WASTE IN LEAN SOFTWARE DEVELOPMENT: A ROOT CAUSE ANALYSIS

Prasadbabu Medidi

Faculty of Computing
Blekinge Institute of Technology,
This thesis is submitted to the Faculty of Computing at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science in Software Engineering. This thesis is equivalent to 20 weeks of full time studies.

Contact Information:

Author: Medidi Prasadbabu  
E-mail: prme10@student.bth.se

University advisor:  
Dr. Kai Petersen  
E-mail: Kai.petersen@bth.se  
Department of Software Engineering

Faculty of Computing  
Blekinge Institute of Technology  
SE-371 79 Karlskona, Sweden  
Internet: www.bth.se  
Phone: +46 455 38 50 00  
Fax: +46 455 38 50 57
ABSTRACT

Context: Removal of wastes is a crucial area in lean software development. It has been found that there was little evidence on root causes of wastes in lean software development. Root causes from the state of practice had not being investigated. Furthermore, relations between wastes were now successfully exposed through root cause identifications process.

Objectives: The objective of this study was to perform an in-depth investigation to identify causes which lead to wastes in Lean software development process in the context of medium to large software development. To this end, researcher also identified relationships that exist between wastes.

Methods: The researcher conducted Literature review to look for evidence on waste related activities offered in peer-reviewed literature. Furthermore, the author conducted seven semi-structured interviews and used Grounded Theory method for both literature and interview data analysis.

Results: The researcher identified three categories of factors of wastes. Namely, Technical, Non-technical and Global software product development. In the technical category, factors relating to different technical aspects to build a product such as required resource issues, solving complexity issues among others were identified. Similarly, factors relating to people knowledge, management issues as well as factors that bothered on communication, coordination and temporal distance were identified as non-technical and global software product development respectively. For all seven kinds of wastes the root causes were identified.

Conclusion: The main contribution of the study is to help software development companies to avoid root causes of wastes in lean software development. The study show how wastes are related to each other. Various root causes of wastes were identified. These include partially completed work, extra processes (Relearning), task switching (Handoffs, motion), delays (waiting) and defects. These factors of wastes and relationships between wastes were identified from both state of art and state of practice.

Keywords: 1. Lean thinking. 2. Lean software development. 3. Wastes in LSD. 4. Root Cause Analysis.
ACKNOWLEDGEMENT

I thank Mr Mattias Borgkvist, Mr Rölf Häsänen, Mr Femi, Ms Annie, Mr Michael, Mr Johannes, Mr Benjamin and Mr Daniel in this study for generous participation for this study.
I thank Prof Jurgen Börstler, Mrs Espen Krona, Sister Carley King and Sister Carlson for all their help in certain situations.
I thank M Ramakrishna, Vijay, N Bucheswara Rao, Surya Rao for their financial support in completing my degree.
I thank Joshuah Anjorin for his kind help in starting and completing this work without him I don’t think of starting this work.
Finally I would like express my sincere gratitude and thank Dr. Kai Petersen for accepting me with this work, without his guidance I cannot solve this problem.
DEDICATION

To my mother, father and my brothers.
TABLE OF CONTENTS

Abstract .......................................................................................................................... iii
Acknowledgement ........................................................................................................ iv
Dedication ......................................................................................................................... v
Tables of Contents ........................................................................................................ vi
List of Figures .................................................................................................................. vii
List of Tables .................................................................................................................... x
Acronyms and Notation Description .............................................................................. xii

Prasadbabu Medidi ........................................................................................................ iv

Chapter 1 Introduction .................................................................................................. 3
Chapter 2 Background and Related Work ..................................................................... 5
  2.2 Lean Methodology Fundamental Concept ............................................................. 7
Chapter 3 Method .......................................................................................................... 16
  3.1 Overview of methods and selection ........................................................................ 16
  3.1.1 Motivation of the selected research method ....................................................... 16
  3.1.2 Aims and objectives ......................................................................................... 17
  3.1.3 Research questions ......................................................................................... 18
  3.1.4 Research Question 1 (RQ1) ............................................................................ 18
  3.1.5 Research Question 2 (RQ2) ............................................................................ 18
  3.1.6 Expected outcomes ......................................................................................... 19
  3.1.7 Qualitative Methods ....................................................................................... 19
  3.1.8 Empirical study (Conducting Semi-Structured Interviews) .............................. 20
  3.1.9 Data Analysis .................................................................................................. 20
  3.1.10 Grounded Theory .......................................................................................... 20
  3.2 Literature Review .................................................................................................. 20
    3.2.1 Systematic Literature Review ........................................................................ 20
    3.2.1.1 Motivation and objectives of this SLR ......................................................... 21
    3.2.1.2 Planning and Review .................................................................................. 22
    3.2.1.2.1 Purpose of Systematic Review ............................................................... 22
    3.2.1.2.2 Development of Review protocol .......................................................... 22
    3.2.1.2.3 Search terms and search term construction process .............................. 23
    3.2.1.2.4 Search Databases .................................................................................. 25
    3.2.1.2.5 Search procedure for Automatic search .................................................. 26
    3.2.1.2.6 Paper selection process ....................................................................... 26
    3.2.2 Snowball Sampling ......................................................................................... 29
    3.2.3 Threats: .......................................................................................................... 29
  3.3 Interview study method: ......................................................................................... 30
    3.3.1 Types of Interviews: ..................................................................................... 30
    3.3.1.1 Unstructured Interview: .......................................................................... 30
3.3.1.2 Semi-structured Interview: ..........................................................30
3.3.1.3 Structured Interview: ...............................................................31
3.3.2 Interview Design: .......................................................................31
3.3.3 Thematizing..................................................................................31
3.3.4 Designing....................................................................................32
3.3.5 Interviewing................................................................................33
3.3.6 Transcribing.................................................................................34
3.3.7 Analyzing....................................................................................34
3.3.8 Coding Process: ........................................................................35
3.3.9 Threats: ......................................................................................37
3.3.9.1 Construct validity: ...................................................................37
3.3.9.2 Internal validity: .....................................................................37
3.3.9.3 External validity: .................................................................37
3.3.9.4 Reliability: .............................................................................38
Chapter 4 Results..................................................................................39
4.1 Following are the results of Literature Review on the basis of 67 primary studies selected. .................................................................39
4.2 Different factors related each wastes: ............................................41
  Partial done work: ............................................................................41
  Extra process: ..................................................................................49
  Extra features: ................................................................................54
  Task Switching: ...............................................................................61
  Handover: .......................................................................................65
  Delays: ..........................................................................................68
  Defects: ..........................................................................................78
Chapter 5 Discussion: ..........................................................................88
5.1 Comparative analysis: .................................................................88
  5.1.1 Implications to Researchers: ....................................................88
  5.1.2 Implications to practitioners: ...................................................89
  5.2 Synthesis.....................................................................................90
  5.3 Conclusion...................................................................................90
  5.4 Future work ...............................................................................91
References.........................................................................................92
APPENDIX 1: .....................................................................................I
APPENDIX 2: .....................................................................................II
APPENDIX 3: .....................................................................................IV
APPENDIX 4: .....................................................................................X
APPENDIX 5: .....................................................................................XI
LIST OF FIGURES

Figure 1: Lean principles and wastes.................................................................8
Figure 2: Literature review flow.............................................................................23
Figure 3: Roles of the subjects in industry ...............................................................34
Figure 4: Publication over time .............................................................................39
Figure 5: Partial done work factors .......................................................................41
Figure 6: Extra process factors ............................................................................49
Figure 7: Extra features factors............................................................................54
Figure 8: Task switching factors..........................................................................61
Figure 9: Handovers factors................................................................................65
Figure 10: Delays factors....................................................................................68
Figure 11: Defects factors...................................................................................78
Figure 12: Global product development .................................................................XV
Figure 13: Non-Technical factors ....................................................................... XV
Figure 14: Technical factors.................................................................................XVI
LIST OF TABLES

Table 1: Wastes in manufacturing and their mapping to software development (taken from[31]). ..... 12
Table 2: Lean principles for different areas (taken from [21, 32]). .................................................. 13
Table 3: Shows research questions against methodologies................................................................. 19
Table 4: Search strategy.............................................................................................................. 24
Table 5: Grouping of different terms .......................................................................................... 24
Table 6: Main Keywords used in the Search string ........................................................................ 25
Table 7: Databases searched ........................................................................................................ 26
Table 8: Inclusion and exclusion criteria .................................................................................. 27
Table 9: Number of Studies with respect to Review Stages ......................................................... 28
Table 10: Results of Grounded Theory Coding ............................................................................. 36
Table 11: Different Search Strings ............................................................................................. II
Table 12: Literature review .......................................................................................................... IV
Table 13: Data Extraction Strategy .............................................................................................. XI
## Acronyms and Notation Description

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSD</td>
<td>Lean software development</td>
</tr>
<tr>
<td>SDLC</td>
<td>Software development life cycle</td>
</tr>
<tr>
<td>RCA</td>
<td>Root cause analysis</td>
</tr>
<tr>
<td>GT</td>
<td>Grounded Theory</td>
</tr>
</tbody>
</table>
Chapter 1 Introduction

The concept of lean has gained much recognition among scholars and even in the industry especially in the recent past [1]. The increased awareness has prompted more studies and has led to companies in the adoption of lean practices [2]. The software organizations were faced with the challenge of how to reduce waste in lean software development (LSD) from the commissioning of the project until to the support [1]. Based on this problem different root causes of wastes should be identified. Some common causes related to multiple types of wastes also create relationships between wastes. To understand the complex relationships between wastes and their cause one has to understand the relationships between wastes and identify unique and shared root causes between them. This study investigates the root-causes for the different kinds of wastes through a qualitative investigation of literature and interviews with practitioners. With that, the main objective of the study is to help professionals to minimize the waste.

The contributions from the study are listed as follows. To identify different kinds of root-causes influencing waste in LSD. To identify the different kinds of relations between root causes that occurs between wastes. To identify the different kinds of relations between wastes this will provide information to practitioners to devise strategies for eliminating or lowering waste. In this study the researcher conducted interviews with seven software engineering practitioners, and applied grounded theory analysis to the qualitative data collected. The same analysis approach was used on the literature related to software engineering wastes, which allows comparing literature and the view of the practitioners.

The remainder of this report is structured as follows. Chapter 2: Describes the background related work, chapter 3 explains research methodology used, chapter 4: presents the results obtained chapter 5: discusses the synthesis and conclusion, chapter 6 shows the references used and chapter 7 is appendix.
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Research contribution</th>
<th>Research implications</th>
</tr>
</thead>
<tbody>
<tr>
<td>2. <strong>Background and Related work</strong></td>
<td>The works from previous studies were gathered and studied to identify research gap to map to the current study.</td>
<td>No study was conducted which identifies root causes of wastes through the state of practice or state of art.</td>
</tr>
<tr>
<td>3. <strong>Research Method</strong></td>
<td>In this chapter the motivation for selecting the research methods, the process of how these methods were applied, and the threats associated that were occurred is explained.</td>
<td>From both the literature and interviews data was collected and analyzed through grounded theory analysis method.</td>
</tr>
<tr>
<td>4. <strong>Results</strong></td>
<td>This chapter explains different factors obtained from analyzing and literature and interview.</td>
<td>Three different categories of factors were identified and explained in detail.</td>
</tr>
<tr>
<td>5. <strong>Synthesis and Conclusion</strong></td>
<td>Synthesis of the results with respect research questions and discussions of open issues which lead to future work are presented in these sections</td>
<td>These sections conclude with different factors for seven kinds of wastes were identified and relationships among them were noticed.</td>
</tr>
</tbody>
</table>
Chapter 2 Background and Related Work

The goal of most organizations is to ensure the provision of products with the highest quality, at the reasonable price, and within the shortest time [3]. Lean is a production practice and philosophy that has been successfully applied in various work fields such as services industries, healthcare, aerospace, logistics, product development and general manufacturing, to increase the value to the customers and eliminate waste from the production. Studies in different fields show that lean initiatives have led to radical improvements in quality, cost reduction and delivery time [4-7]. Lean concepts were introduced in manufacturing in Japan in mid-21st century [8]. The term was introduced as an industrial paradigm in a book 'The Machine That Changed the World, The Story of Lean Production' published in MIT International Motor Vehicle Program by MIT researchers [8]. In this book, a new industrial paradigm was introduced on the Toyota Production System. It is a holistic work system credited for the extraordinary rise of Toyota to the most profitable and the largest auto manufacturing company in the world [9].

In 1980’s, Toyota revolutionized the automobile industry with their approach of ‘lean manufacturing’. Reducing the time from customer order to product delivery by eliminating non-value adding activities (waste) in the production stream is the focus of “lean manufacturing” [10]. Lean is a way of thinking of what to do and what not to do. It helps organizations add specific value and does that without interrupting value-adding activities [11]. The way lean works is by creating only value creating activities that are needed, placing them in best sequence and conducting the activities uninterruptedly and in most effective ways. The lean approach tries to achieve quality products and defect-free products. It has a high focus on minimizing wastes in delivering and meeting customer demand. Its core elements include just-in-time, minimizing inventories, minimizing time, considering unique corporate culture of openness, respect of others, teamwork, and empowerment [9]. Lean has emerged as one of the best way to make processes better [4]. In terms of profitability, product quality, time-to-market that lean exhibited had generated greater interests and proved beneficial in other than manufacturing industries [12].

The software industry needs better tools, methods, models, process and techniques to continuously develop reliable software. It is having challenges achieving short time-to-market, global competitiveness and providing faster time-to-market [13]. Toyota, which has been successfully manufacturing vehicles within a very short time and meeting deadlines, uses a
vast amount of embedded software [14]. A study shows how the concepts of lean manufacturing can be successfully applied to software development and from this the phrase “Lean Software Development” was coined. LSD can produce rapid quality and productivity gains with some deep changes in the way organizations are structured and managed. Recently, the lean approach is becoming popular among software projects with tremendous performance benefits including managing the project more efficiently, eliminating waste and reducing lead times [15].

The emerging field of Lean Software Development (LSD) is the application of lean principles, practices, and tools to Software Engineering (SE) in order to increase delivery of value to customers while reducing waste in every level of work. Lean is not restricted to addressing a particular size of company. It can be adopted to any company which is large and global. It can be adjusted to even small and medium enterprises [16]. Lean transformation is a long-term activity which could require even years for a company to adopt completely. There are number of case studies showing implementation of lean in software industry is improving [17] [18]. In particular, some are studying on individual elements of lean such as value [3], flow [19] and so on. However, the application is dependent on the people with their knowledge in the implementation because of the intangible nature of software development. The work of people involves the information about, and understanding the challenges of, the applicability of lean [20]. The core five lean concepts are value, value stream, flow, pull, perfection [21].

- **Value** is a perception of customer on a product of how it should be. It is difficult to understand. Every activity in an organization should be associated with a value and if that value is not perceived by the customer then it is a non-value activity [22].
- **Value stream** consists of all the actions that are to be performed from a customer order to customer care on a product which generates value to customer [22].
- **Flow** are the value adding activities in the value stream organized in a way that there should no interruption in the smooth delivery of product [22].
- **Pull** is working to deliver first only things that are needed and on demand by customer [22].
- **Perfection** is delivering a product without defects while continually applying new thinking to catch defects and delivering a defect-free one [22].

Lean principles are considered as effective process improvement approach to eliminate system’s inefficiencies and wastes. Even the substantial benefits and significant paybacks from lean for a diverse range of industries like manufacturing, it is not easy to utilize same
manufacturing principles for software development[23]. Even after using the lean process, organizations should take challenges that arise and risks that are associated because of changes that need to be performed in the organization operations. Womack and Jones first identified principles of lean processes such as providing the highest customer value, maximizing flow and eliminating waste. A framework with five high-level principles can be used. These principles include:

- Determine what creates value from all stakeholder perspectives,
- Identify all current steps within a process or value stream,
- Continue with steps that create value,
- Discontinue steps that are wasteful,
- Continue to evaluate and remove steps that are wasteful [24, 25].

Adopting these principles is just one of the solutions to software development. LSD as an application of lean principles to software production was first published by Poppendieck’s in 2003. Although several authors have identified several lean principles especially in software development, the applicability of exact lean principles still need to come to consensus.

Poppendieck identified seven key lean principles which are transferred from the core concepts of lean manufacturing. They include: eliminate waste, build quality in, create knowledge, defer commitment, deliver fast, respect people and optimize the whole [21]. All these principles aim to reduce risks by having feedback so often, improve product quality, removal of wastage, and reduction of cost-increasing factors. The first and an important principle of LSD are to recognize and eliminate waste. However, classification and identification of wastes processes in lean practices is to perform in this first principle [22].

2.2 Lean Methodology Fundamental Concept

Three concepts that are fundamental to the understanding of Lean include: Value, Waste and the process of creating value without waste captured into the term Lean Principles.

**Value**

Value is defined as flawless delivery of product with satisfaction of all stakeholders [9]. Another definition emphasizes that value is defined by customer and there should be clear understanding of what it is [3].
Waste

Waste is defined as any unnecessary activities that add cost or time without adding value [26].

The identified seven key Lean principles are: eliminate waste, build quality in, create knowledge, defer commitment, deliver fast, respect people and optimize the whole [27].

All these principles try to reduce the risks by having feedback so often, improve product quality, removal of wastages, and reduction of increasing cost factors. The first and one of the important principles of LSD is to recognize and eliminate waste; however, classification and identification of wastes are practices to perform for this first principle in specific settings [28].

Waste is defined as anything that does not add product value as perceived by the customer [27] [29]. It is everything including unnecessary activities, those adding cost or time, absorb resources and producing output with no value [26]. In lean perspective, the time spent on developing a product that does not meet what the customer really needs or demands is a source of waste. Waste is an important concept of lean. In manufacturing, there are waste elimination strategies that are successfully used to increase the productivity. There are steps that have been followed to increase the productivity of organizations, reduce cost of delivering, and time-to-market of a product to customers. But the feasibility in other different industries has not been fully researched and completely established [30]. Three basic categories of wastes-related elements with Japanese names in lean manufacturing are Muda, Mura and Muri. Muda includes all non-value activities. Mura is all activities with variations in final product e.g. process
quality, coast, delivery etc. Muri are those activities that shouldn’t be eliminated because of restrictions but rather should be performed (overburden).

In lean, process separates wastes into two types. In the first type, waste originates from activities that do not provide value, but the activities are necessary to be performed. An example, may be support activities that are required by regulations but are not really helping the customer value stream, they just have to be done. In the second type, waste arises from activities that create no value, those activities are not necessarily performed and can be avoided. The second type should be eliminated immediately.

In software development, there are several kinds of waste, for example the seven wastes that are transferred by Poppendieck and Poppendieck [4] from manufacturing to software development are: partially completed work, extra processes (Relearning), extra processes, task switching (Handoffs, motion), delays (waiting) and defects. In manufacturing, different wastes can be easily detected because of the physical appearance nature of the wastes in material flows and work activities. But due to invisibility of nature of wastes in software it is hard to identify wastes. Lean helps in complete elimination of waste in all aspects of production.

