EFFICIENCY OF SOFTWARE DEVELOPMENT AFTER IMPROVEMENTS IN REQUIREMENTS ENGINEERING

David Kuhlen[1] and Andreas Speck[2]

[1]Datenlotsen Informationssysteme GmbH, Technical Consultant, Beim Strohhause27, 20095 Hamburg, Germany.
[2]Christian Albrechts Universität zu Kiel, Head of the Business Information Technology Group, Hermann-Rodewald-Straße 3, 24098 Kiel, Germany

ABSTRACT

In the past decade multiple challenges arose from the method of software development [4, 4]. As described by Davenport, the development process needs an overhaul [4, 4]. Different disciplines, like project management, requirements engineering, the development of code or quality assurance have been investigated intensively, in order to improve the productivity of development. To obtain valid results, the overhaul needs to start with the refactoring of the right process at first. Often, it is sensible to start with such processes, which operate at the interface to the customer, because they are perhaps the most critical to an organization's success [3, 270 – 271]. Mainly, software development consists of four sub processes: requirements engineering, development, quality assurance and delivery. Requirements engineering and delivery operate on the interface to the customers. Because of the fact that the analysis of requirements is groundbreaking, we select this process as the starting point of a process innovation initiative. We analyse the impact of requirements engineering in KANBAN development processes. Special emphasis is put on the productivity of the overall development process, after a refactoring of requirements engineering.

KEYWORDS

Software development, Requirements engineering, Operating efficiency.

1. INTRODUCTION

Software development is criticized as being too slow, too expensive and too error-prone [1], [4, 4]. This requires an overhaul of the process [4, 4]. Quality and cost problems emerge from the way how software is usually developed [4, 4].

In agile projects, developers may decide to skip (or shorten) a comprehensive requirements engineering [13]. Unfortunately, in many cases requirements are incomplete and the projects lack of specification. Software producers speculate to save effort in order to finish projects faster by reducing requirements engineering [5]. This reduction is often justified by the fact that requirements are changing continuously. [5] Explains that one to three percent requirements change per month! "Embrace the Change" leads to requirements that are only as far as necessary analysed - and important details are missing to deliver the right quality.

However, requirements engineering is a long-term goal. It is not attractive, particularly in short iterations! Nevertheless, good requirements engineering methods help to reduce the error rate of a

Software system from 0,23 to 0,08 per function point [5]. Early investments in good quality prevent rework - this stops the emerge of waste in (lean development) [12, 143]. Lean approaches like KANBAN require a reliable quality, in order to behave [6, 186]
.

Lean production principles seek to eliminate all kinds of waste from the development [7, 1]. Ikonen et al. declared the adaption of lean methods in software industries to be one of the newest fashions in 2010! The adaption of lean techniques by software manufacturers is economically reasonable. Compared with others, KANBAN leads to an economic ratio of costs to earned function points [9, 11].

However if details are missing about the impact, requirements engineering has on the profitability in KANBAN. We try to investigate the advantages of a comprehensive requirements engineering in iterative development.

2. THEORY

Requirements analysis should provide a basic understanding about customers needs [2]. In order to analyse requirements state diagrams and activity diagrams will be created [5]. The creation of such artefacts could be meaningful - however, it requires some effort.

Stutzke distuniguishes three steps of analysis: (1) logical design describes the problem,(2) physical design adds the description of implementation details and (3) the production process produces the software [15]. The aim of contemporary requirements nearly corresponds to these logical design, described by Stutzke[15].

KANBAN is a pull-system, used in production to organize the sequence and communication of the procedure model in manufacturing [6, 169]. Mathematically, KANBAN consists of a stock in combination with policy rules (Q; r) [6, 186]. Especially in repetitive environments, KANBAN offers simplicity to control the production effectively [6, 169]. Software development operates in a repetitive environment [8, 2]. In general, the productivity of KANBAN is remarkable [7, 1], [9, 11], [6, 169]! To enhance the productivity, requirements specifications have to be qualitative [11, 1]. The quality of requirements influences the progression of development.

