naming conventions and procedures for repository configuration management.

B. Business process modelling

In order to include all relevant information in business models, three aspects must be taken into account [20]:

- Organizational relationships: Who works with whom, how, and in which order?
- Documents and data relationships: Which documents and data are used in the process, and how?
- IT support: Which IT systems and services are used, and how?

The workflow sequences are modelled using the EPC method: functions and events alternate, possibly separated by connectors. The events in EPC are triggers for functions, or for results of functions. They are states with no duration.

For the purpose of modelling the relationship between processes and business environment, the MEDoV uses extended event-driven process chain or eEPC notation. That is an EPC notation extended by symbols which correspond to different aspects of business modelling [21].

Data model diagrams show the structure of inputs/outputs, but MEDoV recommends that business users only collect necessary data and documents (templates, scans of documents, screens etc.), and then that developers model these data in UML Class diagram, if necessary.

C. Business Rules

In many cases, changes in applications affect only decisions, while the structure of business processes remains stable.

Some approaches extract business rules from business models [22][23], but that is suitable for analysis and optimization of business processes, and not for defining of user software requirements.

The MEDoV highly recommends separating description (modelling) of the process logic from description of the actual decision logic. Description of the decision logic, in plain text, has to be attached to appropriate function.

D. Computational models

Both EPC and UML diagrams are very useful and easy to understand, but to different categories of participants and at different levels of modelling [24]. Business stakeholder can hardly understand requirements and process logic in technical terminology, so MEDoV recommends usage of EPC modelling language for business processes modelling. The idea of linking procedural and object-oriented modelling, to take advantage of both approaches, is also accepted in this method, but only in order to understand better the user needs and to ensure correct, complete and unambiguous fulfilment of user requirements. The MEDoV method suggests the use of three types of UML diagrams:

- Use Case diagram is suitable for visualization of small and less structured business functions at a very detailed level, where it is not necessary to define the process workflow.
- Class diagram is used to describe the structure of input and output documents of EPC functions: classes, their fields, and methods, and connections between classes.
- Activity diagram is used to describe the process flow, as well as EPC models, so most EPC diagrams can be more or less easily converted into activity diagrams. Activity diagram should be used for description on a more detailed level.

E. User Suggestions and Remarks

User suggestions and comments can help detect process weak points and collect some good ideas for process optimization and improvement. Also, appreciation increases user satisfaction. Suggestions and comments have to be catalogued according to process/function.

VII. MEDoV – Phase III

A. Analysis

Analysing user requirements is an iterative process which requires constant feedback to elicitation requirements activities. The first analysis of requirements should be done already during Phase II, while the 'As-Is' models are modelled. 'As-Is' models are a good basis for analysing requirements, i.e. for clarifying ambiguities, detecting flaws (missing parts), defining domains and testing requirements feasibilities.

Potential conflicts of interest should be resolved in consultation with stakeholder. Requirements are prioritized according to business and project goals. During the analysis, various legal regulations risks should be identified.

B. Requirements Specification

The MEDoV unites the activity of developing requirements specifications with the activity of system modelling.

Requirements specification has to be comprehensible to business stakeholders, and has to enable changes of requirements in a simple way.

The 'To-Be' model is a proposed diagram of how we hope the future process to be. It is necessary to mark changes in 'To-Be' models in relation to 'As-Is' models.

Users can easily monitor the process flow, so these models are ideal for demonstrating work of the future system. 'To-Be' models are also very suitable for system testing and users training.

The same conventions are used for modelling both 'As-Is' and 'To-Be' models.