By using eliminating strategies, framing strategies that eliminate the root causes of wastes, one can reach highest potential of lean. There is an evidence of previous studies categorizing wasteful processes on specific methods (e.g. VSM), but till now do not have a comprehensive analysis of the causes of wastes independent of particular methodologies, and need to synthesize and systematically acquire those processes, where they exist and how to eliminate them in software development. Waste identification is not an easy work due to lack of identification methods for different root causes of waste. Also lack of any existing studies to help in tackling the root cause identification. Identification of wastes in particular and successful implementation of lean in organizations in general is a challenging task. For organizations it is an endless process to reduce time, efforts, cost, mistakes, space and all other sources of wastes in proving the value to clients. The wastes can be explored even by thoughts which concentrate on what people should find and what contributes to the waste. A team should decide where to focus on and the actual impact of waste by gathering more data on it and the working on it. For elimination of waste, sometimes it requires new product concepts and process technologies to be used, as well as time that is not of certain duration. So lean is not a change to a process to be implemented in within a specific time frame, rather it involves continuous effort to make a process free from any form of waste.
Seven wastes of LSD:

**Partially done work**

Software that is partly done tends to get outdated hence an obstacle to other development works. The major challenge with partly done software is that there is never the certainty that it will work. You may have all the stack requirements, but before integration to the future environment, you have no idea about the problem that might arise. In addition you are never sure if it will solve the business problems. Reducing partly done software reduces waste of resources hence, lowering risk of financial loss [27].

**Extra processes (Relearning)**

The main focus here concerns paper work usage and involvement in software development and it can also be other things, such as spending too much resources and time on an activity, or not having the right level of documentation (producing many documents that will never be read, cannot be understood, etc.), its bulk, expensive, harbors quality problems among others, so the best thing is to try as much as possible to automate this process in order to fasten the response time among other advantages plus proper keeping track of codes[27].

**Extra features**

Extra features can be thought as providing intricate solution rather than simplest one with the use of unnecessary tools or technologies, unnecessary functionality designed and developed into the software, providing more information than is necessary [28]. It is great invention putting extra features on an already complete software. In most cases it is not necessary so its best left in its original way. In most cases, added features get out-date or obsolete hence getting to be a nuisance to the whole system.

**Task switching (Handoffs/ Motion)**

Switching between projects, switching between tasks, switching between maintenance and new development, switching between multiple projects, switching between the task and administration or e-mails, and so forth, can reduce productivity and efficiency. An alternative is to always to direct resources to one project, have it completed then start another one. Rather than having to share resources when developing two projects at a go. Hence it is an affirmation that switching time just clogs down the whole work on the two projects. The movement of resources during the software development is itself an obstacle. It affects coders, project
managers, clients, documents, approvals among others. This greatly affects the software
development because time is wasted within these motion cycles [27].

Delays (Waiting)
The waiting time on almost all projects is a huge problem with projects regarding software
development. Waiting times are exhibited in starting a project, staffing and approvals among
others but also in-between all development activities (e.g. requirements, implementation,
testing, release). Sometimes these delays are invisible but they exist. They not only affect the
developers on a project but also the clients who are always waiting to have the market
advantage with early unleashing of their products on the market [27].

Defects
The defects in the products impacts the software development cycle especially when they are
detected late or remain undetected, hence reaching the customer. The earlier a critical defect
gets detected, the better. The longer they stay undetected, more costly it is to fix them. The best
way to reduce waste is to find them as soon as they happen. This calls for immediate testing,
integrating fast and releasing immediately for production [27].
The following table explains different wastes in lean that were translated from manufacturing to software engineering.

<table>
<thead>
<tr>
<th>Manufacturing</th>
<th>Software Engineering</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inventory: Intermediate work-products and work in process</td>
<td>Partially Done Work: Work-in process that does not have value until it is completed (e.g. code written, but not tested)</td>
</tr>
<tr>
<td>Over-Production: The number of produced items is higher than the number of demanded items (inventory in this case is “dead capital”)</td>
<td>Extra Features: Functionality that has been developed, but does not provide value to the customer</td>
</tr>
<tr>
<td>Extra Processing: Extra work is created in the production due to e.g. poor set-up of machines</td>
<td>Extra processes: Process steps (e.g. Creation of documentation that is not really needed) that can be removed</td>
</tr>
<tr>
<td>Transportation: Transport of intermediate work-products (e.g. due to a poor layout of the production line)</td>
<td>Handovers: Many handovers (e.g. documentation) create overhead</td>
</tr>
<tr>
<td>Motion: People and machines are moved around instead of being used to create value</td>
<td>Motion/Task Switching: People have to move to identify knowledge (e.g. team members that work together are not collocated) or have many disturbances in their work</td>
</tr>
<tr>
<td>Waiting: A machine with free capacity is waiting for input</td>
<td>Delays: There are delays in development that, for example, cause waiting times within a development team (team idles)</td>
</tr>
<tr>
<td>Defects: Fixing of problems in the products</td>
<td>Defects: Fixing of problems in the products</td>
</tr>
</tbody>
</table>
Lean principles proposed by different authors:

Table 2: Lean principles for different areas (taken from [21, 32]).

<table>
<thead>
<tr>
<th>Author</th>
<th>Principles</th>
</tr>
</thead>
</table>
| **Liker [4]** | 1. Base your management decisions on a long-term philosophy, even at the expense of short-term financial goals.  
2. Create a continuous process flow to bring problems to the surface.  
3. Use “pull” systems to avoid overproduction  
4. Level out the workload (heijunka).  
5. Build a culture of stopping to fix problems, to get quality right the first time.  
6. Standardized tasks and processes are the foundation for continuous improvement and employee empowerment.  
7. Use visual control so no problems are hidden.  
8. Use only reliable, thoroughly tested technology that serves your people and processes.  
9. Grow leaders who thoroughly understand the work, live the philosophy, and teach it to others.  
10. Develop exceptional people and teams who follow your company’s philosophy.  
11. Respect your extended network of partners and suppliers by challenging them and helping them improve.  
12. Go and see for yourself to thoroughly understand the situation (genchi genbutsu).  
13. Make decisions slowly by consensus, thoroughly considering all options; implement decisions rapidly (nemawashi).  
14. Become a learning organization through relentless reflection (hansei) and continuous improvement (kaizen). |
| **Womack and Jones 1997 Principles [11]** | 1. Value  
2. Value stream  
3. Flow  
4. Pull  
5. Perfection |
| **Lean principles by Poppendieck and Poppendieck 2003 [4]** | 1. Eliminate waste  
2. Build quality in  
3. Create knowledge  
4. Defer commitment  
5. Deliver fast Apply  
6. Respect people  
7. Optimize the whole |
| **Andersson** | 1. Visualize the workflow  
2. Limit work in progress  
3. Manage flow  
4. Make process policies explicit  
5. Improve collaboratively (using models and the scientific method) |
| **Morgan and Liker [4]** | 1. Establish customer-defined value to separate value-added from waste.  
2. Front-load the product development process to explore thoroughly alternative solutions while there is maximum design space.  
3. Create a level product development process flow.  
4. Utilize rigorous standardization to reduce variation, and create flexibility and predictable outcomes.  
5. Develop a chief engineer system to integrate development from start to finish.  
6. Organize to balance functional expertise and cross-functional integration. |
For the removal of waste it is important to understand the current status in terms of waste, which is supported by rigorous study. Although, some studies have been carried out to identify different factor to wastes, they could not consider all wastes.

In a study some factors were identified to one of the wastes inventory; batching, process and arrival variation, and unsynchronized concurrent tasks [7]. The factors, have a negative effect on other wastes, or even related to affect other wastes as pointed out in the study: batching leads to overproduction; process and arrival variation leads to overproduction and waiting: and unsynchronized tasks lead to waiting [7].

In another study, some factors were identified that influence defect injection, which is one of the waste with the goal to lower the number of defects in software intensive products. The study has identified 16 factors and these factors are grouped together in defect injection group using expert opinion [34]. But there is no empirical evidence on the validity of factors that were provided by the study.

In another study waste was explored (identified) in the Kanban-driven software development project context but even still it does not prevent waste from creeping in. Their empirical evaluation is of on a small size team and the participants are students. The work is only based to lean practice of Kanban and do not consider the factors as a general to waste [1].
In another study wastes were explored only in the software test process using lean methodology (VSM). The study identified different wastes but mainly pointed out delays are cause due to requirements management[35].

To understand the relationships between these different kinds of waste, it is vital to critically know about each kind of waste, how they come about and their impact. The study of the removal of waste from the software development process is of critical importance. One way of identifying waste is seeking to understand what the customer truly values. However, there is necessary to first identify the root cause of the waste in the process and then understand the relationships between the different kinds of waste.

The objectives of the study are to identify and classify factors causing waste and their relations between each other in software development influenced by Lean approaches to product development and use this established knowledge to support industrial partners in decisions on a software process improvement project.
Chapter 3 Method

3.1 Overview of methods and selection

3.1.1 Motivation of the selected research method

The research methodologies which are commonly used in software engineering are case studies [36], survey [37], controlled experiments [38], simulation [39] and action research [39].

Case studies: In this methodology, a specific case is phenomenally analyzed and investigated in-depth. Real world cases are studied in natural environment like software organizations, software projects with software development teams etc. These are conducted by defining the cases to be studied, analyzing the units required and also data collection strategies.

Survey: The process of studying a phenomena of a population by surveying a sample in a specified group. This methodology involves sourcing and collecting of data using interviews and questionnaires. Statistical inference is used to draw conclusions for the data collected from over all information.

Controlled experiments: Controlled environment is used to validate theories in this method. Hypothesis is formulated for cause-effect relationship among one or more independent variable and out-come variables in studies. In this the experiment is conducted in controlled environment, the variables which are not independent shouldn’t have effect on out-come variables.

Simulation: These are executable models in real world phenomena helps to study their pattern of behavior example: software architecture [39].

Action research: The goal is to introduce an investigation to a real world setting and to observe what the effect of the investigation is. In this investigation the researcher plays an active role in the organization by participating in the team stimulated by investigation. In action research the researchers will carry out the observations on effects of an intervention which was introduced in real world settings. The answer to each research questions is associated with proper selection of research methods.
Motivation for selecting this research methodology:

The research methodology that is selected for this thesis is a qualitative research using interviews. This choice is motivated because an in-depth understanding of the wastes and their relationships is sought. This method is more suitable than any other due to its exploratory way of gathering the data to the study in a short time. This excludes survey as a research method because it does not allow the subjects to explain why/how questions. This study does not need large numbers of people (it is good if available) but needs at least a few who are highly experienced. The aim was to gain in-depth understanding because research in the breadth is not an option. A single case study cannot give the solution to this research because the study needs peoples’ responses who are highly experienced with more numbers of cases in industry. Besides a case study is not used because it cannot give maximum generalized factors to all wastes in a short period of time and it is not possible for the researcher to work effectively with many case studies.

A single experiment is not possible to give all possible factors that are causing different wastes. This study requires many experiments with many people from small sized teams to huge team members of varying experience, so it is not possible to replicate such a solution in a lab environment with students.

An in-depth understanding is needed to give factors that are causing wastes where simulation method is eliminated for this research because it focuses primarily on measurable characteristics of a process.

Action research is not selected and it is not suitable for this research because this research is not on a single company and there is no plan of an action or intervention and observing its effect.

3.1.2 Aims and objectives

The objectives of the study are to identify and classify factors causing waste and their relations between each other in software development influenced by lean approaches to product development and use this established knowledge to support industrial partners in decisions on a software process improvement project.

The main aim of the thesis project is to find the root causes of waste in lean software development process and to identify the relationships between them if exists.
In order to meet the aim, the following objectives are defined:

**O1.** Identify different kinds of root causes influencing waste.

**O2.** Identify the relations between root causes that occurs wastes.

**O3.** Identify the relations between wastes.

**O4.** To provide information to practitioners to devise strategies for eliminating or lowering waste.

### 3.1.3 Research questions

### 3.1.4 Research Question 1 (RQ1)

What factors are influencing the wastes defined in lean software development?

**Description**

An efficient planning is required to deliver a software product on time. For this, a company can focus on how to deliver a high quality product in due time and with minimum effort, in order to achieve high value. This can be achieved by lean. However even when everything is planned and lean is used, if a company does not meet all stakeholders high expectations for the product, then the benefits of using lean are not achieved. To achieve this, factors that are causing a waste should be identified, so that lean can be fully taken advantage of.

### 3.1.5 Research Question 2 (RQ2)

What are the relations between the following?

- between root causes and wastes
- between different root causes
- between different wastes
Description

If a root cause of each waste was identified, and a team knew what are the common causes that are effecting different wastes then the team could act on these causes and could achieve leverage on the benefits of lean. These common root causes and their relationships are important to provide information to practitioners to devise strategies for eliminating or lowering waste. The goal of this is to identify these common relationships that exist between wastes.

<table>
<thead>
<tr>
<th>Research Questions</th>
<th>Methodology</th>
</tr>
</thead>
<tbody>
<tr>
<td>RQ1</td>
<td>Interviews &amp; Literature Review</td>
</tr>
<tr>
<td>RQ2</td>
<td>Interviews &amp; Literature Review</td>
</tr>
</tbody>
</table>

Table 3: Shows research questions against methodologies.

3.1.6 Expected outcomes

In general, the outcomes of this research were the relations between wastes. This will be achieved by the other outcomes like reasons, factors, and root causes that occurs wastes.

Established links between different kinds of wastes.

3.1.7 Qualitative Methods

Research methodology is a plan of actions or strategies that relates methods to outcomes. It affects our choice and use of different methods available, e.g. questionnaires, interviews, case study and focus groups, etc. In order, to fulfill the aims and objectives of the study, the research design is based on Qualitative methodology. The main focus of the research is to find the relationships of different wastes, and the primary focus is on their industrial validation. To cope with this research problem a qualitative method is used, the qualitative part of the research deals with the participatory knowledge claims, open ended interviewing and narrative design [40], the research design includes interviews with the support of data from Literature review. RQ1 is answered by interviews and with the literature review. The literature was thoroughly searched to find out what are the factors associated to different wastes and undocumented factors were identified by interviews in industries. RQ2 is answered by grounded theory from
the data obtained by the literature review and with the interviews. The information gathered from the RQ1 was served as an input to RQ2 and some expert opinion was used by the researcher based on the findings.

3.1.8 Empirical study (Conducting Semi-Structured Interviews)

The purpose behind empirical study was to explore wastes, their root causes and relationships from industrial settings according to different stakeholders’ perspectives of the project. For this reason, interviews were conducted on project managers, software developers, and other stakeholders of software organizations with different perspectives. To eliminate the bias of the results from the empirical study the interviews were conducted with different companies.

3.1.9 Data Analysis

The collected data from literature review and semi-structured interviews were analyzed using Grounded Theory. The outcome of this analysis was the relationships.

3.1.10 Grounded Theory

Data gathered from the literature review and the interviews was analyzed using Grounded Theory methodology (GT). The theory is related to the qualitative research approach and is considered as an explicit and systematic approach [41]. It is the best to answer to the question; in particular the study needs deep understanding and also needs to be able to elicit the relationships between wastes. Therefore, Grounded Theory is the best suited approach for study where raw data is collected from two different means and then analyzed accordingly to produce results.

3.2 Literature Review

3.2.1 Systematic Literature Review

A Systematic Literature Review (SLR) is the process of finding relevant literature to the study and reporting the research results in a systematic way. This is done by documenting each and
every step that has been performed to obtain relevant literature. This is done by designing and applying inclusion and exclusion criteria at primary studies, designing a quality criteria for evaluating each obtained studies and extracting data from them [42]. Reviews covering the area focusing on identifying the root causes of wastes in this thesis could not be found, but studies reviewing lean literature in other areas on the SE field were found.

Pernstål and others conducted systematic mapping studies on how lean approaches to large scale software systems development between 1990 up to 2011, identified 38 relevant studies. Of those, only one study was reported regarding lean usage and exploring the source of wastes in Kanban software development [1]. In this study only few sources were identified with a small group of students on a school project.

Beside, Systematic review was chosen because it allow for identification and selection of literature in a specified manner. Thus, it has the advantage of presenting information in an unbiased manner. This is in contrast to the traditional reviews where identification and selection of literature is not usually in a specified manner and have the potential to presents biased information. Sometimes in traditional review appraisal could be variable, and synthesis is most often a qualitative summary [43].

3.2.1.1 Motivation and objectives of this SLR

There are few reasons for carrying out SLR in this thesis. There is no other study conducted focusing on the root causes of each waste in LSD from the start of the project to the support. There is no other study which focuses on finding relationships between different kinds of wastes by utilizing the root causes. With this SLR one can build frameworks for elimination strategies.

The main focus of this research is to find the root causes of each waste in LSD during the development lifecycle as a whole and not during one phase only. The aim of lean and lean principle to eliminate waste is not restricted to one phase or department in product development; it is adapted and applicable to the entire development organization. The common causes of the wastes that are effecting the entire development can be viewed as a tool to eliminate waste.
3.2.1.2 Planning and Review

3.2.1.2.1 Purpose of Systematic Review

The main purpose of this review is to find the relevant literature about the factors causing a waste and about the relation between wastes. In this study, literature reviews results give many factors and relationships that are mentioned in the literature about wastes of lean software development.

3.2.1.2.2 Development of Review protocol

In order to get the primary studies, a review protocol is constructed for conducting the SLR. This protocol, which is a back bone of SLR with a complete plan for conducting SLR, reduces the biasness in the research. SLR is used to identify research gaps and also helps in proposing frameworks to manage the identified research gaps. In this SLR planning and review phase, steps were followed from the guidelines provided by Kitchenham [42] except the quality assessment of the papers was not performed [35, 44]. This step is excluded because the study concentrates only on identifying the factors that are causing wastes in LSD and finding the relationships between wastes. Besides, this the study is not providing any elimination or mitigation strategies. The studies are only selected as published in peer-reviewed literature which were relate to the area of study.
3.2.1.2.3 Search terms and search term construction process

The keywords and their synonyms are used, which were formulated from the research questions. Keywords are selected from the research questions such that it will fulfill the aims and objectives of the study. Search terms were formulated in consultation with a librarian. Pilot search was done to know new features provided in different databases with the help of a librarian. After continuous searches and reading different studies of various authors, many more terms were identified. The terms that are considered important and that were related to the research questions are formulated by identifying population, intervention, outcome, context, method, and comparison.
Search strategy:

Table 4: Search strategy

<table>
<thead>
<tr>
<th>Searching Attributes</th>
<th>Searching data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population</td>
<td>Certain areas such as Software engineering, Software development, Software product, Software project, Software process, Software building, Software improvement.</td>
</tr>
<tr>
<td>Intervention</td>
<td>Grouping of terms and synonyms of different wastes, lean, research method.</td>
</tr>
</tbody>
</table>

For each term that is considered important and related to the research questions synonyms were formed, word spellings were altered, and other alternative search terms were identified which are used by various authors. Some of the keywords were taken from other papers related to the same area in the field of study. Boolean AND, OR operators and quotation marks if necessary are used for combining terms. OR is used for the combination of synonyms and altered spellings. AND is used for combining two different terms for getting the full term meaning, Quotation marks or brackets are required for certain databases. Brackets are used to get the exact phrase or term.