3. METHODS

In order to investigate the profitability of requirements engineering, we need to analyse its impact on the process model. The model of KANBAN consists of a few different activities. In practice, development can be organized by using a pinboard. On this pinboard, companies distinguish phases like requirements engineering, development, quality assurance. Requirements are written on notepads and shifted among these phases. They are put from left to right until quality assurance is finished. The number of requirements which could be performed in the different activities is limited [6, 169, 173].

Figure 1 illustrates a common version of the KANBAN development process. The embedded check if it is possible to move an instance forward is important in the process model. It is possible that requirements have to stay in their current activity, until the load factor of the next activity allows the handling of a new requirement. This stabilizes the workload!

To answer the above mentioned question, we need to consider quality expectations during the development. Software development is an iterative process. As Speck et al. explained, it is very desirable to identify errors and problems in an early state [14, 75–82]. It is strongly required, to identify problems before software is deployed to the customer [16, 6].

The cycles in software development are necessary, because often quality lacks after first iteration. Sometimes, it is necessary to repeat a few cycles, in order to arrive at an effective

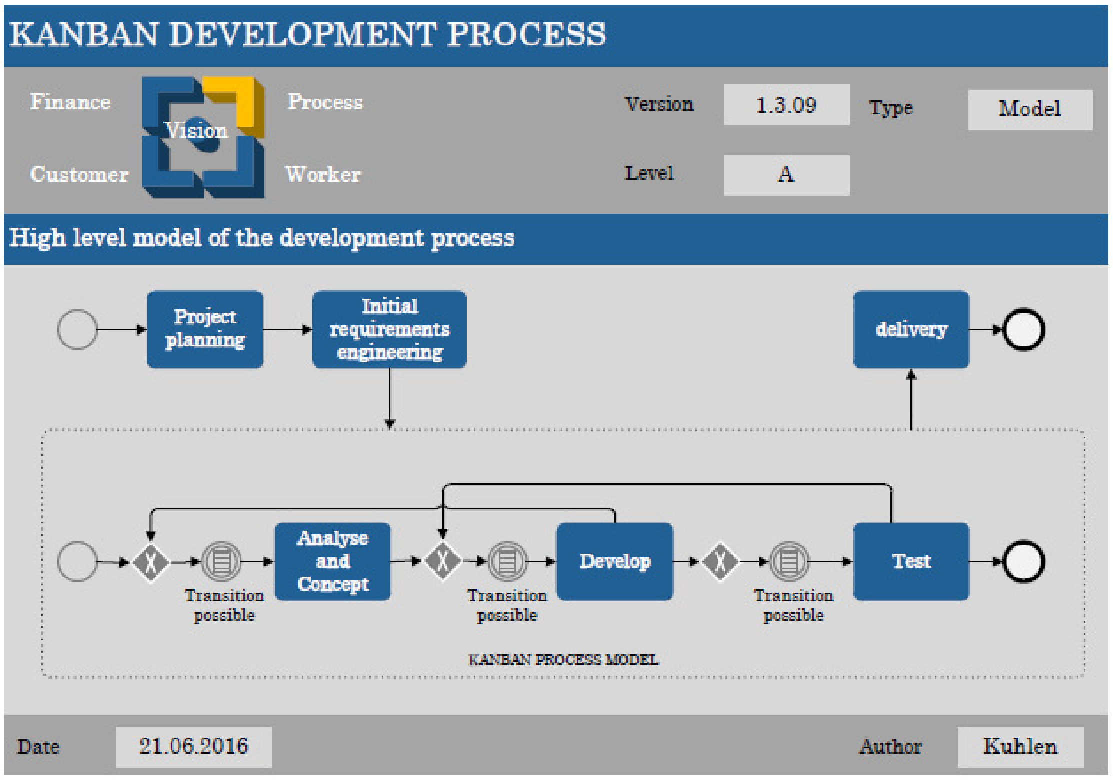

Figure 1: Kanban process model

Design of (target-) processes [3, 158]. During each cycle, we assume that activities which handle a requirement, improve its quality. Furthermore, we add a compliance rule which controls the transition of a requirement from one activity to another. Only if requirements quality fulfils the rule, it is possible to start the next activity. If quality is insufficient, the requirement has to repeat parts of the process.