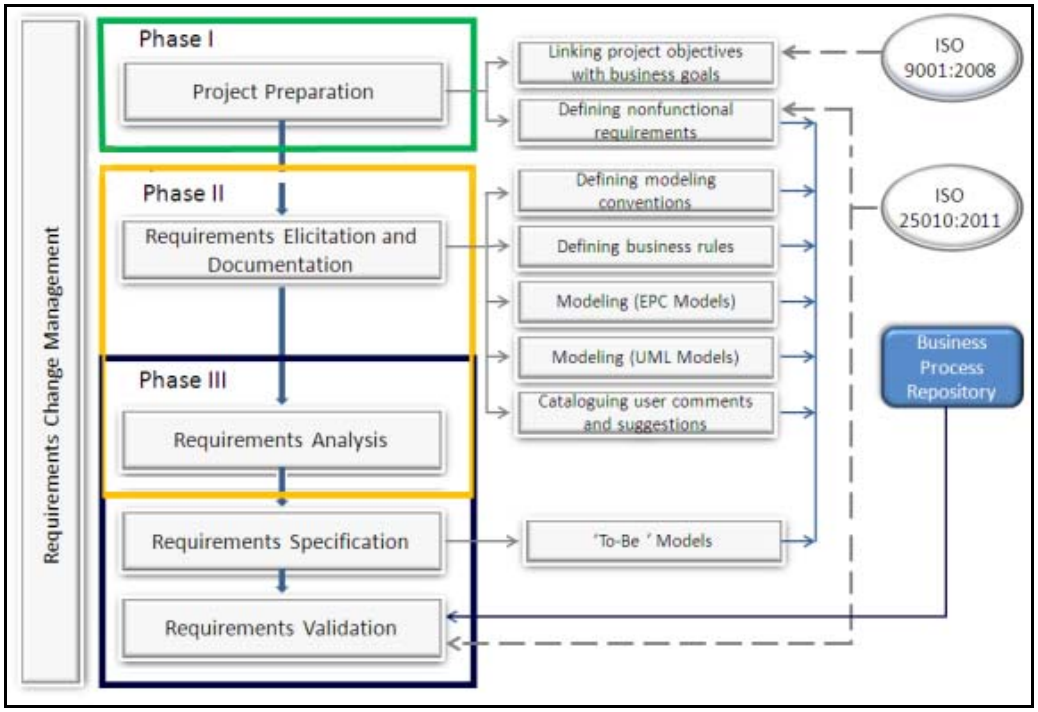

Figure 1 Method for Elicitation, Documentation and Validation of User Requirements (MEDoV)

