A Lean Model Based Outlook on Cost and Quality Optimization in Software Projects

STC 2009

Kalyanam Kannan
Abstract

A large quantum of effort and research is being invested to address the Cost and Quality factors in software projects. Though the solutions, models and methodologies are well established through experimented processes, adoption and optimization of the required parameters for a specific project to obtain predictable and acceptable quality with minimum costs has always remained a challenge.

This paper discusses the Lean process in detail with the help of project data and demonstrates that simple, affordable and adoptable processes are more economical and focused on quality. Also, it is observed that the Lean Model enables a project to be well monitored and controlled by focusing on critical elements, thereby reducing overheads of bulky documentation and irrelevant processes. The above findings are statistically analyzed using the coefficient of variation, which strikes a direct correlation with the predictable quality in a project.
Lean Model – An Overview

The quality and cost of software are two important areas under constant vigil during the process of software development and delivery. The quality of software bears a direct correlation with the thought process and steps followed to achieve the expected output. The costs involved in executing a streamlined work process and repeatable steps for software development are considered to be significant overheads in the industry.

Several teams and groups have been continuously working on a model to achieve high quality with minimum investments of effort and costs. The Lean process takes a simple approach to address this business challenge by eliminating unwanted steps in a project or product development, which translates into ample and required benefits, to remove overheads.

A Lean Model focuses on eliminating activities that do not add any value to the project and fatty processes in a lifecycle. The model also offers multiple advantages by allocating more time and effort for product critical areas and thus minimizing effort for a particular process activity, which would normally be less significant for the entire application.

In other words, elimination of wasteful activities from the quality control processes in a lifecycle helps tailor a process, which is more efficient and can be easily adhered to. The Lean Model implementation is a combined change in the strategic and operational processes. (Not too sure about controls) There are several tools and working models that need to be altered and optimized to remove or minimize wasteful activities to attain efficiency and optimization.

The generic Lean principles of software development are well established and the same is presented below for reference:

1. **Eliminate waste**: Eliminate any process activity that is not adding value to the project delivery

2. **Amplify learning**: A sustainable model to manage knowledge and repeatable process to replicate the knowledge sharing across the projects.
   *(Tools: Feedback, iterations, synchronization and set-based development)*

3. **Decide as late as possible**: Decision to incorporate the required processes for effective delivery so that the cost and effort involved for change can be avoided.
   *
   Tools: Options thinking, the last responsible moment and decision-making.*

4. **Deliver as fast as possible**: Faster delivery resulting in optimized cost and better productivity.
   *
   Tools: Pull systems, Queuing Theory and cost of delay.*
5. **Empower the team**: Encourage people to take responsibility, get motivated and work as a team, and responsibility for the outcome and authority to make it happen.  
*Tools: Self-determination, motivation, leadership and expertise.*

6. **Build integrity**: Customer’s experience with the delivery, whereas conceptual integrity is how well the architecture and system components work together to bring about the perceived integrity. Obviously, testing-unit and integration testing, is a major part of integrity.  
*Tools: Perceived integrity, conceptual integrity, refactoring and testing.*

7. **See the whole**: Holistic and end-to-end approach for projects so the end point implementation and business value is thought through during the initial stage of the project and risks identified and mitigated.

8.  
*Tools: Measurement and contracts.*

This White Paper extensively follows Lean principles in the areas of quality control and quality management by eliminating unwanted processes, and improving team integration and integrity. It also discusses the outcomes of the observations in detail.

An attempt has been made to analyze data from various projects and the effort expended in each of the processes and operational activities. The percentage effort from these activities is mapped to the Lean Model of working and the reduction in effort in the respective activity areas is predicted. The quality of the deliverables is measured using defect detection efficiency and coefficient of variation, which is a measure of risk. Finally, the said parameters and observations are compared between the Lean and Normal Models.

**Sample Study – Data Analysis and Prediction**

The aforementioned Lean Model was applied to a sample of projects from Travel domain, which consisted of Web applications, BI systems, data warehouses, staging area database and mainframe systems. The finite data on effort spent on each of the project activities, listed below for quality control, were obtained from ABTS (Activity-Based Tracking System) and the release data, such as coverage, test cases, defect leakage, were obtained from the test and project management tools.

The sample data from the above mentioned projects were studied in detail by breaking them into Planning, Analysis, Development and Testing phases, as done in any other development model, and all standard processes were monitored throughout this study.

The activity parameters given below were incorporated into the Lean Model and redistribution of the probable effort was carried out with the help of coefficient of variation for the mentioned activities. In all these cases, the risk parameter was maintained at less than 3% for projected Lean activity percentages. The results of the Lean Model were compared with those of the Standard Process Model. The results
found were encouraging and highlighted a substantial amount of efficiency, thereby reducing and eliminating processes and activities that did not value add.

