APPLICABILITY OF VALUE STREAM MAPPING TO SOFTWARE DEVELOPMENT CONTEXT

Maria Dahlman, Ulrika Olsson and Hakan Akillioglu

Industrial Production and Management, The Royal Institute of Technology – Stockholm

dahlma@kth.se, uolss@kth.se, haaki@kth.se

Abstract: The study investigates the applicability of Value Stream Mapping in a specific software development context through a case study. Practically this is done by a transformation of some fundamental lean concepts from production systems into their counterparts in the software development environment. The proposed transformation is experimented by means of a case study in an ICT company in Sweden. The value stream of two software components are studied by applying the lean tool Value Stream Mapping and the results indicate that the lean tool of Value Stream Mapping is applicable to the software development context with the aim of decreasing wastes and improving the efficiency of the value streams.

Key words: Agile, Lean, Software Development, Value Stream Mapping

1. INTRODUCTION

In recent years, software development has mostly evolved around an agile way-of-working to improve the product development process and the operational success of software development companies (Dybå and Dingsøyr, 2008). However, the agile approach has been argued to lack in providing the big picture, which is considered as fundamental in software delivery (Biffl, et al., 2005). Thus, to improve the methods and processes in software development, companies have started to look towards lean practices since the lean approach, in contrast to agile, has a superior holistic approach. On the other hand, applying the lean principles to a software development context is not straightforward since its application depends highly on the context and objectives of the environment (Poppendieck and Poppendieck, 2007; Staats, et al., 2011). The lean tools are primarily developed by and for the manufacturing industry, therefore its principles and practices are adjusted to their repetitive, visible and somewhat rigid production processes (Staats, et al., 2011). However, by adjusting the lean practices to the nature of a specific context, substantial improvements can be attained at the non-manufacturing industries as well (Wang, et al., 2012; Song, et al., 2009; Åhlström, 2004).

The lean concept in the area of software development is a new term that has not yet fully examined in the literature about lean or lean product development. Thus, there is a need of further studies in the applicability of the lean philosophy beyond the manufacturing context (Wang, et al., 2012). The purpose of this study is therefore to investigate the applicability of Value Stream Mapping in a specific software development context by performing a case study. The aim is therefore to evaluate the value contribution of the lean practice when it is adjusted to the nature of software development. In this sense, a thorough analysis is performed in order to characterize the lean principle from a software development perspective in the next section. Being linked to this analysis, the conversion of the relevant lean concept into the software development area is presented in section 3. Its implementation over two software components is presented in section 4 and in the last section concluding remarks are given.

2. THEORETICAL FRAMEWORK

2.1. The background and nature of software development

The software development industry constantly becomes more intense and competitive. Thus, the ability to develop and deliver software solutions in the most efficient way has become a desired target for any company in the industry (Biffl, et al., 2005). In this sense, agile software development paradigm relies on values and principles that enable the development process to respond flexibly and quickly to changes in customer needs (Petersen, 2010). The essence of the agility is presented in the Manifesto for Agile Software Development. This manifesto is a result of 17 software practitioners, which followed a similar software development method, who rationalized a common philosophy named as "Agile" (Misra, et al., 2012). The values and practices form a framework for agile software development that is characterized by short iterative cycles, quick frequent feedback from customers and constant learning. However, the agile concept mainly deals with the activities related to the limited development processes but not with other supporting activities in the development process (Cohen, et al., 2003).

Recent academic research regarding what makes software development companies successful in today's society highlights the importance of meeting customer needs on time and properly delivering value (Hodgetts, 2009). To ensure the fulfillment of the customer needs, the focus cannot only be on improving the product development processes (Poppendieck and Poppendieck, 2007). All steps in the process must be taken into consideration however the tools and frameworks in the agile development approach mostly cover the actual development phase. Hence, researchers state that the popular agile methods are too narrow and feature development oriented to use by themselves, especially when striving towards developing and delivering software solutions that adequately meets customer demands and ensures value to the end-user (Cohen, et al., 2003).