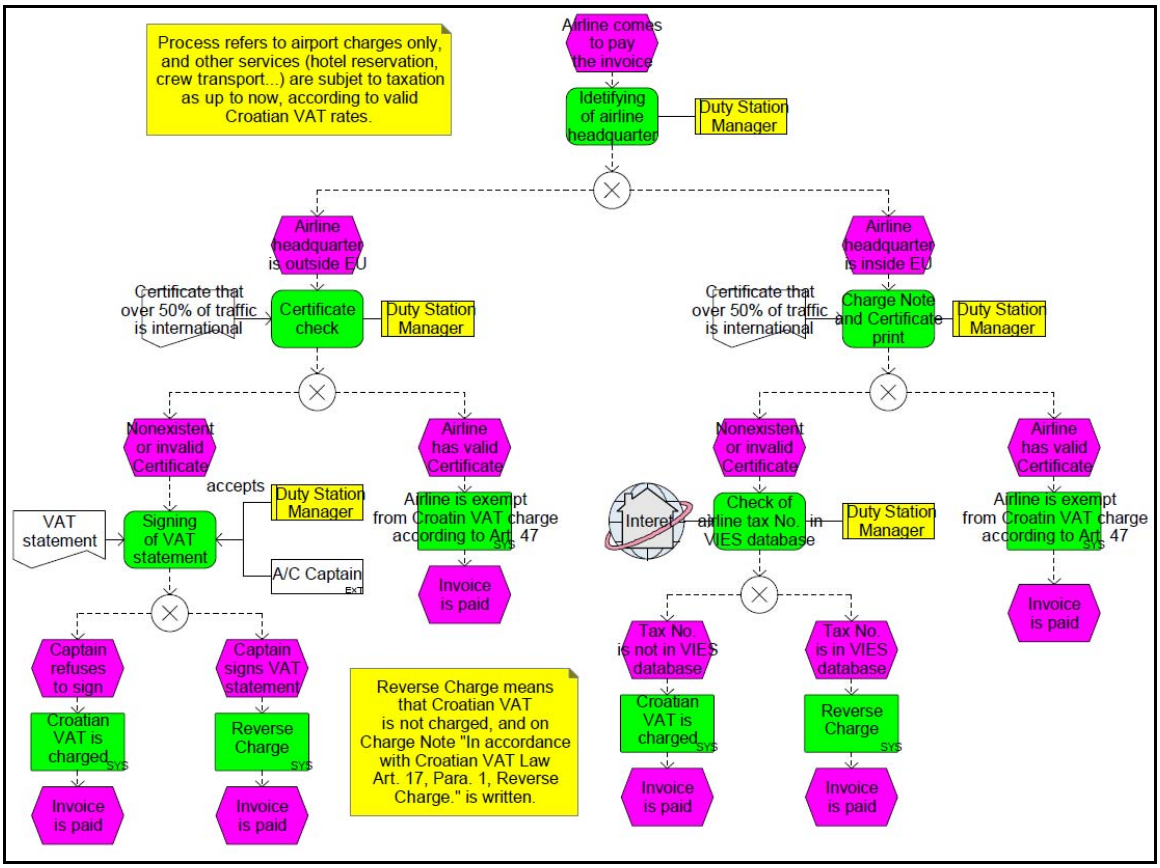

Figure 2 eEPC Model (part)

C. Requirements Validation

The purpose of new system development is to meet user specific needs. Requirements validation checks whether the requirements specification matches the user needs and wishes. Validation includes checking of correctness, completeness, consistency, realism and verifiability of stated requirements. Validation seams to overlap with analysis in the sense that it also deals with finding of problems in requirements, but it is extremely important, because incorrectly specified requirements can cause high costs. It is necessary to analyse and resolve conflicting requirements, to support stakeholder negotiations, and to reason with models that contain inconsistencies. KPIs are the basis for all previously mentioned activities.

D. The MEDoV and Lean Principles

Table I shows the manner in which the MEDoV supports lean principles.

VIII. Case Study

Integrated software system "Airport Software" supports Split Airport key business process, i.e. Passengers and Aircrafts Handling. Manufacturer estimates its size to approximately 50.000 function points. "Airport Software" is constantly updated and upgraded. The main problem of integrated system upgrade is that change in one part results as an unexpected change in other part. Agile development and the MEDoV method are applied and validated on upgrade project of "Airport Software" system.

Croatian accession to the EU changed the tax regulation. Now, any company from EU can choose whether to pay Value Added Tax (VAT) in Croatia or in its own country – as so called “Reverse Charge”.

By Croatian law, an airline is exempt from payment of Value Added Tax if the majority of its air traffic is performed to destinations outside Croatia. In that case, the airline must have a Statement certified by Croatian authority. If an invoice is paid by agent (a company which is not an airline, i.e. which has not the Air Operator Certificate), or by airline which does not have the Statement, VAT is charged:

- When the company is not listed in VIES database (the company is based in the EU).
- When the company does not sign VAT statement (the company is based outside the EU).

For individuals, VAT is always charged at rate prescribed by law.

Two methods of payment are possible: individual, when rendered services are paid prior to take off, and periodical, when calculation and charge of rendered services are performed in accordance with signed contracts. Procedures, documents, user interfaces and business logic are different for each method of payment. Depending on situation, payment can be realised at three different workplaces: Duty Station Manager, Cargo Officer and Commercial Sales Officer. Croatian law prescribes that all forms of payment, except bank transfer payments, must be fiscalized within 48 hours. If fiscalization is not successful, invoice is automatically resent. If invoice is not fiscalized within 24 hours, e-mail alerts are created. Several reports with three level access privileges are also created in this project.

Project team (Figure 3) started with project on 10th June, and ended on 21st June 2013. It lasted two sprints. Agile practices used on project are: night build, code and tests, collective code ownership, continuous integration, sustainable pace, features (statements are replaced by functions in model), incremental design, inspections, short releases, sit together, sprint, stand-up meeting.