Requirements analysis takes place mostly at the beginning of an iteration. Often, it does not result in a fully formulated specification. Thus, the risk of a never ending evelopment process rises! Moreover, a lack of clear specifications may lead to further customer requests and higher costs. The cyclic repetitions particularize the requirements specification [2]. In order to determine the impact of requirements engineering, we performed multiple experiments. Each experiment was performed on the basis of a KANBAN development process. Sole difference between the experiments is the impact requirements engineering has on the quality of specifications.

4. DATA

Each experiment consists of multiple simulations. A simulation describes the development of a team, during six months. Therefore, a progression has 57.601 (one minute in reality conforms to one round). After each progression, key performance indicators were calculated. Process innovation initiatives (like an investment in requirements engineering) needs to improve the financial performance [3, 4]. Therefore, such an improvement of the process has to be economically sensible!

At the beginning of each simulation, a generator creates random requirements. These requirements have any quality. The quality is described as a percentage value which expresses how well the object (requirement, software, etc...) fulfils the wish of the customer. Higher percentage values express better performance. Furthermore, each requirement has an individual size. The size is described as a number of function points. The size determines how long the performance takes in an activity.

Different activities were executed by different employees during the process. The activity combines the requirements (which have a duration) and the employees (which have a salary). This combination (allocation of resources to activities) facilitates to describe the economic performance, for example, by the calculation of process costs [10, 1781].

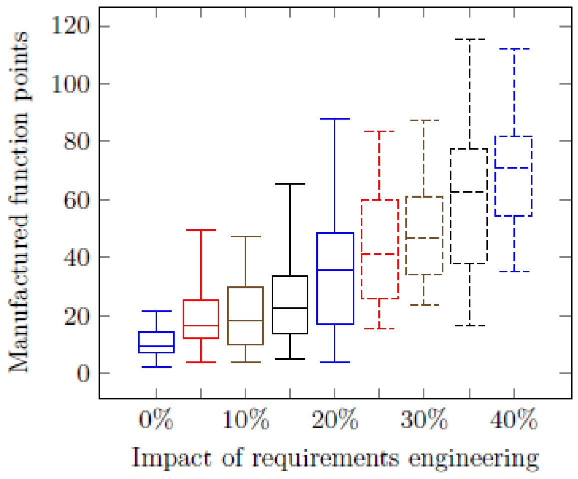

Figure 2: Produced function points, influenced by requirements engineering's impact

Diagram 2 illustrates the relation of requirements engineering efficiency and the produced function points. Multiple simulations belong to the same class. A class bundles different simulations with an equal improvement of requirements engineering. Nine different classes could be distinguished. The impact of requirements engineering is increased from class to class about 5 %. The produced function points of each class are compared by the use of box plot charts. In diagram 2, the correlation of requirements engineering efficiency to realized function points is remarkable. Greater efficiency of requirements engineering increases the number of produced function points. By comparing the median of different classes, it shows the positive correlation. Our assumption seems to be confirmed.

5. Results

In order to investigate our leading question, we calculated a regression analysis. We compared the impact of requirements engineering (axis of abscissas) to the produced function points (ordinate). Both characteristics belong to a cardinal scale (ratio scale). We performed 522 simulations. After each simulation a key performance indicator was calculated. Our calculations base on 520 degrees of freedom. We used an α probability of 5%.

$$n = 522;\ p = 2;\ df_x = 520;\ df_y = 520$$

Expectable about 37, 88 function points could be produced during each simulation (arithmetic average). The median of developments output is about 31,64 finished function points. Its 25.-quantile is 13,49 fp and its 75.-quantile is 56,59 fp.