Table 5: Grouping of different terms

<table>
<thead>
<tr>
<th>Main Keywords</th>
<th>Grouping of terms (Synonyms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wastes</td>
<td>Waste OR Partially Done Work OR Incomplete development OR Extra Features OR Unwanted features OR Additional functions OR Extra Processes OR Handover OR Motion OR Task Switch OR Delay OR Defect</td>
</tr>
<tr>
<td>Software</td>
<td>Software engineering OR Software development OR Software product OR Software project OR Software process OR Software building OR Software improvement</td>
</tr>
<tr>
<td>Factors</td>
<td>RCA OR Cause OR Factor OR Relation OR Influence</td>
</tr>
<tr>
<td>Lean</td>
<td>Lean practice OR Lean OR Value stream OR Kanban</td>
</tr>
<tr>
<td>Empirical</td>
<td>Empirical OR Literature review OR Survey OR Case study OR Experiment OR Action research OR Systematic review OR Comparative study OR Industrial experience report</td>
</tr>
</tbody>
</table>

After conducting many searches the keywords were too general and the area of study was too broad, so to minimize the scope only the following keywords, which have the ability to cover all studies, are considered in the formulation of a search string. The string is created by following the same construction used by another researcher [2]. Once the search terms were
finalized, a final pilot search was performed with guidance of a senior researcher (supervisor) to see that the query is receiving the right population. Changes are made to the search string to support the requirements of each database to be searched. Even Boolean operators such as AND, OR, and Wildcards were changed based upon the database which was searched. The search string is also divided into simple and small search strings of a certain length, because of the restrictions on number of keywords to be used in certain databases. Table 4 contains main keywords and their synonyms that are used for different databases respectively.

<table>
<thead>
<tr>
<th>Keywords</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Waste</td>
<td>Software engineering</td>
</tr>
<tr>
<td>Partially Done Work</td>
<td>Software development</td>
</tr>
<tr>
<td>Incomplete development</td>
<td>Software product</td>
</tr>
<tr>
<td>Extra Features</td>
<td>Software project</td>
</tr>
<tr>
<td>Unwanted features</td>
<td>Software process</td>
</tr>
<tr>
<td>Additional functions</td>
<td>Software building</td>
</tr>
<tr>
<td>Extra Process</td>
<td>Software improvement</td>
</tr>
<tr>
<td>Handover</td>
<td>RCA Cause</td>
</tr>
<tr>
<td>Motion</td>
<td>Factor</td>
</tr>
<tr>
<td>Task Switch</td>
<td>Relation</td>
</tr>
<tr>
<td>Delay</td>
<td>Influence</td>
</tr>
<tr>
<td>Defect</td>
<td>Lean practice</td>
</tr>
<tr>
<td></td>
<td>Lean Value stream</td>
</tr>
<tr>
<td></td>
<td>Kanban</td>
</tr>
<tr>
<td></td>
<td>Empirical</td>
</tr>
<tr>
<td></td>
<td>Literature review</td>
</tr>
<tr>
<td></td>
<td>Survey</td>
</tr>
<tr>
<td></td>
<td>Case study</td>
</tr>
<tr>
<td></td>
<td>Experiment</td>
</tr>
<tr>
<td></td>
<td>Action research</td>
</tr>
<tr>
<td></td>
<td>Systematic review</td>
</tr>
<tr>
<td></td>
<td>Comparative study</td>
</tr>
<tr>
<td></td>
<td>Industrial experience report</td>
</tr>
</tbody>
</table>

### 3.2.1.2.4 Search Databases

Only few databases were used to identify relevant literature which includes articles, journals, work shop papers, and conference papers. These databases were limited and chosen because of the availability of full access to the content. They have been used as a source for other reviews in Lean, but the researcher maintained to choose only these databases which are proposed often
by other researchers in this area and are most commonly used for software engineering studies also [2].

<table>
<thead>
<tr>
<th>Database name</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEEE Xplore</td>
</tr>
<tr>
<td>ACM Digital Library</td>
</tr>
<tr>
<td>Engineering Village(Compendex, Inspec)</td>
</tr>
<tr>
<td>Willey-Inter Science</td>
</tr>
<tr>
<td>Springer Link (access to some paper only)</td>
</tr>
</tbody>
</table>

Table 7: Databases searched

3.2.1.2.5 Search procedure for Automatic search

To reduce the effort and unnecessary search operation this procedure is followed:

- Form a search string for a certain database. Break the search string into small strings if
  the database does not support advanced/big search.
- Type the formulated search string in the search space of the database if provided. Otherwise,
  enter keywords manually by adding or attaching Boolean operators to the key terms. Selecting
  the fields to search depends upon the database being used. The selected field may be the title,
  abstract, or the whole article.
- Run the search.
- Restrict to journals, books and language if possible.
- Download Endnote data with abstract for all papers (all at once or for individual papers depending on database support).

3.2.1.2.6 Paper selection process

This section describes the selection criteria and the selection process used. The following
criteria helped limit the process to only those results that were relevant in this systematic
review. Selection of literature was based on the objectives of this thesis as to identify the factors
that are causing a waste and any relationships between them. This includes the help of the
inclusion/exclusion criteria which were formulated. The important criteria for including a study
is whether or not the study provides the root cause of waste, and explains about waste in detail
or its industrial validation if provided. To ensure that the paper selection meets the aim of the
thesis a list of guiding principles was prepared and followed for the inclusion and exclusion.
Only those studies which fulfilled the criteria were included from the search results. The
process of selecting relevant studies was based on screening the keywords, titles and abstracts of the studies. First keywords were screened and if satisfactory, then the following steps were continued.

Primary selection process:
- All the titles and abstracts of each study are reviewed with the inclusion and exclusion criteria.
- If both title and abstract do not meet the inclusion and exclusion criteria then the study is removed.
- If there is still doubt after reading the title and abstract the study is included for further reading.
- Only studies that are included will be used for further study in extracting data.
- Data extraction strategy (APPENDIX 5) is used to extract root causes and different relationships of wastes.

Table 8: Inclusion and exclusion criteria

<table>
<thead>
<tr>
<th>Inclusion criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>The following inclusion criteria were taken into account when selecting the primary studies:</td>
</tr>
<tr>
<td>• The study is peer reviewed, available in full text.</td>
</tr>
<tr>
<td>• The study discusses a waste of lean in software development.</td>
</tr>
<tr>
<td>• The study discusses any classification of the causes of waste.</td>
</tr>
<tr>
<td>• The study will be included if it gives an overview cause of any waste of lean in software development.</td>
</tr>
<tr>
<td>• The study evaluates, analyzes a waste, or compares two or more wastes of lean in software development.</td>
</tr>
<tr>
<td>• The study evaluates or analyzes an existing relationship between wastes of lean in software development.</td>
</tr>
<tr>
<td>• The study will be included if it discusses a validation of application of lean in software development.</td>
</tr>
<tr>
<td>• The study will be included if it refers to other primary studies.</td>
</tr>
</tbody>
</table>
Study exclusion criteria

- Studies that are not in English.
- Studies that do not meet the inclusion criteria.
- Studies related to application of lean not related to software engineering.
- Studies that describe a waste but do not provide anything further about it.

The following table shows the resultant, included studies from the systematic process at different stages of the review. From 2,761 articles, only 139 articles were selected to read for full text evaluation; this was obtained by using the detailed inclusion and exclusion criteria on all studies. At the end of the full text evaluation only 52 studies were selected.

Table 9: Number of Studies with respect to Review Stages

<table>
<thead>
<tr>
<th>Database</th>
<th>Total</th>
<th>Without Duplicates</th>
<th>After Title and Abstract Review</th>
<th>After Full Review</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compendex/Inspec</td>
<td>1116</td>
<td>725</td>
<td>23</td>
<td>6</td>
</tr>
<tr>
<td>IEEE Xplore</td>
<td>259</td>
<td>257</td>
<td>41</td>
<td>20</td>
</tr>
<tr>
<td>Wiley Inter Science</td>
<td>466</td>
<td>466</td>
<td>29</td>
<td>12</td>
</tr>
<tr>
<td>ACM Digital library</td>
<td>920</td>
<td>918</td>
<td>46</td>
<td>24</td>
</tr>
<tr>
<td>Total</td>
<td>2761</td>
<td>1175</td>
<td>139</td>
<td>52</td>
</tr>
</tbody>
</table>

The researcher do not have full access to the springer database except few studies can be accessed freely. Only four studies were identified in the springer database which have access to the full text were chosen. Five studies were selected which are in conflict with the inclusion/exclusion criteria and are not peer reviewed but still considered important to this work.
3.2.2 Snowball Sampling

In order to find more related studies on sources of each waste and their relationships, the researcher used Snowball sampling. Snowball sampling is proposed by Goodman to recruit future subjects from existing subjects. This is done by gathering more relevant data by building up existing sample data. Snowball sampling is used in sociology and statistics to find relevant work. Snowball has been adapted to software engineering and to other fields of studies. Snowball sampling is a method to either formally or informally reach populations with either individual or network structure for making inference [45]. Hidden studies that cannot be identified through direct search, and studies which are not published in databases that were used for this SLR, were identified with snowball sampling. The method’s process is to read all references of the study by the titles. If relevant, the abstract of the study and further the whole paper is read, applying inclusion/exclusion criteria and data extraction for each study. Time is also spent finding the authors’ profile pages that are relevant and finding more relevant studies if they were done. The researcher performs this process to the studies, which are the final selected papers obtained from using SLR. The resultant of this process was that six additional papers were gathered.

Altogether, sixty seven studies were identified using different methods in entire literature review.

3.2.3 Threats:

Some of the threats identified:

Selecting primary studies by discarding large number of papers is one of the internal threats. To overcome this, the researcher has to read the full text of each and every paper and highlight to see if it satisfies the criteria, and also manually search for the causes of waste in LSD.

Because of this process, the chance of missing the important papers can be reduced. In this thesis construct validity is relevant for the SLR. The search string may not be able to reveal all research data presented in the literature. The researcher have selected the search terms with the direction of a librarian. The supervisor of this work who is a senior researcher in this area also checked and approved the search string.
Coding for textual description during data analysis is another threat. To overcome this problem proper care was taken during filtration of large amounts of data. There can be a problem of misinterpreting the data and applying wrong codes to the data.

3.3  Interview study method:

In this method the researcher formulated questions for the problem he wanted to solve and asked those questions to each subject who participated in the study. The researcher was involved personally in posing questions to the subject. The subjects could be either groups or individuals depending on the design of the interview and situation. Interviews were generally conducted to obtain information in explanation. The advantage of interviews was that it would generate a lot of information with a single question, but it takes a lot of time to plan for an interview and gather subjects to interview. Interviews are categorized into three types. They are unstructured, semi-structured, and structured [46].

3.3.1 Types of Interviews:

3.3.1.1 Unstructured Interview:

In this way of conducting interviews, the interviewer poses the questions which are based on the situation, are not organized and not structured [46]. The interviewer poses the questions without any prior planning and the questions are limited to researcher’s interest [47]. In this way of interviewing with various questions for different participants, researchers may get different answers and could draw conclusions with bias if no proper care is taken.

3.3.1.2 Semi-structured Interview:

This interview is of descriptive and explanatory form [47]. In this way of interviewing the questions are prepared in a structured way in which the interviewer follows a sequence of asking questions. He can stop at any point in the sequence and can concentrate more on a particular question for detailed information and can then continue in the sequence. In this way of interviewing one can get more and detailed information which needs high interaction between the researcher and participant while conducting the interview. Sometimes it takes longer time than planned time to complete the interview.
3.3.1.3 Structured Interview:

In this type of interview before going to conduct interviews everything should be pre-planned and fully organized with all preparations ahead of interviews. Nothing would be changed while conducting the interview this is a questionnaire based interviews. All questions should be asked in the same way as they are constructed and the sequence of asking should not be changed or altered. Misinterpretation of data can be reduced with this method of interviews.

3.3.2 Interview Design

This research was carried out using the semi structured interview method on seven subjects. The study helps to understand a specific case of waste removal in a given context of Lean Software Development.

To know the root causes of different wastes that occurs while working in the industry, The interview was conducted with professionals currently using lean methodology in the software industry. Seven interviews were conducted in a period of one month. The participant’s work with different organizations and are handling different projects. Some of these participants are managers and lean methodology consultants working on different phases of software development. In this study, the seven standards for investigation and interview was followed to examine different relationships between wastes in LSD. This standards and procedure steps were taken from Kvale [48]. These are:

1. Thematizing
2. Designing
3. Interviewing
4. Transcribing
5. Analyzing
6. Validating
7. Reporting

3.3.3 Thematizing

In this step of Thematizing how the interview was being conducted is described. The rationale behind this investigation was to identify all the root causes and common causes between different wastes in LSD from professionals. Participants were identified through searching on
different websites and scanning many profiles of people who are experienced with LSD. The websites like LinkedIn, Twitter, MeetUp, Xing were searched for interviews participants who were then contacted through mails, telephone, and personal contacts. The researcher have identified participants in some companies by calling to HR’s and sending emails to them to provide those who are experienced in working with LSD. The importance of this study and the benefits to the industry is sent to every participant, whose profile matches as a subject to the study. Convenience sampling method was used on the identified participants which involve people who are willing and available to participate. When the participants were willing and interested to participate, an interview protocol with questionnaire was sent in advance of 10 days least. The researcher had to wait for their reply on this “interview questionnaire” mail. At the end, seven participants were interviewed. Given the time constraints, the analysis was based on the responses of these seven participants. Though one can argued that if the number of participants were increased or double, more factors would have been identified, the researcher believed that there would have been no much variations from the view of the experts with whom the interview were conducted. Thus, the researcher believed that the validity of the result is still maintained. To conduct interviews various platform (Skype, face-to-face meeting) were given to choose as an option to fit the researcher and subjects.

3.3.4 Designing

It is a way of getting intended knowledge by using a plan with some methods, all this process is performed in this Designing [48]. For achieving highest quality of the interviews its design is very important because of its direct impact on it [49]. The data obtained through the interview will be dependent on how it is achieved so the researcher took all care that need in designing an interview. The researcher choose semi structured interview and which suits for this study because of its nature to explore more root causes to the wastes of LSD and also it is widely used in software engineering [50]. In this study semi structured interviews were conducted, with questions designed in a way that need to be asked as it is, and also open-ended questions for flexible way of asking [51]. Formulation of Questionnaire for Semi-structured interviews was prepared and refined several times before the interview. Further changes were made to the questionnaires on the comments received from the supervisor (who is a senior researcher on LEAN). Questionnaire was formulated on the basis of literature review and aim of this study which had helped to ask certain causes which are missing in literature. Previously conducted interviews in this area was taken as a protocol to follow in formulation of questionnaire. Semi
structured interviews helped to ask more questions about some factors in detail. This is done because the researcher know some of the factors which were already identified in literature. So the factors identified in the literature review was also helped in asking quality questions in the interview.

3.3.5 Interviewing

Interviews were arranged by the researcher by requesting different participant’s availability. For this study the researcher used a semi-structured interview strategy to collect the data. One interview and one interviewer attended most of the interviews but interviewer used another person in some interviews for taking notes. The interviews are some face to face and some are synchronous online interviews. During the interviews at the starting the purpose of the study and general explanation of LSD were presented to the interviewee. For all the interviews, the time is varied in span of 80 to 105 minutes. The researcher took in the form of, audio recordings, written extensive notes to facilitate and improve analysis process. Written extensive notes were taken for two participants where it was not allowed to take audio recordings. The main advantage of audio recording is the researcher did not bother about taking notes and only concentrated on discussion about different wastes and their root causes and asking more questions which are not considered earlier. It took more time for interviews when audio recording are not allowed and only written notes is permitted. For this most of the times the subjects should repeat the answer again because of speed of taking notes. While conducting the interview the questions were not asked in same order as they were prepared. The flow of asking questions was changed with the development of conversation which has decided of asking questions, but still researcher achieved improvisation and exploration with the interview questions as the interview proceeds. Later transcripts of each interviews were promised to be sent to the respective participant to double check with the reply they have given.
3.3.6 Transcribing

Transcribing is an interpretative process by itself where interviews were transcribed from speech to text. All the data obtained from different subjects through methods like audio recording and written notes were transcribed into digital text form. Keeping track on all the causes and relations of the factors that were exposed or raised during the interview is the main purpose of the recording. For producing consistency of transcription the transcription is done twice and compared each one and concluded with a final one. For validation of transcripts there are many methods example conducting a survey, or sending the transcript to the interviewee to verify the interpretation. In this thesis each transcripts of each interview were sent to the respective participant to double check their reply whether researcher correctly interpreted or not. Researcher has also given assurance that transcript data will be kept anonymous and will not be presented anywhere in the thesis if requested.

3.3.7 Analyzing

Grounded theory (GT) is used to analyze both the interview transcripts date and literature review data in this study. GT is used because it is best suits to this research even though many approaches were available for analyzing interview transcripts for categorization, condensation, ad-hoc, narratives [48]. It provides the valuable tools which of them helps in analyzing the qualitative data in an unbiased way. The concepts applied from the GT to this research are reduction of data through coding, develop a theory that incorporate different root causes of LSD. There are other analysis methods available which also suitable for this research for example Thematic analysis, Content analysis [43]. Thematic analysis it has the limited interpretative power beyond mere description if it is not used in framework. While content
analysis sometimes may fail to reflect structure or importance of underlying phenomenon and the results may be over simplified [43]. These methods were rejected due to their short comes and due to high advantages provided by GT to analyze the data of this research. Because of the nature of this research which to identify the factors of the wastes GT was chosen.

3.3.8 Coding Process:

**Open coding:** Open coding is coding process where temporary codes such as provisional and best grounded in the data. These temporary codes were selected as the conceptualization of the data i.e., to have as much as close meaning to raw data.

**Axial coding:** Axial coding is to build conceptualize structure of the data where data is interrelated and linked to each other which can explain the phenomena more precisely [52].

**Selective coding:** In order to construct theory, a process applies to integrate the concepts to build more general categories to explain theory about a phenomenon, and final results need to be validated by comparing concept relationships data with the raw data and refine the relationship if necessary [52].

Samples of coding process are exemplified in two slightly different forms of data of the literature review and interviews.

Two pieces of data namely x and y were extracted from the literature.

Data X: The most frequent consequence of cross-site problems was delay in the resolution of work issues. By delay, we mean the additional time it takes to resolve an issue when more than one site is involved [53].