The sample consisted of application development projects of the Travel domain. The activities studied, using the Standard Process Model, are given below. The parameters related to quality control are also discussed in this section with their appropriate weightings in percentages.

Table 1: Effort Distribution on Process Activity Parameters

<table>
<thead>
<tr>
<th>Phase</th>
<th>Activity Parameters</th>
<th>% Effort</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis</td>
<td>Requirement Analysis</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Requirement Clarifications</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Technology Feasibility Analysis</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Quality Strategy</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Effort Estimation</td>
<td>4</td>
</tr>
<tr>
<td>Design</td>
<td>Test Strategy</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>High Level Test Scenarios</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Test Reviews</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>RTM</td>
<td>5</td>
</tr>
<tr>
<td>Development</td>
<td>Test Plan Development</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Test Case Development</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>Test Review</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Design-Test Case Updation</td>
<td>5</td>
</tr>
<tr>
<td>Testing</td>
<td>Test Execution</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>Defect Logging</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Defect Analysis</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Test Documentation</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Test Signoff</td>
<td>2</td>
</tr>
</tbody>
</table>

The table above provides effort distribution across various activities to achieve a controlled and monitored quality process for predictable software releases.

The data in the table and the proportion of efforts are correlated with the quality of projects, which is measured using defect detection efficiency as one of the key parameters, while quality index was measured using Test Maturity Model-based indices. The available reports based on data and overall risk, derived from quality index, are represented in the graphs below:
The graphs above provide details of the sample projects in terms of Quality Index (QI), Defect Detection Efficiency (DDE) and Coefficient of Variation (CoV).

Figure 1 provides details of those trends in the projects, which followed standard processes based on the Capability Maturity Model (CMM) and it is observed that the average quality index, DDE and CoV were around 7, 85% and 30 respectively.

Figure 2 provides simulated data for Lean Model and it is observed that the average quality index, DDE and CoV were around 8, 90% and 25, in that order.

The improvement in quality from an overall perspective is discussed in detail in the sections below.

The aforesaid sample projects were fitted into the framework of Lean Development Model, in which the non-value add processes were minimized by using efficient tools and thinner documentation processes.

The proven seven stages of the Lean Model were applied in each and every stage, and the benefits were compounded in each of these areas and the overall efficiency was observed. The overall Lean Process implementation was provided in a tabular format.
The reduction in effort due to usage of the Lean Model was based on iterations, where the CoV (risk) did not exceed a value of 30. Also, it should be noted that the below mentioned effort distribution is prone to change if the absolute value of CoV is altered. In general, higher the risk coefficient, more is the effort reduction using the Lean Model. This also provides a wide range of scope for Risk-Based Testing (RBT) and is a promising area for further analysis and research.

Figure 3 depicts the difference between the Normal Process-based approach and Lean Process-based approach for quality index values. Although both the models use the same set of activities towards delivery, the percentage and amount of focus provided for each of the activity is better managed in the Lean Process, which is likely to yield better quality behavior as displayed above.

**Lean Model Analysis and Discussions**

It is observed by means of this hypothesis that around 29% of effort in quality control can be saved and utilized for various other purposes. The Lean Model gains importance because the effort saved can be used in the areas of coverage and reviews, which enhance the quality and predictability of software. The reduction in effort has direct impact on cost savings, and the processes can be fine-tuned to achieve cost and effort optimization.

The findings on quality of the projects taken as sample provide the following indicators and help us identify the areas of importance to achieve quality and cost reduction.

*Table 2: Percentage effort distribution in normal and lean process models*

<table>
<thead>
<tr>
<th>Activity</th>
<th>% Effort Distribution</th>
<th>Estimate Waste</th>
<th>Effective Learning</th>
<th>Decide Late</th>
<th>Deliver Fast</th>
<th>Empower Team</th>
<th>Integrity</th>
<th>Holistic View</th>
<th>Lean Distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirement Analysis</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>6.0</td>
</tr>
<tr>
<td>Requirement Clarifications</td>
<td>5</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2.9</td>
</tr>
<tr>
<td>Technology Feasibility Analysis</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3.0</td>
</tr>
<tr>
<td>Quality Strategy</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4.0</td>
</tr>
<tr>
<td>Effort Estimation</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>3.0</td>
</tr>
<tr>
<td>Test Strategy</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>2.7</td>
</tr>
<tr>
<td>High Level Test Scenarios</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3.0</td>
</tr>
<tr>
<td>Test Reviews</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>2.1</td>
</tr>
<tr>
<td>RTM</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4.1</td>
</tr>
<tr>
<td>Test Plan Development</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>4.3</td>
</tr>
<tr>
<td>Test Case Development</td>
<td>18</td>
<td>20</td>
<td>15</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>10</td>
<td>14.3</td>
</tr>
<tr>
<td>Test Review</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>3.6</td>
</tr>
<tr>
<td>Design-Test Case Upgradation</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.0</td>
</tr>
<tr>
<td>Test execution</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>10</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>10</td>
<td>10.6</td>
</tr>
</tbody>
</table>
The figure above provides an insight into effort distribution in a Quality Control Model. It is obvious from the graph that the critical steps or processes were accorded more importance and weightings as compared to the Standard Model. This helps address the issues of quality and cost in an effective manner. A greater amount of effort for requirement analysis, quality strategy and effort estimation considerably decreased the test case development and test execution efforts. A holistic approach and understanding business criticality of the application or project under development resulted in decreased effort during the testing phase and minimized the wastages on clarifications.