This insight has turned software development companies towards looking at the advantages of investigating the development process as a whole. Therefore the requirement of a holistic approach for improving the delivery of software has resulted in an increasing interest into lean thinking (Musat and Rodríguez, 2010) since it naturally adopts a holistic approach and it is focused on improving processes as a whole (Hodgetts, 2009). However the applicability of lean in other industries than manufacturing has been questioned (Staats, et al., 2011; Poppendieck and Poppendieck, 2007) even though there are studies presenting successful lean implementations in other industries (Åhlström, 2004; Song, et al., 2009; Bowen and Youngdahl, 1998). It turns out that adjustment of the principles and practices to fit the characteristics of the specific situation is essential (Wang, et al., 2012). Hence, both wider and deeper studies on how the lean concept should be adjusted and applied successfully in non-manufacturing contexts are demanded in order to enhance the conceptual basis regarding the applicability of the lean. This study aims to contribute to this existing knowledge gap by investigating the feasibility of the particular lean concept of Value Stream Mapping into a specific software development process.

2.2. Applicability of Lean in software development

The term "Lean" is not clearly defined in the literature and the available definitions differ. However, in contrast to this academic disagreement, researchers fully agree on that the focus of the lean applications traditionally have been put on production companies. More specifically, on high-volume, low variety suppliers i.e. mass producers (Åhlström, 2004). Due to the fact that the lean thinking is developed by and for the manufacturing industry its principles and practices are to a high extent created to fit repetitive, visible and rigid production processes (Staats, et al., 2011). This allows researchers to question the applicability of the philosophy in other contexts than manufacturing (Poppendieck and Poppendieck, 2007; Staats, et al., 2011; Womack and Jones, 1996). A well suited fit to the manufacturing context also hampers the possibility to successfully apply the methods in Lean directly to organizations and processes that differ from manufacturing (Wang, et al., 2012). Software development and manufacturing are essentially different. Manufacturing value streams include flows of both physical goods and information (Staats, et al., 2011). In contrast, a software development value stream rarely includes flows of material. Software development is mostly about producing a product that consists of code i.e. a non-physical product. The product is in the mind of the developers or stored in a computer and cannot be collected as traditional inventory. This makes it more difficult to follow the flow of the product throughout the value stream and it hinders the understanding and definition of measurements in the process.

A development process where the product evolves in people's minds through the manipulation of information is classified as knowledge work. In the knowledge work information is the basis of value. The information flow should reach the right person in the right form at the right time (May, 2005). Therefore the tasks often require both a high-level perspective and understanding for low-level details (Staats, et al., 2011). This results in a process with a high level of demand and supply uncertainty in comparison to manufacturing processes. This higher

uncertainty complicates the implementation of the lean philosophy (Lee, 2002). The complexity and uncertainty stem from the frequently changing customer requirements regarding the product characteristics (Poppendieck and Poppendieck, 2007). This hampers the process of defining what value is, as the customers rarely know exactly what they want. Additionally, when customers are using new software, their desires tend to shift from the starting conditions (Poppendieck and Poppendieck, 2007). In contrast to manufacturing, this makes repetitive tasks rare in software development processes, which makes standardization of the process difficult (Staats, et al., 2011).

2.3. Lean product development

The value streams that were investigated in this case study include activities related to development, design and execution of concepts related to a specific product or value offering. Hence, not only theory regarding Lean Production but also Lean Product Development (LPD) is relevant to consider while performing the study. The term Lean Product Development is referred to in situations when Lean principles are applied to product development with an aim of develop new or improving current products (Mynott, 2012). LPD includes everything in the development process from idea generation to development and evaluation of concepts. Hence, all activities that take place before the manufacturing process is considered when embracing LPD (Mynott, 2012).

However, implementing LPD demands special consideration according to Nepal et al. (2011) and Reinertsen and Schaeffer (2005). This as the processes included in product development involves the human factor and thus cannot be standardized to the same extent as when lean is applied to a manufacturing process. Though, previous studies state that LPD can contribute with faster time-to-market, fewer engineering hours and higher quality of products. (Karlsson and Åhlström, 1996).Since LPD is sprung from lean production research regarding its applicability is not extensively explored. Additionally, its concepts are fuzzy and no basic principles to follow exist. (Schuh et al., 2008). Further studies on how to implement LPD in different situations are therefore demanded.