Data Y: ”…if a part of the design or code needs to be changed, or if someone needs a better understanding of how some part of the product works, people at more than one site may need to be involved in information exchange, negotiation, and so on,…”

Data X is an excerpt extracted from literature as it was written. From this excerpt, it can be inferred that work issues are the main activities that should be performed which are delaying the cross site performance. Based on this we coded as “work completion issues”. Data Y is a note taken from several articles. From this one may deduce lack of human technical knowledge
lead to cross site collaboration. So it is coded as “human knowledge”. Furthermore those two
challenges were grouped into a general concept as human technical factors. This concept is
again grouped into a general category as “Technical factors”

An example from the interview of how coding process is performed:
Interview transcript 1: “Documentation is reduced to large extent because we are sitting close
to each other if we need anything we are asking one another immediately”.

Interview transcript 2: “I need to check the document again and again sometimes I found it is
unclearly stated about a component. After some time I give a message to them about the
problem they will send me another document with some changes”.

In interview 1 the interviewee stated that how documentation is reduced with the use of
alternative communication method, from this the author coded as “minimizing
documentation”. The same excerpt is also used in another coding “communication medium”.
In the second excerpt the interviewee stated the importance of technical documentation and the
content. From this the author coded as “technical documentation”, “documentation errors”,
“importance of documentation”. The codes from interview 1 & 2 i.e., “minimizing
documentation”, “technical documentation”, “documentation errors”, “importance of
documentation” all together belong to code “content of documentation” then grouped to more
generalized concept as “documentation issues”. This “documentation issues” concept was
finally added to “non-technical factor” category.

The results of GT coding for the SLR and the interviews are shown in Table 10

<table>
<thead>
<tr>
<th>Coding stage</th>
<th>Number of codes (Literature review)</th>
<th>Number of codes (Interviews)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open coding</td>
<td>121</td>
<td>116</td>
</tr>
<tr>
<td>Axial coding</td>
<td>16</td>
<td>10</td>
</tr>
<tr>
<td>Selective coding</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 10: Results of Grounded Theory Coding
3.3.9 Threats:

3.3.9.1 Construct validity:

Construct validity is the researcher’s lack of personal relationship with the subjects. The validation of the interview results is done by sending the interview reply again to the subjects to double check if their reply is correctly interpreted or not. This even helped to know if a subject has changed their opinion on a particular reply and wants to change the answer. This type of validation helped in getting accurate data from the interviews which have increased construct validity.

3.3.9.2 Internal validity:

The questionnaire was designed covering all the wastes in lean for a semi structured interview least 75 minutes of discussion was made in each interview. Manual data is change to digital data. The total interview data was more the 10 hours (audio and written) based on many causes of wastes. To remove irrelevant discussions from the whole data, the data is checked again and again and omitted irrelevant data. All the general discussions were kept out of the study so that only relevant data is used for analysis.

3.3.9.3 External validity:

External validity is the threat, it deals with the ability to generalize the findings of the study to a specific context, that the results are viewed from only specific organizations case study, to generalize results from analytical to broader theory [36, 38]. To mitigate this threat the study is done in different companies with people who are working on different projects on developing various software which makes the results generalizable. With different subjects, and each one is different with another in many areas such as different organization but context they are working is same which was explained in detail and the results are generated from these studies. The participants of this study are having different cultural backgrounds and language. To overcome this threat meetings were conducted according to the agreed meeting time and participant were chose only if they can answer in English.

There was a threat that the participants will give faulty data because of lack of trust to overcome this candidates were given full assurance that their details will be kept anonymous and selected only if they were interested.
Even though wastes in lean is a vast topic of the area of study, it fits in any organization to replicate this study for obtaining similar findings. The results obtained from this study thus can be generalized in the same context for any case organization or companies.

3.3.9.4 Reliability:

The goal of reliability is to eliminate biases in the study if any by minimizing errors in the study. If another investigators does the same study by not replicating this study but doing another case study in the same settings at the end of the study their results, shouldn’t differ much from this study conclusions, and be in line with this study findings [36]. To overcome the reliability issues the study protocol and data collection protocol is designed in detail and documented every step prior to study and followed without any compromise while conducting the study. There is a risk of misinterpretation of the data that were collected by the researcher from different sources which will have great effect on the outcome of the study. To minimize and mitigate this risk the analysis is thoroughly reviewed by the supervisor to suggest and point out any major problems associated with the process of interpretation and revised the outcome. To double check if the data are correctly interpreted the recording of audio was transcribed twice and compared each to a final one.
Chapter 4 Results

4.1 Following are the results of Literature Review on the basis of 67 primary studies selected.

Characteristics of the selected studies:

Statistical data is represented in numerical form to show different perspective representation of primary studies like year of publication.

A presentation of publication years of selected primary studies is shown in below figure 2. From this presentation it is difficult to draw some conclusions about research related to causes of wastes in LSD.

![Articles found](image)

Figure 4 Publication over time

From the above figure shows articles from 2000 to 2014. The reason for obtaining only few articles is because wastes and their causes in LSD studies started evolving since 2000. The empirical evidences on these studies were very few because companies started adaptation of lean very slowly but in recent years there is an increase in lean literature. The following is the studies distributed in various database.
4.2 Different factors related each wastes:

Partial done work:

![Diagram showing factors related to partial done work]

**Requirements Issues**

**Literature:** Managing requirements is an important phase of requirements engineering and software development. It is a fundamental method of how a software would be at the end. If some of the requirements are missing then the function and performance of the software will not meet the customer needs [A2]. Missing requirements always cause delay of project and defected products [A33]. Incomplete requirements specification can introduce defects into systems source code without the knowledge of developer. This might cause by analysts being careless in identifying requirements and omitting certain needs due to ineffective communication with customers. Communication deficiencies with the customer and missing of the true customer requirements create incompleted software [A9]. All efforts to create a software that does not support a customer need is a waste. Lack of requirements traceability and unstable requirements are some of the causes of defective product [A52]. Lack of responsiveness to the customer needs causes incompleteness. When faults are not detected in
early stages and requirements are kept in big batches, errors are hidden could result in uncompleted and defective product [A3, A6, A11]. Even changes to the requirements or late elicitation of requirements which are performed during the design, coding and test processes also cause delays to the project [A36]. Over designing a system to anticipate future requirements changes which will allow extra features to be added to the system results in a huge waste. This is because instead of designing a system that supports the immediate needs of customer, resources are devoted to designing and developing extra features into a product for future use which customer do not use in present time resulting in late product delivery [A19].

Interview: Misinterpretation is a challenging factor between customers and requirement engineers as it could lead to development of the software completely away from the customer’s expectations. Sometimes organization abandons the work in the middle of the project due to misinterpretation. Sometimes inconsistencies within the requirements unidentified before the start of the development put the development on hold until clarification is got from customer. This causes delays in the progress of the project or even abandon in middle of the project. Missing requirements or where sufficient requirements are not gathered in the early phase of software development life cycle can result in unfinished product. Poor analysis of the requirements can lead to inappropriate conclusions and it has negative impacts on the product development. Starting the project ahead of the completion of requirement description can result in the project being stopped half-way. Sometimes requirements contain defects because of failure to write test cases to them. Other times, some requirements are inconsistent or unstable which might not allow project run smoothly. All these different issues arise due poor requirements engineering. Taking and developing unnecessary or unwanted requirements wastes resources.

Work environment Issues

Literature: Employees' errors can be increased with unfamiliar and too complex work operating environment. The problem is increased if the operating environment is confusing and static. This can result in delayed or uncompleted works. Complexity can come with large modules that are inseparable and which are of different variants having only slight changes thereby creating confusion in implementation [A8]. Distributed work on complex environment with more people involved and more inter-dependencies of sites may create delays in development.
cycles. Having modules that are interdependent on each other but are to be implemented in distributed environment may create delays.

**Testing Issues**

**Literature:** Buildup of untested code is the most frequent problem of partial done work. This can be caused by the quality assurance team being busy with other tasks and unable to test the incoming task. This creates huge backlogs of tasks to test [A11]. Rapid feedback can be obtained through mechanisms like continuous testing of the system thereby reducing waste. Continuous testing helps to give frequent deliveries where customers speedily evaluates and approves the developed features [A21]. Testing delays can keep developers idle and waiting for the test results before starting to work on the next piece of code. This is because the developers might think that further coding is required to fix the defects that are identified in the testing process [A3]. To identify defects, only suitable techniques should be used in the test cases because every technique cannot be applicable in each case. The selection of proper approach for test should be based on the resources required for that approach, the resource availability, and features of the approach. Lack of all these leaves defect in the product unidentified, which is a big waste [A48]. Not providing enough resources to identify defects for the testing teams does not help. Testing teams should be equipped with both dynamic and static testing techniques. Environment should be created for testing which should allow smooth testing process [A53]. Changing user requirements always have the possibility to create new code or modify existing code. Significant changes to existing code or adding new code tends to lead defect-prone code [A65]. Proper care should be taken to prevent defects creeping in. Sound testing strategy should be used with mature tests that can avoid defects otherwise, wastes will lead to rework [A67]. If there is no coordination between development groups then it leads to design of poor, ineffective and inefficient tests which may not be able to detect defects.

**Interview:** Conducting or starting the testing process too late leaves the defects undetected in the final product. Early identification and fixing of defects can stop the defects from further spreading. It is easier to identify and fix defects when the system or code is smaller in size than huge in the later stages. Sometimes early stages are less likely to have because of the initial comprehensive test procedures. Testing process should be started as soon as requirements are finalized. Defects can occur by many ways either by people, process, or technological aspects. People aspects might be lack of experience and collaboration between members, less domain knowledge of system, expert’s unavailability, lack of knowledge of frequent updates on
technologies, lack of mature team members on testing process etc. Process aspects like lack of review of code, poor inspection etc. Technology aspects like high degree of dependency between tasks, inconsistent and unsupportive tools, outdated test infrastructure, insufficient testing resources. Testing ability of customer also plays an important role in defect identification. Late testing by customer may lead to dissatisfaction on the product.

**Technical management**

**Literature:** Duration of projects is dependent on many factors. To solve some of the factors, collocation of resources can be very helpful. Facilitating collocation of infrastructure to the group provide many benefits to deal projects durations [A26]. Even though adopting 'collocation' can be difficult for group members at first, members will gradually adjust to it. This can be done by providing and explaining the benefits to the projects of collocating teams including less defects, less delays and reduced extra processes or handovers. Team member with high levels of domain knowledge of the system and all technical aspects that are needed for system to develop can dramatically reduce the duration of the projects. Otherwise, lack of technical and system domain knowledge can always increase the projects duration or even end up leaving the projects as partial done work [A26]. Providing enough resources and sharing limited resources can be facilitated if the team members are collocated. Optimizing the resources is possible to the teams if they are collocated.

**Human Technical factors**

**Literature:** Human technical skills are important in software projects. They always contribute to the projects in one way or the other. A highly skilled member can solve complex problems. Shortage of skills leads to delaying the work. A group should have variety of skills. If a new member joins a group the member should be trained. All the members in the team need to be trained in many different points of view of the project so as to work in the project effectively. Inadequate training can lead to project delay and defective product [A1]. Members of the team should be well trained and should be experts. Experts are always expected to take ultimate decisions which can gradually reduce delay and the handover wastes. Good knowledge of system technical aspects and domain aspects can always eliminate the uncompleted work and reduces the delay in completing the work [A26]. Lower capability of the member of the team (for example in programming, testing, analyzing etc.,) or lack of sufficient experience to solve problems always causes software projects delays. Adding new member who is inexperienced to an already delayed project always leads to further delays [A28]. Members of testing team
being unable to judge the defects or being biased in judging a defect can cause the defect to be undetected and lead to delays [A46]. In the requirements, lack of peer-reviews of the requirements leads to defects. Revision of the requirements many times by peer reviewers can drastically bring down the defect count [A61]. Allowing team members to work on unfamiliar technologies or new technologies where experience is lacking can result in defective product for example due to missing components quality [A51]. Changing models too often during development with matching experiences can result in defective product [A52]. Uneven or unbalanced distribution of developers across locations produces defective products inexperience being the main cause [A62]. The capability of the member in detecting a defect reflects their experience on identifying and removing the severe defects in the product [A63].

**Interview:** Inexperience is a huge problem for a company. High turnover rate put the company in difficult situations. The capability of the company is dependent on the experience of its people. A huge implementation project that is unmatched by adequate high skilled person would be a failure. Shortage of skilled members may result in being unable to complete the work and increase time-to-market. However, good estimation of market can create high value for the customer. Utilization of tools is dependent on human skills but lack of adequate relevant skills always delay the process. Employees’ poor knowledge of the latest technologies can adversely affect the company’s growth rate. Sometimes, there may need to make sudden changes to the process of a finished project, adapting to these unexpected changes can be of benefit to the company. Employee’s ability to resolve problems by immediately resolving the work issues will always generate high value to the organization and eliminate waste or delay. Creating clear process and defining clear procedures in the work established by the employees can also create great benefits to the company, having those people in the company always achieves customer’s expectations.

**Resource management**

**Literature:** Resource estimation and planning are important before the commencement of a project. Resources can be anything that allows completion of the project on time. Lack of resources planning can lead to wastes in “lean software development”. If resources are not available project solutions cannot be framed. Lack of sufficient resources to the team members will always have a negative impact on the performance of the team. Proper estimation of resources required to implement a project and technical factors like features, versions will help to make software development process easy [A53].
**Documentation:**

**Literature:** Documentation helps to save the company when there is high turnover rate. Documentation provides a great aid to a new or inexperienced member to understand a service that has complex issues. Defective software can be created when the requirements are documented unclearly [A3]. Lack of up-to-date documentation affects software maintenance. This will cause the developer to redo the documentation of the entire system which increases waste of extra process [A13]. Documentation is any artifact or the medium that communicate to a user of software system [A13]. Many people have found that inadequate, excessive or missing documentation creates problem to all phases of software development life cycle [A14]. Documentation provides an effective and flexible way of recording informal content [A16]. Documentation is an aid for maintenance which helps in understanding high-level design, implementing, and functional nature of complex applications [A14]. Too much documentation can cause waste of extra process, less documentation increases defects and inventory [A7]. Documentation is an important tool for communication [A17].

**Interview:** Lack of documentation or consequence of no documentation - where instructions or guidelines are missing which are able to aid the developer on implementing a task is a cause of waste or partially completed work. Missing description on how a final state of the system should look like can stop the project half-way. Documentations are created to keep or act as an archive about different aspects of the system such as what are the functionality, technical knowledge and business domain of the system. Not having these documentations will lead to extra processes. Good documentations remove all the misconception regarding the software and keep track of the information regarding all activities. Lack of documentation increases handover waste because of new decisions needed for some problems.

**Experience:**

**Literature:** When an unforeseen change (situation) occurs and requires new information to deal with, this information cannot be searched anywhere. This type of problems can be solved through experience. Lack of this required experience leads to partial done work. Always an experienced staff should guide other staff and new ones. Insufficient experience is the main cause of software project delay [A28]. It is a rational that an experienced person would adjust to the changes or situations in quick time than an inexperienced persons. High levels of experience can significantly minimize the manpower for a project and accelerate the progress of a project. If a team member is has the required experience in analyzing product defect, then
partially done work will be eliminated [A58], if a team member is an expert in defect identification, the product will be free from defects. Experience can greatly impact all wastes of LSD.

**Interview:** People are working on the same task again and again to fix some things in the task because of inexperience on the system. Sometimes experts should guide from the starting to the end or even people should be changed in the middle of the task because of lack of experience. Lack of experience utilizes more resources of the company. Quality of product is always high if more number of experienced people were involved in software product development. Most of the wastes of LSD can be completely minimized with experienced personnel.

**Feedback:**

**Literature:** Feedback can show problems and errors in the process or product. Feedback shows the impact achieved by the quality of the product and productivity of the organizations. Organizations learn from feedback and can continuously reduce the process and product variations from the customer and different stakeholders. Short and frequent feedback can give required and useful features delivered to the external or internal customers or internal customers [A6]. Continuous customer involvement mostly gives the correct products delivery, otherwise lack of feedback increases inventory. Continuous feedback gives continuous testing of the system which can make it free from defects. Frequent deliveries enables continuous feedback from the customers where customers evaluate and approve the system through this, extra features can be eliminated [A19]. Long waiting times between feedbacks can point out many defects in the system which cause rework [A39]. Strong impact of the feedback is that it can decide the duration of the software development process.

**Interview:** Feedback from different stockholders always contributes to give a perfect system. Feedback can point out defects in the product. Quick feedback reduce delays as well as helps in early completion of project. Feedback removed extra feature from the product and help to proceed until completion of software development project without abandoning in the middle.

**Customer issues**

**Literature:** Responsiveness to the customer needs is important to the software development. The value he had received, the delivery received and the time it was received for his needs are all important to the customer. Some practices like on-site visits and frequent deliveries are
important to the customer [A4]. Lack of responsiveness to the customer with respect to timely delivery increases inventory. Customers always want to know whether sufficient quality of the product is delivered or not. Lack of responsiveness to this quality aspect creates defective product which leads to rework [A11].

**Interview:** Customer involvement in phases like requirements and testing can gives great benefit to final software in terms of how the product looks like and how it functions. Customer involvement give company good advantage it remove confusions in the work and give exact system that is required. When the customer involve by delivering feedback and testing each components the product comes without delays, and free from defects. Customer can cope with delays or defects when they were involved from start to end of the project so the company don’t lose business. Isolation of customers from the product development leads to different wastes of LSD.
**Work environment Issues**

*Literature:* Employees' errors can be increased with unfamiliar and too complex work operating environment. The problem is increased if the operating environment is confusing and static. This can result in delayed or uncompleted works. Complexity can come with large modules that are inseparable and which are of different variants having only slight changes thereby creating confusion in implementation [A8]. Distributed work on complex environment with more people involved and more inter-dependencies of sites may create delays in development cycles. Having modules that are interdependent on each other but are to be implemented in distributed environment may create delays.
**Technical management**

**Literature:** Duration of projects is dependent on many factors. To solve some of the factors, collocation of resources can be very helpful. Facilitating collocation of infrastructure to the group provide many benefits to deal projects durations [A26]. Even though adopting 'collocation' can be difficult for group members at first, members will gradually adjust to it. This can be done by providing and explaining the benefits to the projects of collocating teams including less defects, less delays and reduced extra processes or handovers. Team member with high levels of domain knowledge of the system and all technical aspects that are needed for system to develop can dramatically reduce the duration of the projects. Otherwise, lack of technical and system domain knowledge can always increase the projects duration or even end up leaving the projects as partial done work [A26]. Providing enough resources and sharing limited resources can be facilitated if the team members are collocated. Optimizing the resources is possible to the teams if they are collocated.