**Elimination of Waste**

Elimination of waste in the sample projects discussed above was based on the following:

---

**Figure 4: Percentage effort distribution in normal and lean process models**

<table>
<thead>
<tr>
<th></th>
<th>% Effort Distribution</th>
<th>Estimate Waste</th>
<th>Effective Learning</th>
<th>Decide Late</th>
<th>Deliver Fast</th>
<th>Empower Team</th>
<th>Integrity</th>
<th>Holistic View</th>
<th>Lean Distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defect Logging</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2.0</td>
</tr>
<tr>
<td>Defect Analysis</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1.0</td>
</tr>
<tr>
<td>Test Documentation</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.0</td>
</tr>
<tr>
<td>Test Signoff</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2.0</td>
</tr>
<tr>
<td>Overall View</td>
<td>100</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>70.6</td>
</tr>
</tbody>
</table>
- Development of a software / code and a test case, which were waiting to be tested or integrated
- Incapability to test the code due to non-availability of any other components
- Development of irrelevant test cases, which were not to be used immediately
- Wait for clarifications and defect details
- Excess documentation or effort duplication in terms of reporting

In other words, as per Lean thinking, anything not adding any value to the project was considered to be waste. This included:

- unnecessary test cases and excessive unwanted coverage
- Unclear requirements and unnecessary documentation
- Complex communication procedures and non-agreed upon engagement models

In order to be able to eliminate waste, one should be able to recognize it. For example:

- If some activity could be bypassed or the result could be achieved without it, it is waste.
- Any quality control activity that is not completed and needs to be repeated is also a waste.
- Extra processes and features not often used by project teams are waste.
- Waiting for other activities and teams for integration of code and related processes is also considered as waste.
- Defects of lower quality are waste. Managerial overheads not generating real value are waste.

In the sample projects, a value stream mapping was carried out and waste was minimized, as reflected in Table 2.

Most of the waste was eliminated during phases of requirements understanding and requirement mapping to the test cases, ultimately saving more effort for test development and execution. A better understanding of the overall project and realistic effort estimates also proved helpful in elimination of wastages from the project.

As per an analysis, reduction of 30% waste -- in terms of effort due to requirement understanding, proper effort estimation and overall business understanding of the project -- helped trim the process activity efforts. The quality of the deliverables, measured in terms of defect leakage and risk (CoV), also showed satisfactory quality levels.
**Amplify learning – Feedback mechanism**

The best approach to improve quality and amplify learning within the teams is to get timely feedback from the testing and quality teams. Most of the defects should be prevented by running tests as soon as the code is written and delivered. Instead of adding more documentation or making detailed plans, different ideas should be experimented with by writing code. The process of user requirements gathering should also be simplified by presenting screens to the end-users and asking them for inputs through a simple, workable process.

This specific Lean activity plays a major role in Lean quality control. The high-impact area or the major beneficiary of this activity is the development team. A considerable amount of effort can be saved by getting feedback from the testing team immediately after the software release. More the knowledge of the testing and quality teams, higher are the benefits of integration and business-based acceptance of the software with minimum defects.

The sample study predicted that improving the business knowledge of the testing and quality teams benefited the project by:

- Providing faster feedback to the development teams
- Providing end-to-end test scenarios for integration of code
- Reducing the effort in the development of unwanted test conditions for a specific release
- Reducing the time delay in release process during production

The simulations based on the above-mentioned benefits predicted a 15% reduction in cycle time due to integrated and appropriate learning and sharing mechanisms. This also involved setting up a centralized knowledge management system and planned meetings during the project period.

The findings were in line with the definition of learning given below:

“The learning process is sped up by usage of short iteration cycles – each one coupled with refactoring and integration testing. Increasing feedback via short feedback sessions with customers helps when determining the current phase of development and adjusting efforts for future improvements. During those short sessions, both customer representatives and the development team learn more about the domain problem and figure out possible solutions for further development.