2.4. Lean implementations in software development

Even though manufacturing and software development are different in essence, recent research indicates that the applicability of lean concepts might be more extensive than one first think. Several case studies show that it can be applied successfully in software development companies.

Åhlström (1998) suggests that in the early phases focus should be on finding and eliminating defects and delays. When zero defects are achieved it is proposed that focus should shift towards continuous improvements. However, Åhlström (1998) does not promote any hands-on method for which tools to use and in what order they should be used. Furthermore, the benefits from following the advice presented in the article are not assessed. Karlsson and Åhlström (1996) take another approach and the focus is on implementing "quality circles" i.e. small groups of people who regularly meet and discuss possible improvements. This method relies on the involvement among employees and gives some hints on lean implementations. Though, their method lacks an explanation regarding what tools to use and how to measure the effects. Detty and Yingling (2000) suggest an ambitious method that relies on digital simulations of the current and future state of the processes. This results in a detailed visualization of a possible future performance. However, the method is costly and requires a deep analysis of the processes as a first step of preparation. Rother and Shook, (2003) states that Value Stream Mapping (VSM) is a suitable method to use as a first step for any organization that wants to implement the lean concept. VSM indicates what and when a tool should be applied, as well as the current- and future-state map giving a hint of the effects given by implemented changes. On the other hand VSM does not consider any issues with the cultural change, which is an unavoidable issue in an organizational transformation. Finally, Productivity Press (2002) proposes a method that starts with creating awareness throughout the company about the implementation of the lean being performed. Thereafter 5S is used. The 5S implementation is then followed by the usage of several lean tools. The study presents a basic framework for how to implement the lean concepts however it lacks the details regarding the type and the order of the tools to apply.

The method claimed as the most suitable and comprehensive by several researchers, during the first step of a lean implementation, is the VSM (Pavnaskar, et al., 2003; Singh and Sharma, 2009; Vinodh, et al., 2010). This is due to the fact that VSM gives an indication of what actions to take and shortcomings to improve. Additionally, the VSM tool provides the possibility to measure and to evaluate the potential benefits of the actions at a rather low cost. It is a simple visualization of the processes and it also favors the necessary cultural change as it provides essential understanding of the benefits gained by the process improvements. To further strengthen the argumentation for choosing to apply VSM in this specific study, the choice of the particular lean method of VSM is in line with previous research, the case study company was in an early stage of a lean implementation, hence VSM is proposed as a suitable method.

3. TRANSFORMATION OF VSM INTO THE SOFTWARE DEVELOPMENT CONTEXT

The transformation of the VSM principles from a manufacturing context into a software development environment is detailed below in three classes; transformation of flow, waste and entities. The ICT company where the case study was conducted works with a Component Based Architecture (CBA) value offering and the core of their business is software development. The adjustments were, therefore, made to fit the nature of this company in value offering and way-of-working.

3.1. Transformation of flow

A development process includes several kinds of flow. Products, material, people and information create flows in all value streams. However, the nature of the flows is not similar in all contexts and to avoid a misunderstanding the different flows should be identified and redefined for each value stream. The following two sections describe the interpretation of what forms a production and an information flow in the value streams considered in this study.

Product flows can be intermittent due to bottlenecks. In manufacturing the bottlenecks are represented by, for example, machines being ordered to perform more operations or working faster than possible. In the software development context, bottlenecks occur when people or processes experience more work than they can manage or produce. A bottleneck was hence characterized by a long waiting time before the process itself, not that the process is time-consuming.

In manufacturing a pacemaker is the process that regulates the product flow. It controls the takt for the value flow, typically upstream in order to ensure meeting the customer demands regarding when the product is scheduled to be delivered. This process adjusts the ingoing material flow into the process to smooth out the flow based on the customer demand rate (takt time). In software development the customer demand rate will vary a lot due to the uniqueness of all features that is demanded. A pacemaker-process in software development is hence depending on the outline of the production processes. Meaning a pacemaker can be whichever process that sets the takt and controls what is taken into production. Thus, in the software development context a pacemaker can control both upstream and downstream processes since it acts as a hub for the "material" that passes through the value stream.