**Human Technical factors**

**Literature:** Human technical skills are important in software projects. They always contribute to the projects in one way or the other. A highly skilled member can solve complex problems. Shortage of skills leads to delaying the work. A group should have variety of skills. If a new member joins a group the member should be trained. All the members in the team need to be trained in many different points of view of the project so as to work in the project effectively. Inadequate training can lead to project delay and defective product [A1]. Members of the team should be well trained and should be experts. Experts are always expected to take ultimate decisions which can gradually reduce delay and the handover wastes. Good knowledge of system technical aspects and domain aspects can always eliminate the uncompleted work and reduces the delay in completing the work [A26]. Lower capability of the member of the team (for example in programming, testing, analyzing etc..) or lack of sufficient experience to solve problems always causes software projects delays. Adding new member who is inexperienced to an already delayed project always leads to further delays [A28]. Members of testing team being unable to judge the defects or being biased in judging a defect can cause the defect to be undetected and lead to delays [A46]. In the requirements, lack of peer-reviews of the requirements leads to defects. Revision of the requirements many times by peer reviewers can drastically bring down the defect count [A61].
Allowing team members to work on unfamiliar technologies or new technologies where experience is lacking can result in defective product for example due to missing components quality [A51]. Changing models too often during development with matching experiences can result in defective product [A52]. Uneven or unbalanced distribution of developers across locations produces defective products inexperience being the main cause [A62]. The capability of the member in detecting a defect reflects their experience on identifying and removing the severe defects in the product [A63].

**Interview:** Inexperience is a huge problem for a company. High turnover rate put the company in difficult situations. The capability of the company is dependent on the experience of its people. A huge implementation project that is unmatched by adequate high skilled person would be a failure. Shortage of skilled members may result in being unable to complete the work and increase time-to-market. However, good estimation of market can create high value for the customer. Utilization of tools is dependent on human skills but lack of adequate relevant skills always delay the process. Employees’ poor knowledge of the latest technologies can adversely affect the company’s growth rate. Sometimes, there may need to make sudden changes to the process of a finished project, adapting to these unexpected changes can be of benefit to the company. Employee’s ability to resolve problems by immediately resolving the work issues will always generate high value to the organization and eliminate waste or delay. Creating clear process and defining clear procedures in the work established by the employees can also create great benefits to the company, having those people in the company always achieves customer’s expectations.

**Documentation:**

**Literature:** Documentation helps to save the company when there is high turnover rate. Documentation provides a great aid to a new or inexperienced member to understand a service that has complex issues. Defective software can be created when the requirements are documented unclearly [A3]. Lack of up-to-date documentation affects software maintenance. This will cause the developer to redo the documentation of the entire system which increases waste of extra process [A13]. Documentation is any artifact or the medium that communicate to a user of software system [A13]. Many people have found that inadequate, excessive or missing documentation
creates problem to all phases of software development life cycle [A14]. Documentation provides an effective and flexible way of recording informal content [A16]. Documentation is an aid for maintenance which helps in understanding high-level design, implementing, and functional nature of complex applications [A14]. Too much documentation can cause waste of extra process, less documentation increases defects and inventory [A7]. Documentation is an important tool for communication [A17].

**Interview:** Lack of documentation or consequence of no documentation- where instructions or guidelines are missing which are able to aid the developer on implementing a task is a cause of waste or partially completed work. Missing description on how a final state of the system should look like can stop the project half-way. Documentations are created to keep or act as an archive about different aspects of the system such as what are the functionality, technical knowledge and business domain of the system. Not having these documentations will lead to extra processes. Good documentations remove all the misconception regarding the software and keep track of the information regarding all activities. Lack of documentation increases handover waste because of new decisions needed for some problems.

**Geographical issues:**

**Literature:** In geographically separated sites projects where design and implementation takes place in different geographically separated sites. Direct contact of the members with other members is not possible frequently and also takes lot of effort and consumes many resources and increase in cost. The separation of design and implementation consumes and uses many mediums to communicate with each other. Documentation is one of the main medium of knowledge sharing processes [A17]. Geographical separation leads to delay in communication because of time difference. The response time is impacted by the distance between two sites. More number of sites collaboration makes communication complex which lead to delay in the project and takes many handovers for decisions and approvals. Project planning and control and effort estimation is more difficult if the teams were geographically separated [A34].

**Feedback:**

**Literature:** Feedback can show problems and errors in the process or product. Feedback shows the impact achieved by the quality of the product and productivity of the organizations. Organizations
learn from feedback and can continuously reduce the process and product variations from the customer and different stakeholders. Short and frequent feedback can give required and useful features delivered to the external or internal customers or internal customers [A6]. Continuous customer involvement mostly gives the correct products delivery, otherwise lack of feedback increases inventory. Continuous feedback gives continuous testing of the system which can make it free from defects. Frequent deliveries enables continuous feedback from the customers where customers evaluate and approve the system through this, extra features can be eliminated [A19]. Long waiting times between feedbacks can point out many defects in the system which cause rework [A39]. Strong impact of the feedback is that it can decide the duration of the software development process.

Interview: Feedback from different stockholders always contributes to give a perfect system. Feedback can point out defects in the product. Quick feedback reduce delays as well as helps in early completion of project. Feedback removed extra feature from the product and help to proceed until completion of software development project without abandoning in the middle.

Communication:

Literature: Minimizing communication gaps from the starting to the end of the project life cycle gives the product that is likely to reach customer expectations. Lack of good communication can lead to partially done work [A5]. Good communication should start from requirements phase itself because it can impact on the final software product. Good communication between customers and requirements engineers bring about stable requirements. Good communication between requirements engineers, developers and testers gives a non-defective product [A5]. Communication breakdown is one of the root causes of all wastes in LSD. Communication either through formal or informal channel is necessary to the software development. Direct communication reduces documentation process but lack of communication hinders the product delivery. Communication between the members of team reduces the internal documentation where internal documentation is no direct use to the customer [A17]. In large companies, people who are responsible for the work and who are accountable for the decision-making, can be difficult to identify across the organization if poor communication [A22]. Communication barriers with cultural issues can lead to project failure when they come in the middle of critical problems those raised in mission critical system development.
Extra features:

Figure 7: Extra features factors

Requirements Issues

Literature: Managing requirements is an important phase of requirements engineering and software development. It is a fundamental method of how a software would be at the end. If some of the requirements are missing then the function and performance of the software will not meet the customer needs [A2]. Missing requirements always cause delay of project and defected products [A33]. Incomplete requirements specification can introduce defects into systems source code without the knowledge of developer. This might cause by analysts being careless in identifying requirements and omitting certain needs due to ineffective communication with customers. Communication deficiencies with the customer and missing of the true customer requirements create uncompleted software [A9]. All efforts to create a software that does not support a customer need is a waste. Lack of requirements traceability and unstable requirements are some of the causes of defective product [A52]. Lack of responsiveness to the customer needs causes incompleteness. When faults are not detected in early stages and requirements are kept in big batches, errors are
hidden could result in uncompleted and defective product [A3, A6, A11]. Even changes to the requirements or late elicitation of requirements which are performed during the design, coding and test processes also cause delays to the project [A36]. Over designing a system to anticipate future requirements changes which will allow extra features to be added to the system results in a huge waste. This is because instead of designing a system that supports the immediate needs of customer, resources are devoted to designing and developing extra features into a product for future use which customer do not use in present time resulting in late product delivery [A19].

**Interview:** Misinterpretation is a challenging factor between customers and requirement engineers as it could lead to development of the software completely away from the customer’s expectations. Sometimes organization abandons the work in the middle of the project due to misinterpretation. Sometimes inconsistencies within the requirements unidentified before the start of the development put the development on hold until clarification is got from customer. This causes delays in the progress of the project or even abandon in middle of the project. Missing requirements or where sufficient requirements are not gathered in the early phase of software development life cycle can result in unfinished product. Poor analysis of the requirements can lead to inappropriate conclusions and it has negative impacts on the product development. Starting the project ahead of the completion of requirement description can result in the project being stopped half-way. Sometimes requirements contain defects because of failure to write test cases to them. Other times, some requirements are inconsistent or unstable which might not allow project run smoothly. All these different issues arise due poor requirements engineering. Taking and developing unnecessary or unwanted requirements wastes resources.

**Work environment Issues**

**Literature:** Employees' errors can be increased with unfamiliar and too complex work operating environment. The problem is increased if the operating environment is confusing and static. This can result in delayed or uncompleted works. Complexity can come with large modules that are inseparable and which are of different variants having only slight changes thereby creating confusion in implementation [A8]. Distributed work on complex environment with more people involved and more inter-dependencies of sites may create delays in development cycles. Having modules that are interdependent on each other but are to be implemented in distributed environment may create delays.
Testing Issues

Literature: Buildup of untested code is the most frequent problem of partial done work. This can be caused by the quality assurance team being busy with other tasks and unable to test the incoming task. This creates hug backlogs of tasks to test [A11]. Rapid feedback can be obtained through mechanisms like continuous testing of the system thereby reducing waste. Continuous testing helps to give frequent deliveries where customers speedily evaluates and approves the developed features [A21]. Testing delays can keep developers idle and waiting for the test results before starting to work on the next piece of code. This is because the developers might think that further coding is required to fix the defects that are identified in the testing process [A3]. To identify defects, only suitable techniques should be used in the test cases because every technique cannot be applicable in each case. The selection of proper approach for test should be based on the resources required for that approach, the resource availability, and features of the approach. Lack of all these leaves defect in the product unidentified, which is a big waste [A48]. Not providing enough resources to identify defects for the testing teams does not help. Testing teams should be equipped with both dynamic and static testing techniques. Environment should be created for testing which should allow smooth testing process [A53]. Changing user requirements always have the possibility to create new code or modify existing code. Significant changes to existing code or adding new code tends to lead defect-prone code [A65]. Proper care should be taken to prevent defects creeping in. Sound testing strategy should be used with mature tests that can avoid defects otherwise, wastes will lead to rework [A67]. If there is no coordination between development groups then it leads to design of poor, ineffective and inefficient tests which may not be able to detect defects.

Interview: Conducting or starting the testing process too late leaves the defects undetected in the final product. Early identification and fixing of defects can stop the defects from further spreading. It is easier to identify and fix defects when the system or code is smaller in size than huge in the later stages. Sometimes early stages are less likely to have because of the initial comprehensive test procedures. Testing process should be started as soon as requirements are finalized. Defects can occur by many ways either by people, process, or technological aspects. People aspects might be lack of experience and collaboration between members, less domain knowledge of system, expert’s unavailability, lack of knowledge of frequent updates on technologies, lack of mature
team members on testing process etc. *Process* aspects like lack of review of code, poor inspection etc. *Technology* aspects like high degree of dependency between tasks, inconsistent and unsupportive tools, outdated test infrastructure, insufficient testing resources. Testing ability of customer also plays an important role in defect identification. Late testing by customer may lead to dissatisfaction on the product.

**Human Technical factors**

**Literature:** Human technical skills are important in software projects. They always contribute to the projects in one way or the other. A highly skilled member can solve complex problems. Shortage of skills leads to delaying the work. A group should have variety of skills. If a new member joins a group the member should be trained. All the members in the team need to be trained in many different points of view of the project so as to work in the project effectively. Inadequate training can lead to project delay and defective product [A1]. Members of the team should be well trained and should be experts. Experts are always expected to take ultimate decisions which can gradually reduce delay and the handover wastes. Good knowledge of system technical aspects and domain aspects can always eliminate the uncompleted work and reduces the delay in completing the work [A26]. Lower capability of the member of the team (for example in programming, testing, analyzing etc.,) or lack of sufficient experience to solve problems always causes software projects delays. Adding new member who is inexperienced to an already delayed project always leads to further delays [A28]. Members of testing team being unable to judge the defects or being biased in judging a defect can cause the defect to be undetected and lead to delays [A46]. In the requirements, lack of peer-reviews of the requirements leads to defects. Revision of the requirements many times by peer reviewers can drastically bring down the defect count [A61]. Allowing team members to work on unfamiliar technologies or new technologies where experience is lacking can result in defective product for example due to missing components quality [A51]. Changing models too often during development with matching experiences can result in defective product [A52]. Uneven or unbalanced distribution of developers across locations produces defective products inexperience being the main cause [A62]. The capability of the member in detecting a defect reflects their experience on identifying and removing the severe defects in the product [A63].
Interview: Inexperience is a huge problem for a company. High turnover rate put the company in difficult situations. The capability of the company is dependent on the experience of its people. A huge implementation project that is unmatched by adequate high skilled person would be a failure. Shortage of skilled members may result in being unable to complete the work and increase time-to-market. However, good estimation of market can create high value for the customer. Utilization of tools is dependent on human skills but lack of adequate relevant skills always delay the process. Employees’ poor knowledge of the latest technologies can adversely affect the company’s growth rate. Sometimes, there may need to make sudden changes to the process of a finished project, adapting to these unexpected changes can be of benefit to the company. Employee’s ability to resolve problems by immediately resolving the work issues will always generate high value to the organization and eliminate waste or delay. Creating clear process and defining clear procedures in the work established by the employees can also create great benefits to the company, having those people in the company always achieves customer’s expectations.

Global market

Literature: Working with known requirements and implementing only what is needed by the customer in the present situation or what is currently required for the market (dynamic market), leads to high customer values and market satisfaction. Otherwise designing the system with high expectancy and not considering the customer value and market needs lead to overdesigning and extra features [A20]. Any extra work on the system or designing it to allow extra features with anticipated future requirements may often lead to overproduction. Prematurely overdesigning a system for future use can introduce unnecessary complexity in the code finally with more defects and delays of the entire product.

Documentation:

Literature: Documentation helps to save the company when there is high turnover rate. Documentation provides a great aid to a new or inexperienced member to understand a service that has complex issues. Defective software can be created when the requirements are documented unclearly [A3]. Lack of up-to-date documentation affects software maintenance. This will cause the developer to redo the documentation of the entire system which increases waste of extra process [A13]. Documentation is any artifact or the medium that communicate to a user of software
system [A13]. Many people have found that inadequate, excessive or missing documentation creates problem to all phases of software development life cycle [A14]. Documentation provides an effective and flexible way of recording informal content [A16]. Documentation is an aid for maintenance which helps in understanding high-level design, implementing, and functional nature of complex applications [A14]. Too much documentation can cause waste of extra process, less documentation increases defects and inventory [A7]. Documentation is an important tool for communication [A17].

**Interview:** Lack of documentation or consequence of no documentation- where instructions or guidelines are missing which are able to aid the developer on implementing a task is a cause of waste or partially completed work. Missing description on how a final state of the system should look like can stop the project half-way. Documentations are created to keep or act as an archive about different aspects of the system such as what are the functionality, technical knowledge and business domain of the system. Not having these documentations will lead to extra processes. Good documentations remove all the misconception regarding the software and keep track of the information regarding all activities. Lack of documentation increases handover waste because of new decisions needed for some problems.

**Feedback:**

**Literature:** Feedback can show problems and errors in the process or product. Feedback shows the impact achieved by the quality of the product and productivity of the organizations. Organizations learn from feedback and can continuously reduce the process and product variations from the customer and different stakeholders. Short and frequent feedback can give required and useful features delivered to the external or internal customers or internal customers [A6]. Continuous customer involvement mostly gives the correct products delivery, otherwise lack of feedback increases inventory. Continuous feedback gives continuous testing of the system which can make it free from defects. Frequent deliveries enables continuous feedback from the customers where customers evaluate and approve the system through this, extra features can be eliminated [A19]. Long waiting times between feedbacks can point out many defects in the system which cause rework [A39]. Strong impact of the feedback is that it can decide the duration of the software development process.
Interview: Feedback from different stockholders always contributes to give a perfect system. Feedback can point out defects in the product. Quick feedback reduce delays as well as helps in early completion of project. Feedback removed extra feature from the product and help to proceed until completion of software development project without abandoning in the middle.

Customer issues

Literature: Responsiveness to the customer needs is important to the software development. The value he had received, the delivery received and the time it was received for his needs are all important to the customer. Some practices like on-site visits and frequent deliveries are important to the customer [A4]. Lack of responsiveness to the customer with respect to timely delivery increases inventory. Customers always want to know whether sufficient quality of the product is delivered or not. Lack of responsiveness to this quality aspect creates defective product which leads to rework [A11].

Interview: Customer involvement in phases like requirements and testing can gives great benefit to final software in terms of how the product looks like and how it functions. Customer involvement give company good advantage it remove confusions in the work and give exact system that is required. When the customer involve by delivering feedback and testing each components the product comes without delays, and free from defects. Customer can cope with delays or defects when they were involved from start to end of the project so the company don’t lose business. Isolation of customers from the product development leads to different wastes of LSD.

Communication:

Literature: Minimizing communication gaps from the starting to the end of the project life cycle gives the product that is likely to reach customer expectations. Lack of good communication can lead to partially done work [A5]. Good communication should start from requirements phase itself because it can impact on the final software product. Good communication between customers and requirements engineers bring about stable requirements. Good communication between requirements engineers, developers and testers gives a non-defective product [A5]. Communication breakdown is one of the root causes of all wastes in LSD. Communication either through formal or informal channel is necessary to the software development. Direct
communication reduces documentation process but lack of communication hinders the product delivery. Communication between the members of team reduces the internal documentation where internal documentation is no direct use to the customer [A17]. In large companies, people who are responsible for the work and who are accountable for the decision-making, can be difficult to identify across the organization if poor communication [A22]. Communication barriers with cultural issues can lead to project failure when they come in the middle of critical problems those raised in mission critical system development.

Task Switching:

![Figure 8: Task switching factors](image)

Work environment Issues

**Literature:** Employees' errors can be increased with unfamiliar and too complex work operating environment. The problem is increased if the operating environment is confusing and static. This can result in delayed or uncompleted works. Complexity can come with large modules that are inseparable and which are of different variants having only slight changes thereby creating confusion in implementation [A8]. Distributed work on complex environment with more people involved and more inter-dependencies of sites may create delays in development cycles. Having
modules that are interdependent on each other but are to be implemented in distributed environment may create delays.

**Human Technical factors**

**Literature**: Human technical skills are important in software projects. They always contribute to the projects in one way or the other. A highly skilled member can solve complex problems. Shortage of skills leads to delaying the work. A group should have variety of skills. If a new member joins a group the member should be trained. All the members in the team need to be trained in many different points of view of the project so as to work in the project effectively. Inadequate training can lead to project delay and defective product [A1]. Members of the team should be well trained and should be experts. Experts are always expected to take ultimate decisions which can gradually reduce delay and the handover wastes. Good knowledge of system technical aspects and domain aspects can always eliminate the uncompleted work and reduces the delay in completing the work [A26]. Lower capability of the member of the team (for example in programming, testing, analyzing etc.,) or lack of sufficient experience to solve problems always causes software projects delays. Adding new member who is inexperienced to an already delayed project always leads to further delays [A28]. Members of testing team being unable to judge the defects or being biased in judging a defect can cause the defect to be undetected and lead to delays [A46]. In the requirements, lack of peer-reviews of the requirements leads to defects. Revision of the requirements many times by peer reviewers can drastically bring down the defect count [A61]. Allowing team members to work on unfamiliar technologies or new technologies where experience is lacking can result in defective product for example due to missing components quality [A51]. Changing models too often during development with matching experiences can result in defective product [A52]. Uneven or unbalanced distribution of developers across locations produces defective products inexperience being the main cause [A62]. The capability of the member in detecting a defect reflects their experience on identifying and removing the severe defects in the product [A63].