Thus, the customers better understand their needs, based on the existing results of development efforts, and the developers learn how to better satisfy those needs. Another idea in the communication and learning process with a customer is set-based development – this concentrates on communicating the constraints of the future solution and not the possible solutions, thus, promoting the birth of the solution via dialog with the customer.”
**Decide as late as possible – Minimize assumptions and predictions**

There are several aspects to be taken care of in a complex software development environment. The approach of deciding the change as late as possible, will help reduce assumptions and predictions during the course of the project and get real requirements and clarifications based on a simple and effective process. This prevents a lot of rework in the later stages. The iterative approach promotes a simple principle – the ability to adapt to changes and correct mistakes – which might be very costly if discovered after the release of the software.

In a nutshell, this approach prescribes efforts only on a complete and clear understanding of requirements, and advises against investment of any efforts in assumptions and predictions, which may need re-work (end up in re-work?????) during the final stages of the project.

The sample study and simulated data concluded that around 10% of excess effort was being wasted due to commencement of project activities without a clear understanding of the requirements. The same is also reiterated in the Agile Development Model.

**Deliver as fast as possible – Decrease iteration time**

The sample projects developed using Agile Development Model responded well to the dynamic customer needs and demands in a shorter period of time.

Caution should be exercised in communicating and following integrated team approach during development and testing, so there are no discrepancies between the perceived product/project and the actual product/project developed.

Business requirements have become more dynamic due to compelling needs of the industry, and therefore, the processes and methodologies should be adaptable, flexible and effective, to facilitate faster delivery. The Lean Model exactly falls in line with the business needs and helps the customers decide what they really need.

**Empower the team – Decisive team and proper engagement**

Team engagement and working agreement between business owners, development teams and quality teams play a very critical role in the success and quality of delivery. It is important to agree on a simple process, through which requirements and changes will be communicated and clarifications will be addressed. The decisiveness of the identified members of the team adds value and proves to be the central success factor for the Lean Model.

Defect management and acceptance of the project heavily depends on the engagement model as it influences the cycle time and creates a chain effect throughout the lifecycle.

The sample project data revealed the delay and ambiguity during development, testing and acceptance due to lack of decisions at various stages, which resulted in a
cumulative effect leading to delay and shortening of testing cycles. This resulted in lesser test coverage and delays in delivery.

**Building integrity – Deliver business value**

Integrity of software is rated on the basis of various factors like flexibility, maintainability, efficiency and responsiveness. This is normally achieved through face to face discussions and finally delivers a project full of potential that yields the expected outcome and addresses the business requirements with minimum possible efforts. The process of building integrity also proves that understanding of business requirements is critical to test a project or product for its long-term usability and its ability to cater to the changing market needs.

**See the whole – Holistic approach**

Software quality should not be restricted to testing; it should also cover and encompass the entire software development lifecycle. It is interesting to note that all activities in the lifecycle are interdependent. A holistic approach to the entire development model, with minimum processes and small repeatable activities, is the key differentiator of the Lean Model.

The sample study demonstrated that there was a considerable reduction in effort, with faster releases, when quality was controlled through a managed model, brought about by interaction among various project stakeholders. This established the effectiveness of various small processes -- like stand-up meetings, status updates, triage priorities, daily feedback mechanism, etc. -- in a day-to-day working environment.

**Conclusion**

The sample projects provided some vital data, which evidenced that efficiency could be enhanced and processes could be improved by using the Lean Model:

- The cost to quality and subsequently, the effort is likely to be reduced by 30% by using the Lean Model. Focusing on vital tasks like requirement understanding and making strategies also leads to effort reduction in the test case development and execution phases.
- The predicted reduction in effort and increase in productivity is attributed to elimination of wastes like waiting period for clarifications, and planning the test cases and activities in the right sequence, so testing was not affected due to non-availability of integration and systems. At the time of writing this paper, the team was working at around 80-85% productivity and going Lean was expected to enhance this to around 95%.
- It is expected that the cycle time will be reduced by 30% owing to effective feedback from testing, automated build and deployment practices.

The Lean Model seems to be the future of software quality control as it falls in line with the industry slogan “Enhanced Quality and Reduced Cost.” Optimization of this model and its customization for specific projects would lead to benefits as it recommends execution of only those activities that bring value add to the project and thereby, provide more time for and focus more on the critical success factors in a project.
It is also very vital to note that suitable and capable resources are a pre-condition to adopt the Lean Model for a project.

References:

1. Mary Poppendieck, Tom Poppendieck *Lean Software Development: An Agile Toolkit for Software Development Managers*

2. Lean Software Institute - Testing in Lean Software Development