Information flow represents, both in manufacturing and in software development, the flow of information to, from and within the value stream. In manufacturing a distinction between manual and electronic information is made. Manual information is people sharing information, person to person or person to machine, and electronic information is when information is shared through an electronic source. This is usually the case when sending orders or feedback. In a software development context the product was realized in an electronic environment and hence the information trade is most frequently electronic. This was also due to the fact that the company is rather big and geographically divided, which makes manual information flows rare as the physical distance between people makes it impossible for them to meet. Hence, in the software development environment, manual information was seen as information sent from one person to another through e-mail, telephone calls, or face-to-face meetings.

In contrast, electronic information flows are classified as when information was sent between instances and thus was available for a lot of people. Thus, all feedback directly affecting the product and that was available for everyone was classified as electronic information. These definitions are thereby strictly bound to the nature of the sender and receiver of information and very little regarding the channel they arrive through, which is the case in a manufacturing context.

3.2. Transformation of waste

The most distinguishing principle of the lean thinking is the one regarding waste elimination (Monden, 1983). What should be classified as waste depends on the processes. The lean manufacturing highlights seven common wastes that can act as a general framework to rely on when identifying non-value adding activities in any kind of product development processes (Poppendieck and Poppendieck, 2007). Poppendieck and Poppendieck (2007) have investigated and adjusted these seven wastes in order to make them more relevant to a software development value stream. These were further clarified to be utilized in the case study company.

Table 1. Table of seven wastes transformed to software development context.

Seven wastes in software development	**Seven wastes in manufacturing**	**Adjustments of the seven wastes to the case study context**
Partially Done Work	In-process inventory	Prolonged items on backlog
Extra features	Over-production	Extra features
Relearning	Extra processing	Extra processing
Handoffs	Transportation	Handoffs
Task switching	Motion	Task switching
Delays	Waiting	Delays
Defects	Defects	Defects

3.3. Transformation of VSM entities

The standardized symbols and icons that are used in a VSM are made by and for manufacturing and they aim to map a physical flow of material. Thus, some of the icons were adjusted or redefined to fit a software development context. The adjustments mean redefining the meaning and content of the icons to better fit the nature of the context. Figure 1 gives an explanation to the meaning of the icons in the case study.

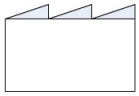
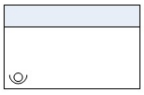
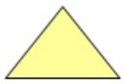
Manufacturing	**Software development**	**Explanation**	**Symbol**
Manual information	Manual information	All exchange of information between two persons was classified as manual information.	
Electronic information	Electronic information	Electronic information was information that was not meant to travel between two people but rather two instances e.g. between the community and the department.	
Shipment arrow	Shipment arrow	A shipment was stated as the delivery of a finished and quality tested product to a customer. This was not context dependent even though the shipment in software development is not physical.	
Shipment truck	Shipment	The symbol for a shipment is a truck, which highly emphasizes the manufacturing relation of the Lean concept. In software delivery no logistics of this kind is required. Hence the symbol of the truck will remain but it will mean a non-physical shipment. The product sent was digital.	
Customer	Customer	The value streams that were taken into consideration in the case study do not include direct contact with the end-customer. Hence, in this VSM an internal department was used as a "conceptual" customer instead.	
Process	Process	A process was when an operation was made on the product and thus adding value to it. During a process someone was actually handling the product. Hence, processes require people to be actively involved. The number within the circle was the number of people directly involved in the process.	
Push	Push	Push was defined as creating or add value to something not asked for, such as acting on own demand i.e. improvements or wishes from the software developers. The entire flow was regarded as push when passing a product, which was not asked for by the upcoming process.	
Pull	Pull	Pull in this case meant not letting the product further down the value stream until the next process is available.	
Inventory	Backlog	Since no physical products exist, the definition of an inventory was changed to; a place where a product in the value stream is waiting to be refined in an activity. Hence, the opportunity inboxes and backlogs were the best representation of "the waiting place".	
Go see	Auditing, testing, FA	Go see was defined as a manual quality control. Software development environment and the nature of the product demand a (to the largest extent) digital control. Thus, steps of quality control of all kind will be marked with a go see anyhow. This just to show where actual batch quality controls were made.	
Supermarket	Release	In production a supermarket holds a fixed variation of products. However, in the software development nature the releases were unique and customer specific. In software development the supermarket were more loosely defined as where ready to ship products were being stored or held. Thus, in out case this meant complete inventory.	