**Interview**: Inexperience is a huge problem for a company. High turnover rate put the company in difficult situations. The capability of the company is dependent on the experience of its people. A huge implementation project that is unmatched by adequate high skilled person would be a failure. Shortage of skilled members may result in being unable to complete the work and increase time-
to-market. However, good estimation of market can create high value for the customer. Utilization of tools is dependent on human skills but lack of adequate relevant skills always delay the process. Employees’ poor knowledge of the latest technologies can adversely affect the company’s growth rate. Sometimes, there may need to make sudden changes to the process of a finished project, adapting to these unexpected changes can be of benefit to the company. Employee’s ability to resolve problems by immediately resolving the work issues will always generate high value to the organization and eliminate waste or delay. Creating clear process and defining clear procedures in the work established by the employees can also create great benefits to the company, having those people in the company always achieves customer’s expectations.

**Human social factors:**

**Literature:** Responding to other colleague is important when the project work is distributed among team members. There are many barriers which could transpire among colleagues such as cultural, language or gender. Overcoming barriers can create smooth flow of work. Lack of responsiveness in helping of the remote colleague creates waste of delay [A24]. In some situation hiding information or assuming it is sensitive information from the remote colleague leads to defect system. A sociological aspect of the team also can create delays to the project [A40]. Sometimes distribution of work will be uneven to some team members; sharing the work overload by other members can deliver quick outcome [A24]. Not all team members will have the same knowledge on the technical and functional aspects of the system. Some members of the team who is experienced can share the experience to the other members. Some members shows displeasure on other colleagues due to lack of trust towards them, to them physical signatures are needed for approvals or decisions made most of time this will increase handovers or even delays [A3].

**Interview:** People hiding information to the remote colleague is caused by lack of trust or can be due to cultural and language difference. Lack of trust or stop in the sharing of knowledge to the remote person causes delays to the project. Colleagues Response time increases if trust decreases. Due to trust, understanding between two colleagues will always differ. Good understanding between members will occur only if they have high levels of trust on one and another which will gradually reflects on the wastes. Interaction barriers such education, designation, gender plays a huge role on decision making which increases handovers and waiting times.
**Time:**

**Interview:** Software project managers take various steps to meet deadlines. The thought of losing market opportunity can lead to less time, pressure to complete the project to meet market. These create many wastes in software development. Project which was delayed consumes more resources and generates most of the wastes of LSD. When time to meet deadline decreases on project, defects start occurring, handovers start increasing, people switch from one task to another by leaving other work as partially done work, documentation starts increasing and at the end loss of value for the customer.

**Coordination:**

**Literature:** When software development projects are globally distributed teams should coordinate with each other in successful implementation of the project. This coordination can be done optimally by splitting work across sites, increasing communication, finding experts, awareness [A33]. Splitting of work between teams, optimally locating the experts before the start of the project, maintaining frequent communication and generating awareness of other member status creates good coordination. Coordination can influence LSD by reduction in defects of final product, meeting deadlines without delays, less handovers, people helping each other by switching between tasks, eliminating extra feature, continuing until project completion.
Handover:

**Figure 9: Handovers factors**

**Geographical issues:**

**Literature:** In geographically separated sites projects where design and implementation takes place in different geographically separated sites. Direct contact of the members with other members is not possible frequently and also takes lot of effort and consumes many resources and increase in cost. The separation of design and implementation consumes and uses many mediums to communicate with each other. Documentation is one of the main medium of knowledge sharing processes [A17]. Geographical separation leads to delay in communication because of time difference. The response time is impacted by the distance between two sites. More number of sites collaboration makes communication complex which lead to delay in the project and takes many handovers for decisions and approvals. Project planning and control and effort estimation is more difficult if the teams were geographically separated [A34].

**Human social factors:**

**Literature:** Responding to other colleague is important when the project work is distributed among team members. There are many barriers which could transpire among colleagues such as cultural, language or gender. Overcoming barriers can create smooth flow of work. Lack of responsiveness in helping of the remote colleague creates waste of delay [A24]. In some situation hiding information or assuming it is sensitive information from the remote colleague leads to defect system. A sociological aspect of the team also can create delays to the project [A40]. Sometimes
distribution of work will be uneven to some team members; sharing the work overload by other members can deliver quick outcome [A24]. Not all team members will have the same knowledge on the technical and functional aspects of the system. Some members of the team who is experienced can share the experience to the other members. Some members shows displeasure on other colleagues due to lack of trust towards them, to them physical signatures are needed for approvals or decisions made most of time this will increase handovers or even delays [A3].

Interview: People hiding information to the remote colleague is caused by lack of trust or can be due to cultural and language difference. Lack of trust or stop in the sharing of knowledge to the remote person causes delays to the project. Colleagues Response time increases if trust decreases. Due to trust, understanding between two colleagues will always differ. Good understanding between members will occur only if they have high levels of trust on one and another which will gradually reflects on the wastes. Interaction barriers such education, designation, gender plays a huge role on decision making which increases handovers and waiting times.

**Organizational issues:**

**Literature:** Organizations are classified into two levels - hierarchical or flat organization levels. On each level people are assigned different roles (top level managers to bottom level people like coders, testers etc.). An organization size depends on the number of levels and departments it consists. Waiting for decisions are dependent on what level in the organization they should be made. Short waiting times occur always if the organization structure is flat than the hierarchical organizations where long waiting time takes place. Flat organizations deliver the software with less delay in contrast to hierarchical organization. Increase of levels in the organization leads to more handovers because of lack of frequent communication to higher management team [A21]. Traceability on any information is dependent on organization size. Traceability effort varies due to the organization structure. Lack of traceability increases the handover process.

**Time:**

**Interview:** Software project managers take various steps to meet deadlines. The thought of loosing market opportunity can lead to less time, pressure to complete the project to meet market. These create many wastes in software development. Project which was delayed consumes more resources and generates most of the wastes of LSD. When time to meet deadline decreases on project, defects
starts occurring, handovers starts increasing, people switch from one task to another by leaving other work as partially done work, documentation starts increasing and at the end loss of value for the customer.

**Communication:**

**Literature:** Minimizing communication gaps from the starting to the end of the project life cycle gives the product that is likely to reach customer expectations. Lack of good communication can lead to partially done work [A5]. Good communication should start from requirements phase itself because it can impact on the final software product. Good communication between customers and requirements engineers bring about stable requirements. Good communication between requirements engineers, developers and testers gives a non-defective product [A5]. Communication breakdown is one of the root causes of all wastes in LSD. Communication either through formal or informal channel is necessary to the software development. Direct communication reduces documentation process but lack of communication hinders the product delivery. Communication between the members of team reduces the internal documentation where internal documentation is no direct use to the customer [A17]. In large companies, people who are responsible for the work and who are accountable for the decision-making, can be difficult to identify across the organization if poor communication [A22]. Communication barriers with cultural issues can lead to project failure when they come in the middle of critical problems those raised in mission critical system development.

**Temporal:**

**Literature:** Due to time zone differences synchronized communication is difficult [A17]. Lack of communication will adversely affect the whole software development in many ways. Less overlap time between sites causes more waiting times for the response for this, more documentation is needed to cope with this. Time zone difference between collaborative sites has the potential to cause delays [A27]. Increase in time-zones creates increase in response time; hence, as the number of sites increases the longer will be the delays.

**Interview:** Temporal distances always generates waiting times and causes delays to the project. Difficult to find communication channel makes documentation as very important. Some interviewees have pointed out that some people are undesirable to communicate with the remote
colleague because of language, physical, cultural barriers. People are having misunderstanding between them when having conversations with temporal distance.

Delays:

![Delays factors diagram]

**Figure 10: Delays factors**

**Requirements Issues**

**Literature:** Managing requirements is an important phase of requirements engineering and software development. It is a fundamental method of how a software would be at the end. If some of the requirements are missing then the function and performance of the software will not meet the customer needs [A2]. Missing requirements always cause delay of project and defected products [A33]. Incomplete requirements specification can introduce defects into systems source code without the knowledge of developer. This might cause by analysts being careless in identifying requirements and omitting certain needs due to ineffective communication with customers. Communication deficiencies with the customer and missing of the true customer requirements
create uncompleted software [A9]. All efforts to create a software that does not support a customer need is a waste. Lack of requirements traceability and unstable requirements are some of the causes of defective product [A52]. Lack of responsiveness to the customer needs causes incompleteness. When faults are not detected in early stages and requirements are kept in big batches, errors are hidden could result in uncompleted and defective product [A3, A6, A11]. Even changes to the requirements or late elicitation of requirements which are performed during the design, coding and test processes also cause delays to the project [A36]. Over designing a system to anticipate future requirements changes which will allow extra features to be added to the system results in a huge waste. This is because instead of designing a system that supports the immediate needs of customer, resources are devoted to designing and developing extra features into a product for future use which customer do not use in present time resulting in late product delivery [A19].

Interview: Misinterpretation is a challenging factor between customers and requirement engineers as it could lead to development of the software completely away from the customer’s expectations. Sometimes organization abandons the work in the middle of the project due to misinterpretation. Sometimes inconsistencies within the requirements unidentified before the start of the development put the development on hold until clarification is got from customer. This causes delays in the progress of the project or even abandon in middle of the project. Missing requirements or where sufficient requirements are not gathered in the early phase of software development life cycle can result in unfinished product. Poor analysis of the requirements can lead to inappropriate conclusions and it has negative impacts on the product development. Starting the project ahead of the completion of requirement description can result in the project being stopped half-way. Sometimes requirements contain defects because of failure to write test cases to them. Other times, some requirements are inconsistent or unstable which might not allow project run smoothly. All these different issues arise due poor requirements engineering. Taking and developing unnecessary or unwanted requirements wastes resources.

Work environment Issues

Literature: Employees' errors can be increased with unfamiliar and too complex work operating environment. The problem is increased if the operating environment is confusing and static. This can result in delayed or uncompleted works. Complexity can come with large modules that are inseparable and which are of different variants having only slight changes thereby creating
confusion in implementation [A8]. Distributed work on complex environment with more people involved and more inter-dependencies of sites may create delays in development cycles. Having modules that are interdependent on each other but are to be implemented in distributed environment may create delays.

**Testing Issues**

**Literature:** Buildup of untested code is the most frequent problem of partial done work. This can be caused by the quality assurance team being busy with other tasks and unable to test the incoming task. This creates hug backlogs of tasks to test [A11]. Rapid feedback can be obtained through mechanisms like continuous testing of the system thereby reducing waste. Continuous testing helps to give frequent deliveries where customers speedily evaluates and approves the developed features [A21]. Testing delays can keep developers idle and waiting for the test results before starting to work on the next piece of code. This is because the developers might think that further coding is required to fix the defects that are identified in the testing process [A3]. To identify defects, only suitable techniques should be used in the test cases because every technique cannot be applicable in each case. The selection of proper approach for test should be based on the resources required for that approach, the resource availability, and features of the approach. Lack of all these leaves defect in the product unidentified, which is a big waste [A48]. Not providing enough resources to identify defects for the testing teams does not help. Testing teams should be equipped with both dynamic and static testing techniques. Environment should be created for testing which should allow smooth testing process [A53]. Changing user requirements always have the possibility to create new code or modify existing code. Significant changes to existing code or adding new code tends to lead defect-prone code [A65]. Proper care should be taken to prevent defects creeping in. Sound testing strategy should be used with mature tests that can avoid defects otherwise, wastes will lead to rework [A67]. If there is no coordination between development groups then it leads to design of poor, ineffective and inefficient tests which may not be able to detect defects.

**Interview:** Conducting or starting the testing process too late leaves the defects undetected in the final product. Early identification and fixing of defects can stop the defects from further spreading. It is easier to identify and fix defects when the system or code is smaller in size than huge in the later stages. Sometimes early stages are less likely to have because of the initial comprehensive
test procedures. Testing process should be started as soon as requirements are finalized. Defects can occur by many ways either by people, process, or technological aspects. People aspects might be lack of experience and collaboration between members, less domain knowledge of system, expert’s unavailability, lack of knowledge of frequent updates on technologies, lack of mature team members on testing process etc. Process aspects like lack of review of code, poor inspection etc. Technology aspects like high degree of dependency between tasks, inconsistent and unsupportive tools, outdated test infrastructure, insufficient testing resources. Testing ability of customer also plays an important role in defect identification. Late testing by customer may lead to dissatisfaction on the product.

**Technical management**

**Literature:** Duration of projects is dependent on many factors. To solve some of the factors, collocation of resources can be very helpful. Facilitating collocation of infrastructure to the group provide many benefits to deal projects durations [A26]. Even though adopting 'collocation' can be difficult for group members at first, members will gradually adjust to it. This can be done by providing and explaining the benefits to the projects of collocating teams including less defects, less delays and reduced extra processes or handovers. Team member with high levels of domain knowledge of the system and all technical aspects that are needed for system to develop can dramatically reduce the duration of the projects. Otherwise, lack of technical and system domain knowledge can always increase the projects duration or even end up leaving the projects as partial done work [A26]. Providing enough resources and sharing limited resources can be facilitated if the team members are collocated. Optimizing the resources is possible to the teams if they are collocated.

**Human Technical factors**

**Literature:** Human technical skills are important in software projects. They always contribute to the projects in one way or the other. A highly skilled member can solve complex problems. Shortage of skills leads to delaying the work. A group should have variety of skills. If a new member joins a group the member should be trained. All the members in the team need to be trained in many different points of view of the project so as to work in the project effectively. Inadequate training can lead to project delay and defective product [A1]. Members of the team
should be well trained and should be experts. Experts are always expected to take ultimate
decisions which can gradually reduce delay and the handover wastes. Good knowledge of system
technical aspects and domain aspects can always eliminate the uncompleted work and reduces the
delay in completing the work [A26]. Lower capability of the member of the team (for example in
programming, testing, analyzing etc..) or lack of sufficient experience to solve problems always
causes software projects delays. Adding new member who is inexperienced to an already delayed
project always leads to further delays [A28]. Members of testing team being unable to judge the
defects or being biased in judging a defect can cause the defect to be undetected and lead to delays
[A46]. In the requirements, lack of peer-reviews of the requirements leads to defects. Revision of
the requirements many times by peer reviewers can drastically bring down the defect count [A61].
Allowing team members to work on unfamiliar technologies or new technologies where
experience is lacking can result in defective product for example due to missing components
quality [A51]. Changing models too often during development with matching experiences can
result in defective product [A52]. Uneven or unbalanced distribution of developers across locations
produces defective products inexperience being the main cause [A62]. The capability of the
member in detecting a defect reflects their experience on identifying and removing the severe
defects in the product [A63].

Interview: Inexperience is a huge problem for a company. High turnover rate put the company in
difficult situations. The capability of the company is dependent on the experience of its people. A
huge implementation project that is unmatched by adequate high skilled person would be a failure.
Shortage of skilled members may result in being unable to complete the work and increase time-
to-market. However, good estimation of market can create high value for the customer. Utilization
of tools is dependent on human skills but lack of adequate relevant skills always delay the process.
Employees’ poor knowledge of the latest technologies can adversely affect the company’s growth
rate. Sometimes, there may need to make sudden changes to the process of a finished project,
adapting to these unexpected changes can be of benefit to the company. Employee’s ability to
resolve problems by immediately resolving the work issues will always generate high value to the
organization and eliminate waste or delay. Creating clear process and defining clear procedures in
the work established by the employees can also create great benefits to the company, having those
people in the company always achieves customer’s expectations.
**Resource management**

**Literature:** Resource estimation and planning are important before the commencement of a project. Resources can be anything that allows completion of the project on time. Lack of resources planning can lead to wastes in “lean software development”. If resources are not available project solutions cannot be framed. Lack of sufficient resources to the team members will always have a negative impact on the performance of the team. Proper estimation of resources required to implement a project and technical factors like features, versions will help to make software development process easy [A53].

**System complexity factors**

**Interview:** Increase in system complexity will make companies to compromise with some of the wastes occurring in LSD. Complexity can be either due to increase of system modules, coupling of system components or dependencies between different tasks. Sometimes increase in complexity can be viewed as increase of source code or functionality. Change to any module might affect the entire system which consumes many resources, time and cost. Also frequent changes can affect the quality of the system. This constitutes a great problem to the company.

**Global market**

**Literature:** Working with known requirements and implementing only what is needed by the customer in the present situation or what is currently required for the market (dynamic market), leads to high customer values and market satisfaction. Otherwise designing the system with high expectancy and not considering the customer value and market needs lead to overdesigning and extra features [A20]. Any extra work on the system or designing it to allow extra features with anticipated future requirements may often lead to overproduction. Prematurely overdesigning a system for future use can introduce unnecessary complexity in the code finally with more defects and delays of the entire product.

**Documentation:**

**Literature:** Documentation helps to save the company when there is high turnover rate. Documentation provides a great aid to a new or inexperienced member to understand a service that has complex issues. Defective software can be created when the requirements are documented
unclearly [A3]. Lack of up-to-date documentation affects software maintenance. This will cause the developer to redo the documentation of the entire system which increases waste of extra process [A13]. Documentation is any artifact or the medium that communicate to a user of software system [A13]. Many people have found that inadequate, excessive or missing documentation creates problem to all phases of software development life cycle [A14]. Documentation provides an effective and flexible way of recording informal content [A16]. Documentation is an aid for maintenance which helps in understanding high-level design, implementing, and functional nature of complex applications [A14]. Too much documentation can cause waste of extra process, less documentation increases defects and inventory [A7]. Documentation is an important tool for communication [A17].

**Interview:** Lack of documentation or consequence of no documentation- where instructions or guidelines are missing which are able to aid the developer on implementing a task is a cause of waste or partially completed work. Missing description on how a final state of the system should look like can stop the project half-way. Documentations are created to keep or act as an archive about different aspects of the system such as what are the functionality, technical knowledge and business domain of the system. Not having these documentations will lead to extra processes. Good documentations remove all the misconception regarding the software and keep track of the information regarding all activities. Lack of documentation increases handover waste because of new decisions needed for some problems.

**Experience:**

**Literature:** When an unforeseen change (situation) occurs and requires new information to deal with, this information cannot be searched anywhere. This type of problems can be solved through experience. Lack of this required experience leads to partial done work. Always an experienced staff should guide other staff and new ones. Insufficient experience is the main cause of software project delay [A28]. It is a rational that an experienced person would adjust to the changes or situations in quick time than an inexperienced persons. High levels of experience can significantly minimize the manpower for a project and accelerate the progress of a project. If a team member is has the required experience in analyzing product defect, then partially done work will be eliminated [A58], if a team member is an expert in defect identification, the product will be free from defects. Experience can greatly impact all wastes of LSD.
Interview: People are working on the same task again and again to fix some things in the task because of inexperience on the system. Sometimes experts should guide from the starting to the end or even people should be changed in the middle of the task because of lack of experience. Lack of experience utilizes more resources of the company. Quality of product is always high if more number of experienced people were involved in software product development. Most of the wastes of LSD can be completely minimized with experienced personnel.