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i = \frac{1}{522}\sum_{i=1}^{522} x_i = 19,8659$$

$$\overline{y} = \frac{1}{n}\sum_{i=1}^{n} y_i = \frac{1}{522}\sum_{i=1}^{522} y_i = 37,8813$$

The variance of produced function points is about 914; 68fp^2. This is in accordance with a standard deviation of 30,24 function points. The standard error of produced function points (leaving out degrees of freedom) amounts to 1,32. With a probability of 90% no more than 115,91 function points will be produced during each simulation (confidence interval). It is expactable that in the population *w* the variance of finished function points conforms to 916; 44fp^2

$$s_x^2 = \sum_{i=1}^{n}(x_i - \overline{x})^2 = \sum_{i=1}^{522}(x_i - 19,8659)^2 = 165,9782$$

$$s_y^2 = \sum_{i=1}^{n}(y_i - \overline{y})^2 = \sum_{i=1}^{522}(y_i - 37,8813)^2 = 914,6817$$

$$s_x = \sqrt{s_x^2} = \sqrt{165,9782} = 12,8833$$

$$s_y = \sqrt{s_y^2} = \sqrt{914,6817} = 30,2437$$

The application of the Cauchy-Schwarz inequality shows that the covariance is less than 389,64 (formula: $| c_{xy} |\leq s_x.s_y$ -> $|c_{xy}|\leq$ 389; 64). The covariance amounts to 259,31. With about 0,67 the correlation coefficient is more than of medium height. This positive correlation coefficient indicates an equal tendency between the characteristics.

$$\begin{aligned} c_{xy} &= \frac{1}{n}\sum_{i=1}^{n}(x_i-\overline{x})(y_i-\overline{y}) \\ &= \frac{1}{522}\sum_{i=1}^{522}(x_i-19,8659)(y_i-37,8813) \\ &= 259,3068 \end{aligned}$$

$$\begin{aligned} r_{xy} &= \frac{c_{xy}}{s_X \cdot s_y} \\ &= \frac{259,3068}{12,8833 \cdot 30,2437} \\ &= 0,6655 \end{aligned}$$

Our model fits to an univariate linear regression. We used the method of the least squares to calculate the best fitting regression line. Our function has a slop of about 1,56 and a y-intercept of about 6,84. Theoretically, the function has a simple zero at -4; 38% which isn't reachable in practice.

$$\begin{aligned} f(x) &= \beta \cdot x + u \\ &= \frac{c_{xy}}{s_x^2} \cdot x + (\overline{y} - (\frac{c_{xy}}{s_x^2} \cdot \overline{x})) \\ &= \frac{259,3068}{165,9782} \cdot x + (37,8813 - (\frac{259,3068}{165,9782} \cdot 19,8659)) \\ &= 1,5623x + 6,8449 \end{aligned}$$

The reliability (ration of explained variance to total variance) amounts to about 0,443. Adjusted by the number of parameters, it amounts to 0,441. Based on this reliability, the quality of the regression is moderate. Further tests are necessary.

$$R^2 = r_{xy}^2 = 0,6655^2 = 0,4429$$

$$\begin{aligned}\overline{R^2} &= 1-(1-R^2)\frac{n-1}{n-p-1}\\ &= 1-(1-0,4429)\frac{522-1}{522-2-1}\\ &= 0,4408\end{aligned}$$

The adjusted reliability suggests that the analysis of the regression should be intensified. In order to determine whether the regression is significant, we calculated the f-test. The f-test allows to check if the correlation of two parameters is also significant also in the basic population *w*. Based on the above mentioned α probability, we determined the critical value of the F-distribution to be about 1,16. We calculated a check sum f, of about 5,51. This check sum exceeds the critical value of the F distribution. Therefore, a significant correlation exists in the basic-population.