Fig. 1. Explanation of symbols and icons used in the VSM for the software development context.

4. IMPLEMENTATION AND RESULTS

4.1. Current state maps of software development components

To verify the applicability of the proposed transformations and to improve the value streams in the case study, the VSM is applied over two components. Their current state maps are given in the appendix A.

The value stream in the maps both starts with an order being submitted by the customer. This submission is placed in an inbox belonging to the separate components. The order is handled in a meeting to decide if it is feasible and relevant or should be rejected, further clarified by the customer or sent for study. The study is a sub-process, which aims to synchronize the order with the IT structure of other relevant parts, and to examine what surroundings an implementation of the order need to have in order to work properly. Thereafter, the processing of the item starts. This is made differently by the two components. Component 1 is handling the evaluating the order in a regular processing meeting. This meetings objective is to strategically align the order with the business side, prioritize and plan for the execution of the order and translate the order to adapt the linguistics of the request. Component 2 evaluates items in a more ad hoc manner, depending on when a singular person have time to make the evaluation.

The processing is followed by a pre-screening with a more technical orientation. The pre-screening is done to break down the order into minor parts and time estimate the order in man-hours and aims to synchronize the order with the underlying IT structure of the component. One of the components also has further preparations before sending the order to actual development. These are backlog grooming and sprint planning, which are processes related the Agile way-of-working. In these meetings the order is prepared to be involved in a sprint, assigned to a specific developer and the final time-estimation and planning is done. Finally, the actual development is implemented and followed by documentation and assessment of the end-quality of the product before releasing it to the customer. Problems occurring when comparing the two maps are the lack of best practice due to differences in their processes. One of the components has a rather undefined value flow with few processes. Many items placed in inventory, or backlogs, and a low ability to time estimate each process, creates an uncertainty in how long an order takes to travel the entire stream. Many sub-processes are connected to the development, which does not favor planning the development or creating an efficient environment for the developers to work in. The assessment step after the development is rather fluctuant in time between the two components. In one component it is evident that development and the following step are the most time consuming together with the study step. The flows are typically push flows and there is little communication with the customer during ongoing development. Furthermore, the clarification arrows simply depict an order or product that has to travel in the opposite direction of the value stream.

4.2. Future state map for all components

The future state value stream represents an improved standardized value stream that contributes to a reduction of wastes and a better flow throughout the value stream. The common future state for both components is given in appendix B. The changes made to the value stream are drawing on Lean thinking. Hence, the first objective was to create a pull flow instead of a push. This is done by setting the processing meeting as a gatekeeper and thereby allowing only current orders or urgent orders to enter the value stream. This to enable an evaluation of the order before entering the value stream and minimizing the risk of creating something out of date or unwanted.

Another big change is the evaluation of best practice and making the components work in a more similar, and structured, manor. This to facilitate internal benchmarking, time-estimates and helping in the structuring of the component 2 value stream. Also to early detection of bottlenecks and thereby minimizing the inventory levels and smoothing out the flow. A production control is set in place to help facilitate planning and synchronization between the customer and the department. The production control together with a communication forum will facilitate and streamline the interaction with the customer to enable creating as much value as possible and working towards a JIT delivery. Doing the documentation simultaneously with the development reduces the time it takes to perform the process after development. And a more efficient development is reached through synchronizing sub processes and enabling a better time-estimation throughout. To enable a better time estimation measurements need to be taken, this is possible after introducing a more standardized structure and the usage of a production control system.

A Leaner value stream with feasible solutions implemented and a holistic picture in mind would result in an efficiency ratio of 18%. This is an improvement of 9 percentage points compared to the current component 1 value stream and probably an even bigger one for component 2. However, this study does not express how the change from current state to future state should be realized. In general, management should be responsible for

driving this kind of radical change. Thus, their understanding and approach to the new structure is essential in order to successfully realize it.