Geographical issues:

Literature: In geographically separated sites projects where design and implementation takes place in different geographically separated sites. Direct contact of the members with other members is not possible frequently and also takes lot of effort and consumes many resources and increase in cost. The separation of design and implementation consumes and uses many mediums to communicate with each other. Documentation is one of the main medium of knowledge sharing processes [A17]. Geographical separation leads to delay in communication because of time difference. The response time is impacted by the distance between two sites. More number of sites collaboration makes communication complex which lead to delay in the project and takes many handovers for decisions and approvals. Project planning and control and effort estimation is more difficult if the teams were geographically separated [A34].

Human social factors:

Literature: Responding to other colleague is important when the project work is distributed among team members. There are many barriers which could transpire among colleagues such as cultural, language or gender. Overcoming barriers can create smooth flow of work. Lack of responsiveness in helping of the remote colleague creates waste of delay [A24]. In some situation hiding information or assuming it is sensitive information from the remote colleague leads to defect system. A sociological aspect of the team also can create delays to the project [A40]. Sometimes distribution of work will be uneven to some team members; sharing the work overload by other members can deliver quick outcome [A24]. Not all team members will have the same knowledge on the technical and functional aspects of the system. Some members of the team who is experienced can share the experience to the other members. Some members shows displeasure on
other colleagues due to lack of trust towards them, to them physical signatures are needed for approvals or decisions made most of time this will increase handovers or even delays [A3].

**Interview:** People hiding information to the remote colleague is caused by lack of trust or can be due to cultural and language difference. Lack of trust or stop in the sharing of knowledge to the remote person causes delays to the project. Colleagues Response time increases if trust decreases. Due to trust, understanding between two colleagues will always differ. Good understanding between members will occur only if they have high levels of trust on one and another which will gradually reflects on the wastes. Interaction barriers such education, designation, gender plays a huge role on decision making which increases handovers and waiting times.

**Feedback:**

**Literature:** Feedback can show problems and errors in the process or product. Feedback shows the impact achieved by the quality of the product and productivity of the organizations. Organizations learn from feedback and can continuously reduce the process and product variations from the customer and different stakeholders. Short and frequent feedback can give required and useful features delivered to the external or internal customers or internal customers [A6]. Continuous customer involvement mostly gives the correct products delivery, otherwise lack of feedback increases inventory. Continuous feedback gives continuous testing of the system which can make it free from defects. Frequent deliveries enables continuous feedback from the customers where customers evaluate and approve the system through this, extra features can be eliminated [A19]. Long waiting times between feedbacks can point out many defects in the system which cause rework [A39]. Strong impact of the feedback is that it can decide the duration of the software development process.

**Interview:** Feedback from different stockholders always contributes to give a perfect system. Feedback can point out defects in the product. Quick feedback reduce delays as well as helps in early completion of project. Feedback removed extra feature from the product and help to proceed until completion of software development project without abandoning in the middle.

**Organizational issues:**
Literature: Organizations are classified into two levels - hierarchical or flat organization levels. On each level people are assigned different roles (top level managers to bottom level people like coders, testers etc.). An organization size depends on the number of levels and departments it consists. Waiting for decisions are dependent on what level in the organization they should be made. Short waiting times occur always if the organization structure is flat than the hierarchical organizations where long waiting time takes place. Flat organizations deliver the software with less delay in contrast to hierarchical organization. Increase of levels in the organization leads to more handovers because of lack of frequent communication to higher management team [A21]. Traceability on any information is dependent on organization size. Traceability effort varies due to the organization structure. Lack of traceability increases the handover process.

Communication:

Literature: Minimizing communication gaps from the starting to the end of the project life cycle gives the product that is likely to reach customer expectations. Lack of good communication can lead to partially done work [A5]. Good communication should start from requirements phase itself because it can impact on the final software product. Good communication between customers and requirements engineers bring about stable requirements. Good communication between requirements engineers, developers and testers gives a non-defective product [A5]. Communication breakdown is one of the root causes of all wastes in LSD. Communication either through formal or informal channel is necessary to the software development. Direct communication reduces documentation process but lack of communication hinders the product delivery. Communication between the members of team reduces the internal documentation where internal documentation is no direct use to the customer [A17]. In large companies, people who are responsible for the work and who are accountable for the decision-making, can be difficult to identify across the organization if poor communication [A22]. Communication barriers with cultural issues can lead to project failure when they come in the middle of critical problems those raised in mission critical system development.

Temporal:

Literature: Due to time zone differences synchronized communication is difficult [A17]. Lack of communication will adversely affect the whole software development in many ways. Less overlap
time between sites causes more waiting times for the response for this, more documentation is needed to cope with this. Time zone difference between collaborative sites has the potential to cause delays [A27]. Increase in time-zones creates increase in response time; hence, as the number of sites increases the longer will be the delays.

**Interview:** Temporal distances always generates waiting times and causes delays to the project. Difficult to find communication channel makes documentation as very important. Some interviewees have pointed out that some people are undesirable to communicate with the remote colleague because of language, physical, cultural barriers. People are having misunderstanding between them when having conversations with temporal distance.

### Defects:

![Figure 11: Defects factors](image)

**Requirements Issues**

**Literature:** Managing requirements is an important phase of requirements engineering and software development. It is a fundamental method of how a software would be at the end. If some of the
requirements are missing then the function and performance of the software will not meet the customer needs [A2]. Missing requirements always cause delay of project and defected products [A33]. Incomplete requirements specification can introduce defects into systems source code without the knowledge of developer. This might cause by analysts being careless in identifying requirements and omitting certain needs due to ineffective communication with customers. Communication deficiencies with the customer and missing of the true customer requirements create uncompleted software [A9]. All efforts to create a software that does not support a customer need is a waste. Lack of requirements traceability and unstable requirements are some of the causes of defective product [A52]. Lack of responsiveness to the customer needs causes incompleteness. When faults are not detected in early stages and requirements are kept in big batches, errors are hidden could result in uncompleted and defective product [A3, A6, A11]. Even changes to the requirements or late elicitation of requirements which are performed during the design, coding and test processes also cause delays to the project [A36]. Over designing a system to anticipate future requirements changes which will allow extra features to be added to the system results in a huge waste. This is because instead of designing a system that supports the immediate needs of customer, resources are devoted to designing and developing extra features into a product for future use which customer do not use in present time resulting in late product delivery [A19].

Interview: Misinterpretation is a challenging factor between customers and requirement engineers as it could lead to development of the software completely away from the customer’s expectations. Sometimes organization abandons the work in the middle of the project due to misinterpretation. Sometimes inconsistencies within the requirements unidentified before the start of the development put the development on hold until clarification is got from customer. This causes delays in the progress of the project or even abandon in middle of the project. Missing requirements or where sufficient requirements are not gathered in the early phase of software development life cycle can result in unfinished product. Poor analysis of the requirements can lead to inappropriate conclusions and it has negative impacts on the product development. Starting the project ahead of the completion of requirement description can result in the project being stopped half-way. Sometimes requirements contain defects because of failure to write test cases to them. Other times, some requirements are inconsistent or unstable which might not allow project runs smoothly. All these different issues arise due poor requirements engineering. Taking and developing unnecessary or unwanted requirements wastes resources.
Work environment Issues

**Literature:** Employees' errors can be increased with unfamiliar and too complex work operating environment. The problem is increased if the operating environment is confusing and static. This can result in delayed or uncompleted works. Complexity can come with large modules that are inseparable and which are of different variants having only slight changes thereby creating confusion in implementation \[A8\]. Distributed work on complex environment with more people involved and more inter-dependencies of sites may create delays in development cycles. Having modules that are interdependent on each other but are to be implemented in distributed environment may create delays.

Testing Issues

**Literature:** Buildup of untested code is the most frequent problem of partial done work. This can be caused by the quality assurance team being busy with other tasks and unable to test the incoming task. This creates huge backlogs of tasks to test \[A11\]. Rapid feedback can be obtained through mechanisms like continuous testing of the system thereby reducing waste. Continuous testing helps to give frequent deliveries where customers speedily evaluates and approves the developed features \[A21\]. Testing delays can keep developers idle and waiting for the test results before starting to work on the next piece of code. This is because the developers might think that further coding is required to fix the defects that are identified in the testing process \[A3\]. To identify defects, only suitable techniques should be used in the test cases because every technique cannot be applicable in each case. The selection of proper approach for test should be based on the resources required for that approach, the resource availability, and features of the approach. Lack of all these leaves defect in the product unidentified, which is a big waste \[A48\]. Not providing enough resources to identify defects for the testing teams does not help. Testing teams should be equipped with both dynamic and static testing techniques. Environment should be created for testing which should allow smooth testing process \[A53\]. Changing user requirements always have the possibility to create new code or modify existing code. Significant changes to existing code or adding new code tends to lead defect-prone code \[A65\]. Proper care should be taken to prevent defects creeping in. Sound testing strategy should be used with mature tests that can avoid defects otherwise, wastes will lead to rework \[A67\]. If there is no coordination between development
groups then it leads to design of poor, ineffective and inefficient tests which may not be able to detect defects.

**Interview:** Conducting or starting the testing process too late leaves the defects undetected in the final product. Early identification and fixing of defects can stop the defects from further spreading. It is easier to identify and fix defects when the system or code is smaller in size than huge in the later stages. Sometimes early stages are less likely to have because of the initial comprehensive test procedures. Testing process should be started as soon as requirements are finalized. Defects can occur by many ways either by people, process, or technological aspects. *People* aspects might be lack of experience and collaboration between members, less domain knowledge of system, expert’s unavailability, lack of knowledge of frequent updates on technologies, lack of mature team members on testing process etc. *Process* aspects like lack of review of code, poor inspection etc. *Technology* aspects like high degree of dependency between tasks, inconsistent and unsupportive tools, outdated test infrastructure, insufficient testing resources. Testing ability of customer also plays an important role in defect identification. Late testing by customer may lead to dissatisfaction on the product.

**Technical management**

**Literature:** Duration of projects is dependent on many factors. To solve some of the factors, collocation of resources can be very helpful. Facilitating collocation of infrastructure to the group provide many benefits to deal projects durations [A26]. Even though adopting 'collocation' can be difficult for group members at first, members will gradually adjust to it. This can be done by providing and explaining the benefits to the projects of collocating teams including less defects, less delays and reduced extra processes or handovers. Team member with high levels of domain knowledge of the system and all technical aspects that are needed for system to develop can dramatically reduce the duration of the projects. Otherwise, lack of technical and system domain knowledge can always increase the projects duration or even end up leaving the projects as partial done work [A26]. Providing enough resources and sharing limited resources can be facilitated if the team members are collocated. Optimizing the resources is possible to the teams if they are collocated.

**Human Technical factors**
**Literature:** Human technical skills are important in software projects. They always contribute to the projects in one way or the other. A highly skilled member can solve complex problems. Shortage of skills leads to delaying the work. A group should have variety of skills. If a new member joins a group the member should be trained. All the members in the team need to be trained in many different points of view of the project so as to work in the project effectively. Inadequate training can lead to project delay and defective product [A1]. Members of the team should be well trained and should be experts. Experts are always expected to take ultimate decisions which can gradually reduce delay and the handover wastes. Good knowledge of system technical aspects and domain aspects can always eliminate the uncompleted work and reduces the delay in completing the work [A26]. Lower capability of the member of the team (for example in programming, testing, analyzing etc.,) or lack of sufficient experience to solve problems always causes software projects delays. Adding new member who is inexperienced to an already delayed project always leads to further delays [A28]. Members of testing team being unable to judge the defects or being biased in judging a defect can cause the defect to be undetected and lead to delays [A46]. In the requirements, lack of peer-reviews of the requirements leads to defects. Revision of the requirements many times by peer reviewers can drastically bring down the defect count [A61]. Allowing team members to work on unfamiliar technologies or new technologies where experience is lacking can result in defective product for example due to missing components quality [A51]. Changing models too often during development with matching experiences can result in defective product [A52]. Uneven or unbalanced distribution of developers across locations produces defective products inexperience being the main cause [A62]. The capability of the member in detecting a defect reflects their experience on identifying and removing the severe defects in the product [A63].

**Interview:** Inexperience is a huge problem for a company. High turnover rate put the company in difficult situations. The capability of the company is dependent on the experience of its people. A huge implementation project that is unmatched by adequate high skilled person would be a failure. Shortage of skilled members may result in being unable to complete the work and increase time-to-market. However, good estimation of market can create high value for the customer. Utilization of tools is dependent on human skills but lack of adequate relevant skills always delay the process. Employees’ poor knowledge of the latest technologies can adversely affect the company’s growth rate. Sometimes, there may need to make sudden changes to the process of a finished project,
adapting to these unexpected changes can be of benefit to the company. Employee’s ability to resolve problems by immediately resolving the work issues will always generate high value to the organization and eliminate waste or delay. Creating clear process and defining clear procedures in the work established by the employees can also create great benefits to the company, having those people in the company always achieves customer’s expectations.

**Resource management**

**Literature:** Resource estimation and planning are important before the commencement of a project. Resources can be anything that allows completion of the project on time. Lack of resources planning can lead to wastes in “lean software development”. If resources are not available project solutions cannot be framed. Lack of sufficient resources to the team members will always have a negative impact on the performance of the team. Proper estimation of resources required to implement a project and technical factors like features, versions will help to make software development process easy [A53].

**System complexity factors**

**Interview:** Increase in system complexity will make companies to compromise with some of the wastes occurring in LSD. Complexity can be either due to increase of system modules, coupling of system components or dependencies between different tasks. Sometimes increase in complexity can be viewed as increase of source code or functionality. Change to any module might affect the entire system which consumes many resources, time and cost. Also frequent changes can affect the quality of the system. This constitutes a great problem to the company.

**Documentation:**

**Literature:** Documentation helps to save the company when there is high turnover rate. Documentation provides a great aid to a new or inexperienced member to understand a service that has complex issues. Defective software can be created when the requirements are documented unclearly [A3]. Lack of up-to-date documentation affects software maintenance. This will cause the developer to redo the documentation of the entire system which increases waste of extra process [A13]. Documentation is any artifact or the medium that communicate to a user of software system [A13]. Many people have found that inadequate, excessive or missing documentation
creates problem to all phases of software development life cycle [A14]. Documentation provides an effective and flexible way of recording informal content [A16]. Documentation is an aid for maintenance which helps in understanding high-level design, implementing, and functional nature of complex applications [A14]. Too much documentation can cause waste of extra process, less documentation increases defects and inventory [A7]. Documentation is an important tool for communication [A17].

**Interview:** Lack of documentation or consequence of no documentation - where instructions or guidelines are missing which are able to aid the developer on implementing a task is a cause of waste or partially completed work. Missing description on how a final state of the system should look like can stop the project half-way. Documentations are created to keep or act as an archive about different aspects of the system such as what are the functionality, technical knowledge and business domain of the system. Not having these documentations will lead to extra processes. Good documentations remove all the misconception regarding the software and keep track of the information regarding all activities. Lack of documentation increases handover waste because of new decisions needed for some problems.

**Experience:**

**Literature:** When an unforeseen change (situation) occurs and requires new information to deal with, this information cannot be searched anywhere. This type of problems can be solved through experience. Lack of this required experience leads to partial done work. Always an experienced staff should guide other staff and new ones. Insufficient experience is the main cause of software project delay [A28]. It is a rational that an experienced person would adjust to the changes or situations in quick time than an inexperienced persons. High levels of experience can significantly minimize the manpower for a project and accelerate the progress of a project. If a team member is has the required experience in analyzing product defect, then partially done work will be eliminated [A58], if a team member is an expert in defect identification, the product will be free from defects. Experience can greatly impact all wastes of LSD.

**Interview:** People are working on the same task again and again to fix some things in the task because of inexperience on the system. Sometimes experts should guide from the starting to the end or even people should be changed in the middle of the task because of lack of experience.
Lack of experience utilizes more resources of the company. Quality of product is always high if more number of experienced people were involved in software product development. Most of the wastes of LSD can be completely minimized with experienced personnel.

**Geographical issues:**

**Literature:** In geographically separated sites projects where design and implementation takes place in different geographically separated sites. Direct contact of the members with other members is not possible frequently and also takes lot of effort and consumes many resources and increase in cost. The separation of design and implementation consumes and uses many mediums to communicate with each other. Documentation is one of the main medium of knowledge sharing processes [A17]. Geographical separation leads to delay in communication because of time difference. The response time is impacted by the distance between two sites. More number of sites collaboration makes communication complex which lead to delay in the project and takes many handovers for decisions and approvals. Project planning and control and effort estimation is more difficult if the teams were geographically separated [A34].

**Human social factors:**

**Literature:** Responding to other colleague is important when the project work is distributed among team members. There are many barriers which could transpire among colleagues such as cultural, language or gender. Overcoming barriers can create smooth flow of work. Lack of responsiveness in helping of the remote colleague creates waste of delay [A24]. In some situation hiding information or assuming it is sensitive information from the remote colleague leads to defect system. A sociological aspect of the team also can create delays to the project [A40]. Sometimes distribution of work will be uneven to some team members; sharing the work overload by other members can deliver quick outcome [A24]. Not all team members will have the same knowledge on the technical and functional aspects of the system. Some members of the team who is experienced can share the experience to the other members. Some members shows displeasure on other colleagues due to lack of trust towards them, to them physical signatures are needed for approvals or decisions made most of time this will increase handovers or even delays [A3].

**Interview:** People hiding information to the remote colleague is caused by lack of trust or can be due to cultural and language difference. Lack of trust or stop in the sharing of knowledge to the
remote person causes delays to the project. Colleagues Response time increases if trust decreases. Due to trust, understanding between two colleagues will always differ. Good understanding between members will occur only if they have high levels of trust on one and another which will gradually reflects on the wastes. Interaction barriers such education, designation, gender plays a huge role on decision making which increases handovers and waiting times.

**Feedback:**

**Literature:** Feedback can show problems and errors in the process or product. Feedback shows the impact achieved by the quality of the product and productivity of the organizations. Organizations learn from feedback and can continuously reduce the process and product variations from the customer and different stakeholders. Short and frequent feedback can give required and useful features delivered to the external or internal customers or internal customers [A6]. Continuous customer involvement mostly gives the correct products delivery, otherwise lack of feedback increases inventory. Continuous feedback gives continuous testing of the system which can make it free from defects. Frequent deliveries enables continuous feedback from the customers where customers evaluate and approve the system through this, extra features can be eliminated [A19]. Long waiting times between feedbacks can point out many defects in the system which cause rework [A39]. Strong impact of the feedback is that it can decide the duration of the software development process.