Project is completed on time and within the estimated budget. Programmer productivity on this project was 1,04 function points per hour (FT/hour), which is slightly better than the average of 1,2 FT/hour in previous projects.

After the implementation (Table II), no errors and no unnecessary features have been found. Requirements specification models (Figure 2) have also been used for user training. Reactions of end users are very positive. Their general opinion is that the model facilitates change tracking in everyday work.

TABLE I HOW MEDOV SUPPORTS LEAN PRINCIPLES

Software Development Lean Principles	MEDoV Supports
Optimize the whole	Project goals Business process models
Eliminate waste	Business process models
Build quality in	MEDoV method
Learn constantly	Top-down approach User suggestions and comments
Deliver fast	Top-down approach
Engage everyone	User approach Business process models User suggestions and comments
Keep getting better	-

IX. Conclusion

The key cause of software project failures is the lack of clarity in eliciting and communicating user requirements. Agile and lean development methodologies require greater user involvement, but they lack structure in requirements engineering activities.

The MEDoV method ensures structure for RE activities with minimal impact on agility. The method also ensures active role of business user, a wide-picture what is usually a problem for agile development, and a definition of non-functional requirements, what is an even bigger problem.

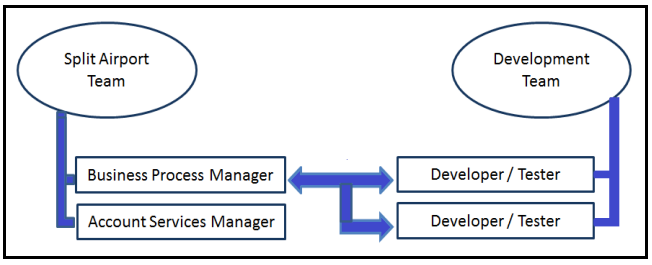

Figure 3 Project Team

TABLE II SOME FACTS ABOUT PROJECT

Some Facts about Project	
Project size (function points)	94
Project on time	Yes
Project within bugdet	Yes
Development duration (work days)	10
Number of errors (after implementation)	0
Unnecessary features (after implementation)	0
Programmer productivity on project	1,04 FP/hour

The MEDoV method uses business models of medium level of details combined with only three different UML diagrams, what produces a synergistic effect. This method allows users to avoid at least one level of detail in their business models, what represents a major time saving in requirements elicitation and documentation. The use of UML diagrams is limited to supplementing and elaborating of requirements, thus improving the understanding of RE process by business users.

Additional savings in business processes modelling are achieved by separating business rules, wherein the prioritization of business rules is made on the basis of prioritization of process parts.

The MEDoV ensures that no unnecessary features are produced, so that no extra code is created. The maintenance is easier, as well as code correction and improvement.

Models which are used for requirements specification also represent the documentation of the system, meaning that there is no need for additional documentation. Using models enables agility in product maintenance, especially for integrated systems where one change can have multiple impacts on different parts of system.

The MEDoV method has been used to elicit, document and validate user requirements for agile software development project at Split Airport.

REFERENCES

[1] B. Boehm, "Get ready for agile methods, with care", IEEE Computer, Vol. 35, No. 1, 2002, pp. 64-69.

[2] M. Poppendieck, and T. Poppendieck, “Lean software development”, Addison-Wesley, 2003.

[3] P. Rodríguez, J. Markkula, M. Oivo, and J. Garbajosa, Analyzing the drivers of the combination of lean and agile in software development companies, PROFES 2012, LNCS 7343, 2012,.pp. 145–159.

[4] A. Sillitti, and G. Succi, “Requirements engineering for agile methods engineering and managing software requirements”, Part 2, Springer Berlin Heidelberg, 2005, pp 309-326, DOI: 10.1007/3-540-28244-0_14.

[5] W. Helmy, and A. Kamel, O. Hegazy, “Requirements engineering methodology in agile environment”, IJCSI ‘12, Vol. 9, Issue 5, No 3, September 2012, pp.293-300, ISSN (Online): 1694-0814.

[6] M. Hammer, “Beyond reengineering: how the processed-centered organization is changing our work and our lives”, Harper Business, New York, 1996.