5. DISCUSSION AND CONCLUSION

Studied application of the proposed VSM transformation indicates that the lean principles can be applicable in order to improve the efficiency and eliminate waste from software development value streams. However, the general tools should preferably be used in a pragmatic way and the transformation of lean practices is necessary in order to reach their intended purpose. The applied VSM tool in this study reveals bottlenecks and inconsistencies in the value streams and to some extent the wastes created throughout the processes. However, all wastes are not detectable, especially when occurring inside a process. On the other hand, the VSM contributes to a holistic view and helps knowledge and information sharing within the company. Moreover, the VSM can be a proper tool for indicating what to measure and facilitating a follow-up of the current value stream and changes made to it. To use the tool of creating a fishbone diagram by conducting a root cause analysis was easy and gave a comprehensive view upon the issues, hence it is feasible in the current software development context.

VSM lacks in its ability to visualize information flows. In contrary to a manufacturing type industry, the information could be done in numerous ways and when using the VSM it needed to be grouped by rather rough estimates.

The future state map lacks in providing all the possible improvements, especially the ones not strictly related to a value stream process. Furthermore to reduce the number of processes, when the process structure is iterative, does not seem to be clearly communicated by the VSM. The software development way-of-working also makes time estimates and backlog levels quite hard to establish because each order is broken down into smaller parts, i.e. the product is divided throughout the value stream. However, the future state map does provide a goal to strive towards and a good basis for communicating the necessary improvements to higher instances. The efficiency ratio is hard to evaluate due to the lack of benchmarking opportunities, and also due to the hard time estimates. However, the VSM will properly show improvements being implemented in the value stream.

The conclusion is that lean principles at least to some extent are applicable in the software development environment. Application of VSM to software development value stream can be made in a satisfying way but not comprehensive in regards to displaying the software development complexity, iterative cycles and information flow. Hence, making necessary adjustments to fit the classic VSM tool to the specific context of the value stream seems to be essential in order to reach intended results.

REFERENCES

Åhlström, P. (2004). Lean service operations: translating lean production principles to service operations. *International Journal of Service Technology and Management , 5*, 545-564.

Alvesson, M., & Sköldberg, K. (1994). *Tolkning och reflektion. Vetenskapsfilosofi och kvalitativ metod.* Lund: Studentlitteratur.

Biffl, S., Aurum, A., Grünbacher, P., & Boehm, B. (2005). *Value Based Software Engineering.* Springer.

Bowen, D. E., & Youngdahl, W. E. (1998). "Lean" service: in defense of a production-line approach. *International Journal of Service industry and Management , 9* (3), 207-225.

Cohen, D., Lindvall, M., & Costa, P. (2003). *Agile Software Development, A DACS State-of-the-art report.* Maryland: Fraunhofer center for experimental software engineering Maryland.

Collins, J., & Hussey, R. (2009). *Business Research, a practical guide for undergraduate & postgraduate students.* Hampshire: Paulgrave Macmillan.

Collin, M. (2012). *Lean Production Development: A Manager's guide.* London, UK: The Institution of Engineering and Technology, IET

Dybå, T., & Dingsøyr, T. (2008). Empirical studies of agile software development: A systematic review. *Information and Software Technology , 50* (9-10), 833-859.

Hodgetts, P. (2009). Retrieved December 4, 2013, from Agile Logic: https://www.agilelogic.com/files/LeanIsMoreWhitepaper.pdf

Karlsson, C. & Åhlström, P. (1996). "The Difficult Path to Lean Product Development". *Journal of Product Innovation Management*, 13, pp. 283-295.

Lee, H. L. (2002). Aligning supply chain strategies with product uncertainties. *California management review* , 105-119.

May, M. (2005). Lean thinking for knowledge work. *Quality Progress* .

Misra, S., Kumar, V., Kumar, U., Fantazy, K., & Akhter, M. (2012). Agile software development practices: evolution, principles, and criticisms. *29*.

Monden, Y. (1983). Toyota Production System: Practical Approach to Production Management. *Industrial Engineering and Management Press* .

Musat, D., & Rodríguez, P. (2010). Value Stream Mapping integration in Software Product Lines. *Proceedings of the 11th International Conference on Product Focused Software* (pp. 110-111). New York: Association of Computing Machinery.