**Interview:** Feedback from different stockholders always contributes to give a perfect system. Feedback can point out defects in the product. Quick feedback reduce delays as well as helps in early completion of project. Feedback removed extra feature from the product and help to proceed until completion of software development project without abandoning in the middle.

**Time:**

**Interview:** Software project managers take various steps to meet deadlines. The thought of loosing market opportunity can lead to less time, pressure to complete the project to meet market. These create many wastes in software development. Project which was delayed consumes more resources and generates most of the wastes of LSD. When time to meet deadline decreases on project, defects
starts occurring, handovers starts increasing, people switch from one task to another by leaving other work as partially done work, documentation starts increasing and at the end loss of value for the customer.
Chapter 5 Discussion:

5.1 Comparative analysis:

The researcher gathered the factors and relationships of wastes from what the academicians have reported in literature and what the practitioners said through interviews. In order to figure out to what extent either of them know about the causes of wastes a qualitative comparative analysis (QCA) was conducted [54]. In QCA analysis method where human analytics are used to analyze instead of completely relying on computer applications for comparing different attributes [55] [54]. The comparison analysis was performed between the findings from literature and case study. To draw meaningful conclusion from the comparison process the comparison is done between open codes of the two. The table

<table>
<thead>
<tr>
<th>Factors</th>
<th>Number of factors found in Literature</th>
<th>Number of factors found in Interview</th>
<th>Common factors from literature and interview</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-Technical</td>
<td>26</td>
<td>18</td>
<td>3</td>
</tr>
<tr>
<td>Technical</td>
<td>32</td>
<td>21</td>
<td>6</td>
</tr>
<tr>
<td>Global software product develop</td>
<td>10</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

5.1.1 Implications to Researchers:

For researchers, this study offers the opportunity to find and solve many problems. Many studies can be formed from this study where each study can take different directions.

For example, documentation. It is not understood how much documentation is sufficient or required, what would be unnecessary documentation that needs to be reduced is not known. Few
practitioners agree and some literature points out the importance of documentation while some literature contradicts this point and relates documenting as a waste of time.

Task switching in practice, people are having different opinions about moving between tasks. Practitioners are more opted to switch tasks in order to deal with different situations that arise while working and the benefits that are realized when working. But the literature contradicts this and says that task switching is a big waste in LSD. The advantages and disadvantages of task switching are still not fully studied especially in software development to provide suggestions to the practitioners.

Geographically separated sites with temporal distance which are in collaboration on a project for achieving benefits like cost saving and utilizing man power have many challenges. These challenges largely affect benefits of LSD which suggests to have more close cooperation between individuals in the project. This is another typical area which needs further studies in lean perspective.

Another factor is providing extra features (gold plating) to the customers. In LSD literature, there is not enough evidence (studies) to support how much the customer benefits or the advantages and disadvantages a company achieves by providing additional features, this is not known. So in all these areas, a number of investigations are required which need different research methods to solve problems and identify additional root causes of each wastes in LSD.

5.1.2 Implications to practitioners:

Practitioners may use this study as the base to reduce the wastes by eliminating or mitigating the effect of root causes. These root causes can also be viewed as technical, non-technical, and global software development point of view. With this one can take particular care in reducing the wastes of their own choice. Practitioners can reduce the wastes also by reducing the effect of common causes that are creating relationships between wastes. There are many causes related to wastes like defect and delays which are identified from the industry that are discussed in the literature, but for the other wastes the evidence is less this is because of lack of realization in the industry. The root causes that are identified in this study are common to software development methodologies and LSD, so there is a need to look beyond existing studies in LSD. Practitioners looking for solutions to eliminating wastes, there are many solutions provided for dealing different root cause of the
wastes in software development from this practitioners can adopt these solution and implement in LSD.

Furthermore, practitioners may use this study as a checklist to conduct their risk assessment while eliminating wastes in LSD.

5.2 Synthesis

The synthesis draws together the obtained results in order to provide answers to the research questions formulated in this thesis.

RQ1. What factors are influencing the wastes defined in lean software development?

RQ2. What are the relations between a) factors and wastes? b) factors? c) wastes?

In the in-depth investigation conducted through the state-of-art and state-of-practice it was shown that lean software development have many benefits which has been evolving over years with significant changes to the company. In order to experience full potential benefits from the lean software development waste removal activities plays a significant portion. Most of benefits can be achieved by taking care in eliminating root causes related to technical, non-technical, and global software product development issues. Technical factors with respective to the various phases of software development life cycle is well covered in this work were mostly affecting the software development. There are also relatively more non-technical factors which deals with the process improvement and people management are covered in this work that can damage business in the software development companies. The factors related to global software product development where communication cooperation and temporal issues which effects the entire software development where identified in this work. Within these factors there are many common factors that are effecting more number of wastes. These factors are much related to each other which are causing many relations between different kinds of wastes in lean software development.

5.3 Conclusion

The main contribution of this study lies in helping software development companies to eliminate root causes of wastes in lean software development. The study shows how wastes are related to each other. Various root causes of wastes are elicited to partially completed work, extra processes
(Relearning), extra processes, task switching (Handoffs, motion), delays (waiting) and defects. Factors and relationships of wastes were identified from both state of art and state of practice. Literature review helped not only in identifying root causes but also in refining of questionnaires related to interviews that are related for finding causes to wastes elimination strategies. The result show various social and technical factors that causes a wastes are effecting whole production in lean software development. Lean software development is a relatively unexplored phenomenon which needs lot of work to explore the elimination of waste. The proposed relationships were elicited through literature review, interviews with industrial practitioners and lean consultants. The validity threats were minimized very much to the proposed relationships because they were obtained from success full practitioners. The report helps, software development organizations from small and medium industries to large organizations, particularly in waste identification and elimination, when they developing software products in general or implementing LSD in particular.

5.4 Future work

- One area of improvement can be a study can be conducted by performing quality assessment on each and every individual factor obtained through literature this can suggest mitigation strategies to those factors.

- Second area of improvement can be a study can be conducted on removing waste on a single phase of software development life cycle instead of more phases or entire cycle, which can help where this study might have lag in identifying some of the in depth factors in some phase.

- Another area can be all the factors and causes that are identified in this study is from industrial point of view with value perceived by the company but not the value perceived by the customer. A future study can include the customers into the study to find more factors from the customer perspective also.
References


APPENDIX 1:

Consent Form which is used in this thesis:

Kindly read and sign following form at the start of interview.

- Participation is voluntary to this study.
- Participants details can be kept confidential or will not mention any were in the study or will not reveal to others (if insists).
- Interviewee has right to answer or skip any question while in the interview, and even cancel whole interview data at any time before dissertation of the thesis.
- Direct audio recordings, or extensive notes will not be published anywhere only transcripts will be used as a data in the study.
- All the answers will be used as anonymous so that the answers will not affect in any way to the interviewee.
- Data gathered from the interview is only used by interviewer and supervisor.
- The interview data will be used only by interviewer for this thesis and it may be published in different platforms with the support of supervisor.

Signature:
Place:
Date:

Thankyou

Prasadbabu M
prme10@student.bth.se
APPENDIX 2:

Table 11: Different Search Strings

<table>
<thead>
<tr>
<th>IEEE</th>
</tr>
</thead>
</table>
| ((waste OR "Partially Done Work" OR "Extra Features" OR "Extra Processes" OR Handover OR Motion OR "Task Switch" OR Delay OR Defect) AND ("software engineering" OR "software development" OR "software product" OR "software project" OR "software process" OR "software building" OR "software improvement") AND (empirical OR "literature review" OR survey OR "case study" OR experiment OR "action research" OR "systematic review" OR "comparative study" OR "industrial experience report") AND (rca OR cause OR factor OR relation OR influence)) OR ("lean practice" OR "lean" OR "value stream" OR "kanban") AND ("software engineering" OR "software development" OR "software project" OR "software process" OR "software building" OR "software improvement") | OR ("lean practice" OR $lean OR "value stream" OR $kanban) AND ("software engineering" OR "software development" OR "software product" OR "software project" OR "software process" OR "software building" OR "software improvement") AND (empirical OR "literature review" OR survey OR "case study" OR experiment OR "action research" OR "systematic review" OR "comparative study" OR "industrial experience report") |}

<table>
<thead>
<tr>
<th>Engineering Village (Compendex &amp; Inspec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Limited to till 2013, English</td>
</tr>
<tr>
<td>(waste OR &quot;Partially Done Work&quot; OR &quot;Extra Features&quot; OR &quot;Extra Processes&quot; OR Handover OR Motion OR &quot;Task Switch&quot; OR Delay OR Defect) AND (&quot;software engineering&quot; OR &quot;software development&quot; OR &quot;software project&quot; OR &quot;software process&quot; OR &quot;software building&quot; OR &quot;software improvement&quot;) AND (empirical OR &quot;literature review&quot; OR survey OR &quot;case study&quot; OR experiment OR &quot;action research&quot; OR &quot;systematic review&quot; OR &quot;comparative study&quot; OR &quot;industrial experience report&quot;) AND (English WN LA) AND (1969-2013 WN YR))</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Wiley</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Limited to till 2014 and journals, searched on full text.</td>
</tr>
<tr>
<td>(waste OR &quot;incomplete development&quot; OR &quot;Partially Done Work&quot; OR &quot;unwanted features&quot; OR &quot;additional functions&quot; OR &quot;Extra Features&quot; OR &quot;Extra Processes&quot; OR Handover OR Motion OR &quot;Task Switch&quot; OR Delay OR Defect) in FullText AND (empirical OR &quot;literature review&quot; OR survey OR &quot;case study&quot; OR experiment OR &quot;action research&quot; OR &quot;systematic review&quot; OR &quot;comparative study&quot; OR &quot;industrial experience report&quot;) in FullText AND (&quot;software engineering&quot; OR &quot;software development&quot; OR &quot;software product&quot; OR &quot;software process&quot; OR &quot;software building&quot; OR &quot;software improvement&quot;)</td>
</tr>
</tbody>
</table>
2. Limited to 1990-2014 and journals.

("lean practice" OR "lean" OR "lean principles" OR "lean thinking" OR "lean enablers" OR "lean production" OR "Lean Methodologies" in Abstract AND ("software engineering" OR "software development" OR "software product" OR "software project" OR "software process" OR "software building" OR "software improvement") in All Fields

ACM

1. (wastes and lean and software)

2. ("software engineering" or "software development" or "software product") and (documentation or handover)

3. ("defect") and ("software development" or "software engineering" or "software product")

4. ("delay" or "speed") and ("software development" or "software engineering" or "software product")

5. ("lean practice" OR "lean" OR "lean principles" OR "lean thinking" OR "lean enablers" OR "lean production" OR "Lean Methodologies" OR "lean software")

6. (((waste OR "Partially Done Work" OR "Extra Features" OR "Extra Processes" OR Handover OR Motion OR "Task Switch" OR Delay OR Defect) AND ("software engineering" OR "software development" OR "software product" OR "software project" OR "software process" OR "software building" OR "software improvement") AND (empirical OR "literature review" OR survey OR "case study" OR experiment OR "action research" OR "systematic review" OR "comparative study" OR "industrial experience report") AND (rca OR cause OR factor OR relation OR influence))

7. ("lean practice" OR "lean" OR "value stream" OR "kanban") AND ("software engineering" OR "software development" OR "software product" OR "software project" OR "software process" OR "software building" OR "software improvement") AND (empirical OR "literature review" OR survey OR "case study" OR experiment OR "action research" OR "systematic review" OR "comparative study" OR "industrial experience report")

SpringerLink

Limited to Articles, Software engineering, Computer Science, 1994-2014 and language as English.

"Lean software" OR "Lean practices" OR "Lean principle*"
### APPENDIX 3:

Table 12: Literature review


[A38] S. Yamada and J. Yamakawa, "An empirical study on statistical analysis based on software process monitoring data with initial project risks," in *17th ISSAT International Conference on*


APPENDIX 4:

Interview Protocol

A. Introduction

Goal of the study:
1. To find the root causes of each waste (Partial done work, Extra processes, Extra features, Handover, Task switching, Delay, Defect) of lean software development practiced in industry from the experts.
2. To find the common causes that can create relationships between different wastes of lean software development from the experts who are working in industry.

B. Warm-up and Experience

1. What are your educational background and how many years of experience in software industry?
2. How long have you been working on lean software development?
3. What is your role in the projects or organization when you are working on lean software development?
4. Do you think that your organization benefited from the lean software development in projects or is it difficult to adopt?
5. What is your team size when you are working with lean?
6. On what projects application domain does your team worked using lean way?
7. If the development is global what are the locations?

C. Main body of the interview

1. What reasons or challenges that are affecting to cause a waste in the lean software development (either global or local development)? Explain why it is a reason?
   A waste is a: - Partial done work (Inventory), Extra processes (Relearning), Extra features, Handover, Task switching (Handoffs/ Motion), Delay (Waiting), Defect.
2. What be the common causes that affecting different wastes in lean software development?
3. What do you think are the important reason(s) which are giving hard time and those should be concentrated most?

D. Closing

Anything that you want to explain or you think that is benefiting to uncover or eliminate waste in lean software development. Do you know anyone else in this field who would be helpful for me to interview?
### APPENDIX 5:

Table 13: Data Extraction Strategy

<table>
<thead>
<tr>
<th>Fundamental information</th>
<th>Targeted Value</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Temp Citation Key</td>
<td>A1,A2….An.</td>
<td>Overview of Literature</td>
</tr>
<tr>
<td>Article title</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Author’s Name</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Year</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Journal/Conference/Conference proceedings</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Specific Information**

<table>
<thead>
<tr>
<th>Study context</th>
<th>Academia</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Industry</td>
</tr>
<tr>
<td></td>
<td>Both Academia &amp; Industry</td>
</tr>
<tr>
<td>Research and data collection</td>
<td>Literature review</td>
</tr>
<tr>
<td></td>
<td>Systematic review</td>
</tr>
<tr>
<td></td>
<td>Case study</td>
</tr>
<tr>
<td></td>
<td>Experiment</td>
</tr>
<tr>
<td></td>
<td>Survey</td>
</tr>
<tr>
<td></td>
<td>Action research</td>
</tr>
<tr>
<td></td>
<td>Industrial report</td>
</tr>
<tr>
<td></td>
<td>Comparative study</td>
</tr>
<tr>
<td></td>
<td>Other</td>
</tr>
<tr>
<td>Study Domain</td>
<td>Software Engineering</td>
</tr>
<tr>
<td></td>
<td>Computer Science</td>
</tr>
<tr>
<td></td>
<td>Other Domain (Mechanical, Construction…)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Study subjects</th>
<th>Professional</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Researcher</td>
</tr>
<tr>
<td>Student</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td></td>
</tr>
<tr>
<td>Final Decision</td>
<td>0 = Exclude</td>
</tr>
<tr>
<td></td>
<td>1 = Primary Study</td>
</tr>
<tr>
<td></td>
<td>2 = Others (Duplicate, not found etc.)</td>
</tr>
</tbody>
</table>

**Wastes**

**Factors/Root causes**

<table>
<thead>
<tr>
<th>Waste-1</th>
<th>Factor 1...2...3...n</th>
<th>Answer to Research question(s).</th>
</tr>
</thead>
<tbody>
<tr>
<td>Waste-2</td>
<td>Factors 1...2...3...n</td>
<td></td>
</tr>
<tr>
<td>.......</td>
<td>........</td>
<td></td>
</tr>
<tr>
<td>Waste-7</td>
<td>Factors 1...2...3...n</td>
<td></td>
</tr>
</tbody>
</table>

**Relationships**

<table>
<thead>
<tr>
<th>Wastes</th>
<th>Answer to Research question 2.</th>
</tr>
</thead>
</table>
APPENDIX 6:

Interview Study

In this appendix description of the interviewees, the questions asked during the interview are presented

a) Description of Interviewees

Below is the brief description of the 7 employees following with their background to increase the understanding of what they have based their answers on the interview questions.

Matthias

Interview1

Date of interview: 19-08-2014

Interviewee is holding Master’s Degree in Computer Science and having ten years of experience in software development industry. Since five years he has been working on Agile & lean software development. His role in the projects as a technical manager working in lean software development. His organization has benefited from the lean software development in projects and it is difficult to make changes. He has worked with around thirty two team members when working with lean. Mostly the development is global and locations in China, Hungary.

Femi

Interview2

Date of interview: 15-08-2014

Interviewee2 is holding Master’s Degree in Telecommunications system and having fourteen years of experience in software development industry. Since three years he has been working on Agile & lean software development. His role in the projects as a test manager working in lean software development. His organization has improved from the lean software development in projects and it is still difficult to be adopted to changes. He has worked with around eighteen team members when working with lean. Mostly the development is global and locations in Scandinavian countries, Nigeria.

Annie

Interview3

Date of interview: 24-08-2014

Interviewee3 is holding Master’s Degree in Software Engineering and having eleven years of experience in software development industry. Since four years she has been working on Agile & lean software development. Her role in the projects as a senior developer. Her organization has not been able to improve much from the lean software development in projects and it is very difficult to be adopted to changes. She has worked with around sixty team members when working with lean. Mostly the development is global and locations in America, India.

Michael

Interview4

Date of interview: 01-09-2014
Interviewee 4 is a senior consultant and Software Architect and having more than twenty five years of experience in software development industry. More than fifteen years he is been working in Agile and few years in lean software development. His role in the projects as a senior consultant for lean software development. His organization has been able to improve much from the lean software development in projects and it is highly recommended to be adopted to changes quickly. He has worked for many projects around fourth team members in each project when working with lean. Mostly the development is global and locations in Europe, North America, Asia.

Johannes

Interview5

Date of interview: 12-08-2014

Interviewee is a senior consultant having more than twenty three years of experience in software development industry. More than fifteen years he is been working on Agile and few years in lean software development. His role in the projects as a senior consultant for lean software development. His organization has been able to improve much from the lean software development in projects and it is highly recommended to be adopted to changes quickly. He has worked for many projects around fourth team members in each project when working with lean. Mostly the development is global and locations in Europe, North America, Asia.

Rölf

Interview6

Date of interview: 27-08-2014

Interviewee is a senior consultant having more than twenty five years of experience in software development industry. More than fifteen years he is been working on Agile and lean software development. His role in the projects as a senior consultant for lean software development and acting as a coach for companies. Many companies were been able to improve much from the lean software development in projects and it is highly recommended to be adopted to changes quickly. He has worked for many projects with small team to big teams when working with lean. Mostly the development is both local but some projects are global and locations in Europe, North America.

Daniel

Interview7

Date of interview: 22-08-2014

Interviewee is a senior consultant having more than 13 five years of experience in software development industry. More than seven years she is been working on Agile and lean software development. His role in the projects as a senior consultant for lean software development and process improvement. The company is been able to improve much from the agile and lean software development in projects and it is recommended to be make changes in company quickly. She has small number of projects with small team to big teams when working with lean. Mostly the development is both local but some projects are global and locations in Sweden, China.
APPENDIX 7:

Figure 12: Global product development

Figure 13: Non-Technical factors
Figure 14: Technical factors