[7] M. Kohlbacher, “The effects of process orientation on customer satisfaction, product quality and time-based performance”, 29th Intern. Conference of the Strategic Management Society, Washington DC, October 2009.

[8] Gartner, “Business Process Management (BPM) Key Initiative Overview“, 2011, https://www.gartner.com/doc/1746423/business-process-management-bpm-key, *Accessed:2014-01-20.*

[9] Software AG, “Intelligent Guide to Enterprise BPM, Chapter 2: Business Process Analysis—Transforming your Business by Transforming your Processes“, June 2012, https://www.softwareag.com/corporate/rc/rc_perma.asp?id=tcm:16-99950, *Accessed:2014-02-06.*

[10] A. Fouad,, K Phalp, J.M Kanyaru, and S. Jeary, “Embedding requirements within Model-Driven Architecture”, Software Quality Journal, Volume 19, No. 2, 2011, pp.411–430, doi: 10.1007/s11219-010-9122-7.

[11] S. Dragicevic, S. Celar, and L. Novak, “Roadmap for requirements engineering process improvement using BPM and UML”, Advanced in Production Engineering & Management (APEM), Vol. 6(3), Sept. 2011, pp.221-231.

[12] S. Dragicevic, and S. Celar, “Method for elicitation, documentation and validation of software user requirements (MEDoV)”, 18th IEEE International Symposium on Computers and Communications (ISCC 2013), 7-10.07.2013, pp. 956-961.

[13] K. Beck, M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, et al., “The agile manifesto”, http://www.agilealliance.org/the-alliance/the-agile-manifesto/, 2001, *Accessed:2014-02-18* .

[14] L. Williams, “Agile software development methodologies and practices”, Advances in Computers, Vol. 80, 2010, pp. 1-44.

[15] J. Nawrocki, M. Jasiński, B. Walter, and A. Wojciechowski, “Extreme programming modified: embrace requirements engineering practices,” Proc. IEEE Joint Int’l Conf. Requirements Eng., IEEE CS Press, 2002, pp. 303–310.

[16] M. Poppendieck, and M. A. Cusumano, “Lean software development: a tutorial”, IEEE Software, Volume 29 Issue 5, september/october 2012, pp.26-32.

[17] A. Abran, J. Moore, P. Bourque, and R. Dupuis, “SWEBOK: guide to the software engineering body of knowledge“, 2004 Version. IEEE Computer Society, California, 2004.

[18] A. Felfernig, M.Schubert, M. Mandl, F. Ricci, and W. Maalej, “Recommendation and decision technologies for requirements engineering”, RSSE '10:, 2010, pp. 11-15.

[19] C. Monsalve, A. April, and A. Abran, “Requirements elicitation using BPM notations: focusing on the strategic level representation,“ ACACOS'11:, 2011, pp. 235-241.

[20] T. Blickle, and H. Hess, “Automatic process discovery with ARIS Process Performance Manager (ARIS PPM)”, October 2010, http://cdn.ariscommunity.com/documents/urelation/SAG-AEP_PPM-Automatic_Process_Discovery_with_PPM.pdf, *Accessed:2014-02-06*

[21] A.W. Scheer, O.Thomas, and O.Adam,“Process modeling using Event-Driven Process Chains“, Process-Aware Information Systems, edited by Dumas, van der Aalst and ter Hofstede, John Wiley and Sons, Hoboken, 2005.

[22] T.T. Pham Thi, M. Helfert, F. Hossain, and T. Le Dinh, “Discovering business rules from business process models“, CompSysTech '11, June 2011, pp. 259-265.

[23] J. Polpinij, A. K. Ghose, and H. K. Dam, “Business rules discovery from process design repositories“, SERVICES '10, July 2010, pp. 614-620.

[24] R. Islam, R. Islam, S. Alam, and S. Azam, “Experiences and comparison study of EPC & UML for business process & IS modeling“, Inter. Jour. of Comp. Science and Information Security, Vol. 9(3), March 2011, pp.125-133.