Nepal, B. P., Yadav, O. P. & Solanki, R. (2011). "Improving the NPD Process by Applying Lean Principles: A Case Study". *Engineering Management Journal*, 23:1, pp. 52-68.

Pavnaskar, S., Gershenson, J., & Jambekar, A. (2003). Classification scheme for lean manufacturing tools . *International Journal of Production Research* , 3075- 3090 .

Petersen, K. (2010). *Implementing lean and agile software development in industry.* Karlskrona: Blekinge Institute of Technology.

Poppendieck, M., & Poppendieck, T. (2007). *Implementing Lean Software Development, from concept to cash.* Boston: Pearson Education.

Reinertsen, D. & Schaeffer, L. (2005). "Making R&D Lean". *Research-Technology Management*, 48:4, pp. 51-57.

Rother, M., & Shook, J. (1998). *Learning to See: Value Stream Mapping to Add Value and Eliminate Muda.* Cambridge: Lean Enterprise Institute .

Schuh, G., Lenders, M. & Hieber, S. (2008). "Lean Innovation: Introducing Value Systems to Product Development". *PICMET 2008 Proceedings*.

Singh, B., & Sharma, S. (2009). Value stream mapping as a versatile tool for lean implementation: an Indian case study of a manufacturing firm. *Measuring Business Excellence* , 58-68.

Song, W., Tan, K. H., & Baranek, A. (2009). Effective toolbox for lean service implementation. *International Journal of Services and Standards* , *5*, 1-16.

Staats, B. R., Brunner, D. J., & Upton, D. M. (2011). Lean principles, learning, and knowledgework: Evidence from a software service provider. *Journal of Operations Management , 29*, 376-390.

Wang, X., Conboy, K., & Cawley, O. (2012). "Leagile" software development: An experience report analysis of the application of lean approaches in agile software development. *The Journal of Systems and Software , 85*, 1287-1299.

Vinodh, S., Arvind, K., & Somanaathan, M. (2010). Application of value stream mapping in an Indian camshaft manufacturing organisation. *Journal of Manufacturing Technology Management* , 888-900.

Womack, J. P., & Jones, D. T. (1996). *Lean thinking, banish waste and create wealth in your corporation.* London: Simon & Schuster, Inc.

APPENDIX A

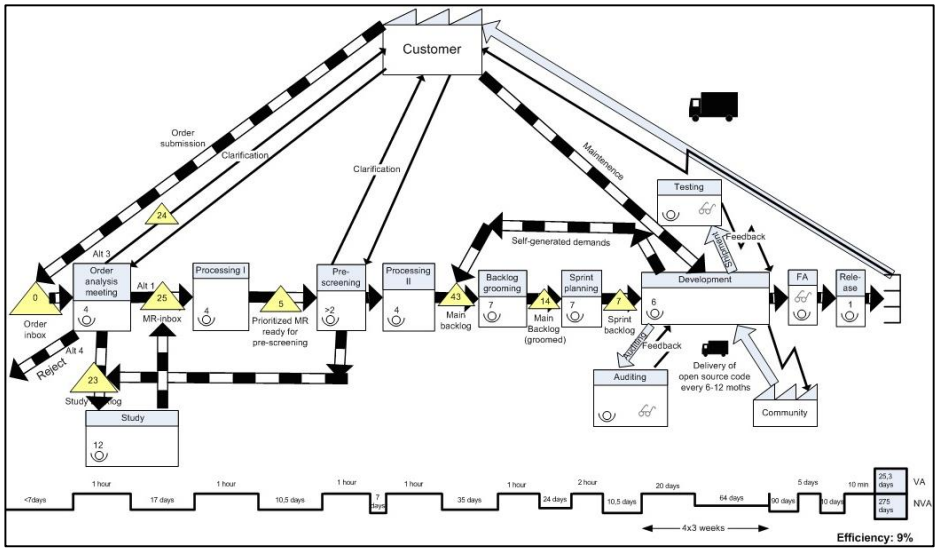


Fig.3. Current state map of Component 1.