$$\begin{aligned}F &= \frac{MAX(s_x^2, s_y^2)}{MIN(s_x^2, s_y^2)}\\ &= 5,5109\end{aligned}$$

$$\begin{aligned}F(\alpha|df_x, df_y) &= F(0,05|520,520)\\ &= 1,1553\end{aligned}$$

$$\begin{aligned}&\frac{MAX(s_x^2, s_y^2)}{MIN(s_x^2, s_y^2)} > F(\alpha|df_x, df_y)\\ &= 5,5109 > 1,1553 \rightarrow significance!\end{aligned}$$

We seek determine, if the independent variable (impact of requirements engineering) influences the dependent variable (finished function points) significant. Within a t-test, we compared the hypothesis that the slop of our regression line is zero, in reality. The variance of the residues is about 511,53. Adjusted under consideration of the degrees of freedom, a standard error of about 0,992 could be estimated in the basic population. This leads to a variance of the slop of about 0,077. The calculation of the t-statistic-value amount to about 20,29. The t-statistic-value exceeds the critical value of the T-distribution (about 1,96, considering a two sided interval). Requirements engineering has a significant impact in the model on the finished function points.

$$\begin{aligned}\hat{\sigma^2} &= \frac{n}{n-p}\cdot(s_y^2-\beta\cdot c_{xy})\\ &= \frac{522}{522-2}\cdot(914,6817-1,5623\cdot 259,3068)\\ &= 511,5281\end{aligned}$$

$$\hat{\sigma} = \sqrt{\hat{\sigma^2}} = \sqrt{511,5281} = 22,617$$

$$MSE = \sqrt{\frac{\hat{\sigma^2}}{n-p}} = \sqrt{\frac{511,5281}{522-2}} = 0,9918$$

$$\sigma_{\hat{\beta}} = \frac{MSE}{s_x} = \frac{0,9918}{12,8833} = 0,077$$

$$t = \frac{\beta}{\sigma_{\hat{\beta}}} = \frac{1,5623}{0,077} = 20,2934$$

$$(T(\alpha, df) = T(0,05,520) \leq 1,9645)$$
$$< (t = 20,2934) \rightarrow significance!$$

The positive contribution of requirements engineering to the productivity is proven by the regression analysis. Next, we need to analyse the workload of the development progression. If the improvement increases the workload, negative cost effects could emerge. We expected the workload to be stable, because of the positive levelling effect of KANBAN.

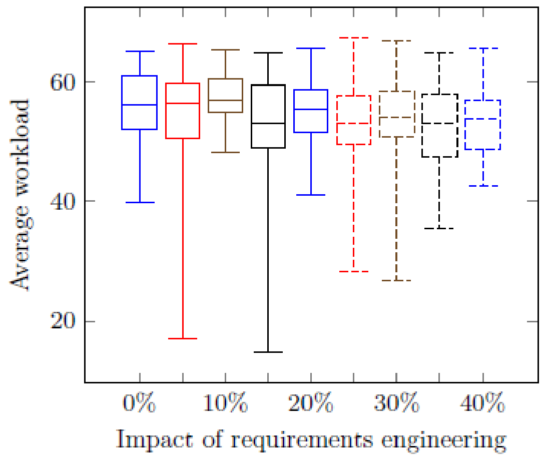

Figure 3: Workload influenced by the impact of requirements engineering

In figure 3 the relation between requirements engineering efficiency to the workload is displayed. More or less the workload is stable in different classes of efficiency improvement.