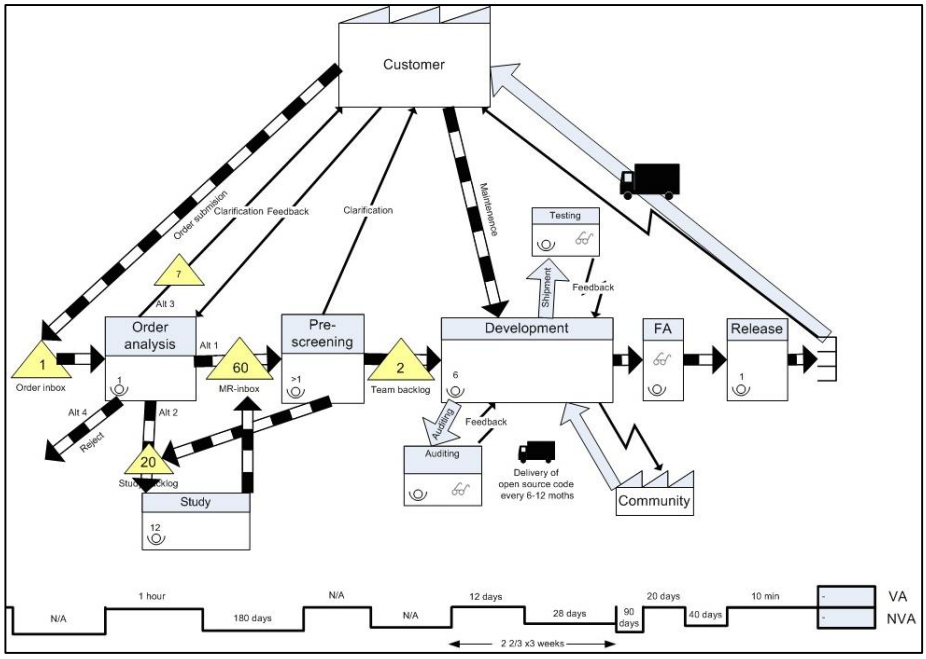


Fig. 4. Current state map of Component 2.

APPENDIX B

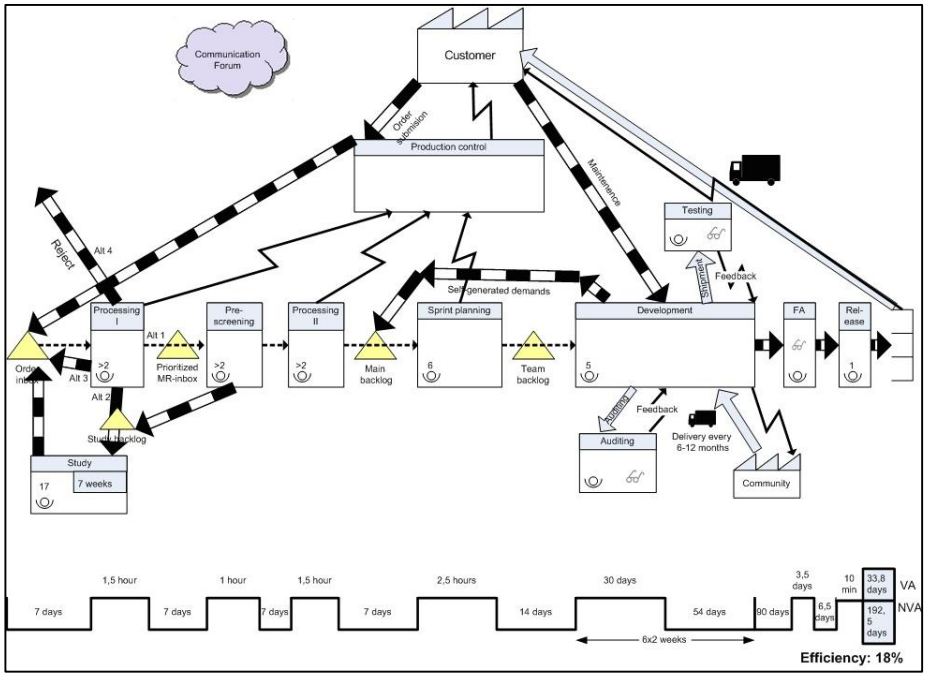


Fig. 5. Future state map of both component 1 and 2.