6. CONCLUSION

Unfortunately, the implementation details determine the productivity. Techniques of the requirements engineering are necessary in development projects. The lack of formal specifications is a challenge in projects [2]. Software producers could increase its output by enhancing the efficiency of requirements engineering. Enhancements of requirements engineering lead to increased numbers of finished function points. It can be expected that the productivity of requirements could not be improved beyond an upper bound. We restricted this upper bound to be 40 %.Furthermore, an upper bound of requirements quality exists (in our model 100 %).In further investigations it could be sensible to determine the maximum limit until an improvement is economically meaningful. This analysis has to consider the unique costs of process innovation, necessary for the improvement of requirements engineering.

7. REFERENCES

[1] P. G. Armour. The business of software estimation is not evil. COMMUNICATIONS OF THE ACM, 57(1):42–43, 2014.

[2] Lan Cao and Balasubramaniam Ramesh. Agile requirements engineering practices:An empirical study. Software, IEEE, 08(1):60–67, February 2008.

[3] Thomas H Davenport. Process Innovation - Reengineering Work through Information Technology. Havard Business School Press, Boston, Massachusetts, 1993. Ernst & Young Center for Information Technology and Strategy.

[4] Thomas H Davenport. The coming commoditization of processes. Harvard business review, 83(6):100–108, 2005.

[5] Dr. Peter Hruschka. Business Analysis und Requirements Engineering. Carl Hanser Verlag München, 2014.

[6] Chun-Che Huang and Andrew Kusiak. Overview of kanban systems. INT. J. COMPUTER INTEGRATED MANUFACTURING, Vol. 9(No. 3):169–189, 1996.

[7] Marko Ikonen, Petri Kettunen, Nilay Oza, and Pekka Abrahamsson. Exploring the sources of waste in kanban software development projects. In 2010 36th EUROMICRO Conference on Software Engineering and Advanced Applications, pages 376–381. IEEE, 2010.

[8] David Kuhlen and Andreas Speck. Business process analysis by model checking. In 5th International Symposium on Data-Driven Process Discovery and Analysis SIMPDA 2015, pages 154–170, Vienna, Austria, December 2015. Ceravolo, Paolo and Rinderle-Ma, Stefanie.

[9] David Kuhlen and Andreas Speck. Magnitude effect of requirements in the development process. May 2016.

[10] Huiping Lin, Yushun Fan, and Stephen T Newman. Manufacturing process analysis with support of workflow modelling and simulation. International Journal of Production Research, 47(7):1773–1790, 2009.

[11] Carlos Monsalve. Business process modeling with levels of abstraction. IEEE COLCOM 2015, (978-1-4799-8834-1), 2015.

[12] Hans-Jürgen Plewan and Benjamin Poensgen. Produktive Softwareentwicklung: Bewertung und Verbesserung von Produktivität und Qualität in der Praxis, volume 1. Auflage. dpunkt. verlag, Heidelberg, Germany, Juli 2011.

[13] J. Schwaber, K.; Sutherland. Der scrum guide der gültige leitfaden für scrum: Die spielregeln, July 2013. Zuletzt Abgerufen am 06.09.2014.

[14] Andreas Speck, Elke Pulvermüller, and Dirk Heuzeroth. Validation of business process models. In Proceedings of ECOOP 2003 Workshop Correctness of Model-based Software Composition (CMC), pages 75–83, 2003.

[15] Richard D Stutzke. Possible uml-based size measures. In Presented at the Thirteenth International Forum on COCOMO and Software Cost Modeling, Los Angeles, 6-8 October 1998., 1998.

[16] Wil MP Van Der Aalst, Arthur HM Ter Hofstede, and Mathias Weske. Business process management: A survey. In Wil M.P. van der Aalst, Arthur ter Hofstede, Mathias Weske, Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen, editors, Business process management, Lecture Notes in Computer Science, pages 1–12, Einhoven, The Netherlands, 2003. Interational Conference BPM, Springer.

[17] Thomas Zink. Class for aircc journal submissions, 2012. This is an unofficial Latex class for Authors of AIRCC Papers. Access timestamp: 2016-10-19.